Guía del usuario

AWS Identity and Access Management

AWS Identity and Access Management: Guía del usuario

Table of Contents

¿Qué es IAM?

Follow us on Twitter

AWS Identity and Access Management (IAM) es un servicio web que lo ayuda a controlar de forma segura el acceso a los recursos de AWS. Con IAM, puede administrar de forma centralizada los permisos que controlan a qué recursos de AWS pueden acceder los usuarios. Utilice IAM para controlar quién está autenticado (ha iniciado sesión) y autorizado (tiene permisos) para utilizar recursos.

Cuando se crea una cuenta de Cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los servicios y recursos de Servicios de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS y se accede a ella iniciando sesión con el email y la contraseña que utilizó para crear la cuenta. Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para obtener la lista completa de tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz.

Contenido

- Uso de IAM con un SDK de AWS

Video de introducción a IAM

La formación y certificación de AWS ofrece una introducción en vídeo de 10 minutos a IAM:

Introducción a AWS Identity and Access Management

Características de IAM

IAM le ofrece las siguientes características:

Acceso compartido a la Cuenta de AWS

Puede conceder permiso a otras personas para administrar y utilizar los recursos de su cuenta de AWS sin tener que compartir su contraseña o clave de acceso.

Permisos detallados

Puede conceder diferentes permisos a diferentes personas para diferentes recursos. Por ejemplo, puede permitir que algunos usuarios completen el acceso a Amazon Elastic Compute Cloud (Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, Amazon Redshift y otros servicios de AWS. En el caso de otros usuarios, puede permitir el acceso de solo lectura a solo algunos buckets de S3 o conceder permiso para administrar solo algunas instancias EC2 o para tener acceso a la información de facturación, pero nada más.

Acceso seguro a los recursos de AWS para aplicaciones que se ejecutan en Amazon EC2

Puede utilizar características de IAM para proporcionar de forma segura credenciales para las aplicaciones que se ejecutan en instancias EC2. Estas credenciales proporcionan permisos a la aplicación para obtener acceso a otros recursos de AWS. Entre los ejemplos se incluyen buckets de S3 y tablas de DynamoDB.

Autenticación multifactor (MFA)

Puede agregar una autenticación de dos factores a la cuenta y a los usuarios individuales para mayor seguridad. Con MFA usted o sus usuarios deben proporcionar no solo una contraseña o clave de acceso para trabajar con la cuenta, sino también un código de un dispositivo configurado específicamente. Si ya utiliza una clave de seguridad FIDO con otros servicios, y tiene una configuración de AWS compatible, puede utilizar WebAuthn para la seguridad MFA. Para obtener más información, consulte Configuraciones admitidas para usar las claves de seguridad FIDO.

Identidad federada

Puede permitir que los usuarios que ya tienen contraseñas en otros lugares, por ejemplo, en la red corporativa o en un proveedor de identidad de Internet, obtengan acceso temporal a la Cuenta de AWS.

Información de identidad para realizar un control

Si utiliza AWS CloudTrail, recibirá registros de logs que incluyen información sobre los usuarios que realizaron solicitudes de recursos en su cuenta. Esta información se basa en identidades de IAM.

Conformidad con DSS de PCI

IAM admite el procesamiento, el almacenamiento y la transmisión de datos de tarjetas de crédito por parte de un comerciante o proveedor de servicios y se ha validado por estar conforme con el Estándar de Seguridad de Datos para la Industria de Tarjeta de Pago (PCI DSS). Para obtener más información acerca de PCI DSS, incluido cómo solicitar una copia del Paquete de conformidad con PCI de AWS, consulte PCI DSS Nivel 1.

Integración con muchos servicios de AWS

Para obtener una lista de servicios de AWS que funcionan con IAM, consulte Servicios de AWS que funcionan con IAM.

Consistencia final

IAM, al igual que muchos otros servicios de AWS, ofrece consistencia final. IAM consigue una alta disponibilidad replicando datos entre varios servidores ubicados en centros de datos de Amazon de todo el mundo. Si se realiza correctamente una solicitud para cambiar algunos datos, el cambio se confirma y se almacena de forma segura. Sin embargo, el cambio se debe replicar en IAM, lo que puede llevar algún tiempo. Estos cambios incluyen la creación o actualización de usuarios, grupos, roles o políticas. Le recomendamos que no incluya esos cambios de IAM en las rutas de código de gran importancia y alta disponibilidad de su aplicación. En su lugar, realice los cambios de IAM en otra rutina de inicialización o configuración que ejecute con menos frecuencia. Además, asegúrese de verificar que los cambios se han propagado antes de que los flujos de trabajo de producción dependan de ellos. Para obtener más información, consulte Los cambios que realizo no están siempre visibles inmediatamente.

Uso gratuito

AWS Identity and Access Management (IAM) y AWS Security Token Service (AWS STS) son características de la cuenta de AWS que se ofrece sin cargo adicional. Solo se le cobrará cuando

acceda a otros servicios AWS utilizando sus usuarios de IAM o AWS STScredenciales de seguridad temporales. Para obtener información acerca de los precios de otros productos de AWS, consulte la Página de precios de Amazon Web Services.

Acceso a IAM

Puede trabajar con AWS Identity and Access Management de cualquiera de las siguientes formas.

AWS Management Console

La consola es una interfaz basada en navegador para administrar los recursos de IAM y AWS. Para obtener más información acerca de cómo acceder a IAM mediante la consola, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In.

Herramientas de línea de comandos de AWS

Puede utilizar las herramientas de línea de comandos de AWS para emitir comandos en la línea de comando de su sistema con el fin de llevar a cabo tareas de IAM y de AWS. El uso de la línea de comandos puede ser más rápido y cómodo que la consola. Las herramientas de línea de comandos también son útiles si desea crear scripts que realicen tareas de AWS.

AWS proporciona dos conjuntos de herramientas de línea de comandos: AWS Command Line Interface (AWS CLI) y la AWS Tools for Windows PowerShell. Para obtener información acerca de la instalación y el uso de la AWS CLI, consulte la Guía del usuario de AWS Command Line Interface. Para obtener información sobre cómo instalar y utilizar Tools for Windows PowerShell, consulte la Guía del usuario de AWS Tools for Windows PowerShell.

Después de iniciar sesión en la consola, puede utilizar AWS CloudShell desde su navegador para ejecutar comandos de CLI o SDK. Los permisos para acceder a los recursos de AWS se basan en las credenciales que utilizó para iniciar sesión en la consola. Según su experiencia, es posible que la CLI le parezca un método más eficiente para administrar su Cuenta de AWS. Para obtener más información, consulte Utilización de AWS CloudShell para trabajar con AWS Identity and Access Management

SDK de AWS

AWS ofrece SDK (kits de desarrollo de software) que se componen de bibliotecas y código de muestra para diversos lenguajes de programación y plataformas (Java, Python, Ruby, .NET, iOS, Android, etc.). Los SDK proporcionan una forma cómoda de crear acceso mediante programación a IAM y AWS. Por ejemplo, los SDK se encargan de tareas como firmar solicitudes

criptográficamente, gestionar los errores y reintentar las solicitudes de forma automática. Para obtener información sobre los SDK de AWS, incluido cómo descargarlos e instalarlos, consulte la página Herramientas para Amazon Web Services.

API de consulta de IAM

Puede acceder a IAM y AWS mediante programación con la API de consulta de IAM, que le permite emitir solicitudes HTTPS directamente al servicio. Cuando utilice la API de consulta, debe incluir un código para firmar de manera digital las solicitudes con sus credenciales. Para obtener más información, consulte Llamar a la API de IAM mediante solicitudes de consulta HTTP y la Referencia IAM API.

¿Cuándo uso IAM?

Cuando lleva a cabo diferentes funciones de trabajo

AWS Identity and Access Management es un servicio de infraestructura central que proporciona la base para el control de acceso basado en identidades en AWS. Utiliza IAM cada vez que accede a su cuenta de AWS.

La forma en que utiliza IAM difiere en función del trabajo que realiza en AWS.

- Usuario de servicio: si utiliza un servicio de AWS para realizar el trabajo, el administrador le proporciona las credenciales y los permisos necesarios. Es posible que a medida que utilice características más avanzadas para realizar su trabajo, necesite permisos adicionales. Entender cómo se administra el acceso puede ayudarlo a solicitar los permisos correctos a su administrador.
- Administrador de servicio: si está a cargo de un recurso de AWS en la empresa, probablemente tenga acceso completo a IAM. Su trabajo consiste en determinar a qué características y recursos de IAM deben acceder los usuarios del servicio. Luego, debe enviar solicitudes a su administrador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM.
- Administrador de IAM: si es un administrador de IAM, administra identidades de IAM y escribe políticas para administrar el acceso a IAM.

Cuando tiene autorización para acceder a los recursos de AWS

La autenticación es la manera de iniciar sesión en AWS mediante credenciales de identidad. Debe estar autenticado (haber iniciado sesión en AWS) como rootlong, como un usuario de IAM o asumiendo un rol de IAM.

Puede iniciar sesión en AWS como una identidad federada mediante las credenciales proporcionadas a través de un origen de identidad. AWS IAM Identity Center Los usuarios (del Centro de identidades de IAM), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su administrador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accede a AWS mediante la federación, está asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en AWS Management Console o en el portal de acceso AWS. Para obtener más información sobre el inicio de sesión en AWS, consulte Cómo iniciar sesión en su cuenta de AWS en la Guía del usuario de AWS Sign-In.

Si accede a AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de la línea de comandos (CLI) para firmar criptográficamente las solicitudes mediante el uso de las credenciales. Si no usa las herramientas de AWS, debe firmar usted mismo las solicitudes. Para obtener más información sobre la firma de solicitudes, consulte Firma de solicitudes API de AWS en la Guía del usuario de IAM.

Independientemente del método de autenticación que utilice, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, AWS le recomienda el uso de la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactor en la Guía del usuario de AWS Single Sign-On y Uso de la autenticación multifactor (MFA) en AWS en la Guía del usuario de IAM.

Cuando inicia sesión como un usuario de IAM

Un usuario de IAM es una identidad de la cuenta de AWS que dispone de permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del Usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, podría tener un grupo cuyo nombre fuese IAMAdmins y conceder permisos a dicho grupo para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales permanentes a largo plazo y los roles proporcionan credenciales temporales. Para obtener más información, consulte Cuándo crear un usuario de IAM (en lugar de un rol) en la Guía del usuario de IAM.

Cuando asume un rol de IAM

Un rol de IAM es una identidad de la Cuenta de AWS que dispone de permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una determinada persona. Puede asumir temporalmente un rol de IAM en la AWS Management Console cambiando de roles. Puede asumir un rol llamando a una operación de AWS CLI o de la API de AWS, o utilizando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulte Uso de roles de IAM en la Guía del Usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir permisos para este. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos que están definidos en este. Para obtener información acerca de roles para federación, consulte Creación de un rol para un proveedor de identidades de terceros en la Guía del Usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué pueden acceder las identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS Single Sign-On.
- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. No obstante, con algunos Servicios de AWS se puede asociar una política directamente a un recurso (en lugar de utilizar un rol como representante). Para

obtener información sobre la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulte Cómo los roles de IAM difieren de las políticas basadas en recursos en la Guía del usuario de IAM.

- Acceso entre servicios: algunos servicios de AWS utilizan características de otros Servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado a servicios.
  - Sesiones de acceso directo (FAS): cuando utiliza un usuario o rol de IAM para realizar acciones en AWS, se considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS usa los permisos de la entidad principal que llama un Servicio de AWS, junto con la solicitud de Servicio de AWS, para realizar solicitudes a los servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros recursos o Servicios de AWS para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información detallada sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar las sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un servicio de AWS en la Guía del usuario de IAM.
  - Rol vinculado a los servicios: un rol vinculado a servicios es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.
- Aplicaciones que se ejecutan en Amazon EC2: puede utilizar un rol de IAM que le permita administrar credenciales temporales para las aplicaciones que se ejecutan en una instancia EC2 y realizan solicitudes a la AWS CLI o a la API de AWS. Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia EC2. Para asignar un rol de AWS a una instancia de EC2 y ponerla a disposición de todas las aplicaciones, cree un perfil de instancia asociado a la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia EC2 obtener credenciales temporales. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias de Amazon EC2 en la Guía del usuario de IAM.

Para obtener información sobre el uso de los roles de IAM, consulte Cuándo crear un rol de IAM (en lugar de un usuario) en la Guía del usuario de IAM.

Cuando crea políticas y permisos

Los permisos se conceden a un usuario creando una política, que es un documento que elabora una lista de las acciones que puede realizar un usuario y los recursos que afectan a dichas acciones. De forma predeterminada, todas las acciones o recursos que no se permiten de forma explícita se deniegan. Las políticas se pueden crear y asociar a entidades principales (usuarios, grupos de usuarios, roles asumidos por usuarios y recursos).

Las políticas se utilizan con un rol de IAM:

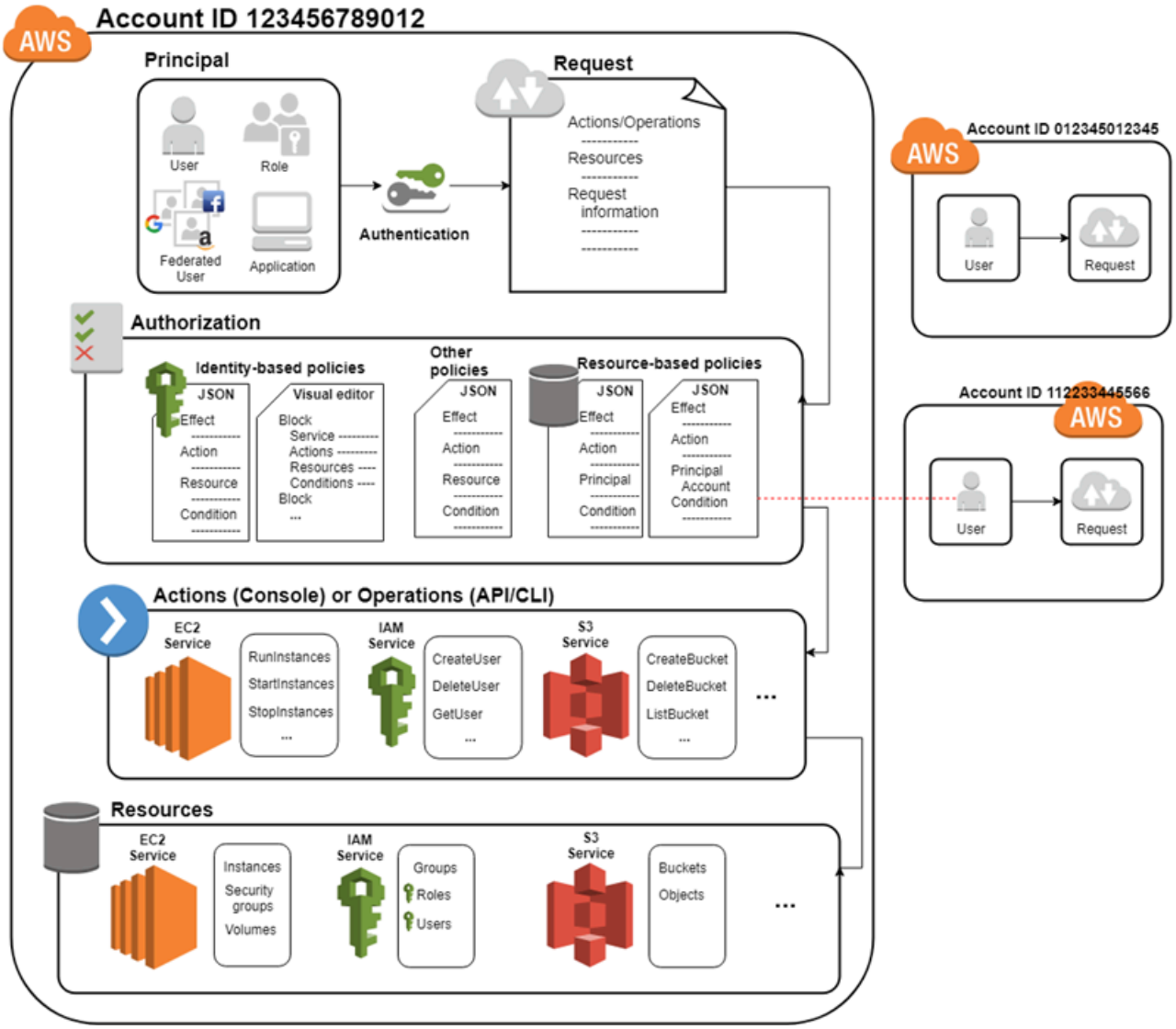
- Política de confianza: define qué entidades principales pueden asumir el rol y en qué condiciones. Una política de confianza es un tipo específico de política basada en recursos para roles de IAM. Un rol solo puede tener una política de confianza.
- Políticas basadas en identidad (insertadas y administradas): estas políticas definen los permisos que se le conceden (o se le deniegan) al usuario del rol y en qué recursos puede o no implementarlos.

Utilice Ejemplos de políticas basadas en identidad de IAM para definir los permisos para las identidades de IAM. Cuando encuentre la política que necesita, elija ver la política para ver el JSON de la política. Puede utilizar el documento de política JSON como plantilla de sus propias políticas.

> **Note**
>
> Si utiliza IAM Identity Center para administrar los usuarios, asigna conjuntos de permisos en IAM Identity Center en lugar de asociar una política de permisos a una entidad principal. Cuando asigna un conjunto de permisos a un grupo o usuario en AWS IAM Identity Center, IAM Identity Center crea los roles de IAM correspondientes en cada cuenta y adjunta a esos roles las políticas especificadas en el conjunto de permisos. IAM Identity Center administra el rol y permite que los usuarios autorizados que usted definió asuman el rol. Si modifica el conjunto de permisos, IAM Identity Center garantiza que las políticas y los roles de IAM correspondientes se actualicen en consecuencia.
> Para obtener más información, consulte What is IAM Identity Center? (¿Qué es el Centro de identidades de IAM?) en la Guía del usuario de AWS IAM Identity Center.

Cómo funciona IAM

IAM proporciona la infraestructura necesaria para controlar la autenticación y la autorización de su Cuenta de AWS. El siguiente diagrama ilustra la infraestructura de IAM:



En primer lugar, un usuario humano o una aplicación utiliza las credenciales de inicio de sesión para autenticarse en AWS. La autenticación se proporciona cuando coinciden las credenciales de inicio de sesión con una entidad principal (un usuario de IAM, un usuario federado, un rol de IAM o una aplicación) en la que confía la Cuenta de AWS.

Luego, se realiza una solicitud para conceder a la entidad principal acceso a los recursos. El acceso se concede en respuesta a una solicitud de autorización. Por ejemplo, cuando inicia sesión en la consola por primera vez y se encuentra en la página de inicio de la consola, no accede a un servicio

específico. Cuando selecciona un servicio, la solicitud de autorización se envía a ese servicio y se comprueba si su identidad aparece en la lista de usuarios autorizados, qué políticas se aplican para controlar el nivel de acceso concedido, y cualquier otra política que pueda estar en vigor. Las entidades principales de su Cuenta de AWS o de cualquier otra Cuenta de AWS de confianza pueden realizar solicitudes de autorización.

Una vez autorizada, la entidad principal puede actuar o llevar a cabo operaciones en los recursos de su Cuenta de AWS. Por ejemplo, la entidad principal puede lanzar una instancia de Amazon Elastic Compute Cloud nueva, modificar las suscripciones a un grupo de IAM o eliminar buckets de Amazon Simple Storage Service.

Conceptos básicos

- Términos
- Entidad principal
- Solicitud
- Autenticación
- Autorización
- Acciones u operaciones
- Recursos

Términos

Estos términos de IAM se utilizan habitualmente cuando se trabaja con AWS:

Recurso de IAM

Los recursos de IAM se almacenan en IAM. Puede agregarlos, editarlos y eliminarlos de IAM.

- user
- grupo
- role
- política
- objeto de proveedor de identidades

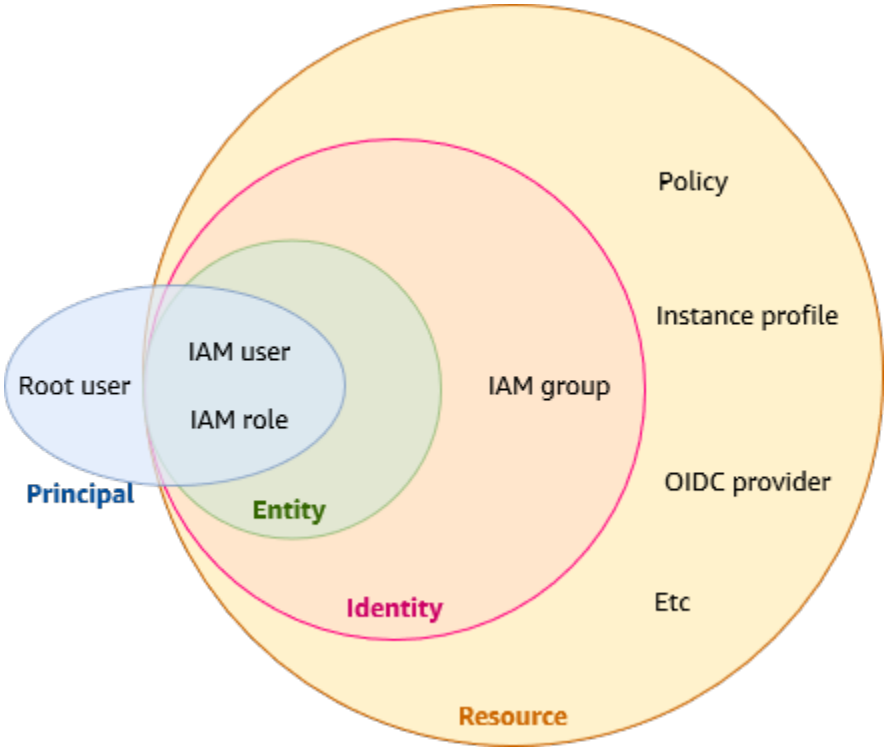
Entidad de IAM

Recursos de IAM que AWS utiliza para la autenticación. Las entidades se pueden especificar como entidad principal en una política basada en recursos.

- user
- role

Identidad de IAM

Recurso de IAM que se puede autorizar en políticas para realizar acciones y acceder a recursos. Las identidades incluyen usuarios, grupos y roles.



Entidades principales

Una persona o una aplicación que utiliza el Usuario raíz de la cuenta de AWS, un usuario de IAM o un rol de IAM para iniciar sesión y realizar solicitudes a AWS. Las entidades principales incluyen usuarios federados y roles asumidos.

Usuarios humanos

También conocidas como identidades humanas; las personas, administradores, desarrolladores, operadores y consumidores de sus aplicaciones.

Carga de trabajo

Un conjunto de recursos y código que aporta valor empresarial, como una aplicación o un proceso backend. Puede incluir aplicaciones, herramientas operativas y componentes.

Entidad principal

Una entidad principal es un usuario humano o una carga de trabajo que puede solicitar una acción u operación en un recurso de AWS. Después de la autenticación, la entidad principal de seguridad puede recibir credenciales permanentes o temporales para llevar a cabo solicitudes a AWS, en función del tipo de entidad principal de seguridad. A los usuarios de IAM y al usuario raíz se les otorgan credenciales permanentes, mientras que a los roles se les otorgan credenciales temporales. Como práctica recomendada, le recomendamos que exija a los usuarios humanos y a las cargas de trabajo que accedan a los recursos de AWS mediante credenciales temporales.

Solicitud

Cuando una entidad principal intenta utilizar la AWS Management Console, la API de AWS o la AWS CLI, la entidad principal envía una solicitud a AWS. La solicitud incluye la información siguiente:

- Acciones u operaciones: las acciones u operaciones que la entidad principal desea realizar. Puede tratarse de una acción en la AWS Management Console o una operación en la AWS CLI o la API de AWS.
- Recursos: el objeto de recurso de AWS sobre el que se realizan las acciones u operaciones.
- Principal: persona o aplicación que utilizó una entidad (usuario o rol) para enviar la solicitud. La información sobre el principal incluye las políticas asociada a la entidad que el principal ha utilizado para iniciar sesión.
- Datos de entorno: información sobre la dirección IP, el agente de usuario, el estado de habilitación de SSL o la hora del día.
- Datos de recursos: datos relacionados con el recurso que se está solicitando. Esto puede incluir información como, por ejemplo, un nombre de tabla de DynamoDB o una etiqueta de una instancia Amazon EC2.

AWS recopila la información sobre la solicitud en un contexto de solicitud, que se utiliza para evaluar y autorizar la solicitud.

Autenticación

Una entidad principal autenticarse (con sesión iniciada en AWS) utilizando sus credenciales para enviar una solicitud a AWS. Algunos servicios, como, por ejemplo, Amazon S3 y AWS STS, permiten algunas solicitudes de los usuarios anónimos. Sin embargo, son la excepción a la regla.

Para autenticarse desde la consola como usuario raíz, debe iniciar sesión con su dirección de correo electrónico y contraseña. Como usuario federado, su proveedor de identidades le ha autenticado y se le concede acceso a los recursos de AWS con roles de IAM. Como usuario de IAM, proporcione el ID de la cuenta o alias y, a continuación, su nombre de usuario y contraseña. Para autenticar cargas de trabajo desde la API o la AWS CLI, puede utilizar credenciales temporales mediante la asignación de un rol o puede utilizar credenciales a largo plazo proporcionando su clave de acceso y clave secreta. También es posible que tenga que proporcionar información de seguridad adicional. Como práctica recomendada, AWS recomienda que utilice la autenticación multifactor (MFA) y credenciales temporales para aumentar la seguridad de la cuenta. Para obtener más información acerca de las entidades de IAM que AWS puede autenticar, consulte Usuarios de IAM y Roles de IAM.

Autorización

Asimismo, debe ser autorizado (admitido) para completar su solicitud. Durante la autorización, AWS utiliza los valores del contexto de la solicitud para comprobar las políticas que se aplican a la solicitud. A continuación, utiliza las políticas para determinar si se debe permitir o denegar la solicitud. La mayoría de las políticas se almacenan en AWS como documentos JSON y especifican los permisos de las entidades principales. Existen varios tipos de políticas que pueden afectar a la autorización de una solicitud. Para proporcionar a los usuarios los permisos necesarios para tener acceso a los recursos de AWS de su propia cuenta, solo necesitará políticas basadas en identidad. Las políticas basadas en recursos se suelen utilizar para conceder acceso entre cuentas. Los demás tipos de políticas son características avanzadas y deben utilizarse con cuidado.

AWS comprueba cada política que se aplica al contexto de una solicitud. Si una sola política de permisos incluye una acción denegada, AWS deniega toda la solicitud y deja de evaluarla. Esto se denomina una denegación explícita. Dado que las solicitudes se deniegan de forma predeterminada, AWS autoriza una solicitud únicamente si las políticas de permisos aplicables permiten todas las partes de la solicitud. La lógica de evaluación de una solicitud dentro de una cuenta individual se rige por las normas generales siguientes:

- De forma predeterminada, se deniegan todas las solicitudes. (Por lo general, las solicitudes realizadas con las credenciales de Usuario raíz de la cuenta de AWS para los recursos de la cuenta siempre se autorizan).
- Un permiso explícito en cualquier política de permisos (basada en identidad o en recursos) anula esta opción predeterminada.

- La existencia de una SCP de Organizaciones, un límite de permisos de IAM o una política de sesión anula el permiso. Si existen uno o varios de estos tipos de políticas, todos ellos deben permitir la solicitud. De lo contrario, se deniega implícitamente.
- Una denegación explícita en cualquier política invalida cualquier permiso concedido.

Para obtener más información acerca de cómo se evalúan todos los tipos de políticas, consulte Lógica de evaluación de políticas. Si debe realizar una solicitud referida a otra cuenta, una política de la otra cuenta debe permitirle el acceso al recurso y, además, la entidad de IAM que utilice para realizar la solicitud debe tener una política basada en identidad que permita la solicitud.

Acciones u operaciones

Una vez que la solicitud se ha autenticado y se ha autorizado, AWS aprueba las acciones u operaciones de la solicitud. Las operaciones se definen mediante un servicio e incluyen las cosas que se pueden hacer con un recurso, como visualizar, crear, editar y eliminar dicho recurso. Por ejemplo, IAM admite aproximadamente 40 acciones para un recurso de usuario, incluidas las siguientes:

- `CreateUser`
- `DeleteUser`
- `GetUser`
- `UpdateUser`

Para permitir a una entidad principal realizar una operación, debe incluir las acciones necesarias en una política que se aplica a la entidad principal o al recurso afectado. Para ver una lista de acciones, tipos de recursos y claves de condición compatibles con cada servicio, consulte Acciones, Recursos y Claves de condición para servicios deAWS.

Recursos

Después de que AWS aprueba las operaciones de una solicitud, estas se pueden realizar en los recursos relacionados dentro de la cuenta. Un recurso es un objeto que existe dentro de un servicio. Entre los ejemplos se incluyen una instancia Amazon EC2, un usuario de IAM y un bucket de Amazon S3. El servicio define un conjunto de acciones que se pueden llevar a cabo en cada recurso. Si crea una solicitud para llevar a cabo una acción independiente en un recurso, dicha solicitud se deniega. Por ejemplo, si solicita eliminar un rol de IAM, pero proporciona un recurso de grupo de

IAM, la solicitud fallará. Para ver tablas de servicio de AWS que identifican qué recursos se ven afectados por una acción, consulte Acciones, Recursos y Claves de condición para servicios de AWS.

Información general sobre la AWS administración de identidades: los usuarios

Puede conceder acceso a su Cuenta de AWS a usuarios específicos y proporcionarles permisos específicos para acceder a los recursos de la Cuenta de AWS. Puede utilizar IAM y AWS IAM Identity Center para crear usuarios nuevos o federar usuarios existentes en AWS. La diferencia principal entre ambos es que los usuarios de IAM reciben credenciales a largo plazo para los recursos de AWS, mientras que los usuarios de IAM Identity Center tienen credenciales temporales que se establecen cada vez que el usuario inicia sesión en AWS. Como práctica recomendada, exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales en lugar de como usuario de IAM. Uno de los principales usos de los usuarios de IAM es ofrecer a las cargas de trabajo que no pueden utilizar roles de IAM la posibilidad de llevar a cabo solicitudes programáticas a servicios de AWS mediante la API o la CLI.

Temas

Solo para el primer acceso: sus credenciales de usuario raíz

Cuando se crea una Cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los Servicios de AWS y recursos de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS y se accede a ella iniciando sesión con el email y la contraseña que utilizó para crear la cuenta. Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para obtener la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM. Solo las políticas de control de servicios (SCP) de las organizaciones pueden restringir los permisos concedidos al usuario raíz.

Usuarios de IAM y usuarios de IAM Identity Center

Los usuarios de IAM no son cuentas separadas, sino que son usuarios dentro de su cuenta. Cada usuario puede tener su propia contraseña para obtener acceso a la AWS Management Console. También puede crear una clave de acceso individual para cada usuario, de modo que el usuario puede realizar solicitudes programadas para trabajar con recursos de su cuenta.

Los usuarios de IAM reciben credenciales a largo plazo para los recursos de AWS. Como práctica recomendada, no cree usuarios de IAM con credenciales a largo plazo para usuarios humanos. En su lugar, solicite a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS.

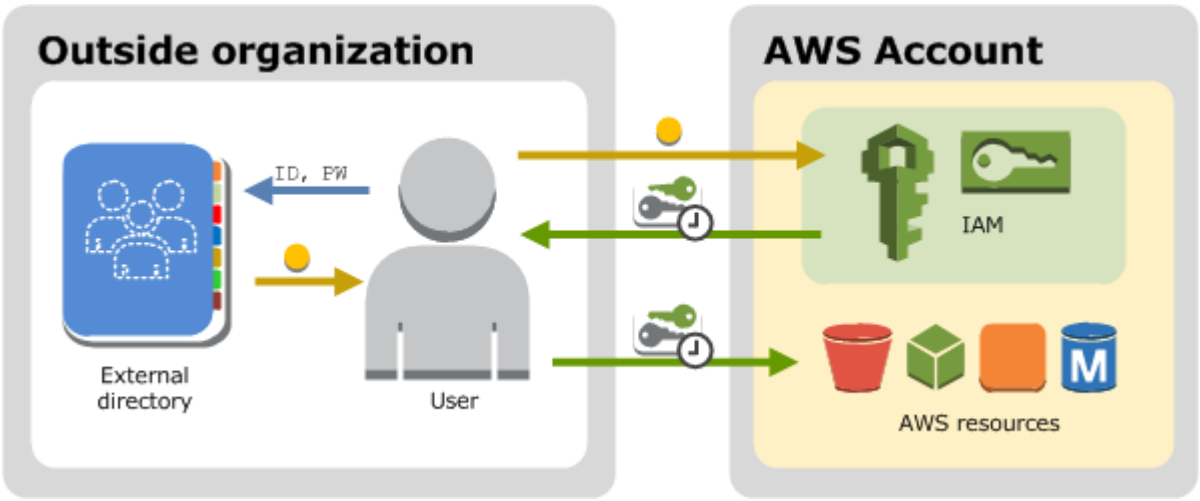
> **Note**
>
> En aquellas situaciones en las que necesite usuarios de IAM con acceso mediante programación y credenciales a largo plazo, recomendamos actualizar las claves de acceso cuando sea necesario. Para obtener más información, consulte Actualización de las claves de acceso.

Por el contrario, a usuarios en AWS IAM Identity Center se le otorgan credenciales a corto plazo para los recursos de AWS. Si desea administrar el acceso de manera centralizada, se recomienda utilizar AWS IAM Identity Center (IAM Identity Center) para administrar el acceso a las cuentas y los permisos de esas cuentas. IAM Identity Center se configura de manera automática con un directorio de Identity Center como fuente de identidad predeterminada, donde usted puede crear usuarios y grupos y asignar el nivel de acceso a los recursos de AWS. Para obtener más información, consulte ¿Qué es AWS IAM Identity Center? en la Guía del usuario de AWS IAM Identity Center.

Federación de usuarios ya existentes

Si los usuarios de su organización ya tienen una forma de autenticarse, por ejemplo, mediante el inicio de sesión en la red de la empresa, no es necesario que cree usuarios de IAM o usuarios de IAM Identity Center para ellos. En su lugar, puede federar esas identidades de usuario en AWS mediante IAM o AWS IAM Identity Center.

En el siguiente diagrama se muestra cómo un usuario puede obtener credenciales de seguridad temporales de AWS para acceder a los recursos de su Cuenta de AWS.



La federación es especialmente útil en los casos siguientes:

- Sus usuarios ya existen en un directorio corporativo.

  Si su directorio corporativo es compatible con Security Assertion Markup Language 2.0 (SAML 2.0), puede configurar su directorio corporativo para proporcionar acceso de inicio de sesión único (SSO) a la AWS Management Console a sus usuarios. Para obtener más información, consulte Escenarios habituales en las credenciales temporales.

  Si su directorio corporativo no es compatible con SAML 2.0, puede crear una aplicación de agente de identidades para proporcionar acceso de inicio de sesión único (SSO) a la AWS Management Console a sus usuarios. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

  Si su directorio corporativo es Microsoft Active Directory, puede utilizar AWS IAM Identity Center para conectar un directorio autoadministrado en Active Directory o un directorio en AWS Directory Service para establecer una relación de confianza entre su directorio corporativo y su Cuenta de AWS.

  Si utiliza un proveedor de identidades (IdP) externo como Okta o Microsoft Entra para administrar usuarios, puede utilizar AWS IAM Identity Center para establecer una relación de confianza entre su IdP y su Cuenta de AWS. Para obtener más información, consulte Conexión a un proveedor de identidad externo en la Guía del usuario de AWS IAM Identity Center.

- Sus usuarios ya disponen de identidades de Internet.

  Si está creando una aplicación móvil o una aplicación basada en web que permita que los usuarios se identifiquen mediante un proveedor de identidades de Internet como Login with Amazon,

Facebook, Google o cualquier proveedor de identidades compatible con OpenID Connect (OIDC), la aplicación puede utilizar una federación para obtener acceso a AWS. Para obtener más información, consulte Federación OIDC.

> **Tip**
>
> Para utilizar las identidades federadas con proveedores de identidades de Internet, le recomendamos que utilice Amazon Cognito.

Métodos de control de acceso

A continuación, se muestran las formas en las que puede controlar el acceso a sus recursos de AWS.

Tipo de acceso de usuario	¿Por qué debería utilizarlo?	¿Dónde puedo obtener más información?
Acceso de inicio de sesión único para usuarios humanos, como los usuarios de su plantilla, a los recursos de AWS mediante IAM Identity Center	IAM Identity Center proporciona un lugar central que reúne la administración de usuarios y su acceso a Cuentas de AWS y aplicaciones en la nube. Puede configurar un almacén de identidades en IAM Identity Center o puede configurar la federación con un proveedor de identidades (IdP) existente. Una práctica recomendada de seguridad es conceder a sus usuarios humanos credenciales limitadas a los recursos de AWS según sea necesario.	Para más información sobre la configuración de IAM Identity Center, consulte Introducción en la Guía del usuario de AWS IAM Identity Center Para más información sobre el uso de MFA en IAM Identity Center, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center

Tipo de acceso de usuario	¿Por qué debería utilizarlo?	¿Dónde puedo obtener más información?
	Los usuarios tienen una experiencia de inicio de sesión más sencilla y se mantiene el control sobre su acceso a los recursos desde un único sistema. IAM Identity Center admite la autenticación multifactor (MFA) para una mayor seguridad de la cuenta.	
Acceso federado para usuarios humanos, como los usuarios de su plantilla , a servicios de AWS que utilizan proveedores de identidad es de IAM	IAM admite proveedores de identidades (IdP) que son compatibles con OpenID Connect (OIDC) o SAML 2.0 (Security Assertion Markup Language 2.0). Después de crear un proveedor de identidades IAM, debe crear uno o más roles de IAM que se puedan asignar dinámicam ente a un usuario federado.	Para más información sobre los proveedores de identidades IAM y la federación, consulte Federación y proveedores de identidades.

Tipo de acceso de usuario	¿Por qué debería utilizarlo?	¿Dónde puedo obtener más información?
Acceso entre cuentas de Cuentas de AWS	Desea compartir el acceso a determinados recursos de AWS con usuarios de otras Cuentas de AWS. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, algunos servicios de AWS permiten asociar una política directamente a un recurso (en lugar de utilizar un rol como proxy). Se denominan políticas basadas en recursos.	Para más información acerca de los roles de IAM, consulte Roles de IAM. Para obtener más información acerca de los roles vinculados a servicios, consulte Uso de roles vinculados a servicios. Para obtener información acerca de qué servicios admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM. Busque los servicios para los que se indique Sí en la columna Roles vinculados a servicios. Para ver la documenta ción de roles vinculados a servicios de ese servicio, seleccione el enlace asociado al Sí en esa columna.

Tipo de acceso de usuario	¿Por qué debería utilizarlo?	¿Dónde puedo obtener más información?
Credenciales de larga duración para usuarios de IAM designados en su Cuenta de AWS	Es posible que tenga casos de uso específicos que requieran credenciales a largo plazo con usuarios de IAM en AWS. Puede utilizar IAM para crear estos usuarios de IAM en su Cuenta de AWS y administrar sus permisos. Estos son algunos de los casos de uso: • Cargas de trabajo que no pueden utilizar roles de IAM • Los clientes de AWS de terceros que requieran acceso programático mediante claves de acceso • Credenciales específicas del servicio para AWS CodeCommit o Amazon Keyspaces • AWS IAM Identity Center no está disponible para su cuenta y no dispone de otro proveedor de identidades Como práctica recomendada en situaciones en las que necesita usuarios de IAM con acceso mediante programación y credenciales a largo plazo, le recomendamos actualizar las claves de	Para obtener más información acerca de cómo configurar un usuario de IAM, consulte Creación de un usuario de IAM en su Cuenta de AWS. Para obtener más información sobre las claves de acceso de usuario de IAM, consulte Administración de las claves de acceso de los usuarios de IAM. Para obtener más información sobre las credenciales específicas del servicio para AWS CodeCommit o Amazon Keyspaces, consulte Uso de IAM con CodeCommit: credenciales de Git, claves SSH y claves de acceso de AWS y Utilizar IAM con Amazon Keyspaces (for Apache Cassandra).

Tipo de acceso de usuario	¿Por qué debería utilizarlo?	¿Dónde puedo obtener más información?
	acceso cuando sea necesario. Para obtener más información, consulte Actualización de las claves de acceso.	

Información general sobre la administración del acceso: permisos y políticas

La parte de administración de acceso de AWS Identity and Access Management (IAM) le ayuda a definir qué puede hacer una entidad principal en una cuenta. Una entidad principal es una persona o aplicación autenticada con una entidad de IAM (usuario o rol). La gestión del acceso a menudo se denomina autorización. Puede administrar el acceso en AWS creando políticas y asignándoselas a identidades de IAM (usuarios, grupos de usuarios o roles) o a recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal utiliza una entidad de IAM (usuario o rol) para realizar una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. Las mayoría de las políticas se almacenan en AWS como documentos JSON. Para obtener más información sobre los tipos de políticas y sus usos, consulte Políticas y permisos en IAM.

Políticas y cuentas

Si administra una única cuenta en AWS, definirá los permisos dentro de dicha cuenta mediante políticas. Si administra permisos entre varias cuentas, le resultará más difícil la administración de permisos para los usuarios. Puede utilizar roles de IAM, políticas basadas en recursos o listas de control de acceso (ACL) para gestionar los permisos entre cuentas. Sin embargo, si es propietario de varias cuentas, es recomendable utilizar el servicio AWS Organizations para administrar esos permisos. Para obtener más información, consulte Qué es AWS Organizations en la Guía del usuario de Organizations.

Políticas y usuarios

Los usuarios de IAM son identidades en el servicio. Cuando se crean usuarios de IAM, estos no pueden obtener acceso a ningún elemento de la cuenta hasta que se les conceda permiso. Para

conceder permisos a un usuario se crea una política basada en identidad, que es una política que se asocia al usuario o a un grupo al que el usuario pertenece. En el siguiente ejemplo, se muestra una política JSON que permite al usuario realizar todas las acciones de Amazon DynamoDB (`dynamodb:*`) en la tabla `Books` de la cuenta `123456789012` que está dentro de la región `us-east-2`.

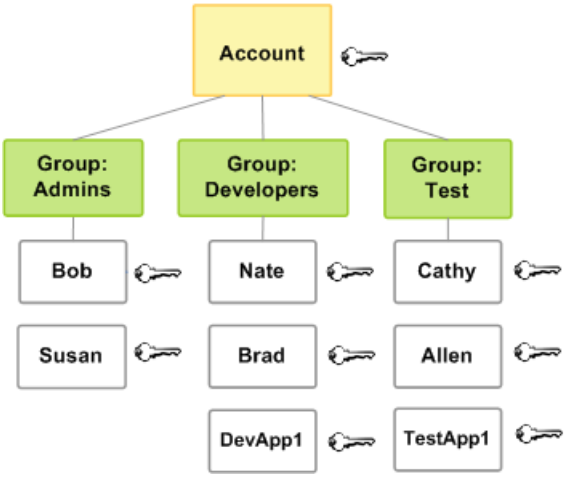
```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Una vez asociada la política al usuario de IAM, este solo tendrá esos permisos de DynamoDB. La mayoría de los usuarios tienen varias políticas que, en conjunto, representan los permisos de dicho usuario.

De forma predeterminada, las acciones o los recursos que no se permiten de forma explícita se deniegan. Por ejemplo, si la política anterior es la única asociada a un usuario, este solo tendrá permiso para realizar acciones de DynamoDB en la tabla `Books`. Las acciones en el resto de las tablas están prohibidas. Del mismo modo, el usuario no podrá realizar ninguna acción en Amazon EC2, Amazon S3 ni en ningún otro servicio de AWS. El motivo es que los permisos para trabajar con esos servicios no están incluidos en la política.

Políticas y grupos

Puede organizar a los usuarios de IAM en grupos de IAM y asociar una política a un grupo. En ese caso, los usuarios individuales siguen teniendo sus propias credenciales, pero todos los usuarios de un grupo tienen los permisos que se asocian al grupo. Utilice grupos para facilitar la administración de los permisos y siga nuestras Prácticas recomendadas de seguridad en IAM.



Los usuarios o los grupos pueden tener asociadas varias políticas que conceden permisos diferentes. En ese caso, los permisos para los usuarios se calculan basándose en las políticas combinadas. No obstante, sigue aplicándose el principio básico: si no se ha concedido al usuario un permiso explícito para una acción y un recurso, el usuario no tiene dichos permisos.

Usuarios federados y roles

Los usuarios federados no tienen identidades permanentes en su Cuenta de AWS tal y como las tienen los usuarios de IAM. Para asignar permisos a usuarios federados, puede crear una entidad a la que se hace referencia como rol y definir permisos para el rol. Cuando un usuario federado inicia sesión en AWS, se asocia el usuario al rol y se le conceden los permisos que están definidos en el rol. Para obtener más información, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

Políticas basadas en identidad y políticas basadas en recursos

Las políticas basadas en identidad son políticas de permisos que se asocian a una identidad de IAM, como un usuario, grupo o rol de IAM. Las políticas basadas en recursos son políticas de permisos que se asocian a un recurso, como un bucket de Amazon S3 o una política de confianza de un rol de IAM.

Las políticas basadas en identidad controlan qué acciones puede realizar la identidad, en qué recursos y en qué condiciones. Las políticas basadas en la identidad pueden clasificarse así:

- Políticas administradas: políticas independientes basadas en la identidad que puede adjuntar a varios usuarios, grupos y funciones en su Cuenta de AWS. Puede utilizar dos tipos de políticas administradas:

- Políticas administradas de AWS – Políticas administradas creadas y administradas por AWS. Si es la primera vez que utiliza políticas, le recomendamos que empiece a utilizar las políticas administradas por AWS.
- Políticas administradas por el cliente: políticas administradas que crea y administra en su Cuenta de AWS. Las políticas administradas por el cliente ofrecen un control más preciso sobre las políticas que las políticas administradas por AWS. Puede crear, editar y validar una política de IAM en el editor visual o mediante la creación del documento de política de JSON directamente. Para obtener más información, consulte Crear políticas de IAM y Edición de políticas de IAM.

- Políticas insertadas – Políticas que crea y administra y que están integradas directamente en un único usuario, grupo o rol. En la mayoría de los casos no es recomendable el uso de políticas insertadas.

Las políticas basadas en recursos controlan qué acciones puede realizar una entidad principal en ese recurso y en qué condiciones. Las políticas basadas en recursos son políticas en línea y no hay políticas administradas basadas en recursos. Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política basada en recursos.

El servicio IAM solo admite un tipo de política basada en recursos, el llamado política de confianza de rol, que se asocia a un rol de IAM. Al ser un rol de IAM al mismo tiempo una identidad y un recurso que admite las políticas basadas en recursos, es necesario asociarle tanto una política de confianza como una política basada en un rol de IAM. Las políticas de confianza definen qué entidades principales (cuentas, usuarios, roles y usuarios federados) puede asumir el rol. Para obtener información sobre cómo difieren los roles de IAM con respecto a otras políticas basadas en recursos, consulte Acceso a recursos entre cuentas en IAM.

Para ver qué servicios admiten políticas basadas en recursos, consulte Servicios de AWS que funcionan con IAM. Para obtener más información sobre las políticas basadas en recursos, consulte Políticas basadas en identidad y políticas basadas en recursos.

¿Qué es ABAC para AWS?

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puede asociar etiquetas a recursos de IAM, incluidas entidades de IAM (usuarios o roles) y recursos de AWS. Puede crear una única política ABAC o un conjunto pequeño de políticas para sus entidades

principales de IAM. Estas políticas ABAC se pueden diseñar para permitir operaciones cuando la etiqueta de la entidad principal coincida con la etiqueta del recurso. ABAC es útil en entornos que crecen con rapidez y ayuda en situaciones en las que la administración de las políticas resulta engorrosa.

Por ejemplo, puede crear tres roles con la clave de etiqueta `access-project`. Establezca el valor de etiqueta del primer rol en `Heart`, el segundo en `Lightning` y el tercero en `Star`. A continuación, puede utilizar una única política que permita el acceso cuando el rol y el recurso estén etiquetados con el mismo valor para `access-project`. Para ver un tutorial detallado que muestra cómo utilizar ABAC en AWS, consulte Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas. Para obtener información sobre los servicios que respaldan a ABAC, consulte Servicios de AWS que funcionan con IAM.

Comparación de ABAC con el modelo RBAC tradicional

El modelo de autorización tradicional utilizado en IAM se denomina control de acceso basado en roles (RBAC). RBAC define permisos en función de la función laboral de una persona, conocida fuera de AWS como rol. Dentro de AWS un rol generalmente se refiere a un rol de IAM, que es una identidad en IAM que usted puede asumir. IAM sí incluye políticas administradas para funciones de trabajo que alinean los permisos a una función de trabajo en un modelo RBAC.

En IAM, implemente RBAC creando diferentes políticas para diferentes funciones de trabajo. A continuación, adjunte las políticas a identidades (usuarios de IAM, grupos de usuarios o roles de IAM). Como práctica recomendada, debe conceder los permisos mínimos necesarios para la función de trabajo. Esto se conoce como concesión de privilegios mínimos. Para ello, enumeren los recursos específicos a los que puede obtener acceso la función de trabajo. La desventaja de utilizar el modelo RBAC tradicional es que cuando los empleados añaden nuevos recursos, debe actualizar las políticas para permitir el acceso a dichos recursos.

Por ejemplo, suponga que tiene tres proyectos, denominados `Heart`, `Star` y `Lightning`, en los que trabajan los empleados. Puede crear un rol de IAM para cada proyecto. A continuación, asocie políticas a cada rol de IAM para definir los recursos a los que cualquier persona que tenga permiso para asumir el rol puede acceder. Si un empleado cambia de trabajo dentro de su empresa, usted les asigna un rol de IAM diferente. Es posible asignar personas o programas a más de un rol. Sin embargo, el proyecto `Star` puede requerir recursos adicionales, como un nuevo contenedor de Amazon EC2. En ese caso, debe actualizar la política adjunta al rol `Star` para especificar el nuevo recurso del contenedor. De lo contrario, los miembros del proyecto `Star` no pueden obtener acceso al nuevo contenedor.

ABAC ofrece las siguientes ventajas con respecto al modelo RBAC tradicional:

- Los permisos ABAC se escalan con innovación. Ya no es necesario que un administrador actualice las políticas existentes para permitir el acceso a nuevos recursos. Por ejemplo, suponga que ha diseñado su estrategia ABAC con la etiqueta `access-project`. Un desarrollador utiliza el rol con la etiqueta `access-project` = `Heart`. Cuando las personas del proyecto `Heart` necesitan recursos de Amazon EC2 adicionales, el desarrollador puede crear nuevas instancias Amazon EC2 con la etiqueta `access-project` = `Heart`. A continuación, cualquier persona en el proyecto `Heart` puede iniciar y detener esas instancias porque sus valores de etiqueta coinciden.
- ABAC requiere menos políticas. Dado que no tiene que crear diferentes políticas para diferentes funciones de trabajo, debe crear menos políticas. Estas políticas son más fáciles de administrar.
- Con ABAC, los equipos pueden cambiar y crecer rápidamente. Esto se debe a que los permisos para nuevos recursos se conceden automáticamente de acuerdo con los atributos. Por ejemplo, si su empresa ya admite los proyectos `Heart` y `Star` con ABAC, es fácil añadir un nuevo proyecto `Lightning`. Un administrador de IAM crea un nuevo rol con la etiqueta `access-project` = `Lightning`. No es necesario cambiar la política para admitir un nuevo proyecto. Cualquier persona que tenga permisos para asumir el rol puede crear y ver instancias etiquetadas con `access-project` = `Lightning`. Además, un miembro del equipo podría pasar del proyecto `Heart` al proyecto `Lightning`. El administrador de IAM asigna al usuario a un rol de IAM diferente. No es necesario cambiar las políticas de permisos.
- Los permisos granulares son posibles con ABAC. Al crear políticas, se recomienda conceder privilegios mínimos. Con RBAC tradicional, debe escribir una política que permita el acceso solo a recursos específicos. Sin embargo, cuando utiliza ABAC, puede permitir acciones en todos los recursos, pero solo si la etiqueta del recurso coincide con la etiqueta de la entidad principal.
- Utilice los atributos de los empleados de su directorio corporativo con ABAC. Puede configurar su proveedor SAML o OIDC para que pase las etiquetas de sesión a AWS. Cuando los empleados se federan en AWS, sus atributos se aplican a su entidad principal resultante en AWS. Entonces puede utilizar ABAC para permitir o denegar permisos basados en esos atributos.

Para ver un tutorial detallado que muestra cómo utilizar ABAC en AWS, consulte Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas.

Características de seguridad fuera de IAM

Utilice IAM para controlar el acceso a las tareas efectuadas con la AWS Management Console, las herramientas de línea de comandos de AWS o las operaciones de API de servicios con los SDK

de AWS. Algunos productos de AWS también tienen otras formas de proteger sus recursos. En la siguiente lista se ofrecen algunos ejemplos, aunque no de forma exhaustiva.

Amazon EC2

En Amazon Elastic Compute Cloud (EC2) inicie sesión en una instancia con un par de claves (si son instancias de Linux) o mediante un nombre de usuario y contraseña (para instancias de Microsoft Windows).

Para obtener más información, consulte la documentación siguiente:

- Introducción a las instancias de Linux de Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux
- Introducción a las instancias de Windows de Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Windows

Amazon RDS

En Amazon Relational Database Service inicie sesión en el motor de base de datos con un nombre de usuario y contraseña que estén vinculados a la base de datos.

Para obtener más información, consulte Introducción a Amazon RDS en la Guía del usuario de Amazon RDS.

Amazon EC2 y Amazon RDS

En Amazon EC2 y Amazon RDS utilice los grupos de seguridad para controlar el tráfico a una instancia o base de datos.

Para obtener más información, consulte la documentación siguiente:

- Grupos de seguridad de Amazon EC2 para instancias de Linux en la Guía del usuario de Amazon EC2 para instancias de Linux
- Grupos de seguridad de Amazon EC2 para las instancias de Windows en la Guía del usuario de Amazon EC2 para instancias de Windows
- Grupos de seguridad de Amazon RDS en la Guía del usuario de Amazon RDS

WorkSpaces

En Amazon WorkSpaces, los usuarios inician sesión en un escritorio con un nombre de usuario y contraseña.

Para obtener más información, consulte Introducción a WorkSpaces en la Guía del administrador de Amazon WorkSpaces.

Amazon WorkDocs

En Amazon WorkDocs, los usuarios obtienen acceso a documentos compartidos iniciando sesión con un nombre de usuario y contraseña.

Para obtener más información, consulte Introducción a WorkDocs en la Guía del administrador de Amazon WorkDocs.

Estos métodos de control de acceso no forman parte de IAM. IAM le permite controlar la forma en que estos productos de AWS se administran: mediante la creación o finalización de una instancia de Amazon EC2, la configuración de nuevos escritorios de WorkSpaces, etc. Es decir, IAM le ayuda a controlar las tareas que se realizan al realizar solicitudes en Amazon Web Services y le ayuda a controlar el acceso a la AWS Management Console. Sin embargo, IAM no le ayuda a administrar la seguridad de tareas como el inicio de sesión en un sistema operativo (Amazon EC2), base de datos (Amazon RDS), escritorio (Amazon WorkSpaces) o sitio de colaboración (Amazon WorkDocs).

Cuando trabaje con un producto específico de AWS, asegúrese de leer la documentación para conocer las opciones de seguridad para todos los recursos que pertenecen a dicho producto.

Enlaces rápidos a tareas comunes

Utilice los siguientes enlaces para obtener ayuda con las tareas comunes asociadas a IAM.

Iniciar sesión para diferentes tipos de usuarios

Inicie sesión en la consola de IAM; para ello, elija IAM user (Usuario de IAM) y escriba su ID de Cuenta de AWS o el alias de la cuenta. En la siguiente página, ingrese su nombre de usuario y su contraseña de IAM.

Para iniciar sesión con el usuario del IAM Identity Center, utilice la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario del Centro de identidades de IAM.

Para obtener ayuda para iniciar sesión con un usuario del IAM Identity Center, consulte Inicio de sesión en el portal de acceso de AWS en la Guía del usuario de AWS Sign-In.

Inicie sesión en la AWS Management Console como propietario de la cuenta; para ello, elija Usuario raíz e introduzca el correo electrónico de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

Consulte ¿Qué es el inicio de sesión en AWS en la Guía del usuario de AWS Sign-In para obtener ayuda acerca de cómo determinar el tipo de usuario y la página de inicio de sesión.

Administrar las contraseñas de los usuarios de

Necesita una contraseña para obtener acceso a la AWS Management Console, incluido el acceso a la información de facturación.

Para tu Usuario raíz de la cuenta de AWS, consulte Cambiar la contraseña del Usuario raíz de la cuenta de AWS en la AWS Account Management Guía de referencia

En el caso del usuario de IAM, consulte Administración de las contraseñas de los usuarios de IAM.

Administrar los permisos para los usuarios de

Puede utilizar políticas para conceder permisos a los usuarios de IAM en la Cuenta de AWS. Los usuarios de IAM no tienen permisos cuando se crean, de modo que debe agregar permisos con el fin de permitirles utilizar los recursos de AWS.

Para dar acceso, añada permisos a los usuarios, grupos o roles:

- Usuarios y grupos en AWS IAM Identity Center:

  Cree un conjunto de permisos. Siga las instrucciones de Creación de un conjunto de permisos en la Guía del usuario de AWS IAM Identity Center.
- Usuarios administrados en IAM a través de un proveedor de identidades:

  Cree un rol para la federación de identidades. Siga las instrucciones de Creación de un rol para un proveedor de identidades de terceros (federación) en la Guía del usuario de IAM.
- Usuarios de IAM:
  - Cree un rol que el usuario pueda aceptar. Siga las instrucciones descritas en Creación de un rol para un usuario de IAM en la Guía del usuario de IAM.
  - (No recomendado) Adjunte una política directamente a un usuario o añada un usuario a un grupo de usuarios. Siga las instrucciones descritas en Adición de permisos a un usuario (consola) de la Guía del usuario de IAM.

Para obtener más información, consulte Administración de políticas de IAM.

Enumerar los usuarios de la Cuenta de AWS y obtener información sobre sus credenciales

Consulte Obtención de informes de credenciales para su cuenta de Cuenta de AWS.

Añadir un dispositivo Multi-Factor Authentication (MFA)

Para añadir un dispositivo MFA virtual, consulte uno de los siguientes temas:

- Habilitación de un dispositivo MFA virtual para su Usuario raíz de la cuenta de AWS (consola)
- Habilite un dispositivo MFA virtual para un usuario de IAM (Consola)

Para agregar una clave de seguridad FIDO, consulte uno de los siguientes temas:

- Habilitación de una clave de seguridad FIDO para el usuario raíz de la Cuenta de AWS (consola)
- Habilitación de una clave de seguridad FIDO para otro usuario de IAM (consola)

Para añadir un dispositivo MFA físico, consulte uno de los siguientes temas:

- Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola).
- Habilitar un token TOTP de hardware para otro usuario de IAM (consola)

Obtener una clave de acceso

Necesita una clave de acceso si desea realizar solicitudes de AWS con los SDK de AWS, las herramientas de línea de comandos de AWS o las operaciones de la API.

> **Important**
>
> Como práctica recomendada, utilice credenciales de seguridad temporales (por ejemplo, roles de IAM) en lugar de crear credenciales a largo plazo como claves de acceso. Antes de crear claves de acceso, revise las alternativas a las claves de acceso a largo plazo.

Para obtener instrucciones que lo ayuden a proteger sus claves de acceso, consulte Protección de las claves de acceso.

Para obtener información sobre cómo administrar las claves de acceso de un usuario de IAM, consulte Administración de las claves de acceso de los usuarios de IAM.

Para obtener más información acerca de las credenciales de seguridad disponibles para su Cuenta de AWS, consulte Credenciales de seguridad de AWS.

Etiquetado de recursos de IAM

Puede etiquetar los siguientes recursos de IAM:

- Usuarios de IAM
- Roles de IAM
- Políticas administradas por el cliente
- Proveedores de identidades
- Certificados de servidor
- Dispositivos MFA virtuales

Para obtener más información acerca de las etiquetas en IAM, consulte Etiquetado de recursos de IAM.

Para obtener más información sobre cómo utilizar etiquetas para controlar el acceso a recursos de AWS, consulte Control de acceso a los recursos de AWS mediante etiquetas.

Ver las acciones, los recursos y las claves de condición de todos los servicios

Este conjunto de documentación de referencia puede ayudarle a escribir políticas de IAM detalladas. Cada servicio de AWS define las claves de contexto de condición, los recursos y las acciones que se utilizan en las políticas de IAM. Para obtener más información, consulte Claves de condición, recursos y acciones de los servicios de AWS.

Introducción a todo AWS

Este juego de documentación aborda principalmente el servicio de IAM. Para obtener información acerca de cómo empezar a utilizar AWS y varios servicios para solucionar un problema, como crear y lanzar su primer proyecto, consulte Centro de recursos introductorios.

Búsqueda de la consola de IAM

Utilice la página de búsqueda de la consola de IAM como una opción más rápida para encontrar recursos de IAM. Puede utilizar la búsqueda de la consola para encontrar claves de acceso relacionadas con su cuenta, las entidades de IAM (como usuarios, grupos, roles, proveedores de identidad), las políticas por nombre, etc.

La característica de búsqueda de la consola de IAM puede encontrar cualquiera de los siguientes elementos:

- Nombres de entidades de IAM que coincidan con sus palabras clave de búsqueda (para usuarios, grupos, roles, proveedores de identidad y políticas)
- Tareas que coincidan con sus palabras clave de búsqueda

La característica de búsqueda de la consola de IAM no devuelve información sobre IAM Access Analyzer.

Cada línea del resultado de búsqueda es un enlace activo. Por ejemplo, puede elegir el nombre de usuario en el resultado de búsqueda, que le lleva a la página de detalles del usuario. O también puede elegir el enlace de una acción, por ejemplo Crear usuario, para ir a la página Crear usuario.

> **Note**
>
> La búsqueda de clave de acceso exige que escriba el ID de clave de acceso completo en el campo de búsqueda. El resultado de búsqueda muestra el usuario asociado a dicha clave. A partir de aquí, puede ir directamente a la página de dicho usuario, donde puede administrar la clave de acceso.

Uso de la búsqueda de la consola de IAM

Utilice la página Búsqueda en la consola de IAM para encontrar elementos relacionados con la cuenta.

Para buscar elementos en la consola de IAM

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS.
2. En la página principal de la consola, seleccione el servicio de IAM.
3. En el panel de navegación, elija Buscar.
4. En el cuadro Búsqueda, escriba las palabras clave de búsqueda.
5. Elija un enlace en la lista de resultados de búsqueda para ir a la parte correspondiente de la consola.

Iconos de los resultados de búsqueda de la consola de IAM

Los siguientes iconos identifican los tipos de elementos que se encuentran mediante una búsqueda:

Icono	Descripción
	Usuarios de IAM
	Grupos de IAM
	Roles de IAM
	Políticas de IAM
	Tareas, tales como "crear usuario" o "asociar política"
	Resultados obtenidos con la palabra clave **`delete`**

Ejemplos de frases de búsqueda

Puede utilizar las siguientes frases en la búsqueda de IAM. Sustituya los términos en cursiva por los nombres reales de los usuarios, los grupos, los roles, las claves de acceso, las políticas o los proveedores de identidad de IAM que desea encontrar.

- ***`user_name`*** o ***`group_name`*** o ***`role_name`*** o ***`policy_name`*** o ***`identity_provider_name`***
- ***`access_key`***
- **`add user`** ***`user_name`*** **`to groups`** o **`add users to group`** ***`group_name`***
- **`remove user`** ***`user_name`*** **`from groups`**
- **`delete`** ***`user_name`*** o **`delete`** ***`group_name`*** o **`delete`** ***`role_name`*** o **`delete`** ***`policy_name`*** o **`delete`** ***`identity_provider_name`***
- **`manage access keys`** ***`user_name`***
- **`manage signing certificates`** ***`user_name`***
- **`users`**
- **`manage MFA for`** ***`user_name`***

- **manage password for *user_name***
- **create role**
- **password policy**
- **edit trust policy for role *role_name***
- **show policy document for role *role_name***
- **attach policy to *role_name***
- **create managed policy**
- **create user**
- **create group**
- **attach policy to *group_name***
- **attach entities to *policy_name***
- **detach entities from *policy_name***

Crear recursos de AWS Identity and Access Management con AWS CloudFormation

AWS Identity and Access Management está integrado con AWS CloudFormation, un servicio que le ayuda a modelar y configurar sus recursos de AWS para que pueda dedicar menos tiempo a crear y administrar sus recursos e infraestructura. Cree una plantilla que describa todos los recursos de AWS que desee (como las claves de acceso, los grupos, las políticas de grupo, los perfiles de instancia, las políticas administradas, los proveedores de OIDC, las políticas en línea, los roles, políticas de rol, los proveedores de SAML, los certificados de servidor, los roles vinculados a servicios, los usuarios y su adición a los grupos, las políticas de usuario y los dispositivos MFA virtuales), y AWS CloudFormation aprovisiona y configura esos recursos.

Cuando se utiliza AWS CloudFormation, se puede volver a utilizar la plantilla para configurar los recursos de IAM de forma coherente y repetida. Solo tiene que describir los recursos una vez y luego aprovisionar los mismos recursos una y otra vez en varias Cuentas de AWS y regiones.

IAM y plantillas de AWS CloudFormation

Para aprovisionar y configurar los recursos para IAM y los servicios, debe entender las plantillas de AWS CloudFormation. Las plantillas son archivos de texto con formato JSON o YAML. Estas

plantillas describen los recursos que desea aprovisionar en sus pilas de AWS CloudFormation. Si no conoce bien JSON o YAML, puede utilizar Designer de AWS CloudFormation para comenzar a utilizar las plantillas de AWS CloudFormation. Para obtener más información, consulte ¿Qué es Designer de AWS CloudFormation? en la Guía del usuario de AWS CloudFormation.

IAM es compatible con la creación de las claves de acceso, los grupos, las políticas de grupo, los perfiles de instancia, las políticas administradas, los proveedores de OIDC, las políticas en línea, los roles, políticas de rol, los proveedores de SAML, los certificados de servidor, los roles vinculados a servicios, los usuarios y su adición a los grupos, las políticas de usuario y los dispositivos MFA virtuales en AWS CloudFormation. Para más información y ejemplos de plantillas JSON y YAML para los recursos de IAM, consulte la AWS Identity and Access Management referencia del tipo de recurso de en la AWS CloudFormation guía de usuario de .

También puede crear plantillas que creen recursos relacionados, como las funciones y las políticas administradas.

Obtener más información sobre AWS CloudFormation

Para conocer más información acerca de AWS CloudFormation, consulte los siguientes recursos:

- AWS CloudFormation
- Guía del usuario de AWS CloudFormation
- Referencia de la API de AWS CloudFormation
- Guía del usuario de la interfaz de la línea de comandos de AWS CloudFormation

Utilización de AWS CloudShell para trabajar con AWS Identity and Access Management

AWS CloudShell es un shell previamente autenticado y basado en el navegador, que se puede lanzar directamente desde la página web de la AWS Management Console. Puede ejecutar comandos de AWS CLI contra servicios de AWS (incluido AWS Identity and Access Management) mediante el uso de su intérprete de comandos preferido (Bash, PowerShell o intérprete de comandos Z). Y puede hacerlo sin necesidad de descargar o instalar herramientas de línea de comandos.

Inicia AWS CloudShell desde la AWS Management Console y las credenciales de AWS que usó para iniciar sesión en la consola están automáticamente disponibles en una nueva sesión de intérprete de comandos. Esta autenticación previa de usuarios de AWS CloudShell le permite omitir

la configuración de las credenciales cuando interactúa con servicios de AWS, como IAM, mediante la versión 2 de AWS CLI (preinstalada en el entorno de computación del intérprete de comandos).

Obtención de permisos de IAM para AWS CloudShell

Con los recursos de administración de acceso que proporciona AWS Identity and Access Management, los administradores pueden conceder permisos a los usuarios de IAM para que puedan acceder a AWS CloudShell y utilizar las características del entorno.

La forma más rápida de que un administrador conceda acceso a los usuarios es mediante una política administrada de AWS. Una política administrada de AWS es una política independiente creada y administrada por AWS. La siguiente política administrada de AWS para CloudShell se puede adjuntar a las identidades de IAM:

- `AWSCloudShellFullAccess`: concede permiso para usar AWS CloudShell con acceso completo a todas las características.

Si desea limitar el alcance de las acciones que un usuario de IAM puede realizar con AWS CloudShell, puede crear una política personalizada que utilice la política administrada de `AWSCloudShellFullAccess` como plantilla. Para obtener más información sobre cómo limitar las acciones disponibles para los usuarios en CloudShell, consulte Administrar el AWS CloudShell acceso y el uso con políticas de IAM en la Guía del usuario de AWS CloudShell.

Interacción con IAM mediante AWS CloudShell

Tras lanzar AWS CloudShell desde la AWS Management Console, podrá empezar a interactuar inmediatamente con IAM mediante la interfaz de línea de comandos.

> **Note**
>
> Al usar AWS CLI en AWS CloudShell, no necesita descargar o instalar recursos adicionales. Además, dado que ya está autenticado en el intérprete de comandos, no tiene que configurar las credenciales antes de realizar llamadas.

Creación de un grupo de IAM y adición de un usuario de IAM al grupo mediante AWS CloudShell

En el siguiente ejemplo, se utiliza CloudShell para crear un grupo de IAM, agregar un usuario de IAM al grupo y, a continuación, verificar que el comando se ejecutó correctamente.

1. Desde la AWS Management Console, puede seleccionar las siguientes opciones disponibles en la barra de navegación para iniciar CloudShell:

   - Elija el icono de CloudShell.
   - Escriba “cloudshell” en el cuadro de búsqueda y, a continuación, elija la opción CloudShell.

2. Para crear un grupo de IAM, introduzca el siguiente comando en la línea de comandos de CloudShell. En este ejemplo, denominamos al grupo east_coast:

   ```
   aws iam create-group --group-name east_coast
   ```

   Si la llamada se realiza correctamente, la línea de comandos muestra una respuesta del servicio similar a la siguiente salida:

   ```
   {
      "Group": {
          "Path": "/",
          "GroupName": "east_coast",
          "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
          "Arn": "arn:aws:iam::111122223333:group/east_coast",
          "CreateDate": "2023-09-11T21:02:21+00:00"
       }
   }
   ```

3. Para agregar un usuario al grupo que creó, utilice el siguiente comando, especificando el nombre del grupo y del usuario. En este ejemplo, denominamos al grupo east_coast y al usuario johndoe:

   ```
   aws iam add-user-to-group --group-name east_coast --user-name johndoe
   ```

4. Para verificar que el usuario está en el grupo, utilice el siguiente comando, especificando el nombre del grupo. En este ejemplo, continuamos usando el grupo east_coast:

   ```
   aws iam get-group --group-name east_coast
   ```

   Si la llamada se realiza correctamente, la línea de comandos muestra una respuesta del servicio similar a la siguiente salida:

```
{
  "Users": [
    {
        "Path": "/",
        "UserName": "johndoe",
        "UserId": "AIDAYBDBW4JBXGEXAMPLE",
        "Arn": "arn:aws:iam::552108220995:user/johndoe",
        "CreateDate": "2023-09-11T20:43:14+00:00",
        "PasswordLastUsed": "2023-09-11T20:59:14+00:00"
    }
  ],
  "Group": {
        "Path": "/",
        "GroupName": "east_coast",
        "GroupId": "AGPAYBDBW4JBY3EXAMPLE",
        "Arn": "arn:aws:iam::111122223333:group/east_coast",
        "CreateDate": "2023-09-11T21:02:21+00:00"
     }
 }
```


Uso de IAM con un SDK de AWS

Los kits de desarrollo de software (SDK) de AWS se encuentran disponibles en muchos lenguajes de programación populares. Cada SDK proporciona una API, ejemplos de código y documentación que facilitan a los desarrolladores la creación de aplicaciones en su lenguaje preferido.

Documentación de SDK	Ejemplos de código
AWS SDK for C++	Ejemplos de código de AWS SDK for C++
AWS SDK for Go	Ejemplos de código de AWS SDK for Go
AWS SDK for Java	Ejemplos de código de AWS SDK for Java
AWS SDK for JavaScript	Ejemplos de código de AWS SDK for JavaScript

Documentación de SDK	Ejemplos de código
AWS SDK para Kotlin	Ejemplos de código de AWS SDK para Kotlin
AWS SDK for .NET	Ejemplos de código de AWS SDK for .NET
AWS SDK for PHP	Ejemplos de código de AWS SDK for PHP
AWS SDK for Python (Boto3)	Ejemplos de código de AWS SDK for Python (Boto3)
AWS SDK for Ruby	Ejemplos de código de AWS SDK for Ruby
AWS SDK para Rust	Ejemplos de código de AWS SDK para Rust
AWS SDK para SAP ABAP	Ejemplos de código de AWS SDK para SAP ABAP
AWS SDK para Swift	Ejemplos de código de AWS SDK para Swift

Para ver ejemplos específicos de IAM, consulte Ejemplos de código de IAM con SDK de AWS.

> **Ejemplo de disponibilidad**
>
> ¿No encuentra lo que necesita? Solicite un ejemplo de código a través del enlace de Provide feedback (Enviar comentarios) que se encuentra al final de esta página.

Configuración de IAM

> **Important**
>
> Las prácticas recomendadas de IAM sugieren que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales, en lugar de utilizar usuarios de IAM con credenciales a largo plazo.

AWS Identity and Access Management (IAM) le ayuda a controlar de forma segura el acceso a Amazon Web Services (AWS) y a los recursos de su cuenta. IAM también puede mantener la privacidad de sus credenciales de inicio de sesión. No es preciso registrarse específicamente para utilizar IAM. No se cobra por utilizar IAM.

Utilice IAM para dar a las identidades, como usuarios y roles, acceso a los recursos de su cuenta. Por ejemplo, puede utilizar IAM con usuarios existentes en su directorio corporativo que administra de forma externa a AWS o puede crear usuarios en AWS con AWS IAM Identity Center. Las identidades federadas asumen roles de IAM definidos para acceder a los recursos que necesitan. Para obtener más información, consulte What is IAM Identity Center? (¿Qué es el Centro de identidades de IAM?) en la Guía del usuario de AWS IAM Identity Center.

> **Note**
>
> IAM está integrada con varios productos de AWS. Para obtener una lista de los servicios compatibles con IAM, consulte Servicios de AWS que funcionan con IAM.

Temas

Registro para obtener una Cuenta de AWS

Si no dispone de una Cuenta de AWS, siga estos pasos para crear una.

Creación de una Cuenta de AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica e indicar un código de verificación en el teclado del teléfono.

   Al registrarse en una Cuenta de AWS, se crea un Usuario raíz de la cuenta de AWS. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario administrativo y utilice únicamente el usuario raíz para realizar tareas que requieran acceso de usuario raíz.

AWS le enviará un correo electrónico de confirmación después de completar el proceso de registro. Puede ver la actividad de la cuenta y administrar la cuenta en cualquier momento entrando en https://aws.amazon.com/ y seleccionando Mi cuenta.

Crear un usuario administrativo

Después de registrarse para obtener una Cuenta de AWS, proteja su Usuario raíz de la cuenta de AWS, habilite AWS IAM Identity Center y cree un usuario administrativo para no utilizar el usuario raíz en las tareas cotidianas.

Protección de su Usuario raíz de la cuenta de AWS

1. Inicie sesión en AWS Management Console como propietario de cuenta eligiendo Usuario raíz e introduzca el correo electrónico de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In.
2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitar un dispositivo MFA virtual para el usuario raíz Cuenta de AWS (consola) en la Guía del usuario de IAM.

Creación de un usuario administrativo

1. Activar IAM Identity Center

   Para obtener instrucciones, consulte Activación de AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center.

2. En el Centro de identidades de IAM, conceda acceso administrativo a un usuario administrativo.

   Para ver un tutorial sobre cómo utilizar Directorio de IAM Identity Center como origen de identidad, consulte Configuración del acceso de los usuarios con el Directorio de IAM Identity Center predeterminado en la Guía del usuario de AWS IAM Identity Center.

Cómo iniciar sesión como usuario administrativo

- Para iniciar sesión con el usuario del Centro de identidades de IAM, utilice la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario del IAM Identity Center.

  Para obtener ayuda para iniciar sesión con un usuario del IAM Identity Center, consulte Iniciar sesión en el portal de acceso de AWS en la Guía del usuario de AWS Sign-In.

Preparación para los permisos de privilegio mínimo

El uso de permisos de privilegio mínimo es una recomendación de prácticas recomendadas de IAM. El concepto de permisos de privilegio mínimo consiste en conceder a los usuarios solo los permisos necesarios para realizar una tarea y ningún permiso adicional. Mientras lleva a cabo la configuración, considere cómo admitirá los permisos de privilegio mínimo. Tanto el usuario raíz como el usuario administrador tienen permisos potentes que no son necesarios para las tareas cotidianas. Mientras aprende acerca de AWS y prueba diferentes servicios, le recomendamos crear, al menos, un usuario adicional en IAM Identity Center con menos permisos que pueda utilizar en diferentes escenarios. Puede utilizar las políticas de IAM para definir las acciones que se pueden realizar en recursos específicos en condiciones específicas y luego, conectarse a los recursos con su cuenta con menos privilegios.

Si utiliza IAM Identity Center, considere la posibilidad de utilizar los conjuntos de permisos de IAM Identity Center para comenzar. Para obtener más información, consulte Crear un conjunto de permisos en la Guía del usuario de IAM Identity Center.

Si no utiliza IAM Identity Center, use los roles de IAM para definir los permisos de las diferentes entidades de IAM. Para obtener más información, consulte Creación de roles de IAM.

Los roles de IAM y los conjuntos de permisos de IAM Identity Center pueden utilizar políticas administradas por AWS basadas en funciones de trabajo. Para obtener más información acerca de los permisos que otorgan estas políticas, consulte Managed Policies de AWS para funciones de trabajo.

> **Important**
>
> Tenga presente que es posible que las políticas administradas de AWS no concedan permisos de privilegio mínimo para sus casos de uso concretos, ya que están disponibles para que las utilicen todos los clientes de AWS. Una vez que haya finalizado la configuración, se recomienda utilizar el Analizador de acceso de IAM para generar políticas de privilegio mínimo en función de la actividad de acceso que se haya registrado en AWS CloudTrail. Para obtener más información acerca de la generación de políticas, consulte Generación de políticas de IAM Access Analyzer.

Métodos de administración de IAM

Puede administrar IAM mediante la consola de AWS, la interfaz de línea de comandos de AWS o las interfaces de aplicaciones (API) en los SDK asociados. A medida que realiza la configuración, tenga en cuenta qué métodos desea admitir y cómo piensa asistir a los diferentes usuarios.

Temas

Consola de AWS

La consola de administración de AWS es una aplicación web que engloba y hace referencia a un amplio conjunto de consolas de servicios para la administración de recursos de AWS. La primera vez que inicie sesión, verá la página de inicio de la consola. La página de inicio proporciona acceso a cada consola de servicio y ofrece un único lugar para acceder a la información que necesita para realizar las tareas relacionadas con AWS. Los servicios y aplicaciones que tendrá a su disposición luego de iniciar sesión en la consola dependerán de los recursos de AWS a los que tenga permiso

de acceso. Puede obtener permisos para acceder a los recursos ya sea asumiendo un rol, siendo miembro de un grupo al que se le hayan concedido permisos u obteniendo permiso de forma explícita. En el caso de una cuenta de AWS independiente, el usuario raíz o el administrador de IAM configura el acceso a los recursos. Para AWS Organizations, la cuenta de administración o el administrador delegado configura el acceso a los recursos.

Si tiene previsto que las personas usen la consola de administración de AWS para administrar los recursos de AWS, le recomendamos configurar los usuarios con credenciales temporales como práctica recomendada de seguridad. Los usuarios de IAM que han asumido un rol, los usuarios federados y los usuarios de IAM Identity Center tienen credenciales temporales, mientras que el usuario de IAM y el usuario raíz tienen credenciales a largo plazo. Las credenciales de usuario raíz proporcionan acceso total a la Cuenta de AWS, mientras que los demás usuarios tienen credenciales que permiten acceder a los recursos que les otorgan las políticas de IAM.

La experiencia de inicio de sesión es diferente para los distintos tipos de usuarios de AWS Management Console.

- Los usuarios de IAM y el usuario raíz inician sesión desde la URL principal de AWS (https://signin.aws.amazon.com). Una vez que inician sesión, tienen acceso a los recursos en la cuenta para los que se les ha concedido permiso.

  Para iniciar sesión como usuario raíz, debe tener la dirección de correo electrónico y contraseña del usuario raíz.

  Para iniciar sesión como usuario de IAM, debe tener el número o alias de la Cuenta de AWS, el nombre de usuario de IAM y la contraseña de usuario de IAM.

  Le recomendamos restringir los usuarios de IAM en su cuenta a situaciones específicas que requieran credenciales a largo plazo (como, por ejemplo, acceso de emergencia) y utilizar el usuario raíz únicamente para las tareas que requieren credenciales de usuario raíz.

  Para su comodidad, en la página de inicio de sesión de AWS se utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. La próxima vez que el usuario vaya a cualquier página en AWS Management Console, la consola utiliza la cookie para redirigir al usuario a la página de inicio de sesión de la cuenta.

  Cierre sesión en la consola cuando finaliza la sesión para evitar que se vuelva a utilizar el inicio de sesión anterior.

- Los usuarios del IAM Identity Center inician sesión mediante un portal de acceso de AWS específico que es exclusivo de su organización. Una vez que inician sesión, pueden elegir a qué cuenta o aplicación acceder. Si eligen acceder a una cuenta, eligen qué conjunto de permisos quieren utilizar para la sesión de administración.
- Los usuarios federados que se administran en un proveedor de identidad externo vinculado a una Cuenta de AWS inician sesión mediante un portal de acceso empresarial personalizado. Los recursos de AWS disponibles para los usuarios federados dependen de las políticas seleccionadas por su organización.

> **Note**
>
> Para proporcionar un nivel de seguridad adicional, el usuario raíz, los usuarios de IAM y los usuarios de IAM Identity Center pueden hacer que AWS verifique la autenticación multifactor (MFA) antes de conceder acceso a los recursos de AWS. Cuando la MFA está habilitada, usted también debe tener acceso al dispositivo de MFA para iniciar sesión.

Para obtener más información sobre cómo los diferentes usuarios inician sesión en la consola de administración, consulte Inicie sesión en la consola de administración de AWS en la AWSGuía del usuario de inicio de sesión.

Interfaz de línea de comandos (CLI) de AWS y kits de desarrollo de software (SDK)

IAM Identity Center y los usuarios de IAM utilizan diferentes métodos para autenticar sus credenciales cuando se autentican a través de la CLI o las interfaces de aplicación (API) en los SDK asociados.

Las credenciales y las opciones de configuración se encuentran en varios lugares, como las variables de entorno del sistema o del usuario, los archivos de configuración de AWS locales o declarados explícitamente en la línea de comandos como un parámetro. Ciertas ubicaciones tienen prioridad sobre otras.

Tanto IAM Identity Center como IAM proporcionan claves de acceso que se pueden utilizar con la CLI o el SDK. Las claves de acceso de IAM Identity Center son credenciales temporales que se pueden actualizar automáticamente y su uso se recomienda en lugar de las claves de acceso a largo plazo asociadas a los usuarios de IAM.

Para administrar su Cuenta de AWS mediante el uso de la CLI o el SDK, puede utilizar AWS CloudShell desde su navegador. Si usa CloudShell para ejecutar comandos de CLI o SDK, primero debe iniciar sesión en la consola. Los permisos para acceder a los recursos de AWS se basan en las credenciales que utilizó para iniciar sesión en la consola. Según su experiencia, es posible que la CLI le parezca un método más eficiente para administrar su Cuenta de AWS.

Para el desarrollo de aplicaciones, puede descargar la CLI o el SDK en su equipo e iniciar sesión desde el símbolo del sistema o una ventana de Docker. En este escenario, se configuran las credenciales de autenticación y acceso como parte del script de la CLI o de la aplicación SDK. Puede configurar el acceso mediante programación a los recursos de diferentes maneras, según el entorno y el acceso disponibles.

- Las opciones recomendadas para autenticar el código local con el servicio de AWS son IAM Identity Center y Funciones de IAM en cualquier lugar.
- Las opciones recomendadas para autenticar el código que se ejecuta dentro de un entorno de AWS son el uso de las funciones de IAM o de las credenciales de IAM Identity Center.

Si utiliza IAM Identity Center, puede obtener credenciales a corto plazo en la página de inicio del portal de acceso de AWS, donde elige su conjunto de permisos. Estas credenciales tienen una duración definida y no se actualizan automáticamente. Si desea usar estas credenciales, después de iniciar sesión en el portal de AWS, elija la Cuenta de AWS y, a continuación, seleccione el conjunto de permisos. Seleccione Línea de comandos o acceso mediante programación para ver las opciones que puede usar para acceder a los recursos de AWS mediante programación o desde la CLI. Para obtener más información sobre estos métodos, consulte Obtener y actualizar las credenciales temporales en la Guía del usuario del IAM Identity Center. Estas credenciales se suelen utilizar durante el desarrollo de aplicaciones para probar rápidamente el código.

Recomendamos utilizar las credenciales de IAM Identity Center, que se actualizan automáticamente al automatizar el acceso a los recursos de AWS. Si ha configurado usuarios y conjuntos de permisos en IAM Identity Center, utilice el comando `aws configure sso` para utilizar un asistente de línea de comandos que lo ayudará a identificar las credenciales disponibles y almacenarlas en un perfil. Para obtener más información sobre la configuración de su perfil, consulte Configurar su perfil con el `aws configure sso` wizard en la versión 2 de la Guía del usuario de la interfaz de línea de comandos de AWS.

> **Note**
>
> Muchas aplicaciones de ejemplo utilizan claves de acceso a largo plazo asociadas a usuarios de IAM o usuario raíz. Solo debe utilizar credenciales a largo plazo dentro de un entorno aislado como parte de un ejercicio de aprendizaje. Tan pronto como sea posible, revise las alternativas a las claves de acceso a largo plazo y planifique la transición de su código para utilizar credenciales alternativas, como las credenciales del IAM Identity Center o los roles de IAM. Tras realizar la transición del código, elimine las claves de acceso.

Para obtener más información sobre la configuración de la CLI, consulte Instalar o actualizar la última versión de la AWS CLI en la versión 2 de la Guía del usuario de la interfaz de línea de comandos de AWS y Credenciales de autenticación y acceso en la Guía del usuario de la interfaz de línea de comandos de AWS.

Para obtener más información sobre la configuración del SDK, consulte Autenticación de IAM Identity Center en la Guía de referencia de herramientas y AWS SDK y Funciones de IAM en cualquier lugar en la Guía de referencia de herramientas y AWS SDK.

ID y alias de su cuenta de Cuenta de AWS

Los usuarios de IAM de la cuenta inician sesión mediante una URL web que incluye el alias de la cuenta o un identificador de cuenta. Si no tiene la URL, la AWS página de inicio de sesión requiere que proporcione el Cuenta de AWS alias o ID de cuenta.

Si no sabe el seudónimo o el alias de su cuenta:

- Compruebe el historial de su navegador. Si ha iniciado sesión anteriormente, podría almacenarse en sus sitios web recientes.
- Si ha configurado el AWS CLI o un AWS SDK con las credenciales de su cuenta, puede obtener su ID de cuenta de sus archivos de configuración.
- Pregúntele al administrador local o al propietario de la cuenta, AWS no puede proporcionar identificadores de cuenta a los usuarios.

> **Tip**
>
> Para crear un marcador para la página de inicio de sesión de su cuenta en el explorador web, debe escribir manualmente la URL de inicio de sesión en la entrada de marcador. No utilice la función «marcar esta página» de su navegador web, ya que captura información específica de su sesión actual del navegador que interfiere con futuras visitas a la página de inicio de sesión.

Temas

Vea su ID de Cuenta de AWS

Puede encontrar el ID de su cuenta de Cuenta de AWS con los siguientes métodos.

Búsqueda del ID de cuenta con la consola

El ID de la cuenta se muestra en el panel de control de IAM de la sección Cuenta de AWS. Hay maneras adicionales de ver el ID de su cuenta en la consola en función del tipo de usuario. Si asumió un rol, Credenciales de seguridad no está disponible.

Tipo de usuario	Procedimiento
Usuario raíz	En la esquina superior derecha de la barra de navegación, seleccione el nombre de usuario y, a continuación, Credenciales de seguridad. El número de cuenta aparece en Identificadores de cuenta.
Usuario de IAM	En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y el ID de la cuenta se mostrará encima de su nombre de usuario. Elija Security Credentia

Tipo de usuario	Procedimiento
	ls (Credenciales de seguridad). El número de cuenta aparece en Detalles de la cuenta.
Usuario federado	En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y el ID de la cuenta se mostrará encima de su nombre de usuario.
Rol asumido	En la barra de navegación de la parte superior derecha, elija Support (Soporte) y, a continuac ión, Support center (Centro de soporte). El número de cuenta (ID) actual de 12 dígitos que ha iniciado sesión aparece en el panel de navegación Centro de asistencia.

Búsqueda del ID de cuenta usando AWS CLI

Utilice el siguiente comando para ver el ID de usuario, el ID de cuenta y el ARN de usuario:

- aws sts get-caller-identity

Búsqueda del ID de cuenta con la API

Utilice la siguiente API para ver el ID de usuario, el ID de cuenta y el ARN de usuario:

- GetCallerIdentity

Acerca de los alias de cuenta

Si quiere que la dirección URL de la página de inicio de sesión contenga el nombre de su empresa (u otro identificador intuitivo) en lugar del ID de su cuenta de Cuenta de AWS, puede crear un alias de cuenta. En esta sección se proporciona información sobre los alias de la cuenta de Cuenta de AWS y se muestra una lista de las operaciones de la API que se utilizan para crear un alias.

La dirección URL de su página de inicio de sesión tiene, de forma predeterminada, siguiente formato.

```
https://Your_Account_ID.signin.aws.amazon.com/console/
```

Si crea un alias de cuenta de Cuenta de AWS para su ID de cuenta de Cuenta de AWS, la URL de la página de inicio de sesión aparece como se muestra en el siguiente ejemplo.

```
https://Your_Account_Alias.signin.aws.amazon.com/console/
```

Consideraciones

- Su cuenta de Cuenta de AWS puede tener únicamente un alias. Si crea un alias nuevo para su cuenta de AWS, el nuevo alias sobrescribe el alias anterior y la dirección URL que contiene dicho alias anterior dejará de funcionar.
- Debe contener solo dígitos, letras en minúsculas y guiones. Para obtener más información sobre las limitaciones de las entidades de cuentas de AWS, consulte IAM y cuotas de AWS STS.
- El alias de una cuenta debe ser único en todos los productos de Amazon Web Services dentro de una partición de red determinada.

  Una partición es un grupo de regiones de AWS. Cada cuenta de AWS está limitada a una partición.

  Las siguientes son las particiones admitidas:

  - `aws`: regiones de AWS
  - `aws-cn` - Regiones de China
  - `aws-us-gov`: regiones de AWS GovCloud (US)

Creación, eliminación y descripción de alias de cuenta de Cuenta de AWS

Puede utilizar la AWS Management Console, la API de IAM o la interfaz de línea de comandos para crear o eliminar el alias de su cuenta de Cuenta de AWS.

> **Note**
>
> Los alias de cuenta no son secretos y aparecerán en la URL de su página de inicio de sesión pública. No incluya información confidencial en el alias de su cuenta.
> La dirección URL original que contiene el ID de su cuenta de Cuenta de AWS permanece activa y se puede utilizar después de que cree su alias de cuenta de Cuenta de AWS.

Para crear o editar un alias de cuenta (consola)

Puede crear, editar y eliminar un alias de cuenta desde la AWS Management Console.

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`

Para crear o editar un alias de cuenta (consola)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Panel.
3. En la sección Cuenta de AWS, busque Alias de cuenta y elija Crear. Si ya existe un alias, elija Editar.
4. Escriba el nombre que desea utilizar para el alias y, a continuación, elija Guardar cambios.

> **Note**
>
> Solo puede tener un alias asociado a su Cuenta de AWS a la vez. Si crea un alias nuevo, se elimina el alias anterior y la URL de inicio de sesión que estaba asociada al alias anterior deja de funcionar.

Eliminar un alias de cuenta (consola)

Puede crear y eliminar un alias de cuenta desde la AWS Management Console.

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - `iam:ListAccountAliases`

- `iam:CreateAccountAlias`
- `iam:DeleteAccountAlias`

Para eliminar un alias de cuenta (consola)

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Panel.
3. En la sección AWS Cuenta, junto a Alias de cuenta, seleccione Eliminar.

> **Note**
>
> La única URL de inicio de sesión de su cuenta se basa en su ID de cuenta. Cualquier intento de conexión a la URL del alias no se redirige.

Creación, eliminación y descripción de alias (AWS CLI)

> **Note**
>
> Para usar los siguientes comandos, debe tener al menos los siguientes permisos de IAM:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Para crear un alias para la URL de la página de inicio de sesión de la AWS Management Console, ejecute el siguiente comando:

- `aws iam create-account-alias`

Para eliminar un alias de ID de cuenta de Cuenta de AWS, ejecute el siguiente comando:

- `aws iam delete-account-alias`

Para mostrar el alias de ID de cuenta de Cuenta de AWS, ejecute el siguiente comando:

- `aws iam list-account-aliases`

Example Comandos de alias

Para mostrar el alias de ID de cuenta de Cuenta de AWS, ejecute el siguiente comando.

```
$ aws iam list-account-aliases
{
    "AccountAliases": [
        "myaccountalias"
    ]
}
```

Para crear un alias para su inicio de sesión de AWS Management Console, ejecute el siguiente comando:

```
$ aws iam create-account-alias \
    --account-alias myaliasname
```

Este comando no genera ningún resultado si se utiliza correctamente.

Para eliminar un alias de ID de cuenta de Cuenta de AWS, ejecute el siguiente comando.

```
$ aws iam delete-account-alias \
--account-alias myaliasname
```

Este comando no genera ningún resultado si se utiliza correctamente.

Creación, eliminación y descripción de alias (API de AWS)

> **Note**
>
> Para utilizar las siguientes operaciones de la API, debe tener al menos los siguientes permisos de IAM:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`

- `iam:DeleteAccountAlias`

Para crear un alias para la URL de la página de inicio de sesión de la AWS Management Console, llame a la siguiente operación:

- `CreateAccountAlias`

Para eliminar un alias de ID de cuenta de Cuenta de AWS, llame a la siguiente operación:

- `DeleteAccountAlias`

Para mostrar su alias de ID de cuenta de Cuenta de AWS, llame a la siguiente operación:

- `ListAccountAliases`

Introducción a IAM

Aproveche este tutorial para comenzar con AWS Identity and Access Management (IAM). Aprenderá a crear roles, usuarios y políticas mediante la AWS Management Console.

AWS Identity and Access Management es una característica de su Cuenta de AWS que se ofrece sin cargo adicional. Solo se le cobrará por otros productos de AWS que utilicen los usuarios de IAM. Para obtener información acerca de los precios de otros productos de AWS, consulte la Página de precios de Amazon Web Services.

> **Note**
>
> Este juego de documentación aborda principalmente el servicio de IAM. Para obtener información acerca de cómo empezar a utilizar AWS y varios servicios para solucionar un problema, como crear y lanzar su primer proyecto, consulte Centro de recursos introductorios.

Contenido

Requisitos previos

Antes de comenzar, asegúrese de que ha realizado los pasos que se detallan en Configuración de IAM. En este tutorial se utiliza la cuenta de administrador que creó en ese procedimiento.

Cree su primer usuario de IAM

Un usuario de IAM es una identidad dentro de su Cuenta de AWS que dispone de permisos específicos para una sola persona o aplicación. Los usuarios se pueden organizar en grupos que comparten los mismos permisos.

> **Note**
>
> Como práctica recomendada de seguridad, le recomendamos que proporcione acceso a los recursos mediante la federación de identidades en lugar de crear usuarios de IAM. Para obtener más información acerca de situaciones específicas en las que se requiere un usuario de IAM, consulte Cuándo crear un usuario de IAM (en lugar de un rol).

Para que se familiarice con el proceso de creación de un usuario de IAM, este tutorial explica cómo crear un grupo y un usuario de IAM para obtener acceso de emergencia.

Para crear su primer usuario de IAM

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS.
2. En la página principal de la consola, seleccione el servicio de IAM.
3. En el panel de navegación, seleccione Usuarios y luego, elija Agregar usuarios.

   > **Note**
   >
   > Si tiene habilitado el IAM Identity Center, la AWS Management Console muestra un recordatorio de que es mejor administrar el acceso de los usuarios en IAM Identity Center. En este tutorial, el usuario de IAM que cree se utilizará únicamente cuando las credenciales del usuario de IAM Identity Center no estén disponibles.

4. En Nombre de usuario, escriba **`EmergencyAccess`**. Los nombres no pueden contener espacios.
5. Seleccione la casilla de verificación junto a Proporcionar al usuario acceso a la AWS Management Console: optional y luego, elija Quiero crear un usuario de IAM.
6. En Contraseña de la consola, seleccione Contraseña generada de manera automática.
7. Desmarque la casilla de verificación junto a El usuario debe crear una contraseña nueva la próxima vez que inicie sesión (recomendado). Dado que este usuario de IAM es para acceso de emergencia, un administrador de confianza conserva la contraseña y solo la proporciona cuando es necesario.
8. En la página Establecer permisos, en Opciones de permisos, seleccione Agregar usuario al grupo. Luego, en Grupos de usuarios, seleccione Crear grupo.

9. En la página Crear grupo de usuarios, en Nombre del grupo de usuarios, ingrese **`EmergencyAccessGroup`**. Luego, en Políticas de permisos, seleccione AdministratorAccess.
10. Seleccione Crear grupo de usuarios para volver a la página Establecer permisos.
11. En Grupos de usuarios, seleccione el nombre del **`EmergencyAccessGroup`** que creó anteriormente.
12. Seleccione Siguiente para dirigirse a la página Revisar y crear.
13. En la página Revisar y crear, revise la lista de suscripciones a grupos de usuarios que se agregarán al usuario nuevo. Cuando esté listo para continuar, seleccione Crear usuario.
14. En la página Recuperar contraseña, seleccione Descargar archivo .csv para guardar un archivo .csv con la información de las credenciales del usuario (URL de conexión, nombre del usuario y contraseña).
15. Guarde este archivo para usarlo si necesita iniciar sesión en IAM y no tiene acceso a su proveedor de identidad federada.

El usuario de IAM nuevo aparece en la lista Usuarios. Seleccione el enlace Nombre del usuario para ver los detalles del usuario. En Resumen, copie el ARN del usuario en el portapapeles. Pegue el ARN en un documento de texto para poder usarlo en el siguiente procedimiento.

Cree su primer rol

Los roles de IAM son una forma segura de conceder permisos a las entidades en las que confía. Un rol de IAM tiene algunas similitudes con un usuario de IAM. Los roles y los usuarios son entidades principales con políticas de permisos que determinan lo que la identidad puede y no puede hacer en AWS. No obstante, en lugar de asociarse exclusivamente a una persona, la intención es que cualquier usuario pueda asumir un rol que necesite. Además, un rol no tiene asociadas credenciales a largo plazo estándar, como una contraseña o claves de acceso. En su lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol. El uso de roles le ayuda a seguir las prácticas recomendadas de IAM. Puede usar un rol para:

- Permitir que las identidades de los trabajadores y las aplicaciones habilitadas para Identity Center accedan a la AWS Management Console mediante AWS IAM Identity Center.
- Delegar el permiso a un servicio de AWS para que lleve a cabo acciones en su nombre.
- Habilitar el código de la aplicación que se ejecuta en una instancia de Amazon EC2 para acceder a los recursos de AWS o modificarlos.

- Otorgamiento de acceso a otra Cuenta de AWS.

> **Note**
>
> Puede usar Roles de AWS Identity and Access Management en cualquier lugar para dar acceso a identidades de máquinas. El uso de Roles de IAM en cualquier lugar significa que no es necesario administrar credenciales a largo plazo para cargas de trabajo que se ejecutan fuera de AWS. Para obtener más información, consulte ¿Qué es Roles de AWS Identity and Access Management en cualquier lugar? en la Guía del usuario de Roles de AWS Identity and Access Management en cualquier lugar.

IAM Identity Center y otros servicios de AWS crean de forma automática roles para sus servicios. Si utiliza usuarios de IAM, le recomendamos que cree roles para que los usuarios los asuman cuando inicien sesión. De esta manera, se les concederán permisos temporales durante la sesión en lugar de permisos a largo plazo.

El asistente de la AWS Management Console que lo guía en la creación de un rol muestra pasos que varían ligeramente en función de si se crea un rol para un usuario de IAM, un servicio de AWS o un usuario federado. El acceso regular a las Cuentas de AWS dentro de una organización debe proporcionarse mediante acceso federado. Si crea usuarios de IAM para fines específicos, como acceso de emergencia o acceso programático, solo conceda a esos usuarios de IAM permiso para que asuman un rol y coloque a esos usuarios de IAM en grupos específicos de roles.

En este procedimiento, crea un rol que proporciona acceso SupportUser para el usuario de IAM EmergencyAccess. Antes de iniciar el procedimiento, copie el ARN del usuario de IAM en el portapapeles.

Para crear un rol para un usuario de IAM

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS.
2. En la página principal de la consola, seleccione el servicio de IAM.
3. En el panel de navegación de la consola de IAM, elija Roles y, a continuación, elija Crear rol.
4. Elija el tipo de rol de Cuenta de AWS.
5. En Seleccionar entidad de confianza, en Tipo de entidad de confianza, elija Política de confianza personalizada.

6. En la sección Política de confianza personalizada, revise la política de confianza básica. Es la que utilizaremos para este rol. Utilice el editor Editar instrucción para actualizar la política de confianza:

   1. En Agregar acciones para STS, seleccione Asumir rol.
   2. Junto a Agregar una entidad principal, seleccione Agregar. Se abrirá la ventana Agregar entidad principal.

      En Tipo de entidad principal, seleccione Usuarios de IAM.

      En ARN, pegue el ARN del usuario de IAM que copió en el portapapeles.

      Seleccione Agregar entidad principal.
   3. Compruebe que la línea `Principal` de la política de confianza ahora contenga el ARN que especificó:

      ```
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/username" }
      ```
7. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.
8. En Agregar permisos, seleccione la casilla de verificación junto a la política de permisos que desea aplicar. Para este tutorial seleccionaremos la política de confianza SupportUser. Puede utilizar este rol para solucionar problemas con la Cuenta de AWS y abrir casos de soporte con AWS. No vamos a establecer un límite de permisos en este momento.
9. Elija Siguiente.
10. En Nombrar, revisar y crear complete los siguientes ajustes:

    - En Nombre del rol, ingrese un nombre que identifique al rol, como SupportUserRole.
    - En Descripción, explique el uso previsto del rol.

    Dado que es posible que otros recursos de AWS hagan referencia al rol, no se puede editar el nombre del rol después de crearlo.
11. Seleccione Crear rol.

    Una vez creado el rol, comparta la información del rol con las personas que lo necesiten. Puede compartir la información del rol de la siguiente manera:

- Enlace de rol: envíe a los usuarios un enlace que los lleve a la página Switch Role (Cambiar el rol) con todos los detalles ya completados.
- ID de cuenta o alias: proporcione a cada usuario el nombre de la función junto con el número de ID de cuenta o alias de cuenta. El usuario se dirige a la página Switch Role (Cambiar rol) y agrega los detalles manualmente.
- La información del enlace del rol se puede guardar junto con las credenciales del usuario EmergencyAccess.

Para más información, consulte Proporcionar información al usuario.

Cree su primera política de IAM

Las políticas de IAM se asocian a identidades de IAM (usuarios, grupos de usuarios o roles) o a recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos.

Para crear su primera política de IAM

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS.
2. En la página principal de la consola, seleccione el servicio de IAM.
3. En el panel de navegación, seleccione Policies (Políticas).

   Si es la primera vez que elige Policies (Políticas), aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Get Started (Comenzar).
4. Elija Create Policy (Crear política).
5. En la página Crear política, elija Acciones y, a continuación, elija Importar política.
6. En la ventana Importar política, en el cuadro Buscar políticas, ingrese **`power`** para reducir la lista de políticas. Seleccione la política PowerUserAccess.
7. Seleccione Importar política. La política se muestra en la pestaña JSON.
8. Elija Siguiente.
9. En la página Revisar y crear, en Nombre de la política, ingrese **`PowerUserExamplePolicy`**. En Description (Descripción), escriba **`Allows full access to all services except those for user management`**. Luego, elija Crear política para guardar la política.

Puede asociar esta política a un rol y así proporcionar a los usuarios que asumirán el rol los permisos asociados a la política. La política PowerUserAccess se utiliza normalmente para proporcionar acceso a los desarrolladores.

Acceso programático

Los usuarios necesitan acceso programático si desean interactuar con AWS fuera de la AWS Management Console. La forma de conceder el acceso programático depende del tipo de usuario que acceda a AWS:

- Si administra las identidades en el IAM Identity Center, las API de AWS requieren un perfil y AWS Command Line Interface requiere un perfil o una variable de entorno.
- Si tiene usuarios de IAM, las API de AWS y AWS Command Line Interface requieren claves de acceso. En la medida de lo posible, cree credenciales temporales que incluyan un ID de clave de acceso y una clave de acceso secreta, pero, además, un token de seguridad que indique cuándo caducan las credenciales.

Para conceder acceso programático a los usuarios, elija una de las siguientes opciones.

¿Qué usuario necesita acceso programático?	Para	Mediante
Identidad del personal (Usuarios administrados en el IAM Identity Center)	Utilice credenciales a corto plazo para firmar las solicitud es programáticas a las API de AWS CLI o AWS (directam ente o mediante los AWS SDK).	Siga las instrucciones de la interfaz que desea utilizar: • Para la AWS CLI, siga las instrucciones que se indican en Getting IAM role credentials for CLI access (Obtener las credencia les de rol de IAM para el acceso a la CLI) de la Guía del usuario de AWS IAM Identity Center. • Para las API de AWS, siga las instrucciones en credenciales de SSO de la

¿Qué usuario necesita acceso programático?	Para	Mediante
		Guía de referencia de SDK y herramientas de AWS.
IAM	Utilice credenciales a corto plazo para firmar las solicitudes programáticas a la AWS CLI o las API de AWS (directamente o mediante los SDK de AWS).	Siga las instrucciones que se detallan en Uso de credenciales temporales con recursos de AWS.
IAM	Utilice credenciales a largo plazo para firmar las solicitudes programáticas a las API de AWS CLI o AWS (directamente o mediante los AWS SDK). (no recomendado)	Siga las instrucciones que se detallan en Administración de las claves de acceso de los usuarios de IAM.

Prácticas de seguridad recomendadas y casos de uso en AWS Identity and Access Management

AWS Identity and Access Management (IAM) cuenta con una serie de características de seguridad que debe tener en cuenta a la hora de desarrollar e implementar sus propias políticas de seguridad. Las siguientes prácticas recomendadas son directrices generales y no suponen una solución de seguridad completa. Puesto que es posible que estas prácticas recomendadas no sean adecuadas o suficientes para el entorno, considérelas como consideraciones útiles en lugar de como normas.

Para obtener los mayores beneficios de IAM, dedique tiempo a aprender las prácticas recomendadas. Una forma de hacerlo es comprobar cómo IAM se usa en situaciones reales con el fin de utilizarlo con otros servicios deAWS .

Temas

Prácticas recomendadas de seguridad en IAM

Follow us on Twitter

Las prácticas recomendadas para AWS Identity and Access Management se actualizaron el 14 de julio de 2022.

Para contribuir a proteger los recursos de AWS, siga estas prácticas recomendadas para AWS Identity and Access Management (IAM).

Temas

- Actualizar las claves de acceso cuando sea necesario para casos de uso que requieren credenciales de larga duración
- Siga las prácticas recomendadas para proteger las credenciales de usuario raíz
- Aplicar permisos de privilegios mínimos
- Introducción a las políticas administradas de AWS y el objetivo de los permisos de privilegios mínimos
- Utilizar IAM Access Analyzer para generar políticas de privilegios mínimos basadas en la actividad de acceso
- Revisar y eliminar periódicamente usuarios, roles, permisos, políticas y credenciales no utilizados
- Utilizar condiciones en las políticas de IAM para restringir aún más el acceso
- Verificar el acceso público y entre cuentas a los recursos con IAM Access Analyzer
- Utilizar IAM Access Analyzer para validar las políticas de IAM con objeto de garantizar la seguridad y funcionalidad de los permisos
- Establecer barreras de protección de permisos en varias cuentas
- Utilizar límites de permisos para delegar la administración de permisos de una cuenta

Exigir que los usuarios humanos utilicen la federación con un proveedor de identidades para acceder a AWScon credenciales temporales

Los usuarios humanos, que también reciben el nombre de identidades humanas, son las personas, los administradores, los desarrolladores, los operadores y los consumidores de las aplicaciones. Deben tener una identidad para acceder a los entornos y aplicaciones de AWS. Los usuarios humanos que son miembros de su organización también reciben el nombre de identidades de personal. Los usuarios humanos también pueden ser usuarios externos con los que colabora y que interactúan con los recursos de AWS. Pueden hacer esto a través de un navegador web, una aplicación cliente, una aplicación móvil o herramientas de línea de comandos interactivas.

Exija a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. Puede utilizar un proveedor de identidades con sus usuarios humanos para proporcionar acceso federado a las cuentas de Cuentas de AWS asumiendo roles, que proporcionan credenciales temporales. Si desea administrar el acceso de manera centralizada, se recomienda utilizar AWS IAM Identity Center (IAM Identity Center) para administrar el acceso a las cuentas y los permisos de esas cuentas. Puede administrar las identidades de los usuarios con IAM Identity Center, o bien administrar los permisos de acceso para las identidades de los usuarios en IAM Identity Center de un proveedor de

identidades externo. Para obtener más información, consulte ¿Qué es AWS IAM Identity Center? en la Guía del usuario de AWS IAM Identity Center.

Para obtener más información acerca de los roles de , consulte Términos y conceptos de roles.

Exigir que las cargas de trabajo utilicen credenciales temporales con roles de IAM para acceder a AWS

Una carga de trabajo es un conjunto de recursos y código que ofrece valor comercial, como una aplicación o un proceso de backend. Una carga de trabajo puede tener aplicaciones, herramientas operativas y componentes que requieren una identidad para realizar solicitudes a Servicios de AWS, tales como solicitudes de lectura de datos. Estas identidades incluyen máquinas que se ejecutan en los entornos de AWS, como instancias de Amazon EC2 o funciones de AWS Lambda.

También se pueden administrar identidades de máquina para las partes externas que necesiten acceso. Para dar acceso a las identidades de máquina, puede utilizar roles de IAM. Los roles de IAM tienen permisos específicos y ofrecen una forma de acceder a AWS empleando credenciales de seguridad temporales con una sesión de rol. Además, es posible que tenga máquinas fuera de AWS que necesiten acceso a los entornos de AWS. Para máquinas que se ejecuten fuera de AWS, puede utilizar AWS Identity and Access Management Roles Anywhere. Para obtener más información acerca de los roles de , consulte Roles de IAM. Para obtener detalles sobre cómo utilizar roles para delegar el acceso en Cuentas de AWS, consulte Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.

Exigir autenticación multifactor (MFA)

Se recomienda utilizar roles de IAM para los usuarios humanos y las cargas de trabajo que accedan a los recursos de AWS con objeto de que utilicen credenciales temporales. Sin embargo, para escenarios en los que necesite un usuario de IAM o usuario raíz en su cuenta, requiera MFA para seguridad adicional. Con MFA, los usuarios tienen un dispositivo que genera una respuesta a un reto de autenticación. Las credenciales de cada usuario y la respuesta generada por el dispositivo son necesarias para completar el proceso de inicio de sesión. Para obtener más información, consulte Uso de autenticación multifactor (MFA) en AWS.

Si utiliza IAM Identity Center para la administración centralizada del acceso de los usuarios humanos, puede emplear las capacidades MFA de IAM Identity Center cuando el origen de identidades esté configurado con el almacén de identidades de IAM Identity Center, AWS Managed Microsoft AD o AD Connector. Para obtener más información sobre MFA en IAM Identity Center, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center.

Actualizar las claves de acceso cuando sea necesario para casos de uso que requieren credenciales de larga duración

Siempre que sea posible, se recomienda emplear credenciales temporales, en lugar de crear credenciales de larga duración tales como claves de acceso. No obstante, en aquellas situaciones en las que necesite usuarios de IAM con acceso mediante programación y credenciales a largo plazo, recomendamos actualizar las claves de acceso cuando sea necesario, como, por ejemplo, cuando un empleado deja la empresa. Se recomienda utilizar la información sobre los últimos accesos de IAM para actualizar y eliminar las claves de acceso de manera segura. Para obtener más información, consulte Actualización de las claves de acceso.

Hay casos de uso específicos que requieren credenciales de larga duración con usuarios de IAM en AWS. Estos son algunos de los casos de uso:

- Cargas de trabajo que no pueden utilizar roles de IAM: puede ejecutar una carga de trabajo desde una ubicación que necesite acceder a AWS. En algunas situaciones, no se pueden utilizar roles de IAM para proporcionar credenciales temporales; por ejemplo, en el caso de los complementos de WordPress. En esas situaciones, utilice claves de acceso a largo plazo de usuarios de IAM para que la carga de trabajo se autentique en AWS.
- Clientes de AWS de terceros: si utiliza herramientas que no admiten el acceso con IAM Identity Center, como clientes de AWS de terceros o proveedores que no están alojados en AWS, utilice claves de acceso de larga duración de usuarios de IAM.
- Acceso a AWS CodeCommit: si utiliza CodeCommit para almacenar el código, puede emplea un usuario de IAM con claves SSH o credenciales específicas del servicio para que CodeCommit se autentique en los repositorios. Se recomienda hacer esto además de utilizar un usuario de IAM Identity Center para la autenticación normal. Los usuarios de IAM Identity Center son el personal que necesita acceso a sus Cuentas de AWS o a sus aplicaciones en la nube. Para dar acceso a los usuarios a los repositorios de CodeCommit sin configurar usuarios de IAM, puede configurar la utilidad git-remote-codecommit. Para obtener más información sobre IAM y CodeCommit, consulte Uso de IAM con CodeCommit: credenciales de Git, claves SSH y claves de acceso de AWS. Para obtener más información sobre cómo configurar la utilidad git-remote-codecommit, consulte Conexión a repositorios de AWS CodeCommit con credenciales rotativas en la Guía del usuario de AWS CodeCommit.
- Acceso a Amazon Keyspaces (para Apache Cassandra): en una situación en la que no pueda utilizar usuarios de IAM Identity Center, como, por ejemplo, para probar la compatibilidad con Cassandra, puede utilizar un usuario de IAM con credenciales específicas del servicio para realizar

la autenticación en Amazon Keyspaces. Los usuarios de IAM Identity Center son el personal que necesita acceso a sus Cuentas de AWS o a sus aplicaciones en la nube. También puede conectarse a Amazon Keyspaces con credenciales temporales. Para obtener más información, consulte Uso de credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4 en la Guía para desarrolladores de Amazon Keyspaces (para Apache Cassandra).

Siga las prácticas recomendadas para proteger las credenciales de usuario raíz

Cuando se crea una Cuenta de AWS, se establecen credenciales de usuario raíz para iniciar sesión en la AWS Management Console. Proteja sus credenciales de usuario raíz del mismo modo que protegería otra información personal confidencial. Para comprender mejor cómo proteger y escalar los procesos de usuario raíz, consulte Prácticas recomendadas para el usuario raíz para la Cuenta de AWS.

Aplicar permisos de privilegios mínimos

Cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para llevar a cabo una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Puede empezar con permisos amplios mientras va conociendo los permisos que se necesitan para su carga de trabajo o caso de uso. A medida que su caso de uso vaya madurando, puede ir reduciendo los permisos que concede para alcanzar el objetivo de privilegio mínimo. Para obtener más información sobre cómo utilizar IAM para aplicar permisos, consulte Políticas y permisos en IAM.

Introducción a las políticas administradas de AWS y el objetivo de los permisos de privilegios mínimos

Para empezar a conceder permisos a los usuarios y las cargas de trabajo, utilice las políticas administradas de AWS, que conceden permisos para muchos casos de uso habituales. Están disponibles en la Cuenta de AWS. Tenga presente que es posible que las políticas administradas de AWS no concedan permisos de privilegios mínimos para sus casos de uso concretos, ya que están disponibles para que las utilicen todos los clientes de AWS. En consecuencia, se recomienda reducir aún más los permisos definiendo políticas administradas por el cliente específicas para sus

casos de uso. Para obtener más información, consulte Políticas administradas de AWS. Para obtener más información acerca de las políticas administradas AWS que están diseñadas para funciones de trabajo específicas, consulte Managed Policies de AWS para funciones de trabajo.

Utilizar IAM Access Analyzer para generar políticas de privilegios mínimos basadas en la actividad de acceso

Para conceder solo los permisos necesarios para llevar a cabo una tarea, puede generar políticas que se basen en la actividad de acceso que haya iniciado sesión en AWS CloudTrail. IAM Access Analyzer analiza los servicios y las acciones que utilizan los roles de IAM, y luego genera una política detallada que se puede emplear. Después de probar cada política generada, puede implementarla en el entorno de producción. Eso garantiza que solo se concedan los permisos necesarios a las cargas de trabajo. Para obtener más información acerca de la generación de políticas, consulte Generación de políticas de IAM Access Analyzer.

Revisar y eliminar periódicamente usuarios, roles, permisos, políticas y credenciales no utilizados

Es posible que tenga usuarios, roles, permisos, políticas o credenciales de IAM que ya no necesite en la Cuenta de AWS. IAM ofrece información sobre último acceso para ayudar a identificar usuarios, roles, permisos, políticas y credenciales que ya no se necesitan y poder eliminarlos. Esto ayuda a reducir la cantidad de usuarios, roles, permisos, políticas y credenciales que hay que monitorear. También puede utilizar esta información para ajustar las políticas de IAM de modo que cumplan mejor con el objetivo de permisos de privilegios mínimos. Para obtener más información, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Utilizar condiciones en las políticas de IAM para restringir aún más el acceso

Puede especificar las condiciones en las que se aplica una instrucción de política. De esa forma, puede conceder acceso a acciones y recursos, pero solo si la solicitud de acceso cumple con determinadas condiciones. Por ejemplo, puede escribir una condición de política para especificar que todas las solicitudes deben enviarse utilizando SSL. También puede utilizar condiciones para conceder acceso a acciones de servicios, pero solo si se emplean a través de un determinado Servicio de AWS, como, por ejemplo, AWS CloudFormation. Para obtener más información, consulte Elementos de política JSON de IAM: Condition.

Verificar el acceso público y entre cuentas a los recursos con IAM Access Analyzer

Antes de conceder permisos de acceso público o entre cuentas en AWS, se recomienda verificar si tal acceso es necesario. Puede utilizar IAM Access Analyzer como ayuda para obtener una vista previa y analizar el acceso público y entre cuentas de los tipos de recursos admitidos. Para ello, consulte los hallazgos que genera IAM Access Analyzer. Estos hallazgos ayudan a verificar que los controles de acceso a los recursos conceden el acceso que se espera. Además, a medida que actualice los permisos públicos y entre cuentas, puede verificar el efecto de los cambios antes de implementar nuevos controles de acceso en los recursos. IAM Access Analyzer también monitorea continuamente los tipos de recursos admitidos y genera un hallazgo para aquellos recursos que permitan el acceso público o entre cuentas. Para obtener más información, consulte Vista previa del acceso con las API de IAM Access Analyzer.

Utilizar IAM Access Analyzer para validar las políticas de IAM con objeto de garantizar la seguridad y funcionalidad de los permisos

Valide las políticas que cree para asegurarse de que respetan el lenguaje de las políticas de IAM (JSON) y las prácticas recomendadas para IAM. Puede validar las políticas mediante la validación de políticas de IAM Access Analyzer. IAM Access Analyzer proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. A medida que se crean nuevas políticas o se editan políticas existentes en la consola, IAM Access Analyzer proporciona recomendaciones para ayudar a ajustarlas y validarlas antes de guardarlas. Además, se recomienda revisar y validar todas las políticas existentes. Para obtener más información, consulte Validación de políticas de IAM Access Analyzer. Para obtener más información sobre las verificaciones de políticas que proporciona IAM Access Analyzer, consulte Referencia de la verificación de políticas de IAM Access Analyzer.

Establecer barreras de protección de permisos en varias cuentas

A medida que vaya ampliando las cargas de trabajo, sepárelas utilizando varias cuentas que se gestionen con AWS Organizations. Se recomienda utilizar políticas de control de servicio (SCP) de Organizations para establecer barreras de protección de permisos con objeto de controlar el acceso de todos los usuarios y roles de IAM en las cuentas. Las SCP son un tipo de política de organización que se puede utilizar para administrar los permisos en la organización en el nivel de la organización, unidad organizativa o cuenta de AWS. Las barreras de protección de permisos que se establecen se aplican a todos los usuarios y roles de las cuentas relacionadas. No obstante, las SCP por sí solas

no bastan para conceder permisos a las cuentas de la organización. Con ese fin, el administrador debe adjuntar políticas basadas en identidad o en recursos a los usuarios de IAM, los roles de IAM o los recursos de las cuentas. Para obtener más información, consulte AWS Organizations, cuentas y barreras de protección de IAM.

Utilizar límites de permisos para delegar la administración de permisos de una cuenta

En algunas situaciones, es posible que desee delegar la administración de permisos de una cuenta en otras personas. Por ejemplo, puede dejar que los desarrolladores creen y administren roles para sus cargas de trabajo. Cuando delegue permisos en otras personas, utilice límites de permisos para establecer los permisos máximos que delega. Un límite de permisos es una característica avanzada para utilizar una política administrada con el fin de establecer los permisos máximos que una política basada en identidad puede conceder a un rol de IAM. Un límite de permisos no concede permisos por sí mismo. Para obtener más información, consulte Límites de permisos para las entidades de IAM.

Prácticas recomendadas para el usuario raíz para la Cuenta de AWS

Cuando crea una Cuenta de AWS por primera vez, comienza con un conjunto predeterminado de credenciales con acceso completo a todos los recursos de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS. Se recomienda encarecidamente no acceder al usuario raíz de la Cuenta de AWS a menos que exista una tarea que requiera credenciales de usuario raíz. Debe proteger las credenciales de usuario raíz y los mecanismos de recuperación de la cuenta para asegurarse de no exponer las credenciales dotadas de muchos privilegios a un uso no autorizado.

En lugar de acceder al usuario raíz, cree un usuario administrativo para las tareas cotidianas.

- En el caso de una Cuenta de AWS única e independiente, consulte Crear un usuario administrativo.
- Si se trata de varias Cuentas de AWS administradas mediante AWS Organizations, consulte Set up Cuenta de AWS access for an IAM Identity Center administrative user.

Con el usuario administrativo podrá crear identidades adicionales para los usuarios que necesiten acceder a los recursos de la Cuenta de AWS. Se recomienda encarecidamente exigir que los usuarios se autentiquen con credenciales temporales al acceder a AWS.

- En el caso de una Cuenta de AWS única e independiente, utilice Roles de IAM para crear identidades en la cuenta con permisos específicos. Los roles están pensados para que los pueda asumir cualquier persona que los necesite. Además, un rol no tiene asociadas credenciales a largo plazo estándar, tales como una contraseña o claves de acceso. En su lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol. A diferencia de los roles de IAM, los Usuarios de IAM tienen credenciales a largo plazo, tales como contraseñas y claves de acceso. Siempre que sea posible, las prácticas recomendadas sugieren emplear credenciales temporales, en lugar de crear usuarios de IAM con credenciales a largo plazo como contraseñas y claves de acceso.
- En el caso de varias Cuentas de AWS administradas mediante Organizations, utilice usuarios de personal de IAM Identity Center. Con IAM Identity Center, puede administrar de manera centralizada los usuarios de sus Cuentas de AWS y los permisos de esas cuentas. Administre las identidades de los usuarios con IAM Identity Center o desde un proveedor de identidades externo. Para obtener más información, consulte ¿Qué es AWS IAM Identity Center? en la Guía del usuario de AWS IAM Identity Center.

Temas

- Proteja las credenciales de usuario raíz para evitar el uso no autorizado
- Utilizar una contraseña de usuario raíz segura para ayudar a proteger el acceso
- Proteja el inicio de sesión del usuario raíz con autenticación multifactor (MFA)
- No crear claves de acceso para el usuario raíz
- Utilice la aprobación de varias personas para el inicio de sesión del usuario raíz siempre que sea posible
- Utilice una dirección de correo de un grupo para las credenciales de usuario raíz
- Limite el acceso a los mecanismos de recuperación de cuentas
- Proteja las credenciales de usuario raíz de su cuenta de Organizations
- Supervise el acceso y el uso

Proteja las credenciales de usuario raíz para evitar el uso no autorizado

Proteja las credenciales de usuario raíz y utilícelas solo para las tareas que las requieren. Para evitar el uso no autorizado, no comparta la contraseña de usuario raíz, la MFA, las claves de acceso, los pares de claves de CloudFront ni los certificados de firma con nadie, excepto con aquellas personas que tengan una necesidad estrictamente empresarial de acceder al usuario raíz.

No guarde la contraseña de usuario raíz con herramientas que dependan de Servicios de AWS en una cuenta a la que se acceda con la misma contraseña. Si pierde u olvida la contraseña de usuario raíz, no podrá acceder a estas herramientas. Se recomienda que priorice la resiliencia y considere la posibilidad de solicitar que dos o más personas autoricen el acceso a la ubicación de almacenamiento. Se debe registrar y supervisar el acceso a la contraseña o a su ubicación de almacenamiento.

Utilizar una contraseña de usuario raíz segura para ayudar a proteger el acceso

Se recomienda utilizar una contraseña segura y única. Herramientas como los administradores de contraseñas con algoritmos de generación de contraseñas seguras pueden ayudar a lograr estos objetivos. AWS requiere que la contraseña cumpla las siguientes condiciones:

- Debe tener 8 caracteres como mínimo y 128 como máximo.
- Debe incluir, como mínimo, tres de estos tipos de caracteres combinados: mayúsculas, minúsculas, números y símbolos ! @ # $ % ^ & * () <> [] {} | _+-=.
- No debe ser idéntica al nombre de la Cuenta de AWS ni a la dirección de correo electrónico.

Para obtener más información, consulte Cambiar la contraseña para Usuario raíz de la cuenta de AWS.

Proteja el inicio de sesión del usuario raíz con autenticación multifactor (MFA)

Dado que un usuario raíz puede realizar acciones privilegiadas, es fundamental agregar MFA para el usuario raíz como segundo factor de autenticación, además de la dirección de correo electrónico y la contraseña como credenciales de inicio de sesión. Se recomienda encarecidamente habilitar varias MFA para las credenciales de usuario raíz, con el fin de dotar de flexibilidad y resiliencia adicionales

a su estrategia de seguridad. Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el usuario raíz de la Cuenta de AWS.

- Las claves de seguridad de hardware certificadas por FIDO las proporcionan proveedores externos. Para obtener más información, consulte Habilitación de una clave de seguridad FIDO para el usuario raíz de la Cuenta de AWS.
- Dispositivo de hardware que genera un código numérico de seis dígitos basado en el algoritmo de contraseña temporal de un solo uso (TOTP). Para obtener más información, consulte Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS.
- Aplicación de autenticador virtual que se ejecuta en un teléfono u otro dispositivo y emula un dispositivo físico. Para obtener más información, consulte Habilitación de un dispositivo MFA virtual para su usuario raíz de la Cuenta de AWS.

No crear claves de acceso para el usuario raíz

Las claves de acceso permiten ejecutar comandos en la interfaz de línea de comandos de AWS (AWS CLI) o utilizar operaciones de la API de alguno de los AWS SDK. Se recomienda encarecidamente que no cree pares de claves de acceso para el usuario raíz, ya que el usuario raíz tiene acceso total a todos los Servicios de AWS y recursos de la cuenta, incluida la información de facturación.

Dado que el usuario raíz solo es necesario en pocas tareas y, por lo general, esas tareas se realizan con poca frecuencia, se recomienda iniciar sesión en la AWS Management Console para realizar las tareas del usuario raíz. Antes de crear claves de acceso, revise las alternativas a las claves de acceso a largo plazo.

Utilice la aprobación de varias personas para el inicio de sesión del usuario raíz siempre que sea posible

Considere la posibilidad de utilizar la aprobación de varias personas para garantizar que ninguna persona pueda acceder tanto a la MFA como a la contraseña del usuario raíz. Algunas empresas agregan una capa de seguridad adicional configurando un grupo de administradores con acceso a la contraseña y otro grupo de administradores con acceso a la MFA. Un miembro de cada grupo debe dar su visto bueno para iniciar sesión con las credenciales de usuario raíz.

Utilice una dirección de correo de un grupo para las credenciales de usuario raíz

Utilice una dirección de correo electrónico que esté administrada por su empresa y reenvíe los mensajes recibidos directamente a un grupo de usuarios. Si AWS debe contactar con el propietario de la cuenta, este enfoque reduce el riesgo de retrasos en la respuesta, incluso si la persona en cuestión está de vacaciones, se enferma o deja la empresa. La dirección de correo electrónico utilizada para el usuario raíz no debe utilizarse para otros fines.

Limite el acceso a los mecanismos de recuperación de cuentas

Asegúrese de desarrollar un proceso para administrar los mecanismos de recuperación de credenciales de usuario raíz en caso de que necesite acceder en caso de emergencia, como, por ejemplo, una apropiación de su cuenta administrativa.

- Asegúrese de tener acceso a la bandeja de entrada de correo electrónico del usuario raíz para poder restablecer la contraseña de usuario raíz en caso de pérdida u olvido.
- Si la MFA del usuario raíz de la Cuenta de AWS se pierde, se daña o no funciona, puede iniciar sesión con otra MFA registrada con las mismas credenciales de usuario raíz. Si pierde el acceso a todas sus MFA, necesitará que tanto el número de teléfono como el correo electrónico que utilizó para registrar su cuenta estén actualizados y sean accesibles para recuperar la MFA. Para obtener más información, consulte Recuperación de un dispositivo MFA de usuario raíz.
- Si decide no almacenar la contraseña de usuario raíz y la MFA, se puede utilizar el número de teléfono registrado en su cuenta como forma alternativa de recuperar las credenciales de usuario raíz. Asegúrese de tener acceso al número de teléfono de contacto, mantenga ese número de teléfono actualizado y limite quién tiene acceso para administrarlo.

Ninguna persona debe tener acceso tanto a la bandeja de entrada de correo electrónico como al número de teléfono, ya que ambas cosas son canales de verificación para recuperar la contraseña de usuario raíz. Es importante contar con dos grupos de personas encargadas de administrar estos canales. Un grupo que tenga acceso a la dirección de correo electrónico principal y otro grupo que tenga acceso al número de teléfono principal para recuperar el acceso a la cuenta como usuario raíz.

Proteja las credenciales de usuario raíz de su cuenta de Organizations

Al pasar a una estrategia de múltiples cuentas con Organizations, cada una de esas Cuentas de AWS tiene sus propias credenciales de usuario raíz que se deben proteger. La cuenta que se utiliza

para crear la organización es la cuenta de administración, y el resto de cuentas de la organización son cuentas miembro.

Proteja las credenciales de usuario raíz de las cuentas miembro

Si utiliza Organizations para administrar varias cuentas, existen dos estrategias que puede adoptar para proteger el acceso del usuario raíz en Organizations.

- Proteja las credenciales de usuario raíz de las cuentas de Organizaciones con MFA.
- No restablezca la contraseña de usuario raíz de sus cuentas, y solo recupere el acceso cuando sea necesario mediante el proceso de restablecimiento de la contraseña. Cuando crea una cuenta miembro en su organización, Organizations crea automáticamente un rol de IAM en la cuenta miembro que permite a la cuenta de administración acceder temporalmente a la cuenta miembro.

Para obtener más información, consulte Acceso a las cuentas miembro de la organización en la Guía del usuario de Organizations.

Establezca controles de seguridad preventivos en Organizations mediante una política de control de servicio (SCP)

Si utiliza Organizations para administrar varias cuentas, puede aplicar una SCP para restringir el acceso al usuario raíz de las cuentas miembro. Impedir todas las acciones relacionadas con el usuario raíz en las cuentas miembro, excepto algunas acciones específicas que lo necesitan, ayuda a evitar el acceso no autorizado. Para obtener más información, consulte Utilice una SCP para restringir lo que puede hacer el usuario raíz en sus cuentas de miembro

Supervise el acceso y el uso

Se recomienda que utilice sus mecanismos de seguimiento actuales para supervisar, alertar e informar sobre el inicio de sesión y el uso de credenciales de usuario raíz, incluyendo alertas que anuncien el inicio de sesión y el uso del usuario raíz. Los siguientes servicios pueden ayudar a garantizar el seguimiento del uso de las credenciales de usuario raíz, así como a realizar controles de seguridad que pueden ayudar a evitar el uso no autorizado.

- Si desea recibir notificaciones sobre actividad de inicio de sesión del usuario raíz en su cuenta, puede servirse de Amazon CloudWatch para crear una regla de Eventos que detecte cuándo se utilizan las credenciales de usuario raíz y active una notificación al administrador de seguridad. Para obtener más información, consulte Monitor and notify on Cuenta de AWS root user activity.

- Si desea configurar notificaciones que le avisen de acciones aprobadas para el usuario raíz, puede servirse de Amazon EventBridge junto con Amazon SNS para escribir una regla de EventBridge que realice un seguimiento del uso del usuario raíz para la acción en cuestión y le notifique mediante un tema de Amazon SNS. Para ver un ejemplo, consulte Send a notification when an Amazon S3 object is created.
- Si ya utiliza GuardDuty como servicio de detección de amenazas, puede ampliar su capacidad para que le notifique cuando se utilicen credenciales de usuario raíz en su cuenta.

Las alertas deben incluir, entre otras cosas, la dirección de correo electrónico del usuario raíz. Establezca procedimientos sobre cómo responder a las alertas para que el personal que reciba una alerta de acceso de usuario raíz sepa cómo comprobar si se espera el acceso de usuario raíz y cómo escalar la cuestión si piensa que se está produciendo un incidente de seguridad. Para ver un ejemplo sobre cómo configurar las alertas, consulte Monitor and notify on Cuenta de AWS root user activity.

Evalúe la conformidad con la MFA del usuario raíz

- AWS Config utiliza reglas para ayudar a aplicar las mejores prácticas para el usuario raíz. Puede utilizar reglas administradas de AWS para exigir que los usuarios raíz tengan habilitada la autenticación multifactor (MFA). AWS Config también puede identificar claves de acceso del usuario raíz.
- Security Hub ofrece una vista integral del estado en cuanto a seguridad en AWS y ayuda a evaluar el entorno de AWS con respecto a prácticas recomendadas y estándares del sector de seguridad, tales como disponer de MFA en el usuario raíz y no tener claves de acceso de usuario raíz. Para obtener más información sobre las reglas disponibles, consulte AWS Identity and Access Management controls en la Guía del usuario de Security Hub.
- Trusted Advisor proporciona un control de seguridad para saber si la MFA no está activada en la cuenta de usuario raíz. Para obtener más información, consulte MFA on Root Account en la Guía del usuario de AWS Support.

Si necesita comunicar un problema de seguridad en su cuenta, consulte Informar acerca de correos electrónicos sospechosos o Informes de vulnerabilidades. Como alternativa, puede ponerse en contacto con AWS para obtener ayuda y orientación adicional.

Casos de uso empresarial para IAM

Un caso de uso empresarial sencillo para IAM puede ayudarle a comprender formas básicas de implementar el servicio para controlar el acceso a AWS que los usuarios tienen. El caso de uso se describe en términos generales, sin los mecanismos de cómo utilizaría la API de IAM para lograr los resultados que desea.

Este caso de uso se centra en dos formas habituales en las que una compañía ficticia, llamada Example Corp, podría utilizar IAM. El primer escenario considera Amazon Elastic Compute Cloud (Amazon EC2). El segundo considera Amazon Simple Storage Service (Amazon S3).

Para obtener más información sobre cómo utilizar IAM con otros servicios de AWS, consulte Servicios de AWS que funcionan con IAM.

Temas

Configuración inicial de Example Corp

Nikki Wolf y Mateo Jackson son los fundadores de Example Corp. Al iniciar la empresa, crean una cuenta de Cuenta de AWS y configuran AWS IAM Identity Center (IAM Identity Center) para crear cuentas administrativas que utilizarán con sus recursos de AWS. Al configurar el acceso a la cuenta para el usuario administrativo, IAM Identity Center crea el rol de IAM correspondiente. Este rol, controlado por IAM Identity Center, se crea en la cuenta de Cuenta de AWS correspondiente y se le adjuntan las políticas especificadas en el conjunto de permisos AdministratorAccess.

Como ahora tienen cuentas de administrador, Nikki y Mateo ya no tienen que usar su usuario raíz para acceder a su cuenta de Cuenta de AWS. Planean utilizar el usuario raíz únicamente para hacer las tareas que solo él puede llevar a cabo. Después de revisar las prácticas recomendadas de seguridad, configuran la autenticación multifactor (MFA) para sus credenciales de usuario raíz y deciden cómo proteger sus credenciales de usuario raíz.

A medida que su empresa crece, contratan empleados para trabajar como desarrolladores, administradores, evaluadores, gestores y administradores de sistemas. Nikki se encarga de las operaciones, mientras que Mateo dirige los equipos de ingeniería. Configuraron un servidor de

dominio de Active Directory para administrar las cuentas de los empleados y gestionar el acceso a los recursos internos de la empresa.

Para que sus empleados tengan acceso a los recursos de AWS, utilizan IAM Identity Center para conectar Active Directory de su empresa a su cuenta de Cuenta de AWS.

Al haber conectado Active Directory con IAM Identity Center, los usuarios, el grupo y la pertenencia a grupos están sincronizados y definidos. -Deben asignar conjuntos de permisos y roles a los distintos grupos para dar a los usuarios el nivel correcto de acceso a los recursos AWS. Utilizan Managed Policies de AWS para funciones de trabajo en la AWS Management Console para crear estos conjuntos de permisos:

- Administrador
- Facturación
- Desarrolladores
- Administradores de red
- Administradores de base de datos
- Administradores de sistemas
- Usuarios de asistencia

A continuación, asignan estos conjuntos de permisos a los roles asignados a sus grupos de Active Directory.

Para obtener una guía paso a paso que describe la configuración inicial de IAM Identity Center, consulte Getting started (Introducción) en la Guía del usuario de AWS IAM Identity Center. Para obtener más información sobre el aprovisionamiento del acceso de usuario de IAM Identity Center, consulte Single sign-on access to AWS accounts (Acceso de inicio de sesión único a cuentas de AWS) en la AWS IAM Identity Center User Guide (Guía del usuario de AWS IAM Identity Center).

Caso de uso para IAM con Amazon EC2

Una compañía como Example Corp suele utilizar IAM para interactuar con servicios como Amazon EC2. Para comprender esta parte del caso de uso, necesita tener unos conocimientos básicos de Amazon EC2. Para obtener más información sobre Amazon EC2, consulte la Guía del usuario de Amazon EC2 para instancias de Linux.

Permisos de Amazon EC2 para los grupos de usuarios

Para proporcionar un control del "perímetro", Nikki asocia una política al grupo de usuarios AllUsers. Esta política deniega cualquier solicitud de AWS de un usuario si la dirección IP de origen se encuentra fuera de la red corporativa de Example Corp.

En Example Corp, los distintos grupos de usuarios requieren permisos diferentes:

- Administradores del sistema - Necesitan permiso para crear y administrar imágenes AMI, instancias, instantáneas, volúmenes, grupos de seguridad, etc. Nikki asocia una política administrada `AmazonEC2FullAccess` de AWS al grupo de usuarios SysAdmins que concede a sus miembros permiso para utilizar todas las acciones de Amazon EC2.
- Desarrolladores - Necesitan la capacidad de trabajar únicamente con instancias. Por lo tanto, Mateo crea y asocia al grupo de usuarios de desarrolladores una política que permite a estos llamar a `DescribeInstances`, `RunInstances`, `StopInstances`, `StartInstances` y `TerminateInstances`.

  > **Note**
  >
  > Amazon EC2 utiliza claves SSH, contraseñas de Windows y grupos de seguridad para controlar quién tiene acceso al sistema operativo de determinadas instancias Amazon EC2. No existe ningún método en el sistema de IAM para permitir o denegar el acceso al sistema operativo de una determinada instancia.

- Asistencia a los usuarios: no deberían llevar a cabo ninguna acción de Amazon EC2 excepto la enumeración de los recursos de Amazon EC2 actualmente disponibles. Por lo tanto, Nikki crea y asocia al grupo de asistencia a los usuarios una política que solo les permite invocar operaciones de API “Describe” de Amazon EC2.

Para obtener ejemplos de cómo podrían parecer estas políticas, consulte Ejemplos de políticas basadas en identidad de IAM y Utilización de AWS Identity and Access Management en la Guía del usuario de Amazon EC2 para instancias de Linux.

Cambio de función de trabajo del usuario

En algún momento, uno de los desarrolladores, Paulo Santos, cambia de función de trabajo y se convierte en administrador. Como gestor, Paulo pasa a formar parte del grupo de usuarios de asistencia para poder abrir casos de asistencia para sus desarrolladores. Mateo traslada a Paulo

desde el grupo de usuarios de desarrolladores al grupo de usuarios de asistencia. Como resultado de este movimiento, su capacidad para interactuar con las instancias de Amazon EC2 es limitada. No puede lanzar ni iniciar instancias. Tampoco puede detener o terminar las instancias existentes, aunque fue el usuario que lanzó o inició la instancia. Solo puede enumerar las instancias que los usuarios de Example Corp han lanzado.

Caso de uso para IAM con Amazon S3

Las compañías como Example Corp normalmente también utilizarían IAM con Amazon S3. John ha creado un bucket de Amazon S3 para la empresa denominado aws-s3-bucket.

Creación de otros usuarios y grupos de usuarios

Como empleados, Zhang Wei y Mary Major deben poder crear sus propios datos en el bucket de la empresa. También necesitan leer y escribir datos compartidos en los que trabajan todos los desarrolladores. Para ello, Mateo organiza de forma lógica los datos de aws-s3-bucket con un esquema de prefijo de clave de Amazon S3 como el que se muestra en la siguiente figura.

```
/aws-s3-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo divide el `/aws-s3-bucket` en un conjunto de directorios principales para cada empleado y un área compartida para los grupos de desarrolladores y administradores.

Ahora, Mateo crea un conjunto de políticas para asignar permisos a los usuarios y grupos de usuarios:

- Acceso al directorio principal de Zhang: Mateo asocia una política a Wei que le permite leer, escribir y enumerar cualquier objeto con el prefijo de clave `/aws-s3-bucket/home/zhang/` de Amazon S3
- Acceso al directorio principal de Major: Mateo asocia una política a Mary que le permite leer, escribir y enumerar cualquier objeto con el prefijo de clave `/aws-s3-bucket/home/major/` de Amazon S3

- Acceso al directorio compartido para el grupo de usuarios de desarrolladores: Mateo asocia una política al grupo de usuarios que permite a los desarrolladores leer, escribir y enumerar cualquier objeto de `/aws-s3-bucket/share/developers/`
- Acceso al directorio compartido para el grupo de usuarios de administradores: Mateo asocia una política al grupo de usuarios que permite a los administradores leer, escribir y enumerar objetos de `/aws-s3-bucket/share/managers/`

> **Note**
>
> Amazon S3 no concede automáticamente a un usuario que crea un bucket u objeto permiso para realizar otras acciones en dicho bucket u objeto. Por lo tanto, en sus políticas de IAM debe conceder de forma explícita a los usuarios permiso para utilizar los recursos de Amazon S3 que creen.

Para obtener ejemplos de cómo pueden lucir estas políticas, consulte Control de acceso en la Guía del usuario de Amazon Simple Storage Service. Para obtener información sobre cómo las políticas se evalúan en tiempo de ejecución, consulte Lógica de evaluación de políticas.

Cambio de función de trabajo del usuario

En algún momento, uno de los desarrolladores, Zhang Wei, cambia de función de trabajo y se convierte en administrador. Supongamos que ya no necesita acceso a los documentos del directorio `share/developers`. Mateo, como administrador, traslada a Wei al grupo de usuarios `Managers` desde el grupo de usuarios `Developers`. Con tan solo una sencilla reasignación, Wei consigue automáticamente todos los permisos concedidos al grupo de usuarios `Managers`, pero ya no puede obtener acceso a los datos del directorio `share/developers`.

Integración con un negocio de terceros

Las organizaciones suelen trabajar con compañías asociadas, asesores y contratistas. Example.Corp tiene un socio denominado Widget Company y una empleada de esta empresa llamada Shirley Rodriguez necesita colocar datos en un bucket para que los utilice Example Corp. Nikki crea un grupo de usuarios denominado WidgetCo y un usuario denominado `Shirley` y agrega a Shirley al grupo de usuarios WidgetCo. Nikki también crea un bucket especial llamado aws-s3-bucket1 para que Shirley lo use.

Nikki actualiza las políticas existentes o agrega otras nuevas para integrar al socio Widget Company. Por ejemplo, Nikki puede crear una política nueva que deniega a los miembros del grupo de usuarios WidgetCo la posibilidad de utilizar acciones que no sean de escritura. Esta política solo sería necesaria si hubiera una política amplia que ofreciera a todos los usuarios acceso a un amplio conjunto de acciones de Amazon S3.

Tutoriales de IAM

Los siguientes tutoriales presentan procedimientos integrales completos para tareas comunes de AWS Identity and Access Management (IAM). Están pensados para un entorno de laboratorio, con los nombres de empresa ficticios, los nombres de usuario y así sucesivamente. Su finalidad es proporcionar orientación general. No deben utilizarse directamente en un entorno de producción sin antes realizar una revisión y adaptación exhaustivas para satisfacer las necesidades únicas del entorno de la organización.

Tutoriales

- Tutorial de IAM: conceder acceso a la consola de facturación
- Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM
- Tutorial de IAM: crear y asociar su primera política administrada por el cliente
- Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas
- Tutorial de IAM: permitir a los usuarios administrar sus credenciales y configuración de MFA

Tutorial de IAM: conceder acceso a la consola de facturación

El propietario de la Cuenta de AWS (Usuario raíz de la cuenta de AWS) puede conceder a los usuarios y roles de IAM acceso a los datos de AWS Billing and Cost Management de su Cuenta de AWS. Las instrucciones que aparecen en este tutorial le ayudarán a configurar un escenario probado previamente. Esta situación le ayudará a obtener experiencia práctica en la configuración de permisos de facturación sin preocuparse si afecta a su cuenta de producción de AWS principal.

Requisitos previos

Lleve a cabo los siguientes procedimientos de preparación antes de realizar los pasos de este tutorial:

- Cree una Cuenta de AWS de prueba.
- Inicie sesión en la Cuenta de AWS como el usuario raíz.
- Registre el número de su cuenta de Cuenta de AWS de prueba para poder usarla en el tutorial. En este tutorial utilizamos el número de cuenta de ejemplo 111122223333. Siempre que un paso utilice ese número de cuenta, sustitúyalo por el número de cuenta de prueba.

Paso 1: Activar el acceso IAM a la información de facturación en la cuenta de prueba de AWS

En este escenario, inicie sesión en la prueba de Cuenta de AWS como usuario raíz para conceder acceso de IAM a la información de facturación. Cuando concedes acceso de IAM a la información de facturación, los usuarios y roles de IAM pueden acceder a la consola de AWS Billing and Cost Management. Esta configuración no concede a los usuarios y roles de IAM los permisos necesarios para estas páginas de la consola, sino que habilita el acceso a los usuarios o roles de IAM que cuentan con las políticas de IAM requeridas. Si las políticas ya están asociadas a los usuarios o roles de IAM, pero esta configuración no está habilitada, los permisos otorgados por esas políticas no estarán en vigor.

> **Note**
>
> Cuentas de AWS creadas utilizando el AWS Organizations, tienen acceso de IAM a la información de facturación activada de forma predeterminada.

Paso 2: Crear usuarios y grupos de prueba

En este escenario, concede a los usuarios de IAM acceso a la consola de facturación y creas dos usuarios:

- Pat Candella

  Pat es miembro del departamento de finanzas y trabaja con facturación y pagos. Pat requiere acceso completo a la información de facturación que figura en su Cuenta de AWS.

- Terry Whitlock

  Terry forma parte de su departamento de soporte de TI. La mayoría de las veces, Terry no necesita acceso a la consola de facturación, pero a veces lo necesita para responder a las preguntas de los empleados del departamento de finanzas.

Paso 3: Crear un rol que conceda acceso a la consola de AWS Billing

Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. Un rol de IAM es similar a un usuario de IAM, ya que se trata de una identidad de AWS con políticas de permisos que determinan lo que la identidad puede hacer y lo que no puede hacer en AWS. No obstante, en lugar de asociarse exclusivamente a una persona, la intención es que cualquier usuario pueda asumir un rol que necesite. Además, un rol no tiene asociadas credenciales estándar a largo plazo, como una contraseña o claves de acceso. En su

lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol. Puede utilizar roles para delegar el acceso a usuarios, aplicaciones o servicios que normalmente no tendrían acceso a los recursos de AWS. En este escenario, crea un rol que Terry Whitlock pueda asumir para acceder a la consola de facturación.

Paso 4: Probar el acceso a la consola de facturación

Una vez que haya agotado las tareas esenciales, estará listo para probar la política. Las pruebas garantizan que la política funciona tal como usted desea. Al probar el acceso de cada usuario, puede comparar las experiencias de los usuarios.

Requisitos previos

Lleve a cabo los siguientes procedimientos de preparación antes de realizar los pasos de este tutorial:

- Cree una Cuenta de AWS de prueba.
- Inicie sesión en la Cuenta de AWS como el usuario raíz.
- Registre el número de su cuenta de Cuenta de AWS de prueba para poder usarla en el tutorial. En este tutorial utilizamos el número de cuenta de ejemplo 111122223333. Siempre que un paso utilice ese número de cuenta, sustitúyalo por el número de cuenta de prueba.

Paso 1: Activar el acceso IAM a la información de facturación en la cuenta de prueba de AWS

En este escenario, inicie sesión en la prueba de Cuenta de AWS como usuario raíz para conceder acceso de IAM a la información de facturación. Cuando concede acceso de IAM a la información de facturación, los usuarios y roles de IAM pueden acceder a la consola de AWS Billing and Cost Management. Esta configuración no concede a los usuarios y roles de IAM los permisos necesarios para estas páginas de la consola, sino que únicamente habilita el acceso a los usuarios o roles de IAM que cuentan con las políticas de IAM requeridas.

> **Note**
>
> Cuentas de AWS creadas utilizando el AWS Organizations, tienen acceso de IAM a la información de facturación activada de forma predeterminada.

Para activar el acceso del usuario de IAM y del rol a la consola de Billing and Cost Management

1. Inicie sesión en la AWS Management Console con sus credenciales de cuenta raíz (la dirección de email y la contraseña que utilizó para crear su AWS).
2. En la barra de navegación, seleccione el nombre de su cuenta y, a continuación, elija Cuenta.
3. Desplácese hacia abajo en la página hasta encontrar la sección Acceso de usuarios y roles de IAM a la información de facturación y, a continuación, seleccione Editar.
4. Seleccione la casilla de verificación Activate IAM Access (Activar acceso de IAM) para activar el acceso a las páginas de la consola de Billing and Cost Management.
5. Elija Update (Actualizar).

   La página muestra el mensaje de que el el acceso del usuario/rol de IAM a la información de facturación está activado.

En el siguiente paso de este tutorial adjuntará políticas de IAM para conceder o denegar acceso para características de facturación específicas.

Paso 2: Crear usuarios y grupos de prueba

Su cuenta de AWS de prueba no tiene ninguna identidad definida, excepto la del usuario raíz. Para proporcionar acceso a la información de facturación, creamos identidades adicionales a las que podemos conceder permiso para acceder a la información de facturación.

Crear usuarios y grupos de prueba

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   > **Note**
   >
   > Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En el panel de navegación, seleccione Usuarios y luego, elija Agregar usuarios.

> **Note**
>
> Si tiene habilitado el IAM Identity Center, la AWS Management Console muestra un recordatorio de que es mejor administrar el acceso de los usuarios en IAM Identity Center. En este tutorial, los usuarios de IAM que creamos deben aprender a proporcionar acceso a la información de facturación. Si ha creado usuarios en IAM Identity Center, asigne el conjunto de permisos de facturación a esos usuarios o grupos que utilicen IAM Identity Center en lugar de IAM.

3. En User name (Nombre de usuario), escriba **`pcandella`**. Los nombres no pueden contener espacios.
4. Seleccione la casilla de selección junto a Proporcionar al usuario acceso a la AWS Management Console: optional y luego, elija Quiero crear un usuario de IAM.
5. En Contraseña de la consola, seleccione Contraseña generada de manera automática.
6. Desmarque la casilla de selección junto a El usuario debe crear una contraseña nueva la próxima vez que inicie sesión (recomendado) y luego seleccione Siguiente. Como este usuario de IAM es para realizar pruebas, descargaremos la contraseña para utilizarla durante el procedimiento de verificación.
7. En la página Establecer permisos, en Opciones de permisos, seleccione Agregar usuario al grupo. Luego, en Grupos de usuarios, seleccione Crear grupo.
8. En la página Crear grupo de usuarios, en Nombre del grupo de usuarios, ingrese **`BillingGroup`**. A continuación, en Políticas de permisos, seleccione la política de funciones de trabajo administradas por AWS Billing.
9. Seleccione Crear grupo de usuarios para volver a la página Establecer permisos.
10. En Grupos de usuarios, seleccione la casilla de selección del nombre del **`BillingGroup`** que creó anteriormente.
11. Seleccione Siguiente para dirigirse a la página Revisar y crear.
12. En la página Revisar y crear, revise la lista de suscripciones a grupos de usuarios para el usuario nuevo. Cuando esté listo para continuar, seleccione Crear usuario.
13. En la página Recuperar contraseña, seleccione Descargar archivo .csv para guardar un archivo .csv con la información de inicio de sesión del usuario (URL de conexión, nombre del usuario y contraseña).

Guarde este archivo para usarlo como referencia cuando inicie sesión en AWS como este usuario de IAM

14. Seleccione Volver a la lista de usuarios
15. Repita este procedimiento con las siguientes modificaciones para crear el usuario para Terry Whitlock y un grupo para los usuarios de soporte.

    a. En el paso 3, en Nombre de usuario, ingrese **`twhitlock`**.

    b. En el paso 8, en Nombre de grupo de usuarios, ingrese **`SupportGroup`**. A continuación, en Políticas de permisos, seleccione la política de funciones de trabajo administradas por AWS SupportUser.

Puede revisar los nuevos usuarios, grupos y roles de IAM en las listas de la consola. Para cada elemento que haya creado, puede seleccionar el nombre para ver sus detalles. Al ver los detalles del usuario, la consola muestra Facturación en Políticas de permisos para **`pcandella`** y SupportUser en Políticas de permisos para **`twhitlock`**.

Para obtener más información sobre el uso de políticas con objeto de conceder acceso a los usuarios de IAM a las características de AWS Billing and Cost Management, consulte Uso de políticas basadas en identidad (políticas de IAM) para AWS Billing en la Guía del usuario de AWS Billing.

Paso 3: Crear un rol que conceda acceso a la consola de AWS Billing

Puede usar un rol para conceder a los usuarios de IAM acceso a la consola de facturación. Los roles proporcionan credenciales temporales que los usuarios pueden asumir cuando las necesiten. En este tutorial, el usuario **`twhitlock`** debe poder acceder a la información de facturación cuando una solicitud de soporte del departamento de finanzas requiera que investigue un problema.

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   > **Note**
   >
   > Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda

para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En el panel de navegación, seleccione Usuario y luego seleccione el usuario **`twhitlock`** para ver los detalles de usuario. Copie el ARN del usuario **`twhitlock`** al portapapeles.
3. En el panel de navegación, seleccione Roles y después Crear rol.
4. En la página Seleccione una entidad de confianza, seleccione Política de confianza personalizada y, a continuación, en Editar declaración, complete los siguientes elementos:
   - Agregar acciones para STS: compruebe que esté seleccionada AssumeRole.
   - Agregar una entidad principal, seleccione Agregar para mostrar el cuadro de diálogo Agregar entidad principal. En Tipo de entidad principal, seleccione Usuarios de IAM y, en ARN, pegue el ARN del usuario twhitlock que copió en el portapapeles en el paso 16. A continuación, seleccione Agregar entidad principal.
5. Selecciona Siguiente para ir a la página Agregar permisos.
6. En Políticas de permisos, en la casilla de filtro, ingrese **`Billing`** y seleccione la política de funciones de trabajo administradas por AWS Facturación.
7. Seleccione Siguiente dos veces para ir a la página Asignar nombre, revisar y crear. En Nombre de rol, ingrese **`TempBillingAccess`** y luego seleccione Crear rol.

   Se le notificará que se ha creado el rol. Vea el rol para que se muestren los detalles del rol. En la sección Resumen, tome nota de la siguiente información:

   - La duración máxima de la sesión es de 1 hora de forma predeterminada. Transcurrido ese tiempo, el usuario que asumió el rol vuelve a los permisos de su cuenta base. Si el usuario quiere seguir usando los permisos de rol, debe volver a cambiar de rol. Puede editar el rol para aumentar la duración máxima. La duración más larga posible de la sesión es de 12 horas.
   - Enlace para cambiar de rol en la consola. Puede copiar el enlace para proporcionárselo directamente a los usuarios que agregue como directores en la política de confianza. Puede ver y editar la política de confianza desde la pestaña Relaciones de confianza.

Paso 4: Probar el acceso a la consola de facturación

Le recomendamos que pruebe el acceso iniciando sesión como los usuarios de prueba para saber lo que los usuarios ven y cómo lo ven. Utilice los siguientes pasos para iniciar sesión con las cuentas de prueba y ver la diferencia entre los derechos de acceso.

Para probar el acceso de facturación iniciando sesión en ambas cuentas de usuario de prueba

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

   > **Note**
   >
   > Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

2. Inicie sesión con cada cuenta siguiendo los pasos indicados a continuación para que pueda comparar las diferentes experiencias de usuario.

   Acceso completo

   a. Inicie sesión en la Cuenta de AWS como el usuario **`pcandella`**.
   b. En la barra de navegación, seleccione pcandella@111122223333 y, a continuación, elija Panel de facturación.
   c. Navegue por las páginas y elija los diferentes botones para asegurarse de que dispone de permisos de modificación completos.

   Sin acceso

   a. Inicie sesión en la Cuenta de AWS como el usuario **`twhitlock`**.
   b. En la barra de navegación, seleccione twhitlock@111122223333 y, a continuación, elija Panel de facturación.
   c. Aparecerá un mensaje con el mensaje Necesita permisos. No hay datos de facturación visibles.

Cambiar de rol para elevar el acceso

a. Inicie sesión en la Cuenta de AWS como el usuario **`twhitlock`**.

b. En la barra de navegación, seleccione twhitlock@111122223333 y, a continuación, elija Cambiar rol.

   Se abrirá la página Cambiar rol. Completa la información tal como se indica a continuación:

   - Cuenta: 111122223333
   - Rol: **`TempBillingAccess`**

   Seleccione Cambiar rol

   Como alternativa, puede utilizar la URL proporcionada en Enlace para cambiar de rol en la consola para abrir la página Cambiar rol.

c. La consola muestra el Panel de control de AWS Billing y la barra de navegación muestra TempBillingAccess@111122223333.

Resumen

Ya ha completado los pasos necesarios para proporcionar a los usuarios de IAM acceso a la consola de AWS Billing. Como resultado, ha visto de primera mano cómo es la experiencia de la consola de facturación para los usuarios. Ahora puede proceder a implementar esta lógica en su entorno de producción cuando lo desee.

Recursos relacionados

Para obtener información relacionada en la Guía del usuario de AWS Billing, consulte los siguientes recursos:

- Activación del acceso a la consola de AWS Billing
- Ejemplos de políticas de AWS Billing
- Uso de políticas basadas en identidad (políticas de IAM) en Facturación de AWS
- Migración del control de acceso para AWS Billing

Para obtener información relacionada en la Guía del usuario de IAM, consulte los siguientes recursos:

- Políticas administradas y políticas insertadas
- Controlar el acceso de los usuarios de IAM a la AWS Management Console
- Asociación de una política a un grupo de usuarios de IAM

Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM

En este tutorial, se enseña a utilizar un rol para delegar el acceso a recursos en Cuentas de AWS diferentes y que son de su propiedad llamadas producción y desarrollo. Los recursos de una cuenta los comparte con usuarios de otra cuenta. Al configurar de esta forma un acceso entre cuentas, no deberá crear usuarios de IAM individuales en cada cuenta. Además, los usuarios no tendrán que cerrar sesión en una cuenta e iniciar sesión en otra cuenta para obtener acceso a los recursos en Cuentas de AWS diferentes. Cuando haya configurado el rol, aprenderá a utilizarlo en la AWS Management Console, la AWS CLI y la API.

> **Note**
>
> Los roles de IAM y las políticas basadas en recursos delegan el acceso entre cuentas solo dentro de una única partición. Suponga, por ejemplo, que tiene una cuenta en EE. UU. Oeste (Norte de California) en la partición estándar `aws`. También tiene una cuenta en China (Pekín) en la partición `aws-cn`. No puede utilizar una política de Amazon S3 basada en recursos en su cuenta en China (Pekín) para permitir el acceso a los usuarios de su cuenta `aws` estándar.

En este tutorial, la cuenta de producción es donde se administran las aplicaciones en directo. La cuenta de desarrollo es un entorno aislado de pruebas en el que los desarrolladores y los evaluadores pueden probar libremente las aplicaciones. En ambas cuentas, la información de las aplicaciones se almacena en buckets de Amazon S3. Usted administra los usuarios de IAM en la cuenta de desarrollo, donde tiene dos grupos de usuarios de IAM: desarrolladores y evaluadores. En ambos grupos de usuarios, los usuarios tienen permiso para trabajar en la cuenta Desarrollo y obtener acceso a los recursos de esta. De vez en cuando, un desarrollador debe actualizar las

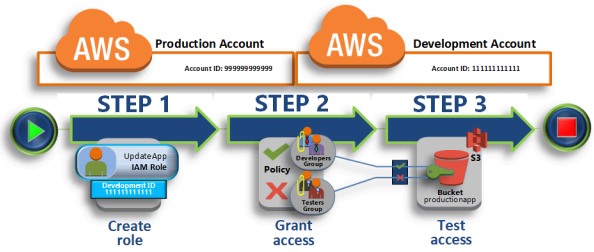
aplicaciones en funcionamiento de la cuenta de producción. Los desarrolladores almacenan dichas aplicaciones en un bucket de Amazon S3 llamado `productionapp`.

Al final de este tutorial, dispone de lo siguiente:

- Los usuarios de la cuenta de desarrollo (la cuenta de confianza) que tienen permiso para asumir un rol específico en la cuenta de producción.
- Un rol de la cuenta de producción (la cuenta que confía) que tiene permiso para obtener acceso a un bucket de Amazon S3 específico.
- Un bucket `productionapp` en la cuenta de producción.

Los desarrolladores pueden utilizar el rol en la AWS Management Console para obtener acceso al bucket `productionapp` de la cuenta de producción. También pueden obtener acceso al bucket mediante llamadas a la API autenticadas con credenciales temporales que el rol proporciona. Si un evaluador realiza intentos similares, obtendrá un error.

Este flujo de trabajo incluye tres pasos básicos:



Crear un rol en la cuenta de producción

En primer lugar, la AWS Management Console se utiliza para establecer una relación de confianza entre la cuenta de producción (número de ID 999999999999) y la cuenta de desarrollo (número de ID 111111111111). Comience creando un rol de IAM llamado UpdateApp. Cuando cree el rol, defina la cuenta de desarrollo como una entidad de confianza y especifique una política de permisos que permita a los usuarios de confianza actualizar el bucket `productionapp`.

Conceder acceso al rol

En este paso del tutorial, debe modificar la política de grupo de usuarios de IAM para que se deniegue el acceso de los evaluadores al rol `UpdateApp`. Como en este caso los evaluadores tienen acceso de usuario avanzado, debe denegar explícitamente la posibilidad de utilizar el rol.

[Probar el acceso alternando roles]

Por último, como desarrollador, utilice el rol `UpdateApp` para actualizar el bucket `productionapp` en la cuenta de producción. Puede ver cómo obtener acceso al rol mediante la consola de AWS, la AWS CLI y la API.

Requisitos previos

En este tutorial se presupone que los elementos siguientes ya existen:

- Dos Cuentas de AWS independientes que puede utilizar, una para representar la cuenta de desarrollo y la otra para representar la cuenta de producción.
- Los usuarios y grupos de usuarios de la cuenta de desarrollo creados y configurados como se indica a continuación:

Usuario	Grupos de usuarios	Permisos
David	Desarrolladores	Ambos usuarios pueden iniciar sesión y utilizar la AWS Management Console en la cuenta de desarrollo.
Jane	Evaluadores	

- No necesita tener usuarios o grupos de usuarios creados en la cuenta de producción.
- Un bucket de Amazon S3 creado en la cuenta de producción. Puede denominarlo `ProductionApp` en este tutorial, pero debido a que los nombres de bucket de S3 deben ser únicos globalmente, deberá utilizar un bucket con otro nombre.

Crear un rol en la cuenta de producción

Puede permitir que los usuarios de una Cuenta de AWS obtengan acceso a los recursos de otra Cuenta de AWS. Para ello, cree un rol que defina quién puede obtener acceso a ella y qué permisos concede a los usuarios que cambian a ella.

En este paso del tutorial, creará el rol en la cuenta de producción y especificará la cuenta de desarrollo como entidad de confianza. También limitará los permisos del rol a un acceso de solo lectura y escritura para el bucket `productionapp`. Todos los que tengan permiso para utilizar el rol podrán leer y escribir en el bucket `productionapp`.

Para poder crear un rol, necesitará el ID de cuenta de la cuenta de Development (Desarrollo) de Cuenta de AWS. Cada Cuenta de AWS tiene un identificador de ID de cuenta exclusivo asignado.

Para obtener el ID de Cuenta de AWS de Desarrollo

1. Inicie sesión en la AWS Management Console como administrador de la cuenta de desarrollo y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En la barra de navegación, elija Support (Asistencia) y, a continuación, Support Center (Centro de asistencia). El número de cuenta (ID) actual de 12 dígitos que ha iniciado sesión aparece en el panel de navegación Centro de asistencia. En este escenario, puede utilizar el ID de cuenta 111111111111 para la cuenta de desarrollo. Sin embargo, debe utilizar un ID de cuenta válido si utiliza el escenario en su entorno de prueba.

Para crear un rol en la cuenta de producción que pueda utilizarse en la cuenta Development (Desarrollo)

1. Inicie sesión en la AWS Management Console como administrador de la cuenta de producción y abra la consola de IAM.
2. Antes de crear el rol, prepare la política administrada que define los permisos para los requisitos del rol. Más tarde, en otro paso, la asociará al rol.

   Debe configurar el acceso de lectura y escritura al bucket `productionapp`. Aunque AWS proporciona algunas políticas administradas por Amazon S3, ninguna proporciona acceso de lectura y escritura a un solo bucket de Amazon S3. En su lugar, puede crear su propia política.

   En el panel de navegación, elija Policies (Políticas) y, a continuación, seleccione Create policy (Crear política).
3. Seleccione la pestaña JSON y copie el texto del siguiente documento de política JSON. Pegue este texto en el cuadro de texto JSON y reemplace el ARN del recurso (`arn:aws:s3:::productionapp`) por el ARN real de su bucket de Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
```

```
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

La acción `ListAllMyBuckets` concede permiso para enumerar todos los buckets propiedad del remitente autenticado de la solicitud. El permiso `ListBucket` permite a los usuarios ver objetos en el bucket `productionapp`. Los permisos `GetObject`, `PutObject`, `DeleteObject` permiten a los usuarios ver, actualizar y eliminar contenido del bucket `productionapp`.

4. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

5. En la página Revisar y crear, escriba **`read-write-app-bucket`** como nombre de la política. Revise los permisos concedidos por la política y, a continuación, seleccione Crear política para guardar su trabajo.

   La nueva política aparece en la lista de políticas administradas.

6. En el panel de navegación, seleccione Roles y, a continuación, seleccione Create role.

7. Elija el tipo de rol Una Cuenta de AWS.
8. En Account ID (ID de cuenta), escriba el ID de la cuenta de desarrollo.

   En este tutorial, se utiliza el ID de cuenta de ejemplo **111111111111** para la cuenta Development (Desarrollo). Debe utilizar un ID de cuenta válido. Si utiliza un ID de cuenta que no es válido, como **111111111111**, IAM no le permitirá crear el nuevo rol.

   Por ahora, no necesita exigir un ID externo ni solicitar a los usuarios que utilicen autenticación multifactor (MFA) para asumir el rol. Deje estas opciones desmarcadas. Para obtener más información, consulte Uso de autenticación multifactor (MFA) en AWS.
9. Seleccione Next: Permissions (Siguiente: Permisos) para establecer los permisos asociados al rol.
10. Seleccione la casilla situada junto a la política que ha creado anteriormente.

    > **Sugerencia**
    >
    > En Filter (Filtro), elija Customer managed (Administrado por el cliente) para filtrar la lista e incluir únicamente las políticas que creó. Esto ocultará las políticas creadas por AWS y facilitará en gran medida encontrar la política que necesita.

    A continuación, haga clic en Next.
11. De manera opcional, agregue metadatos al rol al asociar etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
12. (Opcional) En Description (Descripción), ingrese una descripción para el nuevo rol.
13. Después de revisar el rol, seleccione Create Role (Crear rol).

    El rol `UpdateApp` se muestra en la lista de roles.

Ahora debe obtener el nombre de recurso de Amazon (ARN) del rol, un identificador exclusivo del rol. Si modifica la política de grupo de usuarios de los desarrolladores y los evaluadores, debe especificar el ARN del rol para conceder o denegar permisos.

Para obtener el ARN de UpdateApp

1. En el panel de navegación de la consola de IAM, elija Roles (Roles).
2. En la lista de roles, elija el rol `UpdateApp`.
3. En la sección Summary (Resumen) del panel de detalles, copie el valor de Role ARN (ARN del rol).

   La cuenta de producción tiene el ID de cuenta 999999999999, por lo que el ARN del rol es `arn:aws:iam::999999999999:role/UpdateApp`. Asegúrese de dar el ID de Cuenta de AWS real para la cuenta de producción.

En este momento, ha establecido una relación de confianza entre la cuenta de producción y la cuenta de desarrollo . Para ello, creó un rol en la cuenta de producción que identifica la cuenta de desarrollo como entidad principal de confianza. También ha definido qué pueden hacer los usuarios que cambian al rol `UpdateApp`.

A continuación, va a modificar los permisos de los grupos de usuarios.

Conceder acceso al rol

En este punto, los miembros de los grupos de usuarios de evaluadores y desarrolladores tienen permisos que les permiten probar libremente aplicaciones en la cuenta de desarrollo. Estos son los pasos necesarios para agregar permisos que permitan cambiar al rol.

Para modificar el grupo de usuarios de Desarrolladores para que puedan cambiar al rol UpdateApp

1. Inicie sesión como administrador en la cuenta de desarrollo y abra la consola de IAM.
2. Elija Grupos de usuarios y, a continuación, Desarrolladores.
3. Elija la pestaña de Permisos, elija Agregar permisos y luego Crear política insertada.
4. Seleccione la pestaña JSON.
5. Añada la siguiente declaración de política para permitir la acción `AssumeRole` en el rol `UpdateApp` en la cuenta Producción. Asegúrese de que cambia *`PRODUCTION-ACCOUNT-ID`* en el elemento `Resource` por el ID de Cuenta de AWS real de la cuenta de producción.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
```

```
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
  }
}
```

El efecto `Allow` permite explícitamente al grupo Desarrolladores obtener acceso al rol `UpdateApp` en la cuenta Producción. Todos los desarrolladores que intenten obtener acceso al rol lo consiguen.

6. Elija Review policy (Revisar política).
7. Escriba un Nombre; por ejemplo, **`allow-assume-S3-role-in-production`**.
8. Elija Create Policy (Crear política).

En la mayoría de los entornos, quizás no sea necesario el siguiente procedimiento. Si, por el contrario, usa permisos de PowerUserAccess, es posible que algunos grupos ya puedan cambiar de rol. En el siguiente procedimiento se muestra cómo agregar un permiso `"Deny"` al grupo Evaluadores para garantizar que no puedan asumir el rol. Si este procedimiento no es necesario en su entorno, le recomendamos que no lo agregue. Los permisos `"Deny"` hacen que el panorama general de los permisos sea más difícil de administrar y de entender. Utilice los permisos `"Deny"` solo cuando no tenga una opción mejor.

Para modificar el grupo de usuarios de evaluadores para denegarle el permiso de asumir el rol **`UpdateApp`**

1. Elija Grupos de usuarios y, a continuación, Probadores.
2. Elija la pestaña de Permisos, elija Agregar permisos y luego Crear política insertada.
3. Seleccione la pestaña JSON.
4. Añada la siguiente declaración de política para denegar la acción `AssumeRole` en el rol `UpdateApp`. Asegúrese de que cambia *`PRODUCTION-ACCOUNT-ID`* en el elemento `Resource` por el ID de Cuenta de AWS real de la cuenta de producción.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
  }
```

```
}
```

El efecto `Deny` deniega explícitamente al grupo Evaluadores obtener acceso al rol `UpdateApp` en la cuenta Producción. Todos los evaluadores que intenten obtener acceso al rol reciben un mensaje de acceso denegado.

5. Elija Review policy (Revisar política).
6. Escriba un Nombre; por ejemplo, **`deny-assume-S3-role-in-production`**.
7. Elija Create Policy (Crear política).

Ahora, el grupo de usuarios de Desarrolladores tiene permisos para utilizar el rol `UpdateApp` en la cuenta de producción. El grupo de usuarios Evaluadores no podrá utilizar el rol `UpdateApp`.

A continuación, puede ver cómo David, un desarrollador, puede obtener acceso al bucket `productionapp` en la cuenta de producción. David puede obtener acceso al bucket desde la AWS Management Console, la AWS CLI, o la API de AWS.

Probar el acceso alternando roles

Después de completar los dos primeros pasos de este tutorial, tiene un rol que concede acceso a un recurso en la cuenta de producción. También tiene un grupo de usuarios en la cuenta de desarrollo cuyos usuarios tienen permiso para usar dicho rol. En este paso se explica cómo probar el cambio a este rol desde la AWS Management Console, la AWS CLI, y la API de AWS.

> **Important**
>
> Puede cambiar a un rol únicamente después de iniciar sesión como usuario de IAM o usuario federado. Además, si lanza una instancia de Amazon EC2 para ejecutar una aplicación, la aplicación puede asumir un rol mediante su perfil de instancias. No puede cambiar a un rol si inicia sesión como el Usuario raíz de la cuenta de AWS.

Cambio de roles (consola)

Si David necesita trabajar en el entorno de producción en la AWS Management Console, puede hacerlo a través de Switch Role (Cambiar rol). Indica el ID de cuenta o el alias, y el nombre del rol, y sus permisos cambian inmediatamente a los que están permitidos por el rol. A continuación, puede utilizar la consola para trabajar con el bucket `productionapp`, pero no puede trabajar con ningún

otro recurso en producción. Aunque David utiliza el rol, tampoco puede hacer uso de sus privilegios de usuario avanzado en la cuenta de desarrollo. Esto se debe a que no puede haber más de un conjunto de permisos en vigor a la vez.

> **Important**
>
> El cambio de roles mediante la AWS Management Console solo funciona con cuentas que no requieran un `ExternalId`. Por ejemplo, suponga que concede acceso a su cuenta a un tercero y requiere un `ExternalId` en un elemento `Condition` de su política de permisos. En ese caso, el tercero puede obtener acceso a su cuenta solo a través de la API de AWS o una herramienta de línea de comandos. El tercero no puede utilizar la consola, ya que no puede proporcionar un valor para `ExternalId`. Para obtener más información sobre este caso, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros y Cómo habilitar el acceso entre cuentas a la AWS Management Console en el Blog de seguridad de AWS.

Gracias a IAM, David puede entrar en la página Switch Role (Cambiar rol) de dos formas:

- David recibe un enlace de su administrador que apunta a una configuración de Switch Role (Cambiar rol) predefinida. El enlace se proporciona al administrador en la página final del asistente Create role (Crear rol) y en la página Role Summary (Resumen del rol) a un rol con permisos entre cuentas. Si David elige este enlace, obtendrá acceso a la página Switch Role (Cambiar rol) con los campos Account ID (ID de cuenta) y Role name (Nombre del rol) ya completados. David solo tiene que elegir Switch Roles (Cambiar roles).
- El administrador no envía el enlace por correo electrónico, sino que envía los valores de Account ID (ID de cuenta) y Role Name (Nombre del rol). Para cambiar de roles, David tiene que ingresar manualmente los valores. Esto se muestra en el procedimiento siguiente.

Para asumir un rol

1. David inicia sesión en la AWS Management Console con su usuario normal que se encuentra en el grupo de usuarios de desarrollo.
2. Ellos eligen el vínculo que el administrador les envía. Esto lleva a David a la página Switch Role (Cambiar rol) con la información del ID de cuenta o alias y el nombre de rol ya completados.

   —o bien—

David elige su nombre, en el menú Identity (Identidad), en la barra de navegación y, a continuación, elige Switch Role (Cambiar rol).

Si es la primera vez que David intenta obtener acceso a la página Switch Role (Cambiar rol) de esta manera, primero entrará a la página Switch Role (Cambiar rol) predeterminada. Esta página proporciona información adicional acerca de cómo el cambio de rol puede permitir a los usuarios para que administren recursos entre Cuentas de AWS. David debe hacer clic en Switch Role (Cambiar rol) en esta página para completar el resto de este procedimiento.

3. A continuación, para poder obtener acceso al rol, David debe escribir manualmente el número de ID de la cuenta Producción (999999999999) y el nombre del rol (`UpdateApp`).

   Además, David quiere monitorear qué roles y permisos asociados están activos actualmente en IAM. Para realizar un seguimiento de esta información, escribe `PRODUCTION` en el cuadro de texto Nombre de visualización, selecciona la opción de color rojo y, a continuación, elige Cambiar rol.
4. Ahora puede utilizar la consola de Amazon S3 para trabajar con el bucket de Amazon S3 o cualquier otro recurso sobre el que el rol `UpdateApp` tenga permisos.
5. Al terminar, David puede volver a sus permisos originales. Para ello, eligen el nombre de visualización del rol PRODUCTION (PRODUCCIÓN) en la barra de navegación y, a continuación, eligen Back to David @111111111111 (Devolver a David @111111111111).
6. La siguiente vez que David quiera cambiar de rol y elija el menú Identity (Identidad) en la barra de navegación, verá que la entrada PRODUCTION (Producción) sigue estando ahí. Solo tiene que elegir esa entrada para cambiar de rol inmediatamente sin tener que volver a escribir el ID de cuenta y el nombre de rol.

Cambio de roles (AWS CLI)

Si David necesita trabajar en la línea de comandos en el entorno de producción, puede hacerlo mediante la AWS CLI. Ejecuta el comando `aws sts assume-role` y pasa el ARN del rol para obtener las credenciales de seguridad temporales de dicho rol. Luego configura esas credenciales en las variables de entorno para que los comandos de la AWS CLI posteriores utilicen los permisos del rol. Aunque David utiliza el rol, no puede utilizar sus privilegios de usuario avanzado en la cuenta de desarrollo, ya que solo puede haber en vigor un conjunto de permisos a la vez.

Tenga en cuenta que todas las claves de acceso y tokens solo son ejemplos y no se pueden utilizar tal y como se muestran. Tiene que sustituirlos por los valores adecuados de su entorno real.

Para asumir un rol

1. David abre una ventana de símbolo de sistema y confirma que el cliente de la AWS CLI está trabajando, ejecutando el comando:

```
aws help
```

> **Note**
>
> El entorno predeterminado de David utiliza las credenciales de usuario de `David` de su perfil predeterminado que creó con el comando `aws configure`. Para obtener más información, consulte Configuración de AWS Command Line Interface en la Guía del usuario de AWS Command Line Interface.

2. Comienza el proceso de cambio de rol con la ejecución del siguiente comando para cambiar al rol `UpdateApp` en la cuenta de producción. El administrador que ha creado el rol le proporcionó el ARN del rol. El comando necesita que indique también un nombre de sesión; para ello puede elegir cualquier texto.

```
aws sts assume-role --role-arn "arn:aws:iam::999999999999:role/UpdateApp" --role-
session-name "David-ProdUpdate"
```

Después, David ve lo siguiente en la salida:

```
{
    "Credentials": {
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": "AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLE
CvSRyh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDy
EXAMPLE9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3Uuysg
sKdEXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLEsnf87e
NhyDHq6ikBQ==",
        "Expiration": "2014-12-11T23:08:07Z",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
```

```
}
```

3. David ve los tres elementos que necesitan en la sección Credentials (Credenciales) de la salida.

   - `AccessKeyId`
   - `SecretAccessKey`
   - `SessionToken`

   David necesita configurar el entorno de la AWS CLI para utilizar estos parámetros en las llamadas posteriores. Para obtener información sobre las distintas formas de configurar sus credenciales, consulte Configuración de la AWS Command Line Interface. No puede ejecutar el `aws configure`, ya que no admite la captura del token de sesión. Sin embargo, puede ingresar manualmente la información en un archivo de configuración. Dado que se trata de credenciales temporales con una fecha de vencimiento relativamente corta, es más fácil añadirlas al entorno de la sesión de la línea de comandos actual.

4. Para añadir los tres valores al entorno, David corta y pega la salida del paso anterior en los comandos siguientes. Puede interesarle cortar y pegar en un editor de texto sencillo para tratar los problemas de ajuste de línea de la salida del token de la sesión. Debe añadirse como una cadena única y larga, aunque se muestre con ajustes de línea para aportar mayor claridad.

   > **Note**
   >
   > En el siguiente ejemplo se muestran los comandos indicados en el entorno de Windows, donde "set" es el comando para crear una variable de entorno. En un equipo Linux o macOS, se utiliza el comando "export". Las demás partes del ejemplo son válidas en los tres entornos.
   >
   > Para obtener más información sobre el uso de las herramientas para Windows Powershell, consulte Para cambiar a un rol de IAM (Tools for Windows PowerShell)

```
set AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
set AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
set AWS_SESSION_TOKEN=AQoDYXdzEGcaEXAMPLE2gsYULo
+Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvS
Ryh0FW7jEXAMPLEW+vE/7s1HRpXviG7b+qYf4nD00EXAMPLEmj4wxS04L/
uZEXAMPLECihzFB5lTYLto9dyBgSDyEXA
MPLEKEY9/
g7QRUhZp4bqbEXAMPLENwGPyOj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UusKd
```

```
EXAMPLE1TVastU1A0SKFEXAMPLEiywCC/Cs8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP
+4eZScEXAMPLENhykxiHen
DHq6ikBQ==
```

En este punto, todos los comandos siguientes se ejecutan en los permisos del rol que las credenciales identifican. En el caso de David, el rol `UpdateApp`.

5. Ejecute el comando para obtener acceso a los recursos de la cuenta Producción. En este ejemplo, David genera una lista del contenido de su bucket de S3 con el siguiente comando.

```
aws s3 ls s3://productionapp
```

Dado que los nombres de bucket de Amazon S3 son únicos universalmente, no es necesario especificar el ID de la cuenta que posee el bucket. Para obtener acceso a los recursos de otros servicios de AWS, consulte la documentación de la AWS CLI correspondiente a dicho servicio para informarse de los comandos y la sintaxis que son necesarios para hacer referencia a sus recursos.

Uso de AssumeRole (API de AWS)

Cuando David necesita realizar una actualización en la cuenta de producción desde el código, llama a `AssumeRole` para asumir el rol `UpdateApp`. La llamada devuelve credenciales temporales que puede utilizar para obtener acceso al bucket `productionapp` de la cuenta de producción. Con estas credenciales, David puede realizar llamadas a la API para actualizar el bucket `productionapp`. Sin embargo, no puede realizar llamadas a la API para obtener acceso a otros recursos de la cuenta de producción, aunque tenga permisos de usuario avanzado en la cuenta de desarrollo.

Para asumir un rol

1. David llama a `AssumeRole` como parte de una aplicación. Deben especificar el ARN `UpdateApp`: `arn:aws:iam::999999999999:role/UpdateApp`.

   La respuesta de la llamada `AssumeRole` incluye las credenciales temporales con un `AccessKeyId` y una `SecretAccessKey`. También incluye una hora de `Expiration` que indica cuándo caducan las credenciales y debe solicitar otras nuevas.
2. Con las credenciales temporales, David realiza una llamada `s3:PutObject` para actualizar el bucket `productionapp`. Transfieren las credenciales a la llamada API como el parámetro

`AuthParams`. Dado que las credenciales temporales del rol tienen acceso de solo lectura y escritura al bucket `productionapp`, se deniegan todas las demás acciones de la cuenta Producción.

Para obtener código de ejemplo (mediante Python), consulte Cambio a un rol de IAM (API de AWS).

Recursos relacionados

- Para obtener más información acerca de los grupos de usuarios y usuarios de IAM, consulte Identidades de IAM (usuarios, grupos de usuarios y roles).
- Para obtener más información acerca los buckets de Amazon S3, consulte Creación de un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Resumen

Acaba de completar el tutorial de acceso entre varias cuentas mediante API. Ha creado un rol para establecer una relación de confianza con otra cuenta y definir qué acciones pueden realizar entidades de confianza. Después, ha modificado una política de grupo de usuarios para controlar qué usuarios de IAM pueden obtener acceso al rol. Como resultado, los desarrolladores de la cuenta de desarrollo pueden realizar actualizaciones en el bucket `productionapp` de la cuenta de producción mediante el uso de credenciales temporales.

Tutorial de IAM: crear y asociar su primera política administrada por el cliente

En este tutorial, utilice la AWS Management Console para crear una política administrada por el cliente y asociar dicha política a un usuario de IAM de su Cuenta de AWS. La política que cree permite a un usuario de prueba de IAM iniciar sesión directamente en la AWS Management Console con permisos de solo lectura.

Este flujo de trabajo incluye tres pasos básicos:

Paso 1: crear la política

De forma predeterminada, los usuarios de IAM no tienen permisos para realizar ninguna actividad. No pueden obtener acceso a Management Console de AWS ni administrar los datos, a no ser que usted lo permita. En este paso, debe crear una política administrada por el cliente que permita a cualquier usuario asociado iniciar sesión en la consola.

Paso 2: asociar la política

Al asociar una política a un usuario, el usuario hereda todos los permisos de acceso asociados a dicha política. En este paso, debe asociar la nueva política a una cuenta de usuario de prueba.

Paso 3: Probar el acceso de los usuarios

Una vez que haya asociado la política, puede iniciar sesión como el usuario y probar la política.

Requisitos previos

Para realizar los pasos en este tutorial, tendrá que ya tiene lo siguiente:

- Una Cuenta de AWS en la que puede iniciar sesión como usuario de IAM con permisos administrativos.
- Un usuario de prueba de IAM que no tenga permisos asignados ni suscripciones a grupos, tal y como se indica a continuación:

Nombre de usuario	Grupo	Permisos
PolicyUser	<ninguno>	<ninguno>

Paso 1: crear la política

En este paso, debe crear una política administrada por el cliente que permita a cualquier usuario asociado iniciar sesión en la AWS Management Console con acceso de solo lectura a los datos de IAM.

Para crear la política para un usuario de prueba

1. Inicie sesión en la consola de IAM en https://console.aws.amazon.com/iam/ con un usuario que tenga permisos de administrador.

2. En el panel de navegación, seleccione Policies (Políticas).
3. En el panel de contenido, elija Create policy (Crear política).
4. Seleccione la opción JSON y copie el texto del siguiente documento de política de JSON. Pegue el texto en el cuadro de texto JSON.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    } ]
}
```

5. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. Sin embargo, si realiza cambios o elige Revisar política en la pestaña Editor Visual, IAM podría reestructurar la política para optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

6. En la página Revisar y crear, escriba **`UsersReadOnlyAccessToIAMConsole`** como nombre de la política. Revise los permisos concedidos por la política y, a continuación, seleccione Crear política para guardar su trabajo.

   La nueva política aparece en la lista de las políticas administradas y está lista para asociar.

Paso 2: asociar la política

A continuación, debe asociar la política que acaba de crear al usuario de IAM de prueba.

Para asociar la política a un usuario de prueba

1. En la consola de IAM, en el panel de navegación, elija Políticas.
2. En la parte superior de la lista de políticas, en el cuadro de búsqueda, comience a escribir **`UsersReadOnlyAccesstoIAMConsole`** hasta que pueda ver la política. Después, seleccione el botón de radio situado junto a UsersReadOnlyAccessToIAMConsole en la lista.
3. Elija el botón Actions (Acciones) y, a continuación, elija Attach (Asociar).
4. En las entidades de IAM, seleccione la opción para filtrar por Usuarios.
5. En el cuadro de búsqueda, comience a escribir **`PolicyUser`** hasta que el usuario aparezca en la lista. A continuación, marque la casilla situada junto a ese usuario en la lista.
6. Elija Attach policy (Asociar política).

Ha asociado la política al usuario de prueba de IAM, lo que significa que el usuario ahora tiene acceso de solo lectura a la consola de IAM.

Paso 3: Probar el acceso de los usuarios

En este tutorial, le recomendamos que pruebe el acceso iniciando sesión como usuario de prueba para comprobar lo que los usuarios ven y cómo.

Para probar el acceso iniciando sesión en la cuenta de usuario de prueba

1. Inicie sesión en la consola de IAM en https://console.aws.amazon.com/iam/ con su usuario `PolicyUser` de prueba.
2. Examine las páginas de la consola e intente crear un nuevo usuario o grupo. Tenga en cuenta que `PolicyUser` puede mostrar datos, pero no puede crear ni modificar datos de IAM existentes.

Recursos relacionados

Para obtener información relacionada, consulte los recursos siguientes:

- Políticas administradas y políticas insertadas
- Controlar el acceso de los usuarios de IAM a la AWS Management Console

Resumen

Acaba de completar correctamente todos los pasos necesarios para crear y asociar una política administradas por el cliente. Como resultado, puede iniciar sesión en la consola de IAM con su cuenta de prueba para ver cómo es la experiencia de los usuarios.

Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puede asociar etiquetas a recursos de IAM, incluidas entidades de IAM (usuarios o roles) y recursos de AWS. Puede definir políticas que utilicen claves de condición de etiqueta para conceder permisos a sus entidades principales en función de sus etiquetas. Cuando utiliza etiquetas para controlar el acceso a sus recursos de AWS, permite que sus equipos y recursos crezcan con menos cambios en las políticas de AWS. Las políticas de ABAC son más flexibles que las políticas tradicionales de AWS, en las que debe enumerar cada recurso individual. Para obtener más información acerca de ABAC y su ventaja sobre las políticas tradicionales, consulte ¿Qué es ABAC para AWS?.

> **Note**
>
> Debe pasar un solo valor para cada etiqueta de sesión. AWS Security Token Service no admite etiquetas de sesión de varios valores.

Temas

- Resumen
- Recursos relacionados
- Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC

Información general del tutorial

En este tutorial se muestra cómo crear y probar una política que permite a los roles de IAM con etiquetas principales obtener acceso a los recursos con etiquetas coincidentes. Cuando una entidad principal realiza una solicitud a AWS, sus permisos se conceden en función de si la entidad principal y las etiquetas de recursos coinciden. Esta estrategia permite a las personas ver o editar solo los recursos de AWS necesarios para sus trabajos.

Escenario

Supongamos que es un desarrollador líder de una gran empresa denominada Empresa Ejemplo y que es un administrador de IAM con experiencia. Está familiarizado con la creación y administración de usuarios, roles y políticas de IAM. Quiere asegurarse de que los ingenieros de desarrollo y los miembros del equipo de control de calidad puedan obtener acceso a los recursos que necesitan. También necesita una estrategia que se escale a medida que su empresa crezca.

Puede elegir utilizar etiquetas de recursos de AWS y etiquetas principales de rol IAM para implementar una estrategia de ABAC para los servicios que la admitan, que comienzan por AWS Secrets Manager. Para saber qué servicios admiten la autorización basada en etiquetas, consulte Servicios de AWS que funcionan con IAM. Para obtener información sobre las claves de condición de etiquetado que pueden utilizarse en políticas con cada acción y recurso de servicio, consulte Acciones, recursos y claves de condición para servicios de AWS. Puede configurar su proveedor de identidad web o basado en SAML para que pase las etiquetas de sesión a AWS. Cuando los empleados se federan en AWS, sus atributos se aplican a su entidad principal resultante en AWS. Entonces puede utilizar ABAC para permitir o denegar permisos basados en esos atributos. Para saber cómo el uso de etiquetas de sesión con una identidad federada SAML difiere de este aprendizaje, consulte Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC.

Los miembros del equipo de ingeniería y control de calidad están en el proyecto Pegasus o Unicorn. Puede elegir los siguientes valores de etiqueta de equipo y proyecto de 3 caracteres:

- `access-project` = `peg` para el proyecto Pegasus
- `access-project` = `uni` para el proyecto Unicorn
- `access-team` = `eng` para el equipo de ingeniería

- `access-team` = `qas` para el equipo de control de calidad

Además, puede solicitar la etiqueta de asignación de costos `cost-center` para habilitar los informes de facturación personalizados de AWS. Para obtener más información, consulte Uso de etiquetas de asignación de costos en la Guía del usuario de AWS Billing and Cost Management.

Resumen de decisiones clave

- Los empleados inician sesión con las credenciales de usuario de IAM y, a continuación, asumen el rol de IAM para su equipo y proyecto. Si su empresa tiene su propio sistema de identidad, puede configurar la federación para permitir a los empleados asumir un rol sin usuarios de IAM. Para obtener más información, consulte Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC.
- La misma política se asocia a todos los roles. Las acciones se permiten o deniegan en función de las etiquetas.
- Los empleados pueden crear nuevos recursos, pero solo si asocian las mismas etiquetas al recurso que se aplica a su rol. Esto garantiza que los empleados puedan ver el recurso después de crearlo. Ya no es necesario que los administradores actualicen las políticas con el ARN de los nuevos recursos.
- Los empleados pueden leer los recursos que pertenecen a su equipo, independientemente del proyecto.
- Los empleados pueden actualizar y eliminar recursos propiedad de su propio equipo y proyecto.
- Los administradores de IAM pueden agregar un nuevo rol para nuevos proyectos. Pueden crear y etiquetar un nuevo usuario de IAM para permitir el acceso al rol adecuado. Los administradores no tienen que editar una política para admitir un nuevo proyecto o miembro del equipo.

En este tutorial, etiquetará cada recurso, etiquetará los roles del proyecto y añadirá políticas a los roles para permitir el comportamiento descrito anteriormente. La política resultante permite a los roles `Create`, `Read`, `Update` y `Delete` acceso a los recursos que están etiquetados con las mismas etiquetas de proyecto y equipo. La política también permite el acceso entre proyectos a `Read` para los recursos que están etiquetados con el mismo equipo.

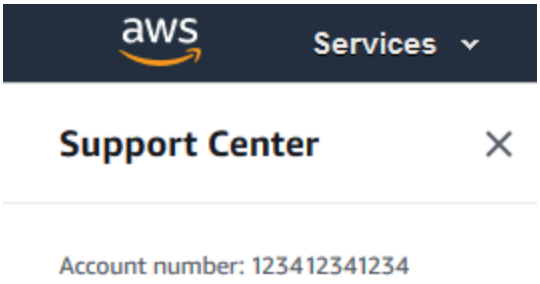
Requisitos previos

Para realizar los pasos en este tutorial, deberá disponer de lo siguiente:

- Una Cuenta de AWS en la que puede iniciar sesión como usuario con permisos administrativos.

- Su ID de cuenta de 12 dígitos, que utilizará para crear los roles en el paso 3.

  Para encontrar el número de ID de cuenta de AWS mediante la AWS Management Console, seleccione Support (Soporte) en la barra de navegación en la parte superior derecha y, a continuación, elija Support Center (Centro de soporte). El número de cuenta (ID) aparece en el panel de navegación de la izquierda.



- Experimente creando y editando usuarios, roles y políticas de IAM en la AWS Management Console. Sin embargo, si necesita ayuda para recordar un proceso de administración de IAM, este tutorial proporciona enlaces en los que puede ver las instrucciones paso a paso.

Paso 1: crear usuarios de prueba

Para realizar pruebas, cree cuatro usuarios de IAM con permisos para asumir roles con las mismas etiquetas. Esto facilita añadir más usuarios a sus equipos. Al etiquetar los usuarios, estos obtienen acceso automáticamente para asumir el rol correcto. No es necesario agregar los usuarios a la política de confianza del rol si solo trabajan en un proyecto y en un equipo.

1. Cree la siguiente política administrada por el cliente denominada `access-assume-role`. Para obtener más información acerca de la creación de un política JSON, consulte Crear políticas de IAM.

   Política de ABAC: asume cualquier rol de ABAC, pero solo cuando coinciden las etiquetas de usuario y rol.

   La siguiente política permite a un usuario asumir cualquier rol de su cuenta con el prefijo de nombre `access-`. El rol también debe estar etiquetado con las mismas etiquetas de proyecto, equipo y centro de costos que el usuario.

   Para utilizar esta política, sustituya el texto del marcador de posición en cursiva por la información de la cuenta.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::account-ID-without-hyphens:role/access-*",
            "Condition": {
                "StringEquals": {
                    "iam:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                    "iam:ResourceTag/access-team": "${aws:PrincipalTag/access-
team}",
                    "iam:ResourceTag/cost-center": "${aws:PrincipalTag/cost-
center}"
                }
            }
        }
    ]
}
```

Para escalar este tutorial a un gran número de usuarios, puede asociar la política a un grupo y añadir cada usuario al grupo. Para obtener más información, consulte Creación de un grupo de usuarios de IAM y Agregar y eliminar usuarios de un grupo de usuarios de IAM.

2. Cree los siguientes usuarios de IAM, asocie la política de permisos `access-assume-role`. Asegúrese de seleccionar Conceder acceso de usuario a la AWS Management Console y, luego, agregue las siguientes etiquetas. Para obtener más información acerca de cómo crear y etiquetar un nuevo usuario, consulte Creación de usuarios de IAM (consola).

   Usuarios de ABAC

Nombre de usuario	Clave de la etiqueta de usuario	Valor de la etiqueta de usuario
`access-Arnav-peg-eng`	`access-project`	`peg`
	`access-team`	`eng`

Nombre de usuario	Clave de la etiqueta de usuario	Valor de la etiqueta de usuario
	`cost-center`	`987654`

`access-Mary-peg-qas`	`access-project`
	`access-team`
	`cost-center`
		`peg`
		`qas`
		`987654`

`access-Saanvi-uni-eng`	`access-project`
	`access-team`
	`cost-center`
		`uni`
		`eng`
		`123456`

`access-Carlos-uni-qas`	`access-project`
	`access-team`
	`cost-center`
		`uni`
		`qas`
		`123456`

Paso 2: crear la política de ABAC

Cree la siguiente política con el nombre **`access-same-project-team`**. Añadirá esta política a los roles en un paso posterior. Para obtener más información acerca de la creación de un política JSON, consulte Crear políticas de IAM.

Para obtener más políticas que puede adaptar para este tutorial, consulte las siguientes páginas:

- Control del acceso para entidades principales de IAM
- Amazon EC2: permite iniciar o detener instancias EC2 que un usuario haya etiquetado, mediante programación y en la consola
- EC2: iniciar o detener instancias basándose en etiquetas de recursos y principal coincidentes
- EC2: iniciar o detener instancias en función de las etiquetas
- IAM: asumir funciones que tienen una etiqueta específica

Política de ABAC: acceso a los recursos de Secrets Manager solo cuando coinciden las etiquetas principal y de recursos

La siguiente política permite a las entidades principales crear, leer, editar y eliminar recursos, pero solo cuando dichos recursos están etiquetados con los mismos pares clave-valor que la entidad principal. Cuando una entidad principal crea un recurso, debe añadir etiquetas `access-project`, `access-team` y `cost-center` con valores que coincidan con las etiquetas de la entidad principal. La política también permite añadir etiquetas `Name` u `OwnedBy` opcionales.

```
{
 "Version": "2012-10-17",
 "Statement": [
     {
         "Sid": "AllActionsSecretsManagerSameProjectSameTeam",
         "Effect": "Allow",
         "Action": "secretsmanager:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
                 "aws:ResourceTag/access-project": "${aws:PrincipalTag/access-
project}",
                 "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:ResourceTag/cost-center": "${aws:PrincipalTag/cost-center}"
             },
             "ForAllValues:StringEquals": {
                 "aws:TagKeys": [
                     "access-project",
                     "access-team",
                     "cost-center",
                     "Name",
                     "OwnedBy"
                 ]
             },
             "StringEqualsIfExists": {
                 "aws:RequestTag/access-project": "${aws:PrincipalTag/access-project}",
                 "aws:RequestTag/access-team": "${aws:PrincipalTag/access-team}",
                 "aws:RequestTag/cost-center": "${aws:PrincipalTag/cost-center}"
             }
         }
     },
     {
         "Sid": "AllResourcesSecretsManagerNoTags",
         "Effect": "Allow",
```

```
            "Action": [
                "secretsmanager:GetRandomPassword",
                "secretsmanager:ListSecrets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ReadSecretsManagerSameTeam",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:Describe*",
                "secretsmanager:Get*",
                "secretsmanager:List*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/access-team": "${aws:PrincipalTag/access-team}"
                }
            }
        },
        {
            "Sid": "DenyUntagSecretsManagerReservedTags",
            "Effect": "Deny",
            "Action": "secretsmanager:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringLike": {
                    "aws:TagKeys": "access-*"
                }
            }
        },
        {
            "Sid": "DenyPermissionsManagement",
            "Effect": "Deny",
            "Action": "secretsmanager:*Policy",
            "Resource": "*"
        }
    ]
}
```

¿Qué hace esta política?

- La instrucción `AllActionsSecretsManagerSameProjectSameTeam` permite todas las acciones de este servicio en todos los recursos relacionados, pero solo si las etiquetas de los recursos coinciden con las etiquetas principales. Al agregar `"Action": "secretsmanager:*"` a la política, la política crece a medida que Secrets Manager crece. Si Secrets Manager añade una nueva operación de API, no es necesario que agregue dicha acción a la instrucción. La instrucción implementa el ABAC utilizando tres bloques de condición. La solicitud solo se permite si los tres bloques devuelven true.
  - El primer bloque de condición de esta instrucción devuelve true si las claves de etiqueta especificadas están presentes en el recurso y sus valores coinciden con las etiquetas de la entidad principal. Este bloque devuelve false para etiquetas no coincidentes o para acciones que no admiten el etiquetado de recursos. Para saber qué acciones no permite este bloque, consulte Claves de condición, recursos y acciones de AWS Secrets Manager. En dicha página se muestra que las acciones realizadas en el tipo de recurso Secret admiten la clave de condición `secretsmanager:ResourceTag/tag-key`. Algunas acciones de Secrets Manager no admiten ese tipo de recurso, incluidas `GetRandomPassword` y `ListSecrets`. Debe crear instrucciones adicionales para permitir esas acciones.
  - El segundo bloque de condición devuelve true si todas las claves de etiqueta pasadas en la solicitud se incluyen en la lista especificada. Esto se realiza utilizando `ForAllValues` con el operador de condición `StringEquals`. Si no se pasa ninguna clave ni subconjunto del conjunto de claves, la condición se vuelve verdadera. Esto permite operaciones `Get*` que no permiten pasar etiquetas en la solicitud. Si el solicitante incluye una clave de etiqueta que no está en la lista, la condición devuelve false. Cada clave de etiqueta que se transfiere en la solicitud debe coincidir con un miembro de esta lista. Para obtener más información, consulte Claves de contexto multivalor.
  - El tercer bloque de condición devuelve true si la solicitud admite la transferencia de etiquetas, si las tres etiquetas están presentes y si coinciden con los valores de etiqueta principal. Este bloque también devuelve true si la solicitud no admite la transferencia de etiquetas. Esto se debe a `...IfExists` en el operador de condición. El bloque devuelve false si no se pasa ninguna etiqueta durante una acción que lo admite, o si las claves y los valores de la etiqueta no coinciden.
- La instrucción `AllResourcesSecretsManagerNoTags` permite las acciones `ListSecrets` y `GetRandomPassword` que la primera instrucción no permite.
- La instrucción `ReadSecretsManagerSameTeam` permite operaciones de solo lectura si la entidad principal está etiquetada con la misma etiqueta de equipo de acceso que el recurso. Esto está permitido independientemente del proyecto o de la etiqueta del centro de costos.

- La instrucción `DenyUntagSecretsManagerReservedTags` deniega las solicitudes para eliminar de Secrets Manager etiquetas con claves que comienzan por "access-". Estas etiquetas se utilizan para controlar el acceso a los recursos, por tanto, si se eliminan etiquetas se podrían eliminar permisos.
- La instrucción `DenyPermissionsManagement` deniega el acceso para crear, editar o eliminar políticas basadas en recursos de Secrets Manager. Estas políticas se pueden utilizar para cambiar los permisos del secreto.

> **Important**
>
> Esta política utiliza una estrategia para permitir todas las acciones de un servicio, pero deniega explícitamente las acciones de modificación de permisos. Si se deniega una acción, se anula cualquier otra política que permita a la entidad principal realizar dicha acción. Esto puede tener resultados no deseados. Como práctica recomendada, utilice denegaciones explícitas solo cuando no haya ninguna circunstancia que deba permitir dicha acción. De lo contrario, permita una lista de acciones individuales y las acciones no deseadas se deniegan de forma predeterminada.

Paso 3: crear roles

Cree los siguientes roles de IAM y asocie la política **`access-same-project-team`** creada en el paso anterior. Para obtener más información sobre cómo crear un rol de IAM, consulte Creación de un rol para delegar permisos a un usuario de IAM. Si decide utilizar la federación en lugar de usuarios y roles de IAM, consulte Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC.

Roles de ABAC

Función de trabajo	Nombre de rol	Etiquetas de roles	Descripción del rol
Ingeniería del proyecto Pegasus	`access-peg-engineering`	`access-project = peg` `access-team = eng`	Permite a los ingenieros leer todos los recursos de ingeniería y crear y administrar los recursos de ingeniería de Pegasus.

Función de trabajo	Nombre de rol	Etiquetas de roles	Descripción del rol
		`cost-center = 987654`	

Control de calidad del proyecto Pegasus	`access-peg-quality-assurance`	`access-project = peg`
		`access-team = qas`
		`cost-center = 987654`
			Permite al equipo de control de calidad leer todos los recursos de control de calidad y crear y administrar todos los recursos de control de calidad de Pegasus.

Ingeniería del proyecto Unicorn	`access-uni-engineering`	`access-project = uni`
		`access-team = eng`
		`cost-center = 123456`
			Permite a los ingenieros leer todos los recursos de ingeniería y crear y administrar los recursos de ingeniería de Unicorn.

Control de calidad del proyecto Unicorn	`access-uni-quality-assurance`	`access-project = uni`
		`access-team = qas`
		`cost-center = 123456`
			Permite al equipo de control de calidad leer todos los recursos de control de calidad y crear y administrar todos los recursos de control de calidad de Unicorn.

Paso 4: Probar la creación de secretos

La política de permisos asociada a los roles permite a los empleados crear secretos. Esto solo se permite si el secreto está etiquetado con su proyecto, equipo y centro de costos. Confirme que sus permisos funcionan según lo previsto iniciando sesión como usuarios, asumiendo el rol correcto y probando la actividad en Secrets Manager.

Para probar la creación de un secreto con y sin las etiquetas necesarias

1. En la ventana principal del navegador, mantenga la sesión iniciada como usuario administrador para que pueda revisar los usuarios, los roles y las políticas en IAM. Utilice una ventana de incógnito del navegador o un navegador independiente para las pruebas. Ahí, inicie sesión como usuario de IAM `access-Arnav-peg-eng` en la consola de Secrets Manager en https://console.aws.amazon.com/secretsmanager/.
2. Intente cambiar al rol `access-uni-engineering`.

   Esta operación falla porque los valores de la etiqueta `access-project` y `cost-center` no coinciden con el usuario `access-Arnav-peg-eng` y el rol `access-uni-engineering`.

   Para obtener más información acerca del cambio de roles en la AWS Management Console, consulte Cambio a un rol (Consola)
3. Cambie al rol `access-peg-engineering`.
4. Almacene un nuevo secreto con la siguiente información. Para obtener información sobre cómo almacenar un secreto, consulte Creación de un secreto básico en la Guía del usuario de AWS Secrets Manager.

   > **Important**
   >
   > Secrets Manager muestra avisos indicando que no tiene permisos para servicios de AWS adicionales que funcionan con Secrets Manager. Por ejemplo, para crear credenciales para una base de datos de Amazon RDS, debe tener permiso para describir instancias de RDS, clústeres de RDS y clústeres de Amazon Redshift. Puede ignorar estas alertas, ya que en este tutorial no está utilizando estos servicios de AWS específicos.

1. En la sección Select secret type (Seleccionar tipo de secreto), elija Other type of secrets (Otro tipo de secretos). En los dos cuadros de texto, escriba `test-access-key` y `test-access-secret`.
2. Escriba `test-access-peg-eng` en el campo Secret name (Nombre del secreto).
3. Añada diferentes combinaciones de etiquetas de la siguiente tabla y vea el comportamiento esperado.
4. Elija Store (Almacenar) para intentar crear el secreto. Cuando se produzca un error en el almacenamiento, vuelva a las páginas de la consola de Secrets Manager anteriores y utilice el siguiente conjunto de etiquetas de la siguiente tabla. El último conjunto de etiquetas está permitido y creará correctamente el secreto.

Combinaciones de etiquetas de ABAC para el rol **`test-access-peg-eng`**

`access-project` Valor de etiqueta	**`access-team`** Valor de etiqueta	**`cost-center`** Valor de etiqueta	Etiquetas adicionales	Comportamiento esperado
(ninguno)	(ninguno)	(ninguno)	(ninguno)	Se deniega porque el valor de la etiqueta `access-project` no coincide con el valor del rol de `peg`.
`uni`	`eng`	`987654`	(ninguno)	Se deniega porque el valor de la etiqueta `access-project` no coincide con el valor del rol de `peg`.
`peg`	`qas`	`987654`	(ninguno)	Se deniega porque el valor de la etiqueta `access-team` no coincide con el valor del rol de `eng`.
`peg`	`eng`	`123456`	(ninguno)	Se deniega porque el valor de la etiqueta `cost-center` no coincide con el valor del rol de `987654`.

access-project Valor de etiqueta	**access-team** Valor de etiqueta	**cost-center** Valor de etiqueta	Etiquetas adicionales	Comportamiento esperado
peg	eng	987654	owner = Jane	Se deniega porque la política no permite la etiqueta adicional owner, aunque las tres etiquetas obligatorias estén presentes y sus valores coincidan con los valores del rol.
peg	eng	987654	Name = Jane	Se permite porque las tres etiquetas obligatorias están presentes y sus valores coinciden con los valores del rol. También puede incluir la etiqueta Name opcional.

5. Cierre la sesión y repita los tres primeros pasos de este procedimiento para cada uno de los siguientes roles y valores de etiqueta. En el cuarto paso de este procedimiento, pruebe cualquier conjunto de etiquetas que faltan, etiquetas opcionales, etiquetas no permitidas y valores de etiqueta no válidos que elija. A continuación, utilice las etiquetas necesarias para crear un secreto con las siguientes etiquetas y nombre.

   Roles y etiquetas de ABAC

Nombre de usuario	Nombre de rol	Nombre del secreto	Etiquetas de secretos
access-Mary-peg-qas	access-peg-quality-assurance	test-access-peg-qas	access-project = peg access-team = qas cost-center = 987654

Nombre de usuario	Nombre de rol	Nombre del secreto	Etiquetas de secretos

`access-Saanvi-uni-eng`	`access-uni-engineering`	`test-access-uni-eng`	`access-project = uni`
			`access-team = eng`
			`cost-center = 123456`

`access-Carlos-uni-qas`	`access-uni-quality-assurance`	`test-access-uni-qas`	`access-project = uni`
			`access-team = qas`
			`cost-center = 123456`

Paso 5: Probar la visualización de secretos

La política que ha adjuntado a cada rol permite a los empleados ver cualquier secreto etiquetado con su nombre de equipo, independientemente de su proyecto. Confirme que sus permisos funcionan según lo previsto probando los roles en Secrets Manager.

Para probar la visualización de un secreto con y sin las etiquetas requeridas

1. Inicie sesión como uno de los siguientes usuarios de IAM:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
2. Cambie al rol coincidente:
   - `access-peg-engineering`

- `access-peg-quality-assurance`
- `access-uni-engineering`
- `access-uni-quality-assurance`

Para obtener más información acerca del cambio de roles en la AWS Management Console, consulte Cambio a un rol (Consola).

3. En el panel de navegación de la izquierda, elija el icono de menú para ampliar el menú y, a continuación, elija Secrets (Secretos).
4. Debería ver los cuatro secretos en la tabla, independientemente del rol actual. Esto se espera porque la política con el nombre `access-same-project-team` permite la acción `secretsmanager:ListSecrets` para todos los recursos.
5. Elija el nombre de uno de los secretos.
6. En la página de detalles del secreto, las etiquetas de su rol determinan si puede ver el contenido de la página. Compare el nombre de su rol con el nombre de su secreto. Si comparten el mismo nombre de equipo, las etiquetas `access-team` coinciden. Si no coinciden, el acceso se deniega.

   Comportamiento de visualización de secretos de ABAC para cada rol

Nombre de rol	Nombre del secreto	Comportamiento esperado
`access-peg-engineering`	`test-access-peg-eng`	Permitida
	`test-access-peg-qas`	Denegado
	`test-access-uni-eng`	Permitida
	`test-access-uni-qas`	Denegado
`access-peg-quality-assurance`	`test-access-peg-eng`	Denegado
	`test-access-peg-qas`	Permitida
	`test-access-uni-eng`	Denegado
	`test-access-uni-qas`	Permitida

Nombre de rol	Nombre del secreto	Comportamiento esperado
`access-uni-engineering`	`test-access-peg-eng`	Permitida
	`test-access-peg-qas`	Denegado
	`test-access-uni-eng`	Permitida
	`test-access-uni-qas`	Denegado
`access-uni-quality-assurance`	`test-access-peg-eng`	Denegado
	`test-access-peg-qas`	Permitida
	`test-access-uni-eng`	Denegado
	`test-access-uni-qas`	Permitida

7. En la parte superior de la página, elija Secrets (Secretos) para volver a la lista de secretos. Repita los pasos de este procedimiento utilizando diferentes roles para probar si puede ver cada uno de los secretos.

Paso 6: Probar la escalabilidad

Un motivo importante para utilizar el control de acceso basado en atributos (ABAC) sobre el control de acceso basado en roles (RBAC) es la escalabilidad. A medida que su empresa añade nuevos proyectos, equipos o personas a AWS, no es necesario actualizar sus políticas basadas en ABAC. Por ejemplo, supongamos que la Empresa Ejemplo está financiando un nuevo proyecto, con un código con el nombre Centaur. Un ingeniero llamado Saanvi Sarkar será el ingeniero jefe de Centaur y continuará trabajando en el proyecto Unicorn . Saanvi también revisará el trabajo del proyecto Peg. También hay varios ingenieros recién contratados, incluido Nikhil Jayashankar, que trabajará solo en el proyecto Centaur.

Para añadir el nuevo proyecto a AWS

1. Inicie sesión como usuario administrador de IAM y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación de la izquierda, seleccione Roles y añada un rol de IAM con el nombre `access-cen-engineering`. Asocie la política de permisos **`access-same-project-team`** al rol y agregue las siguientes etiquetas de rol:

   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`

3. En el panel de navegación de la izquierda, elija Usuarios.

4. Agregue un nuevo usuario denominado `access-Nikhil-cen-eng`, asocie la política denominada `access-assume-role` y agregue las siguientes etiquetas de usuario.

   - `access-project` = `cen`
   - `access-team` = `eng`
   - `cost-center` = `101010`

5. Utilice los procedimientos de Paso 4: Probar la creación de secretos y Paso 5: Probar la visualización de secretos. En otra ventana del navegador, pruebe que Nikhil solo puede crear secretos de ingeniería de Centaur y que puede ver todos los secretos de ingeniería.

6. En la ventana principal del navegador en la que ha iniciado sesión como administrador, elija el usuario `access-Saanvi-uni-eng`.

7. En la pestaña Permissions (Permisos), elimine la política de permisos access-assume-role.

8. Añada la siguiente política insertada denominada `access-assume-specific-roles`. Para obtener más información acerca de cómo añadir una política insertada a un usuario, consulte Para integrar una política insertada de un usuario o un rol (consola).

   Política de ABAC: asumir solo roles específicos

   Esta política permite a Saanvi asumir los roles de ingeniería de los proyectos Pegasus y Centaur. Es necesario crear esta política personalizada porque IAM no admite etiquetas con varios valores. No puede etiquetar al usuario de Saanvi con `access-project` = `peg` y `access-project` = `cen`. Además, el modelo de autorización de AWS no puede coincidir con ambos valores. Para obtener más información, consulte Reglas para etiquetar en IAM y AWS STS. En su lugar, debe especificar manualmente los dos roles que puede asumir.

   Para utilizar esta política, sustituya el texto del marcador de posición en cursiva por la información de la cuenta.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
                "arn:aws:iam::account-ID-without-hyphens:role/access-cen-
engineering"
            ]
        }
    ]
}
```

9. Utilice los procedimientos de Paso 4: Probar la creación de secretos y Paso 5: Probar la visualización de secretos. En otra ventana del navegador, confirme que Saanvi puede asumir ambos roles. Compruebe que solo puede crear secretos para su proyecto, equipo y centro de costos, en función de las etiquetas del rol. Confirme también que puede ver detalles sobre cualquier secreto propiedad del equipo de ingeniería, incluidos los que acaba de crear.

Paso 7: Probar la actualización y eliminación de secretos

La política `access-same-project-team` asociada a los roles permite a los empleados actualizar y eliminar los secretos etiquetados con su proyecto, equipo y centro de costos. Confirme que sus permisos funcionan según lo previsto probando los roles en Secrets Manager.

Para probar la actualización y eliminación de un secreto con y sin las etiquetas requeridas

1. Inicie sesión como uno de los siguientes usuarios de IAM:
   - `access-Arnav-peg-eng`
   - `access-Mary-peg-qas`
   - `access-Saanvi-uni-eng`
   - `access-Carlos-uni-qas`
   - `access-Nikhil-cen-eng`

2. Cambie al rol coincidente:

   - `access-peg-engineering`
   - `access-peg-quality-assurance`
   - `access-uni-engineering`
   - `access-peg-quality-assurance`
   - `access-cen-engineering`

   Para obtener más información acerca del cambio de roles en la AWS Management Console, consulte Cambio a un rol (Consola).

3. Para cada rol, intente actualizar la descripción del secreto y, a continuación, intente eliminar los siguientes secretos. Para obtener más información, consulte Modificación de un secreto y Eliminación y restauración de un secreto en la Guía del usuario de AWS Secrets Manager.

   Comportamiento de actualización y eliminación de secretos de ABAC para cada rol

Nombre de rol	Nombre del secreto	Comportamiento esperado
`access-peg-engineering`	`test-access-peg-eng`	Permitida
	`test-access-uni-eng`	Denegado
	`test-access-uni-qas`	Denegado
`access-peg-quality-assurance`	`test-access-peg-qas`	Permitida
	`test-access-uni-eng`	Denegado
`access-uni-engineering`	`test-access-uni-eng`	Permitida
	`test-access-uni-qas`	Denegado
`access-peg-quality-assurance`	`test-access-uni-qas`	Permitida

Resumen

Ha completado correctamente todos los pasos necesarios para utilizar etiquetas para el control de acceso basado en atributos (ABAC). Ha aprendido a definir una estrategia de etiquetado. Ha aplicado dicha estrategia a sus entidades principales y recursos. Ha creado y aplicado una política que aplica la estrategia para Secrets Manager. También ha aprendido que ABAC se escala fácilmente cuando añade nuevos proyectos y miembros del equipo. Como resultado, puede iniciar sesión en la consola de IAM con sus roles de prueba y experimentar cómo utilizar etiquetas para ABAC en AWS.

> **Note**
>
> Ha agregado políticas que permiten acciones solo en determinadas condiciones. Si aplica una política diferente a los usuarios o roles que tiene permisos más amplios, es posible que las acciones no se limiten a requerir el etiquetado. Por ejemplo, si concede a un usuario permisos administrativos completos mediante la política administrada por AWS `AdministratorAccess`, estas políticas no restringen ese acceso. Para obtener más información acerca de cómo se determinan los permisos cuando se aplican varias políticas, consulte Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta.

Recursos relacionados

Para obtener información relacionada, consulte los recursos siguientes:

- ¿Qué es ABAC para AWS?
- Claves de contexto de condición globales de AWS
- Creación de usuarios de IAM (consola)
- Creación de un rol para delegar permisos a un usuario de IAM
- Etiquetado de recursos de IAM
- Control de acceso a los recursos de AWS mediante etiquetas
- Cambio a un rol (Consola)
- Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC

Para obtener información sobre cómo monitorear las etiquetas en su cuenta, consulte Monitorear los cambios de etiquetas en recursos de AWS con flujos de trabajo sin servidor y Amazon CloudWatch Events.

Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos basados en atributos. En AWS, estos atributos se denominan etiquetas. Puede asociar etiquetas a recursos de IAM, incluidas entidades de IAM (usuarios o roles) y recursos de AWS. Cuando las entidades se utilizan para realizar solicitudes de AWS, se convierten en entidades principales y esas entidades incluyen etiquetas.

También puede pasar etiquetas de sesión al asumir un rol o federar un usuario. A continuación, puede definir políticas que utilicen claves de condición de etiqueta para conceder permisos a sus entidades principales en función de sus etiquetas. Cuando utiliza etiquetas para controlar el acceso a sus recursos de AWS, permite que sus equipos y recursos crezcan con menos cambios en las políticas de AWS. Las políticas de ABAC son más flexibles que las políticas tradicionales de AWS, en las que debe enumerar cada recurso individual. Para obtener más información acerca de ABAC y su ventaja sobre las políticas tradicionales, consulte ¿Qué es ABAC para AWS?.

Si su empresa utiliza un proveedor de identidades (IdP) basado en SAML para administrar las identidades de usuarios corporativos, puede utilizar atributos SAML para un control de acceso detallado en AWS. Los atributos pueden incluir identificadores de centros de coste, direcciones de correo electrónico de usuario, clasificaciones de departamento y asignaciones de proyectos. Cuando pasa estos atributos como etiquetas de sesión, puede controlar el acceso a AWS basándose en estas etiquetas de sesión.

Para completar el tutorial de ABAC pasando atributos de SAML a su sesión principal, complete las tareas de Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas, con los cambios que se incluyen en este tema.

Requisitos previos

Para realizar los pasos necesarios para utilizar las etiquetas de sesión de SAML para ABAC, ya debe tener lo siguiente:

- Acceso a un IdP basado en SAML donde puede crear usuarios de prueba con atributos específicos.
- La posibilidad de iniciar sesión como usuario con permisos administrativos.

- Experimente creando y editando usuarios, roles y políticas de IAM en la AWS Management Console. Sin embargo, si necesita ayuda para recordar un proceso de administración de IAM, el tutorial de ABAC proporciona enlaces en los que puede ver las instrucciones paso a paso.
- Experimenta la configuración de un IdP basado en SAML en IAM. Para ver más detalles y vínculos a documentación detallada de IAM, consulte Traspaso de etiquetas de sesión mediante AssumeRoleWithSAML.

Paso 1: crear usuarios de prueba

Omita las instrucciones en Paso 1: crear usuarios de prueba. Dado que sus identidades están definidas en su proveedor, no es necesario que agregue usuarios de IAM para sus empleados.

Paso 2: crear la política de ABAC

Siga las instrucciones de Paso 2: crear la política de ABAC para crear la política administrada especificada en IAM.

Paso 3: crear y configurar el rol de SAML

Cuando utilice el aprendizaje de ABAC para SAML, debe realizar pasos adicionales para crear el rol, configurar el IdP de SAML y habilitar el acceso a la AWS Management Console. Para obtener más información, consulte Paso 3: crear roles.

Paso 3A: crear el rol de SAML

Cree un único rol que confíe en el proveedor de identidades de SAML y en el usuario `test-session-tags` que creó en el paso 1. El tutorial de ABAC utiliza roles independientes con etiquetas de rol diferentes. Dado que está pasando etiquetas de sesión desde su IdP de SAML, solo necesita un rol. Para obtener información sobre cómo crear un rol basado en SAML, consulte Creación de un rol para una federación SAML 2.0 (consola).

Llame al rol `access-session-tags`. Asocie una política de permisos `access-same-project-team` al rol. Edite la política de confianza del rol para utilizar la siguiente política. Para obtener instrucciones detalladas sobre cómo editar la relación de confianza de un rol, consulte Modificación de un rol (consola).

La siguiente política de confianza del rol permite que el proveedor de identidad de SAML y el usuario `test-session-tags` asuman el rol. Cuando asumen el rol, deben pasar las tres etiquetas de sesión especificadas. La acción `sts:TagSession` es necesaria para permitir pasar etiquetas de sesión.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSamlIdentityAssumeRole",
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRoleWithSAML",
                "sts:TagSession"
            ],
            "Principal": {"Federated":"arn:aws:iam::123456789012:saml-
provider/ExampleCorpProvider"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/cost-center": "*",
                    "aws:RequestTag/access-project": "*",
                    "aws:RequestTag/access-team": [
                        "eng",
                        "qas"
                    ]
                },
                "StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/saml"}
            }
        }
    ]
}
```

La instrucción `AllowSamlIdentityAssumeRole` permite a los miembros de los equipos de Ingeniería y Garantía de Calidad asumir este rol cuando se federan en AWS desde el IdP de Example Corporation. El proveedor `ExampleCorpProvider` de SAML se define en IAM. El administrador ya ha configurado la aserción de SAML para pasar las tres etiquetas de sesión requeridas. La aserción puede pasar etiquetas adicionales, pero estos tres deben estar presentes. Los atributos de la identidad pueden tener cualquier valor para las etiquetas `cost-center` y `access-project`. Sin embargo, el valor del atributo `access-team` debe coincidir con `eng` o `qas` para indicar que la identidad está en el equipo de ingeniería o control de calidad.

Paso 3B: configurar el IdP de SAML

Configure su IdP de SAML para que pase los atributos `cost-center`, `access-project` y `access-team` como etiquetas de sesión. Para obtener más información, consulte Traspaso de etiquetas de sesión mediante AssumeRoleWithSAML.

Para pasar estos atributos como etiquetas de sesión, incluya los siguientes elementos en su aserción de SAML.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:cost-center">
  <AttributeValue>987654</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-project">
  <AttributeValue>peg</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:access-team">
  <AttributeValue>eng</AttributeValue>
</Attribute>
```

Paso 3C: Habilitar el acceso a la consola

Habilite el acceso a la consola para los usuarios federados de SAML. Para obtener más información, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 .

Paso 4: Probar la creación de secretos

Federarse en la AWS Management Console mediante el rol `access-session-tags`. Para obtener más información, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 . A continuación, siga las instrucciones de Paso 4: Probar la creación de secretos para crear secretos. Utilice diferentes identidades de SAML con atributos para que coincidan con las etiquetas indicadas en el tutorial de ABAC. Para obtener más información, consulte Paso 4: Probar la creación de secretos.

Paso 5: Probar la visualización de secretos

Siga las instrucciones de Paso 5: Probar la visualización de secretos para ver los secretos que creó en el paso anterior. Utilice diferentes identidades de SAML con atributos para que coincidan con las etiquetas indicadas en el tutorial de ABAC.

Paso 6: Probar la escalabilidad

Siga las instrucciones de Paso 6: Probar la escalabilidad para probar la escalabilidad. Para ello, agregue una nueva identidad en su IdP basado en SAML con los siguientes atributos:

- `cost-center = 101010`
- `access-project = cen`

- `access-team = eng`

Paso 7: Probar la actualización y eliminación de secretos

Siga las instrucciones de Paso 7: Probar la actualización y eliminación de secretos para actualizar y eliminar secretos. Utilice diferentes identidades de SAML con atributos para que coincidan con las etiquetas indicadas en el tutorial de ABAC.

> **Important**
>
> Elimine todos los secretos que ha creado para evitar cargos de facturación. Para obtener más información sobre los precios de Secrets Manager, consulte Precios de AWS Secrets Manager.

Resumen

Ahora ha completado correctamente todos los pasos necesarios para utilizar etiquetas de sesión y etiquetas de recursos de SAML para la administración de permisos.

> **Note**
>
> Ha agregado políticas que permiten acciones solo en determinadas condiciones. Si aplica una política diferente a los usuarios o roles que tiene permisos más amplios, es posible que las acciones no se limiten a requerir el etiquetado. Por ejemplo, si concede a un usuario permisos administrativos completos mediante la política administrada por AWS `AdministratorAccess`, estas políticas no restringen ese acceso. Para obtener más información acerca de cómo se determinan los permisos cuando se aplican varias políticas, consulte Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta.

Tutorial de IAM: permitir a los usuarios administrar sus credenciales y configuración de MFA

Puede permitir que los usuarios administren sus propias credenciales y dispositivos de autenticación multifactor (MFA) en la página Credenciales de seguridad. Puede utilizar la AWS Management Console para configurar credenciales (claves de acceso, contraseñas, certificados de firma y claves

públicas SSH) y eliminar o desactivar credenciales que no son necesarias, además de habilitar los dispositivos MFA para sus usuarios. Esto es útil para pocos usuarios; sin embargo, es una tarea que pronto podría requerir demasiado tiempo si el número de usuarios aumenta. En este tutorial se muestra cómo habilitar estas prácticas recomendadas sin que suponga una carga para los administradores.

Este tutorial muestra cómo conceder acceso a los usuarios a los servicios de AWS, pero solo cuando inicien sesión con MFA. Si no se han iniciado sesión con un dispositivo MFA, entonces los usuarios no pueden acceder a otros servicios.

Este flujo de trabajo incluye tres pasos básicos.

Paso 1: crear una política para que se cumpla el inicio de sesión de MFA

Cree una política administrada por el cliente que prohíba todas las acciones excepto las pocas acciones de IAM. Estas excepciones permiten a un usuario cambiar sus propias credenciales y administrar sus dispositivos MFA en la página Credenciales de seguridad. Para obtener más información sobre cómo obtener acceso a esta página, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

Paso 2: asociar políticas a su grupo de usuarios de prueba

Crear un grupo de usuarios cuyos miembros tengan acceso total a todas las acciones de Amazon EC2 cuando inician sesión con MFA. Para crear dicho grupo de usuarios, asocie la política administrada por AWS denominada `AmazonEC2FullAccess` y la política administrada del cliente que ha creado en el primer paso.

Paso 3: Probar el acceso de usuario

Inicie sesión como usuario de prueba para verificar que el acceso a Amazon EC2 está bloqueado hasta que el usuario crea un dispositivo MFA. A continuación, el usuario puede iniciar sesión con ese dispositivo.

Requisitos previos

Para realizar los pasos en este tutorial, deberá disponer de lo siguiente:

- Una Cuenta de AWS en la que puede iniciar sesión como usuario de IAM con permisos administrativos.
- Su número de ID de cuenta, que debe especificar en la política en el paso 1.

Para encontrar su número de ID de la cuenta, en la barra de navegación situada en la parte superior de la página, elija Support (Soporte) y, a continuación, elija Support Center (Centro de soporte). Puede encontrar el ID de cuenta en el menú Support (Soporte) de esta página.

- Un dispositivo MFA virtual (basado en software), una clave de seguridad FIDO o un dispositivo MFA basado en hardware.
- Un usuario de prueba de IAM que sea miembro de un grupo de usuarios de la siguiente manera:

Creación de usuario		Crear y configurar cuentas de grupo de usuarios		
Nombre de usuario	Otras instrucciones	Nombre del grupo de usuarios	Añadir usuario como miembro	Otras instrucciones
MFAUser	Elija solo la opción Habilitar acceso a la consola: opcional y asigne una contraseña.	EC2MFA	MFAUser	NO adjunte políticas o ni conceda permisos a este grupo de usuarios de ninguna otra manera.

Paso 1: crear una política para que se cumpla el inicio de sesión de MFA

Puede comenzar creando una política administrada por el cliente de IAM que deniegue todos los permisos, salvo los necesarios para los usuarios de IAM para que administren sus propias credenciales y dispositivos MFA.

1. Inicie sesión en Management Console de AWS como usuario con credenciales de administrador. Para cumplir con las prácticas recomendadas de IAM, no inicie sesión con las credenciales de usuario Usuario raíz de la cuenta de AWS.

   > **Important**
   >
   > Las prácticas recomendadas de IAM sugieren que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales, en lugar de utilizar usuarios de IAM con credenciales a largo plazo.

2. Abra la consola de IAM en https://console.aws.amazon.com/iam/.

3. En el panel de navegación, seleccione Policies (Políticas) y, a continuación, seleccione Create policy (Crear política).
4. Seleccione la pestaña JSON y copie el texto del siguiente documento de política JSON: AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.
5. Pegue el texto de la política en el cuadro de texto JSON. Resuelva las advertencias de seguridad, errores o advertencias generales surgidos durante la validación de política y luego seleccione Siguiente.

   > **Note**
   >
   > Puede alternar entre las opciones Editor visual y JSON en todo momento. Sin embargo, la política anterior, que incluye el elemento `NotAction`, no se admite en el editor visual. Para esta política, verá una notificación en la pestaña Visual editor (Editor visual). Vuelva a JSON para seguir trabajando con esta política.
   > Este ejemplo de política no permite a los usuarios restablecer una contraseña al iniciar sesión en la AWS Management Console por primera vez. Recomendamos que no conceda permisos a los nuevos usuarios hasta después de que inicien sesión y restablezcan su contraseña.

6. En la página Revisar y crear, escriba **`Force_MFA`** como nombre de la política. Para la descripción de la política, escriba **`This policy allows users to manage their own passwords and MFA devices but nothing else unless they authenticate with MFA.`**. En el área Etiquetas, puede agregar opcionalmente pares clave-valor de etiquetas a la política administrada por el cliente. Revise los permisos concedidos por la política y, a continuación, seleccione Crear política para guardar su trabajo.

   La nueva política aparece en la lista de las políticas administradas y está lista para asociar.

Paso 2: asociar políticas a su grupo de usuarios de prueba

A continuación, se conectan dos políticas al grupo de usuarios de IAM de prueba, que se utilizarán para conceder los permisos protegidos por MFA.

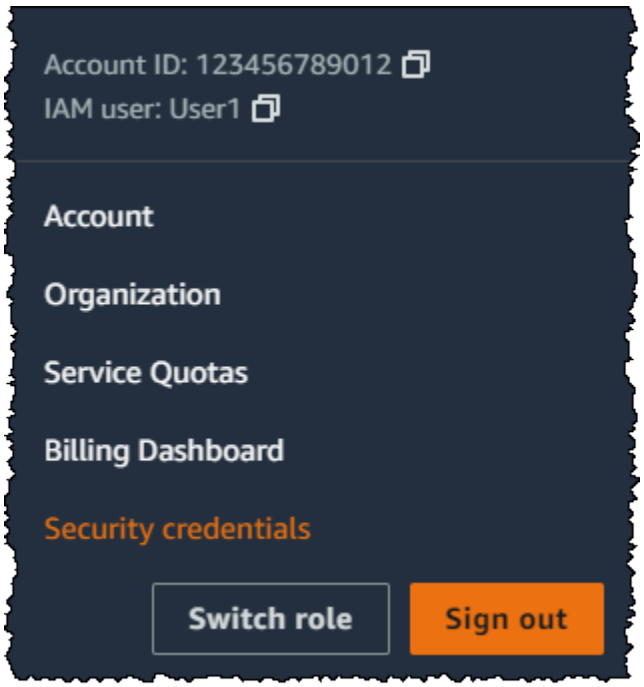
1. En el panel de navegación, elija User groups (Grupos de usuarios).

2. En el cuadro de búsqueda, escriba **`EC2MFA`** y, a continuación, elija el nombre del grupo en la lista (no la casilla de verificación).
3. Elija la pestaña Permisos, elija Agregar permisos y luego, Asociar políticas.
4. En la página Attach permission policies to EC2MFA group (Adjuntar políticas de permisos al grupo EC2MFA), en el campo de búsqueda, escriba **`EC2Full`**. Luego, seleccione la casilla de verificación situada junto a AmazonEC2FullAccess en la lista. No guarde aún los cambios.
5. En el cuadro de búsqueda, escriba **`Force`** y, a continuación, seleccione la casilla de verificación junto a Force_MFA en la lista.
6. Seleccione Asociar políticas.

Paso 3: Probar el acceso de usuario

En esta parte del tutorial, inicie sesión como usuario de prueba y verifique que la política funciona según lo previsto.

1. Inicie sesión en su Cuenta de AWS como **`MFAUser`** con la contraseña que haya asignado en la sección anterior. Use la URL: `https://<alias or account ID number>.signin.aws.amazon.com/console`
2. Elija EC2 para abrir la consola de Amazon EC2 y comprobar que el usuario no tiene permisos para realizar ninguna actividad.
3. En la esquina superior derecha de la barra de navegación, elija el nombre de usuario `MFAUser` y, a continuación, My Security Credentials (Mis credenciales de seguridad).



4. Ahora agregue un dispositivo MFA. En la sección Multi-Factor Authentication (MFA), elija Assign MFA device (Asignar dispositivo MFA).

> **Note**
>
> Es posible que reciba un error que indica que no está autorizado a realizar `iam:DeleteVirtualMFADevice`. Esto podría ocurrir si alguien anteriormente comenzó la asignación de un dispositivo MFA virtual este usuario y canceló el proceso. Para continuar, usted u otro administrador debe eliminar el dispositivo MFA virtual existente sin asignar del usuario. Para obtener más información, consulte No tengo autorización para realizar la operación iam:DeleteVirtualMFADevice.

5. En este tutorial, utilizamos un dispositivo MFA virtual (basado en software), como la aplicación Google Authenticator en un teléfono móvil. Elija Authenticator app (Aplicación del autenticador) y, a continuación, haga clic en Next (Siguiente).

   IAM generará y mostrará la información de configuración del dispositivo MFA virtual, incluido un gráfico de código QR. El gráfico es una representación de la clave de configuración secreta que se puede introducir manualmente en dispositivos que no admiten códigos QR.

6. Abra su aplicación de MFA virtual. (Para ver una lista de las aplicaciones que puede utilizar para alojar dispositivos MFA virtuales, consulte Aplicaciones MFA virtuales). Si la aplicación de MFA

virtual admite varias cuentas (varios dispositivos de MFA virtuales), elija la opción para crear una nueva cuenta (un nuevo dispositivo MFA virtual).

7. Determine si la aplicación MFA admite códigos QR y, a continuación, lleve a cabo alguna de las siguientes operaciones:

   - Desde el asistente, seleccione Show QR code (Mostrar código QR). A continuación, utilice la aplicación para analizar el código QR. Por ejemplo, puede elegir el icono de la cámara o una opción similar a Scan code (Escanear código) y, a continuación, utilizar la cámara del dispositivo para escanear el código.
   - En el asistente Set up device (Configurar el dispositivo), elija Show secret key (Mostrar clave secreta) y, a continuación, escriba la clave secreta en su aplicación MFA.

   Cuando haya terminado, el dispositivo MFA virtual comenzará a generar contraseñas de uso único.

8. En el asistente Set up device (Configurar el dispositivo), en el cuadro Enter the code from your authenticator app (Introducir el código de la aplicación de autenticación), escriba la contraseña de uso único que aparece actualmente en el dispositivo MFA virtual. Elija Register MFA (Registrar MFA).

   > **Important**
   >
   > Envíe su solicitud inmediatamente después de generar el código. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA está correctamente asociado al usuario. Sin embargo, el dispositivo MFA no está sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

   El dispositivo MFA virtual ya está listo para utilizarlo con AWS.

9. Cierre la sesión de la consola y, a continuación, vuelva a iniciar sesión en **`MFAUser`**. En esta ocasión AWS le solicita un código de MFA desde su teléfono. Al hacerlo, escriba el código en la casilla y, a continuación, seleccione Submit (Enviar).
10. Elija EC2 para abrir de nuevo la consola de Amazon EC2. Observe que esta vez puede ver toda la información y realizar cualquier acción que desee. Si va a cualquier otra consola como

este usuario, verá mensajes de acceso denegado. El motivo es que las políticas de este tutorial conceden acceso únicamente a Amazon EC2.

Recursos relacionados

Para obtener más información, consulte los siguientes temas:

- Uso de autenticación multifactor (MFA) en AWS
- Habilitación de dispositivos MFA para usuarios en AWS
- Uso de dispositivos MFA con la página de inicio de sesión de IAM

Identidades de IAM (usuarios, grupos de usuarios y roles)

> **Tip**
>
> ¿Tiene problemas para iniciar sesión en AWS? Asegúrese de que está en la página de inicio de sesión correcta.
>
> - Para iniciar sesión como el Usuario raíz de la cuenta de AWS (propietario de la cuenta), utilice las credenciales que configuró cuando creó la Cuenta de AWS.
> - Para iniciar sesión como usuario de IAM, utilice las credenciales que le brindó el administrador de su cuenta para iniciar sesión en AWS.
> - Para iniciar sesión con el usuario del Centro de identidades de IAM, utilice la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario de IAM Identity Center.
>
>   Para obtener ayuda para iniciar sesión con un usuario del IAM Identity Center, consulte Iniciar sesión en el portal de acceso de AWS en la Guía del usuario de AWS Sign-In.
>
> Para ver los tutoriales de inicio de sesión, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In.

> **Note**
>
> Si necesita solicitar soporte técnico, no utilice el enlace de Comentarios de esta página. Los comentarios que ingrese los recibe el equipo de documentación de AWS, no AWS Support. En su lugar, seleccione el enlace Contáctenos en la parte superior de esta página. Allí encontrará enlaces a recursos que le ayudarán a obtener el apoyo que necesita.

El Usuario raíz de la cuenta de AWS o un usuario administrativo de la cuenta puede crear identidades de IAM. Una identidad de IAM proporciona acceso a una Cuenta de AWS. Un grupo de usuarios de IAM es una colección de usuarios de IAM administrados como una unidad. Una identidad de IAM representa a un usuario o carga de trabajo programática, se puede autenticar y, a continuación, autorizar para realizar acciones en AWS. Cada identidad de IAM se puede asociar a

una o varias políticas. Las políticas determinan qué acciones puede realizar un usuario, un rol o un miembro de un grupo de usuarios, en qué recursos de AWS y en qué condiciones.

Usuario raíz de la Cuenta de AWS

Cuando crea por primera vez una Cuenta de AWS, se comienza con una identidad de inicio de sesión que tiene acceso completo a todos los Servicios de AWS y recursos de la cuenta. Esta identidad recibe el nombre de usuario raíz de la Cuenta de AWS, y se accede a ella iniciando sesión con la dirección de correo electrónico y la contraseña que utilizó para crear la cuenta.

> **Important**
>
> Recomendamos encarecidamente que no utilice el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para obtener la lista completa de tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz.

Usuarios de IAM

Un usuario de IAM es una identidad de la Cuenta de AWS que dispone de permisos específicos para una sola persona o aplicación. Siempre que sea posible, las prácticas recomendadas sugieren emplear credenciales temporales, en lugar de crear usuarios de IAM con credenciales a largo plazo como contraseñas y claves de acceso. Antes de crear claves de acceso, revise las alternativas a las claves de acceso a largo plazo. Si hay casos de uso específicos que requieren claves de acceso, le recomendamos actualizar las claves de acceso cuando sea necesario. Para obtener más información, consulte Actualizar las claves de acceso cuando sea necesario para casos de uso que requieren credenciales de larga duración. Para agregar usuarios de IAM a su Cuenta de AWS, consulte Creación de un usuario de IAM en su Cuenta de AWS.

> **Note**
>
> Como práctica recomendada de seguridad, le recomendamos que proporcione acceso a sus recursos mediante la federación de identidades en lugar de crear usuarios de IAM. Para obtener más información acerca de situaciones específicas en las que se requiere un usuario de IAM, consulte Cuándo crear un usuario de IAM (en lugar de un rol).

Grupos de usuarios de IAM

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puede iniciar sesión como grupo. Puede usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos de grandes conjuntos de usuarios. Por ejemplo, puedes tener un grupo llamado IAMPublishers y conceder a ese grupo los tipos de permisos que las cargas de trabajo de publicación suelen necesitar.

Roles de IAM

Un rol de IAM es una identidad de su Cuenta de AWS que dispone de permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una persona determinada. Puede asumir temporalmente un rol de IAM en la AWS Management Console cambiando de roles. Puede asumir un rol llamando a una operación de la AWS CLI o de la API de AWS, o utilizando una URL personalizada. Si necesita más información sobre los métodos de uso de los roles, consulte Uso de roles de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir permisos para este. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles para federación, consulte Creación de un rol para un proveedor de identidades de terceros en la Guía del usuario de IAM. Si utiliza IAM Identity Center, debe configurar un conjunto de permisos. IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué pueden acceder las identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulte Conjuntos de permisos en la Guía del usuario de AWS Single Sign-On.
- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. No obstante, con algunos Servicios de AWS se puede adjuntar una política directamente a un recurso (en lugar de utilizar un rol como representante). Para conocer la diferencia entre la utilización de roles y políticas basadas en recursos para el acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM.

- Acceso entre servicios: algunos Servicios de AWS utilizan características de otros Servicios de AWS. Por ejemplo, cuando realiza una llamada en un servicio, es común que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado a servicios.
  - Reenviar sesiones de acceso (FAS): cuando utiliza un rol o un usuario de IAM para llevar a cabo acciones en AWS, se le considera una entidad principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos de la entidad principal para llamar a un Servicio de AWS, combinados con el Servicio de AWS solicitante para realizar solicitudes a servicios posteriores. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS o recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.
  - Rol vinculado a servicio: un rol vinculado a servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.
- Aplicaciones que se ejecutan en Amazon EC2: puede utilizar un rol de IAM que le permita administrar credenciales temporales para las aplicaciones que se ejecutan en una instancia de EC2 y realizan solicitudes a AWS CLI o a la API de AWS. Es preferible hacerlo de este modo a almacenar claves de acceso en la instancia EC2. Para asignar un rol de AWS a una instancia de EC2 y ponerla a disposición de todas las aplicaciones, cree un perfil de instancia adjuntado a la instancia. Un perfil de instancia contiene el rol y permite a los programas que se ejecutan en la instancia EC2 obtener credenciales temporales. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2 en la Guía del usuario de IAM.

Credenciales temporales en IAM

Como práctica recomendada, utilice credenciales temporales tanto para usuarios humanos como para cargas de trabajo. Las credenciales temporales se utilizan principalmente con los roles de IAM, pero también tiene otros usos. Puede solicitar credenciales temporales que tienen un conjunto de permisos más restringido que el usuario de IAM estándar. Esto evita que lleve a cabo de forma no intencionada tareas no permitidas por las credenciales más restrictivas. Una ventaja de las credenciales temporales es que vencen después de un periodo de tiempo determinado. Usted controla la duración de la validez de las credenciales.

¿Cuándo se utilizan los usuarios de IAM Identity Center?

Se recomienda que todos los usuarios humanos utilicen IAM Identity Center para acceder a los recursos de AWS. IAM Identity Center permite mejoras significativas sobre el acceso a los recursos de AWS como usuario de IAM. IAM Identity Center proporciona:

- Un conjunto central de identidades y asignaciones
- Acceso a las cuentas de toda la organización de AWS
- Conexión con su proveedor de identidad actual
- Credenciales temporales
- Autenticación multifactor (MFA)
- Configuración de MFA de autoservicio para usuarios finales
- Aplicación administrativa del uso de la MFA
- Acceso único a todos los derechos de la Cuenta de AWS

Para más información, consulte ¿Qué es IAM Identity Center? en la Guía del usuario de AWS IAM Identity Center.

Cuándo crear un usuario de IAM (en lugar de un rol)

Le recomendamos que utilice únicamente usuarios de IAM para los casos de uso no admitidos por los usuarios federados. Estos son algunos de los casos de uso:

- Cargas de trabajo que no pueden utilizar roles de IAM: puede ejecutar una carga de trabajo desde una ubicación que necesite acceder a AWS. En algunas situaciones, no se pueden utilizar roles de

IAM para proporcionar credenciales temporales; por ejemplo, en el caso de los complementos de WordPress. En esas situaciones, utilice claves de acceso a largo plazo de usuarios de IAM para que la carga de trabajo se autentique en AWS.

- Clientes de externos de AWS: si utiliza herramientas que no admiten el acceso con IAM Identity Center, como clientes externos de AWS o proveedores que no están alojados en AWS, utilice claves de acceso a largo plazo de usuarios de IAM.
- Acceso a AWS CodeCommit: si utiliza CodeCommit para almacenar el código, puede emplea un usuario de IAM con claves SSH o credenciales específicas del servicio para que CodeCommit se autentique en los repositorios. Se recomienda hacer esto además de utilizar un usuario de IAM Identity Center para la autenticación normal. Los usuarios de IAM Identity Center son el personal que necesita acceso a sus Cuentas de AWS o a sus aplicaciones en la nube. Para dar acceso a los usuarios a los repositorios de CodeCommit sin configurar usuarios de IAM, puede configurar la utilidad git-remote-codecommit. Para obtener más información sobre IAM y CodeCommit, consulte Uso de IAM con CodeCommit: credenciales de Git, claves SSH y claves de acceso de AWS. Para obtener más información sobre cómo configurar la utilidad git-remote-codecommit, consulte Conexión a repositorios de AWS CodeCommit con credenciales rotativas en la Guía del usuario de AWS CodeCommit.
- Acceso a Amazon Keyspaces (para Apache Cassandra): en una situación en la que no pueda utilizar usuarios de IAM Identity Center, como por ejemplo, para probar la compatibilidad con Cassandra, puede utilizar un usuario de IAM con credenciales específicas del servicio para realizar la autenticación en Amazon Keyspaces. Los usuarios de IAM Identity Center son el personal que necesita acceso a sus Cuentas de AWS o a sus aplicaciones en la nube. También puede conectarse a Amazon Keyspaces con credenciales temporales. Para obtener más información, consulte Uso de credenciales temporales para conectarse a Amazon Keyspaces mediante un rol de IAM y el complemento SigV4 en la Guía para desarrolladores de Amazon Keyspaces (para Apache Cassandra).
- Acceso de emergencia: en caso de que no pueda acceder a su proveedor de identidad y deba realizar alguna acción en su Cuenta de AWS. Puede establecer usuarios de IAM de acceso de emergencia como parte de su plan de resiliencia. Recomendamos que las credenciales de usuario de emergencia estén estrictamente controladas y protegidas mediante autenticación multifactor (MFA).

Cuándo crear un rol de IAM (en lugar de un usuario)

Cree un rol de IAM cuando se encuentre en las siguientes situaciones:

Está por crear una aplicación que se ejecuta en una instancia de Amazon Elastic Compute Cloud (Amazon EC2) y dicha aplicación realiza solicitudes a AWS.

No cree ningún usuario de IAM ni transfiera las credenciales del usuario a la aplicación o incruste las credenciales en la aplicación. En cambio, cree un rol de IAM para adjuntarlo a la instancia EC2 para proporcionar credenciales de seguridad temporales a las aplicaciones que se ejecutan en la instancia. Cuando una aplicación utiliza estas credenciales en AWS, puede realizar todas las operaciones permitidas por las políticas asociadas al rol. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2.

Va a crear una aplicación que se ejecuta en un teléfono móvil y que realiza solicitudes a AWS.

No cree ningún usuario de IAM ni distribuya la clave de acceso del usuario con la aplicación. En cambio, utilice un proveedor de identidad, como Login with Amazon, Amazon Cognito, Facebook o Google, para autenticar a los usuarios y asignar los usuarios a un rol de IAM. La aplicación puede utilizar el rol para obtener credenciales de seguridad temporales que tengan los permisos especificados por las políticas asociadas al rol. Para obtener más información, consulte lo siguiente:

- Guía del usuario de Amazon Cognito
- Federación OIDC

Los usuarios de su compañía se autentican en la red de la compañía y quieren poder utilizar AWS sin necesidad de iniciar sesión de nuevo, es decir, que desea permitir a los usuarios federarse en AWS.

No cree usuarios de IAM. Configure una relación de federación entre el sistema de identidades de la compañía y AWS. Puede hacerlo de dos formas:

- Si el sistema de identidad de su compañía es compatible con SAML 2.0, puede establecer una relación de confianza entre su sistema de identidad y AWS. Para obtener más información, consulte Federación SAML 2.0.
- Cree y utilice un servidor proxy personalizado que convierta las identidades de los usuarios de la empresa en roles de IAM que proporcionen credenciales de seguridad temporales de AWS. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

Comparar las credenciales de Usuario raíz de la cuenta de AWS con las credenciales de usuario de IAM

El usuario raíz es el propietario de la cuenta y se crea cuando se crea la Cuenta de AWS. Otros tipos de usuarios (incluidos los usuarios de IAM) y los usuarios de AWS IAM Identity Center los crea el usuario raíz o un administrador de la cuenta. Todos los usuarios de AWS tienen credenciales de seguridad.

Credenciales de usuario raíz

Las credenciales del propietario de la cuenta permiten el acceso completo a todos los recursos en la cuenta. No puede utilizar las políticas de IAM para denegar al usuario raíz el acceso a los recursos de forma explícita. Solo puede usar una política de control de servicio (SCP) de AWS Organizations para limitar los permisos del usuario raíz de una cuenta miembro. Por ello, le recomendamos crear un usuario administrativo en IAM Identity Center para utilizarlo en las tareas diarias de AWS. A continuación, proteja las credenciales de usuario raíz y utilícelas para realizar solo las pocas tareas de administración de cuentas y servicios que requieren que inicie sesión como usuario raíz. Para ver la lista de esas tareas, consulte Tareas que requieren credenciales de usuario raíz. Para obtener información sobre cómo configurar un administrador para uso diario en IAM Identity Center, consulte Introducción en la Guía del usuario de IAM Identity Center.

Credenciales de IAM

Un usuario de IAM es una entidad que se crea en AWS para representar a la persona o servicio que utiliza el usuario de IAM para interactuar con recursos de AWS. Estos usuarios son identidades dentro de su Cuenta de AWS, que tienen permisos personalizados específicos. Por ejemplo, puede crear usuarios de IAM y darles permisos para crear un directorio en IAM Identity Center. Los usuarios de IAM disponen de credenciales a largo plazo que pueden utilizar para acceder a AWS mediante la AWS Management Console, o mediante el uso de la programación de la AWS CLI o las API de AWS. Para obtener instrucciones paso a paso sobre cómo los usuarios de IAM inician sesión en AWS Management Console, consulte Inicie sesión en la AWS Management Console como usuario de IAM en la Guía del usuario para iniciar sesión en AWS.

En general, le recomendamos que evite crear usuarios de IAM, porque tienen credenciales a largo plazo, como un nombre de usuario y una contraseña. En su lugar, solicite a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. Puede utilizar un proveedor de identidades para los usuarios humanos a fin de proporcionar acceso federado a las Cuentas de AWS asumiendo roles de IAM, que proporcionan credenciales temporales. Si desea administrar el

acceso de manera centralizada, le recomendamos utilizar IAM Identity Center para administrar el acceso a las cuentas y los permisos dentro de esas cuentas. Puede administrar las identidades de los usuarios con IAM Identity Center, o bien administrar los permisos de acceso para las identidades de los usuarios en IAM Identity Center de un proveedor de identidades externo. Para obtener más información, consulte ¿Qué es IAM Identity Center? en la Guía del usuario de IAM Identity Center.

Usuario raíz de la cuenta de AWS

Cuando se crea por primera vez una cuenta de Amazon Web Services (AWS), se comienza con una única identidad de inicio de sesión que tiene acceso completo a todos los servicios y recursos de AWS de la cuenta. Esta identidad recibe el nombre de usuario raíz de la cuenta de AWS y se accede a ella iniciando sesión con la dirección de email y la contraseña que utilizó para crear la cuenta.

> **Important**
>
> Se recomienda encarecidamente no utilizar el usuario raíz para las tareas cotidianas y seguir las prácticas recomendadas para el usuario raíz para la Cuenta de AWS. Proteja las credenciales del usuario raíz y utilícelas sólo para las tareas que el usuario raíz pueda realizar. Para obtener la lista completa de tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz.

En las siguientes secciones, se detallan las tareas de administración asociadas al usuario raíz.

Tareas

- Habilitación de un dispositivo MFA virtual para su Usuario raíz de la cuenta de AWS (consola)
- Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola)
- Habilitación de una clave de seguridad FIDO para el usuario raíz de la Cuenta de AWS (consola)
- Cambiar la contraseña para Usuario raíz de la cuenta de AWS
- Restablecimiento de una contraseña de usuario raíz perdida u olvidada
- Creación de claves de acceso para el usuario raíz
- Eliminación de claves de acceso para el usuario raíz
- Tareas que requieren credenciales de usuario raíz
- Solución de problemas con el usuario raíz
- Información relacionada

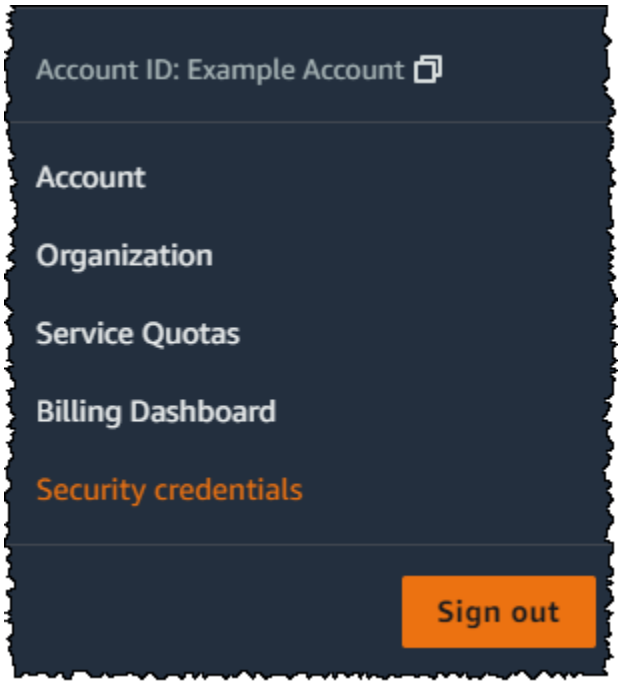
Habilitación de un dispositivo MFA virtual para su Usuario raíz de la cuenta de AWS (consola)

Puede utilizar AWS Management Console para configurar y habilitar un dispositivo MFA virtual para su usuario raíz. Para habilitar dispositivos MFA para la Cuenta de AWS, debe haber iniciado sesión en AWS con las credenciales de usuario raíz.

Antes de habilitar MFA para su usuario raíz, revise la configuración de la cuenta y la información de contacto para asegurarse de que tiene acceso al correo electrónico y al número de teléfono. Si su dispositivo MFA se pierde, se lo roban o no funciona, puede iniciar sesión como usuario raíz mediante la verificación de su identidad con ese correo electrónico y número de teléfono. Para obtener información acerca de cómo iniciar sesión con estos factores de autenticación alternativos, consulte ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?.

Para configurar y habilitar un dispositivo MFA virtual para utilizarlo con su usuario raíz (consola)

1. Inicie sesión en la AWS Management Console.
2. En la parte derecha de la barra de navegación, elija su nombre de cuenta y seleccione Security Credentials (Credenciales de seguridad). Si es necesario, elija Continue to Security Credentials (Seguir en Credenciales de seguridad).



3. En la sección Multi-Factor Authentication (MFA) (Autenticación multifactor [MFA]), elija Assign MFA device (Asignar dispositivo MFA).

4. En el asistente, escriba un Nombre de dispositivo, elija Aplicación del autenticador y luego, Siguiente.

   IAM generará y mostrará la información de configuración del dispositivo MFA virtual, incluido un gráfico de código QR. El gráfico es una representación de la clave de configuración secreta que se puede introducir manualmente en dispositivos que no admiten códigos QR.

5. Abra la aplicación de MFA virtual en el dispositivo.

   Si la aplicación de MFA virtual admite varios dispositivos o cuentas de MFA, elija la opción para crear un nuevo dispositivo o cuenta de MFA virtual.

6. La forma más sencilla de configurar la aplicación consiste en utilizar la aplicación para escanear el código QR. Si no puede analizar el código, puede escribir la información de configuración manualmente. El código QR y la clave de configuración secreta generada por IAM están vinculados a su Cuenta de AWS y no se pueden utilizar con otra cuenta. Sin embargo, se pueden volver a utilizar para configurar un dispositivo MFA nuevo para su cuenta en caso de que pierda el acceso al dispositivo MFA original.

   - En el asistente, para utilizar el código QR para configurar el dispositivo MFA virtual, elija Show QR code (Mostrar código QR). A continuación, siga las instrucciones de la aplicación para escanear el código. Por ejemplo, puede que tenga que elegir el icono de la cámara o un comando similar a Scan account barcode (Escanear código de barras) y, a continuación, utilizar la cámara del dispositivo para analizar el código QR.
   - En el asistente Set up device (Configurar el dispositivo), elija Show secret key (Mostrar clave secreta) y, a continuación, escriba la clave secreta en su aplicación MFA.

   Important

   Haga una copia de seguridad protegida del código QR o la clave de configuración secreta, o asegúrese de habilitar varios dispositivos MFA para su cuenta. Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Un dispositivo MFA virtual puede dejar de estar disponible si, por ejemplo, pierde el smartphone donde se aloja el dispositivo MFA virtual. Si eso ocurre y no puede iniciar sesión en su cuenta sin dispositivos de MFA adicionales asociados al usuario o incluso mediante Recuperación de un dispositivo MFA de usuario raíz, no podrá iniciar sesión en

su cuenta y tendrá que contactarse con el servicio de atención al cliente para eliminar la protección de MFA de la cuenta.

El dispositivo comienza a generar números de seis dígitos.

7. En el asistente, en la casilla MFA code 1 (Código MFA 1), escriba la contraseña de un solo uso que aparece actualmente en el dispositivo MFA virtual. Espere hasta 30 segundos a que el dispositivo genere una nueva contraseña de uso único. A continuación, escriba la otra contraseña de uso único en el cuadro MFA code 2 (Código MFA 2). Elija Add MFA (Agregar MFA).

> **Important**
>
> Envíe su solicitud inmediatamente después de generar el código. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

El dispositivo ya está listo para utilizarlo con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola)

Puede configurar y habilitar un dispositivo MFA virtual para el usuario raíz únicamente desde la AWS Management Console, no desde la AWS CLI ni la API de AWS.

de Si su dispositivo MFA se ha perdido, ha sido robado o no funciona, puede iniciar sesión con otros factores de autenticación. Si no puede iniciar sesión con su dispositivo MFA, puede hacerlo verificando su identidad con el correo electrónico y teléfono que están registrados en su cuenta. Antes de habilitar MFA para su usuario raíz, revise la configuración de la cuenta y la información de contacto para asegurarse de que tiene acceso al correo electrónico y al número de teléfono. Para obtener más información acerca de cómo iniciar sesión con factores de autenticación alternativos,

consulte ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?. Para deshabilitar esta característica, póngase en contacto con AWS Support.

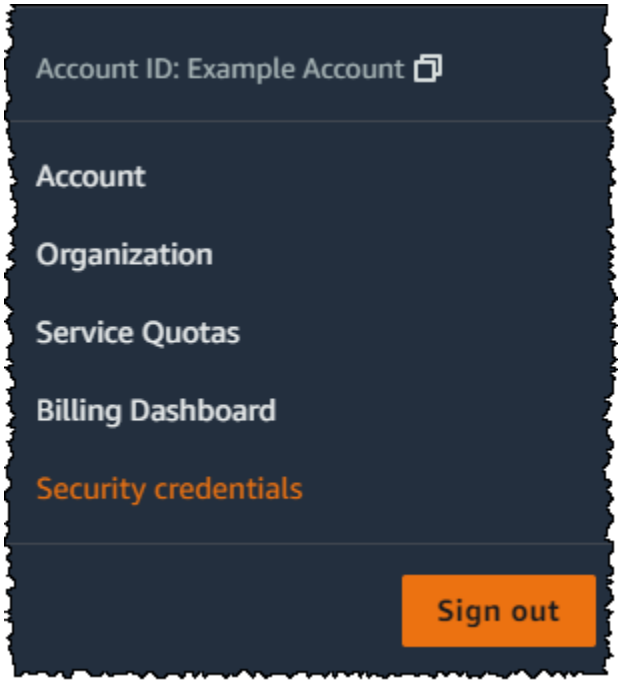
> **Note**
>
> Puede ver texto diferente, como, por ejemplo, Iniciar sesión mediante MFA y Solución de problemas con el dispositivo de autenticación. Sin embargo, se proporcionan las mismas características. En cualquier caso, si no puede verificar la dirección de correo electrónico y el número de teléfono de su cuenta mediante factores de autenticación alternativos, póngase en contacto con AWS Support para desactivar la configuración de MFA.

Para habilitar el dispositivo de MFA para su usuario raíz (consola)

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   > **Note**
   >
   > Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En la parte derecha de la barra de navegación, elija su nombre de cuenta y, a continuación, Security credentials (Credenciales de seguridad). Si es necesario, elija Continue to Security Credentials (Seguir en Credenciales de seguridad).



3. Expanda la sección Autenticación multifactor (MFA).
4. Elija Assign MFA device (Asignar dispositivo MFA).
5. En el asistente, escriba un Device name (Nombre del dispositivo), elija Hardware TOTP token (Token TOTP de hardware) y, a continuación, elija Next (Siguiente).
6. En el campo Serial number (Número de serie), escriba el número de serie que se encuentra en la parte posterior del dispositivo MFA.
7. En el cuadro Código MFA 1, escriba el número de seis dígitos que se encuentra en el dispositivo MFA. Es posible que tenga que pulsar el botón de la parte anterior del dispositivo para mostrar el número.

8. Espere 30 segundos a que el dispositivo actualice el código y, a continuación, escriba el número de seis dígitos siguiente en el cuadro Código MFA 2. Es posible que tenga que volver a pulsar el botón de la parte anterior del dispositivo para mostrar el otro número.
9. Elija Add MFA (Agregar MFA). Ahora el dispositivo MFA está asociado a Cuenta de AWS.

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos de autenticación. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

La siguiente vez que utilice sus credenciales de usuario raíz para iniciar sesión, debe escribir un código del dispositivo de MFA.

Habilitación de una clave de seguridad FIDO para el usuario raíz de la Cuenta de AWS (consola)

Puede configurar y habilitar un dispositivo MFA virtual para su usuario raíz solo desde la AWS Management Console, no desde la AWS CLI ni la API de AWS.

Si pierde su clave de seguridad FIDO, se la roban o no funciona, puede iniciar sesión con otro dispositivo MFA registrado con el mismo Usuario raíz de la cuenta de AWS. Si solo tiene un dispositivo MFA registrado, puede iniciar sesión con factores de identificación alternativos. Para obtener más información acerca de cómo iniciar sesión con factores de autenticación alternativos, consulte ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?. Para deshabilitar esta característica, póngase en contacto con AWS Support.

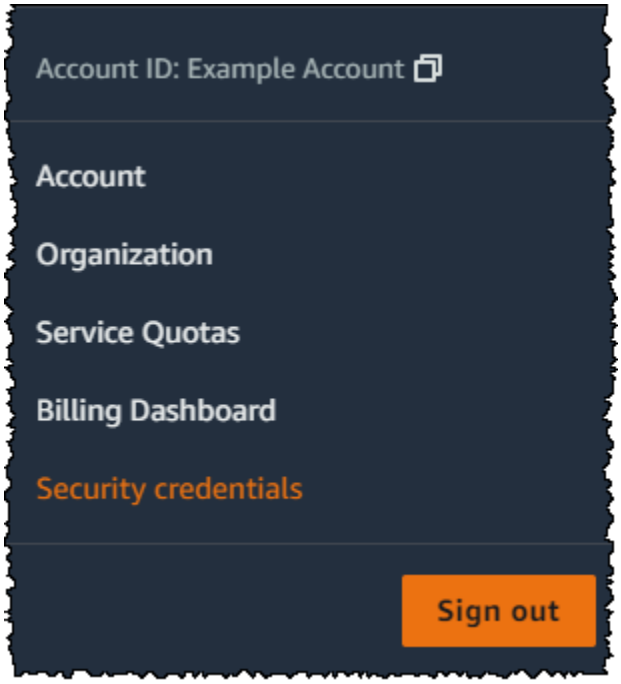
> **Note**
>
> No debe elegir ninguna de las opciones disponibles en la ventana emergente de Google Chrome que le pide verificar su identidad con amazon.com. Solo tiene que tocar la clave de seguridad.

Para habilitar la clave FIDO para su usuario raíz (consola)

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

> **Note**
>
> Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En la parte derecha de la barra de navegación, elija su nombre de cuenta y, a continuación, Credenciales de seguridad. Si es necesario, elija Seguir en Credenciales de seguridad.



3. Expanda la sección Autenticación multifactor (MFA).
4. Elija Asignar dispositivo MFA.
5. En el asistente, escriba un Nombre del dispositivo, seleccione Clave de seguridad y, a continuación, Siguiente.
6. Inserte la clave de seguridad FIDO en el puerto USB de su ordenador.

7. Pulse la clave de seguridad FIDO.

La clave de seguridad FIDO está lista para utilizarse con AWS. La próxima vez que utilice sus credenciales de usuario raíz para iniciar sesión, deberá tocar su clave de seguridad FIDO para completar el proceso de inicio de sesión.

Para obtener ayuda sobre cómo solucionar problemas con su llave de seguridad FIDO, consulte Solución de problemas con claves de seguridad FIDO.

Cambiar la contraseña para Usuario raíz de la cuenta de AWS

Puede cambiar la dirección de correo electrónico y la contraseña en la página Credenciales de seguridad o en la página Cuenta. También puede elegir Forgot password? (¿Ha olvidado la contraseña?) en la página de inicio de sesión de AWS para restablecer la contraseña.

Para cambiar la contraseña del usuario raíz, debe iniciar sesión como Usuario raíz de la cuenta de AWS y no como un usuario de IAM. Para obtener información sobre cómo restablecer la contraseña de usuario raíz olvidada, consulte Restablecimiento de una contraseña de usuario raíz perdida u olvidada.

Para proteger la contraseña, es importante seguir estas prácticas recomendadas:

- Cambie la contraseña periódicamente.
- Mantenga la contraseña en secreto, ya que cualquier persona que la conozca podrá acceder a su cuenta.
- Utiliza una contraseña diferente de AWS a la utilizada en otros sitios.
- Evite utilizar contraseñas que sean fáciles de adivinar. Entre estas se incluyen contraseñas tales como `secret`, `password`, `amazon` o `123456`. Además, evite usar palabras del diccionario, su nombre, dirección de correo electrónico u otra información personal que pueda obtenerse fácilmente.

AWS Management Console

Para cambiar la contraseña para el usuario raíz

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - Debe iniciar sesión como usuario raíz de la Cuenta de AWS, lo cual no requiere permisos adicionales de AWS Identity and Access Management (IAM). No puede realizar estos pasos como usuario o rol de IAM.

1. Utilice la dirección de correo electrónico y contraseña de su Cuenta de AWS para iniciar sesión en AWS Management Console como su Usuario raíz de la cuenta de AWS.
2. En la esquina superior derecha de la consola, elija el nombre o número de cuenta y, a continuación, seleccione Cuenta.
3. En la página Cuenta, junto a Configuración de la cuenta, elija Editar. Se le solicitará que vuelva a autenticarse por motivos de seguridad.

   > **Note**
   >
   > Si no ve la opción Editar, es probable que no haya iniciado sesión como el usuario raíz de la cuenta. No puede modificar la configuración de la cuenta si ha iniciado sesión como usuario o rol de IAM.

4. En la página Actualizar la configuración de la cuenta, en Contraseña, seleccione Editar.
5. En la página Actualizar contraseña, complete los campos Contraseña actual, Contraseña nueva y Confirmar contraseña nueva.

   > **Important**
   >
   > Asegúrese de elegir una contraseña segura. Aunque puede definir una política de contraseñas de cuentas para los usuarios de IAM, dicha política no se aplica al usuario raíz.

AWS exige que la contraseña cumpla con las siguientes condiciones:

- Debe tener 8 caracteres como mínimo y 128 como máximo.
- Debe incluir, como mínimo, tres de estos tipos de caracteres combinados: mayúsculas, minúsculas, números y símbolos ! @ # $ % ^ & * () <> [] {} | _+-=.
- No debe ser idéntica al nombre de la Cuenta de AWS ni a la dirección de correo electrónico.

> **Note**
>
> AWS es el despliegue de mejoras en el proceso de inicio de sesión. Una de esas mejoras consiste en implementar una política de contraseñas más segura para su cuenta. Si AWS ha actualizado la cuenta, usted debe cumplir la política de contraseñas anterior. Si AWS aún no actualizó la cuenta, AWS aún no aplica esta política. Sin embargo, le recomendamos encarecidamente seguir las pautas indicadas en esta para que la contraseña sea más segura.

6. Elija Guardar cambios.

AWS CLI or AWS SDK

Esta tarea no es compatible con la AWS CLI o con una operación de API de uno de los AWS SDK. Solamente puede realizar esta tarea mediante la AWS Management Console.

Restablecimiento de una contraseña de usuario raíz perdida u olvidada

Cuando creó su Cuenta de AWS por primera vez, proporcionó una dirección de correo electrónico y contraseña. Estas son sus credenciales de Usuario raíz de la cuenta de AWS. Si olvida su contraseña de usuario raíz, puede restablecer la contraseña desde la AWS Management Console.

Para restablecer la contraseña de su usuario raíz:

1. Utilice la dirección de correo electrónico de la cuenta de Cuenta de AWS para iniciar sesión en la AWS Management Console como usuario raíz y, a continuación, elija Next (Siguiente).

> **Note**
>
> Si ha iniciado sesión en la AWS Management Console con las credenciales de usuario de IAM, debe cerrar la sesión para poder restablecer la contraseña de usuario raíz. Si aparece la página de inicio de sesión de usuario de IAM específica de la cuenta, elija Iniciar sesión utilizando las credenciales de la cuenta raíz cerca de la parte inferior de la página. Si es necesario, proporcione la dirección de correo electrónico de la cuenta y elija Siguiente para acceder a la página Inicio de sesión de usuario raíz.

2. Elija Forgot your password? (¿Ha olvidado su contraseña?).

> **Note**
>
> Si es usuario de IAM, esta opción no se encuentra disponible. La opción ¿Ha olvidado su contraseña? solo se encuentra disponible para la cuenta de usuario raíz. Los usuarios de IAM deben solicitar a su administrador que restablezca una contraseña olvidada. Para obtener más información, consulte Olvidé la contraseña del usuario de IAM de mi cuenta de AWS. Si inicia sesión a través del Portal de acceso a AWS, consulte Restablecimiento de la contraseña de usuario en IAM Identity Center.

3. Proporcione la dirección de correo electrónico asociada a la cuenta. A continuación, proporcione el texto CAPTCHA y elija Continue (Continuar).
4. Busque un mensaje de Amazon Web Services en la bandeja del correo electrónico asociado con su cuenta de Cuenta de AWS. El correo electrónico procederá de una dirección que termina en `@verify.signin.aws`. Siga las indicaciones del correo electrónico. Si no ve el correo electrónico en su cuenta, compruebe la carpeta de spam. Si ya no tiene acceso al correo electrónico, consulte No tengo acceso al correo electrónico de mi cuenta de AWS en la Guía del usuario de AWS Sign-In.

Creación de claves de acceso para el usuario raíz

> **Warning**
>
> Le recomendamos encarecidamente no crear pares de claves de acceso para el usuario raíz. Dado que solo unas pocas tareas requieren el usuario raíz y, por lo general, las realiza con poca frecuencia, le recomendamos iniciar sesión en la AWS Management Console para

realizar las tareas de usuario raíz. Antes de crear claves de acceso, revise las alternativas a las claves de acceso a largo plazo.

Aunque no lo recomendamos, puede crear claves de acceso para el usuario raíz para poder ejecutar comandos en AWS Command Line Interface (AWS CLI) o utilizar las operaciones de la API desde uno de los AWS SDK con las credenciales de usuario raíz. Cuando crea una clave de acceso, se genera el ID de clave de acceso y la clave de acceso secreta como conjunto. Cuando se crea la clave de acceso, AWS permite ver y descargar la clave de acceso secreta que forma parte de la clave de acceso. Si no la descarga o la pierde, puede eliminar la clave de acceso y, a continuación, crear una nueva. Puede crear claves de acceso de usuario raíz con la consola, la AWS CLI o la API de AWS.

Una clave de acceso recién creada tiene estado activo; esto significa que se puede utilizar para efectuar llamadas de CLI y a la API. Puede asignar un máximo de dos claves de acceso al usuario raíz.

Las claves de acceso que no están en uso deben desactivarse. Una vez que una clave de acceso está inactiva, no se la puede utilizar para las llamadas a la API. Las llaves no activas cuentan igual para su límite. Puede crear o eliminar una clave de acceso en cualquier momento. Sin embargo, cuando se elimina una clave de acceso, desaparece para siempre y ya no se puede recuperar.

AWS Management Console

Para crear una clave de acceso para la Usuario raíz de la cuenta de AWS

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - Debe iniciar sesión como usuario raíz de la Cuenta de AWS, lo cual no requiere permisos adicionales de AWS Identity and Access Management (IAM). No puede realizar estos pasos como usuario o rol de IAM.

1. Utilice la dirección de correo electrónico y contraseña de su Cuenta de AWS para iniciar sesión en Introducción a AWS Management Console como su Usuario raíz de la cuenta de AWS.

2. En la esquina superior derecha de la consola, elija su nombre o número de cuenta y, a continuación, seleccione Credenciales de seguridad.
3. En la sección Access keys (Claves de acceso), haga clic en Create access key (Crear clave de acceso). Si esta opción no está disponible, significa que ya tiene el número máximo de claves de acceso. Debe eliminar una de las claves de acceso existentes antes de poder crear una nueva. Para obtener más información, consulte IAM Object Quotas.
4. En la página Alternativas a las claves de acceso de usuario raíz, revise las recomendaciones de seguridad. Para continuar, seleccione la casilla de verificación y, a continuación, elija Crear clave de acceso.
5. En la página Recuperar clave de acceso, se muestra el ID de clave de acceso.
6. En Clave de acceso secreta, elija Mostrar y, a continuación, copie el ID de clave de acceso y la clave secreta de la ventana del navegador, luego, pegue esos datos en un lugar seguro. También puede elegir Descargar archivo .csv, lo cual descargará un archivo denominado `rootkey.csv` que contiene el ID de clave de acceso y la clave secreta. Guarde el archivo en un lugar seguro.
7. Seleccione Done (Listo). Cuando ya no necesite la clave de acceso, le recomendamos eliminarla o, al menos, considerar desactivarla para que nadie pueda utilizarla de manera indebida.

AWS CLI & SDKs

Para crear una clave de acceso para el usuario raíz

> **Note**
>
> Para ejecutar el siguiente comando u operación de API como usuario raíz, ya debe tener un par de claves de acceso activo. Si no tiene ninguna clave de acceso, cree la primera con AWS Management Console. A continuación, puede usar las credenciales de la primera clave de acceso con la AWS CLI para crear la segunda clave de acceso o eliminar una clave de acceso.

- AWS CLI: aws iam create-access-key

  Example

  ```
  $ aws iam create-access-key
  ```

```
{
    "AccessKey": {
        "UserName": "MyUserName",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "Status": "Active",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "CreateDate": "2021-04-08T19:30:16+00:00"
    }
}
```

- API de AWS: CreateAccessKey en la Referencia de la API de IAM.

Eliminación de claves de acceso para el usuario raíz

Puede utilizar la AWS Management Console, la AWS CLI o la API de AWS para eliminar las claves de acceso de usuario raíz.

AWS Management Console

Para eliminar una clave de acceso para el usuario raíz

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - Debe iniciar sesión como usuario raíz de la Cuenta de AWS, lo cual no requiere permisos adicionales de AWS Identity and Access Management (IAM). No puede realizar estos pasos como usuario o rol de IAM.

1. Utilice la dirección de correo electrónico y contraseña de su Cuenta de AWS para iniciar sesión en Introducción a AWS Management Console como su Usuario raíz de la cuenta de AWS.
2. En la esquina superior derecha de la consola, elija su nombre o número de cuenta y, a continuación, seleccione Credenciales de seguridad.
3. En la sección Claves de acceso, seleccione la clave de acceso que desea eliminar y, a continuación, en Acciones, elija Eliminar.

> **Note**
>
> Como alternativa, puede desactivar una clave de acceso, en lugar de eliminarla de forma permanente. Esto le permitirá volver a utilizarla más tarde, sin tener que cambiar el ID de clave o la clave secreta. Mientras la clave está inactiva, todos los intentos de utilizarla en las solicitudes a la API de AWS, muestran el error acceso denegado.

4. En el cuadro de diálogo Eliminar <ID de clave de acceso>, seleccione Desactivar, introduzca el ID de clave de acceso para confirmar que desea eliminarla y, a continuación, seleccione Eliminar.

AWS CLI & SDKs

Para eliminar una clave de acceso para el usuario raíz

> **Permisos mínimos**
>
> Para realizar los siguientes pasos, debe tener al menos los siguientes permisos IAM:
>
> - Debe iniciar sesión como usuario raíz de la Cuenta de AWS, lo cual no requiere permisos adicionales de AWS Identity and Access Management (IAM). No puede realizar estos pasos como usuario o rol de IAM.

- AWS CLI: aws iam delete-access-key

  Example

  ```
  $ aws iam delete-access-key \
      --access-key-id AKIAIOSFODNN7EXAMPLE
  ```

  Este comando no genera ninguna salida si se realiza correctamente.

- API de AWS: DeleteAccessKey

Tareas que requieren credenciales de usuario raíz

> **Important**
>
> ¿Tiene problemas para iniciar sesión en AWS? Asegúrese de que está en la página de inicio de sesión de AWS correcta para su tipo de usuario. Si es el Usuario raíz de la cuenta de AWS (propietario de la cuenta), puede iniciar sesión en AWS con las credenciales que configuró cuando creó la Cuenta de AWS. Si es usuario de IAM, el administrador de su cuenta puede proporcionarle las credenciales que puede utilizar para iniciar sesión en AWS. Si necesita solicitar soporte técnico, no utilice el enlace de comentarios de esta página, ya que el formulario lo recibe el equipo de documentación de AWS, no AWS Support. En lugar de ello, en la página Contacte con nosotros, elija Todavía no es posible iniciar sesión en la cuenta de AWS y, a continuación, elija una de las opciones de asistencia disponibles.

Le recomendamos configurar un usuario administrativo en AWS IAM Identity Center para realizar las tareas diarias y acceder a los recursos de AWS. Sin embargo, las tareas que se enumeran a continuación únicamente se pueden realizar cuando se inicia sesión como usuario raíz de una cuenta.

Tareas de administración de cuentas

- Cambie la configuración de la cuenta. Esto incluye el nombre de la cuenta y la dirección de correo electrónico, así como la contraseña y las claves de acceso de usuario raíz. Otras configuraciones de la cuenta, como la información de contacto, la moneda de pago preferida y Regiones de AWS, no requieren credenciales de usuario raíz.
- Restaure los permisos de usuario de IAM. Si el único administrador de IAM revoca de manera accidental sus propios permisos, usted puede iniciar sesión como usuario raíz para editar políticas y restaurar esos permisos.
- Cierre su Cuenta de AWS.

  Para obtener más información, consulte los temas siguientes:

  - ¿Cómo asigno la propiedad de mi Cuenta de AWS a otra entidad?
  - ¿Cómo cierro mi Cuenta de AWS?
  - Cerrar una Cuenta de AWS independiente

Etiquetas de facturación.

- Active el acceso de IAM a la consola de Administración de facturación y costos.
- Algunas tareas de facturación están limitadas al usuario raíz. Para obtener más información, consulte la Guía del usuario AWS Billing sobre la gestión de una Cuenta de AWS.
- Consulte ciertas facturas de impuestos. Un usuario de IAM con el permiso aws-portal:ViewBilling puede ver y descargar facturas de IVA de AWS Europa, pero no de AWS Inc o Amazon Internet Services Private Limited (AISPL).

Tareas de AWS GovCloud (US)

- Regístrese en AWS GovCloud (US).
- Solicite las claves de acceso de usuario raíz de la cuenta AWS GovCloud (US) a AWS Support.
- Si una clave de AWS Key Management Service ya no puede administrarse, puede contactar con AWS Support como usuario raíz para recuperarla.

Tarea de Amazon EC2

- Se ha registrado como vendedor en el marketplace de instancias reservadas.

Tareas de Amazon Simple Storage Service

- Configure un bucket de Amazon S3 para habilitar MFA (autenticación multifactor).
- Edite o elimine una política de bucket de Amazon S3 que deniega todas las entidades principales.

Tareas de Amazon Simple Queue Service

- Edite o elimine una política de recursos de Amazon SQS que deniega todas las entidades principales.

Solución de problemas con el usuario raíz

Utilice la información que aquí se incluye para solucionar problemas relacionados con el usuario raíz de una Cuenta de AWS.

No puedo realizar las tareas que espero poder realizar si inicio sesión como usuario raíz de la cuenta.

Si no puede completar las tareas cuando inicia sesión como usuario raíz de la cuenta, es posible que su cuenta sea miembro de una organización en AWS Organizations. En ese caso y si el administrador de la organización utilizó una política de control de servicio (SCP) para limitar los permisos de la cuenta, todos los usuarios, incluido el usuario raíz, se verán afectados. Para obtener más información, consulte Políticas de control de servicios en la Guía del usuario de AWS Organizations.

He olvidado la contraseña del usuario raíz de mi Cuenta de AWS

Si usted es un usuario raíz y ha perdido u olvidado la contraseña de su Cuenta de AWS, puede restablecerla. Debe saber qué dirección de correo electrónico se utilizó para crear la Cuenta de AWS, como así también tener acceso a la cuenta de correo electrónico. Para obtener más información, consulte Restablecimiento de una contraseña de usuario raíz perdida u olvidada.

No tengo acceso al correo electrónico de mi Cuenta de AWS

Al crear una Cuenta de AWS, debe proporcionar una dirección de correo electrónico y una contraseña. Estas son las credenciales del Usuario raíz de la cuenta de AWS. Si no está seguro de la dirección de correo electrónico asociada a su Cuenta de AWS, busque los mensajes que `@signin.aws` o `@verify.signin.aws` envió a alguna dirección de correo electrónico de su organización y que se puedan haber utilizado para abrir la Cuenta de AWS.

Si conoce la dirección de correo electrónico pero ya no tiene acceso a dicho correo electrónico, intente recuperar el acceso al correo electrónico mediante una de las siguientes opciones:

- Si es el propietario del dominio de la dirección de correo electrónico, puede restaurar una dirección de correo electrónico eliminada. Alternativamente, puede configurar un catch-all para su cuenta de correo electrónico. El catch-all captura todos los mensajes enviados a direcciones de correo electrónico que ya no existen en el servidor de correo y los redirige a otra dirección de correo electrónico.
- Si la dirección de correo electrónico de la cuenta forma parte de su sistema de correo electrónico de la empresa, le recomendamos que se ponga en contacto con los administradores del sistema de TI. Estos administradores podrían ayudarle a recuperar el acceso al correo electrónico.

Si sigue sin poder iniciar sesión en su Cuenta de AWS, puede encontrar otras opciones de asistencia en Contáctenos.

Información relacionada

En los siguientes artículos se proporciona información adicional sobre cómo trabajar con el usuario raíz.

- ¿Cuáles son algunas de las prácticas recomendadas para proteger mi Cuenta de AWS y sus recursos?
- ¿Cómo puedo crear una regla de eventos de EventBridge que me notifique que se ha utilizado mi usuario raíz?
- Supervisar y notificar la actividad Usuario raíz de la cuenta de AWS
- Supervisar la actividad del usuario raíz de IAM

Usuarios de IAM

> **Important**
>
> Las prácticas recomendadas de IAM sugieren que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales, en lugar de utilizar usuarios de IAM con credenciales a largo plazo.

Un usuario de AWS Identity and Access Management (IAM) es una entidad que se crea en AWS. El usuario de IAM representa al usuario humano o carga de trabajo que utiliza el usuario de IAM para interactuar con AWS. Un usuario de AWS consta de un nombre y credenciales.

Un usuario de IAM con permisos de administrador no es lo mismo que el Usuario raíz de la cuenta de AWS. Para obtener más información sobre el usuario raíz, consulte Usuario raíz de la cuenta de AWS.

¿Cómo AWS identifica un usuario de IAM?

Al crear un usuario de IAM, IAM crea estas formas de identificar dicho usuario:

- Un “nombre fácil de recordar” para el usuario de IAM, que es el nombre que especificó al crear el usuario de IAM, como `Richard` o `Anaya`. Estos son los nombres que aparecen en la AWS Management Console.

- Un Amazon Resource Name (Nombre de recurso de Amazon [ARN]) para el usuario de IAM. Puede utilizar el ARN cuando necesite identificar de forma inequívoca el usuario de IAM en todo AWS. Por ejemplo, puede utilizar un ARN para especificar el usuario de IAM como entidad `Principal` en una política de IAM para un bucket de Amazon S3. Un ARN para un usuario de IAM podría ser el siguiente:

  `arn:aws:iam::account-ID-without-hyphens:user/Richard`

- Un identificador único para el usuario de IAM. Este ID solo se devuelve cuando utilice la API, Tools for Windows PowerShell o la AWS CLI para crear el usuario de IAM; no podrá verlo en la consola.

Para obtener más información sobre estos identificadores, consulte Identificadores de IAM.

Usuarios de IAM y credenciales

Puede obtener acceso a AWS de diferentes formas en función de las credenciales de usuario de IAM:

- Console password (Contraseña de la consola): una contraseña que el usuario de IAM puede escribir para iniciar sesión en sesiones interactivas, como la AWS Management Console. Al desactivar la contraseña (acceso a la consola) para un usuario de IAM, se impide que inicien sesión en la AWS Management Console con sus credenciales. No cambia sus permisos ni les impide acceder a la consola utilizando un rol asumido.
- Teclas de acceso: se utilizan para hacer llamadas programáticas a AWS. Sin embargo, hay alternativas más seguras que hay que tener en cuenta antes de crear claves de acceso para los usuarios de IAM. Para más información, consulte Consideraciones y alternativas para las claves de acceso a largo plazo en la Guía de la Referencia general de AWS. Si el usuario de IAM dispone de claves de acceso activas, estas seguirán funcionando y permitirán el acceso a través de la AWS CLI, Tools for Windows PowerShell, API de AWS o Console Mobile Application de AWS.
- Claves SSH para utilizar con CodeCommit: una clave SSH pública en formato OpenSSH que puede utilizarse para realizar la autenticación con CodeCommit.
- Certificados de servidor: los certificados SSL/TLS que puede utilizar para realizar la autenticación con algunos servicios de AWS. Le recomendamos que utilice AWS Certificate Manager (ACM) para aprovisionar, administrar e implementar los certificados de servidor. Utilice IAM solo cuando tenga que admitir conexiones HTTPS en una región que no sea compatible con ACM. Para saber qué regiones admiten ACM , consulte puntos finales y cuotas de AWS Certificate Manager en la Referencia general de AWS.

Puede elegir las credenciales adecuadas para el usuario de IAM. Cuando se utiliza la AWS Management Console para crear un usuario de IAM, debe elegir como mínimo la inclusión de una contraseña de consola o claves de acceso. De forma predeterminada, los nuevos usuarios de IAM creados mediante la AWS CLI o API de AWS no tienen credenciales de ningún tipo. Debe crear el tipo de credenciales para un usuario de IAM en función de las necesidades de su usuario.

Dispone de las opciones siguientes para administrar contraseñas, claves de acceso y dispositivos de autenticación multifactor (MFA):

- Administrar las contraseñas de los usuarios de IAM. Cree y cambie las contraseñas que permiten el acceso a la AWS Management Console. Establezca una política de contraseñas para aplicar un mínimo de complejidad a las contraseñas. Permita que los usuarios cambien sus contraseñas.
- Administrar las claves de acceso para los usuarios de IAM. Cree y actualice claves de acceso para acceder a los recursos de la cuenta mediante programación.
- Active la autenticación multifactor (MFA) para el usuario de IAM. Como práctica recomendada, le sugerimos exigir la autenticación multifactor para todos los usuarios de IAM de su cuenta. Con la MFA, los usuarios deben proporcionar dos formas de identificación: en primer lugar, proporcionar las credenciales que forman parte de su identidad del usuario (una contraseña o clave de acceso). Además, proporcionan un código numérico temporal que se genera en un dispositivo de hardware o mediante una aplicación en un smartphone o una tablet.
- Buscar contraseñas y claves de acceso sin utilizar. Cualquier persona que tenga una contraseña o clave de acceso para su cuenta o un usuario de IAM de su cuenta tendrá acceso a sus recursos de AWS. La práctica recomendada de seguridad es eliminar las contraseñas y claves de acceso cuando los usuarios ya no las necesiten.
- Descargar un informe de credenciales para la cuenta. Puede generar y descargar un informe de credenciales que contenga una lista de todos los usuarios de IAM de su cuenta y el estado de sus credenciales, tales como contraseñas, claves de acceso y dispositivos MFA. Para las contraseñas y claves de acceso, el informe de credenciales muestra cuándo se ha utilizado la contraseña o una clave de acceso por última vez.

Usuarios de IAM y permisos

De forma predeterminada, un nuevo usuario de IAM no tiene permisos para realizar ninguna actividad. El usuario no está autorizado a realizar ninguna operación de AWS ni a tener acceso a los recursos de AWS. Un beneficio de tener usuarios de IAM individuales es que puede asignar permisos específicos para cada uno. Es posible asignar permisos administrativos a unos usuarios

para que administren los recursos de AWS e incluso creen y administren otros usuarios de IAM. Sin embargo, en la mayoría de los casos, será necesario limitar los permisos de un usuario para que realice únicamente las tareas (acciones u operaciones de AWS) y utilice los recursos que necesita para su trabajo.

Imagine que tiene un usuario denominado Diego. Al crear el usuario de IAM `Diego` , debe crear una contraseña para dicho usuario y asociar permisos al usuario que le permitan lanzar una determinada instancia de Amazon EC2 y leer información (`GET`) de una tabla en una base de datos de Amazon RDS. Para obtener los procedimientos que indican cómo crear usuarios y concederles permisos y credenciales iniciales, consulte Creación de un usuario de IAM en su Cuenta de AWS. Para obtener los procedimientos que indican cómo cambiar los permisos de los usuarios existentes, consulte Cambio de los permisos de un usuario de IAM. Para obtener los procedimientos que indican cómo cambiar las claves de acceso y la contraseña de un usuario, consulte Administración de las contraseñas de usuarios en AWS y Administración de las claves de acceso de los usuarios de IAM.

También puede agregar un límite de permisos a los usuarios de IAM. Un límite de permisos es una característica avanzada que le permite utilizar políticas administradas de AWS para limitar los permisos máximos que puede conceder una política basada en identidad a un usuario de IAM o rol. Para obtener más información sobre los tipos de políticas y sus usos, consulte Políticas y permisos en IAM.

Usuarios de IAM y cuentas

Cada usuario de IAM está asociado a una única Cuenta de AWS. Dado que los usuarios de IAM se definen en la Cuenta de AWS, no necesitan tener un método de pago válido para AWS. Cualquier actividad de AWS realizada por los usuarios de IAM de la cuenta se factura en su cuenta.

El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Usuarios de IAM como cuentas de servicio

Un usuario de IAM es un recurso en IAM que tiene credenciales y permisos asociados. Un usuario de IAM puede representar una persona o una aplicación que utiliza sus credenciales para realizar solicitudes de AWS. Esto se suele conocer como cuenta de servicio. Si decide utilizar las credenciales a largo plazo de un usuario de IAM en su aplicación, no integre las claves de acceso directamente en el código de la aplicación. Los SDK de AWS y las AWS Command Line Interface le permiten colocar las claves de acceso en ubicaciones conocidas para que no tenga que mantenerlas

en el código. Para obtener más información, consulte Administración correcta de las claves de acceso de los usuarios de IAM en la Referencia general de AWS. De forma alternativa, y como práctica recomendada, puede utilizar credenciales de seguridad temporales (roles de IAM) en lugar de las claves de acceso a largo plazo.

Creación de un usuario de IAM en su Cuenta de AWS

Follow us on Twitter

> **Important**
>
> Las prácticas recomendadas de IAM sugieren que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales, en lugar de utilizar usuarios de IAM con credenciales a largo plazo.

> **Note**
>
> Si ha encontrado esta página porque está buscando información sobre Product Advertising API para vender productos de Amazon en su sitio web, consulte la documentación de la API de publicidad de productos 5.0.
> Si ha llegado a esta página desde la consola de IAM, es posible que su cuenta no incluya usuarios de IAM aunque haya iniciado sesión. Podría haber iniciado sesión como usuario Usuario raíz de la cuenta de AWS utilizando un rol o haber iniciado sesión con credenciales temporales. Para obtener más información acerca de las identidades de IAM consulte Identidades de IAM (usuarios, grupos de usuarios y roles).

El proceso para crear un usuario y habilitarlo para que realice tareas de trabajo consta de los pasos siguientes:

1. Cree el usuario en el AWS Management Console, el AWS CLI, Tools for Windows PowerShell o utilizar una Operación de la API AWS. Si crea el usuario en la AWS Management Console, los pasos 1 a 4 se realizan automáticamente de acuerdo con sus preferencias. Si crea los usuarios de forma programada, debe ejecutar individualmente cada uno de los pasos.
2. Cree las credenciales del usuario en función del tipo de acceso que este requiera:
   - Habilitar el acceso a la consola: opcional: si el usuario necesita acceder a la AWS Management Console, cree una contraseña para el usuario. Al desactivar el acceso a la consola para un

usuario, se impide que inicien sesión en la AWS Management Console con su nombre de usuario y contraseña. No cambia sus permisos ni les impide acceder a la consola utilizando un rol asumido.

> **Tip**
>
> Cree solo las credenciales que necesite el usuario. Por ejemplo, en el caso de un usuario que necesite obtener acceso únicamente mediante la AWS Management Console, no cree claves de acceso.

3. Dé al usuario permisos para realizar las tareas necesarias añadiendo el usuario a uno o varios grupos. También puede otorgar permisos asociando políticas de permisos directamente al usuario. No obstante, le recomendamos que en su lugar ponga a los usuarios en grupos y que administre los permisos mediante las políticas asociadas a dichos grupos. Asimismo, puede utilizar un límite de permisos para limitar los permisos que puede tener un usuario, aunque esto no es frecuente.
4. (Opcional) Añadir metadatos al usuario asociando etiquetas. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
5. Proporcione al usuario la información de inicio de sesión necesaria. Esto incluye la contraseña y la URL de la consola de la página de inicio de sesión de la cuenta en la que el usuario proporciona esas credenciales. Para obtener más información, consulte Cómo inician sesión los usuarios de IAM en AWS.
6. (Opcional) Configure la autenticación multifactor (MFA) para el usuario. MFA requiere que el usuario proporcione un código de un solo uso cada vez que inicia sesión en la AWS Management Console.
7. (Opcional) Conceda a los usuarios permisos para administrar sus propias credenciales de seguridad. (De forma predeterminada, los usuarios no tienen permisos para administrar sus propias credenciales). Para obtener más información, consulte Autorización para que los usuarios de IAM cambien sus contraseñas.

Para obtener información sobre los permisos que necesita para poder crear un usuario, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Temas

- Creación de usuarios de IAM (consola)
- Creación de usuarios de IAM (AWS CLI)
- Creación de usuarios de IAM (API de AWS)

Creación de usuarios de IAM (consola)

Puede utilizar la AWS Management Console para crear usuarios de IAM.

Para crear un usuario de IAM (consola)

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS.
2. En la página principal de la consola, seleccione el servicio de IAM.
3. En el panel de navegación, seleccione Usuarios y luego, elija Agregar usuarios.
4. En la página Especificar detalles del usuario, en Detalles del usuario, en Nombre del usuario, ingrese el nombre del usuario nuevo. Este es el nombre de inicio de sesión para AWS.

   > **Note**
   >
   > El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS. Los nombres de usuario pueden ser una combinación de un máximo de 64 letras, dígitos y los siguientes caracteres: más (+), igual (=), coma (,), punto (.), arroba (@), guion bajo (_) y guion (-). Los nombres deben ser únicos dentro de una cuenta. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear dos usuarios llamados TESTUSER y testuser. Cuando se utiliza un nombre de usuario en una política o como parte de un ARN, el nombre distingue entre mayúsculas y minúsculas. Cuando los clientes ven un nombre de usuario en la consola, por ejemplo, durante el proceso de inicio de sesión, el nombre del usuario no distingue entre mayúsculas y minúsculas.

5. Seleccione Proporcionar al usuario acceso a la AWS Management Console opcional. Se generan credenciales de inicio de sesión en la AWS Management Console para el usuario nuevo.

   Se le pregunta si proporciona acceso a la consola a una persona. Le recomendamos que cree usuarios en IAM Identity Center en lugar de en IAM.

   - Para comenzar a crear usuarios en IAM Identity Center, seleccione Especificar un usuario en Identity Center.

     Si no habilitó IAM Identity Center, cuando seleccione esta opción, accederá a la página de servicio de la consola para que pueda habilitar el servicio. Para obtener más información

acerca de este procedimiento, consulte https://docs.aws.amazon.com/singlesignon/latest/userguide/getting-started.html en la Guía del usuario de AWS IAM Identity Center.

Si habilitó IAM Identity Center, cuando seleccione esta opción, accederá a la página Especificar detalles del usuario en IAM Identity Center. Para obtener más información acerca de este procedimiento, consulte https://docs.aws.amazon.com/singlesignon/latest/userguide/addusers.html en la Guía del usuario de AWS IAM Identity Center.

- Si no puede utilizar IAM Identity Center, seleccione Quiero crear un usuario de IAM y continúe con este procedimiento.

a. En Contraseña de la consola, seleccione una de las siguientes opciones:

   - Contraseña generada de manera automática: el usuario obtiene una contraseña generada de forma aleatoria que cumple con la política de contraseñas de la cuenta. Si ingresa a la página Recuperar contraseña, puede ver o descargar la contraseña.
   - Contraseña personalizada: al usuario se le asigna la contraseña que usted ingresa en el cuadro.

b. (Opcional) Los usuarios deben crear una contraseña nueva la próxima vez que inicien sesión (recomendada) está seleccionada de forma predeterminada para garantizar que el usuario cambie la contraseña la primera vez que inicie sesión.

> **Note**
>
> Si un administrador habilita la configuración de política de contraseñas de cuentas Permitir a los usuarios cambiar su contraseña, esta casilla de verificación no hace nada. De lo contrario, se asociará automáticamente una política de AWS administrada denominada `IAMUserChangePassword` a los nuevos usuarios. La política les otorga permiso para cambiar sus propias contraseñas.

6. Seleccione Siguiente.
7. En la página Establecer permisos, especifique cómo quiere asignar permisos a este usuario. Seleccione una de las siguientes tres opciones:

   - Agregar usuario al grupo: seleccione esta opción si desea asignar el usuario a un grupo o a varios grupos que ya tienen políticas de permisos. IAM muestra una lista de los grupos de la cuenta, junto con sus políticas asociadas. Puede seleccionar un grupo o varios

grupos existentes, o seleccionar Crear grupo para crear un grupo nuevo. Para obtener más información, consulte Cambio de los permisos de un usuario de IAM.

- Copiar permisos: seleccione esta opción para copiar todas las suscripciones a grupos, las políticas administradas asociadas, las políticas insertadas integradas y los límites de permisos de un usuario existente al usuario nuevo. IAM muestra una lista de los usuarios de la cuenta. Seleccione un usuario cuyos permisos se ajusten más a las necesidades del usuario nuevo.
- Asociar políticas de manera directa: seleccione esta opción para ver una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de su cuenta. Seleccione las políticas que desea asociar al usuario o seleccione Crear política para abrir una pestaña nueva del navegador y crear una política nueva. Para obtener más información, consulte el paso 4 del procedimiento Crear políticas de IAM. Una vez creada la política, cierre la pestaña y vuelva a la pestaña original para agregar la política al usuario.

> **Tip**
>
> Siempre que sea posible, adjunte sus políticas a un grupo y, a continuación, haga a los usuarios miembros de los grupos apropiados.

8. (Opcional) Configure un límite de permisos. Esta es una característica avanzada.

   Abra la sección Límite de permisos y seleccione Utilizar un límite de permisos para controlar los permisos máximos. IAM muestra una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de la cuenta. Seleccione la política que desea utilizar para el límite de permisos o seleccione Crear política para abrir una pestaña nueva del navegador y crear una política nueva. Para obtener más información, consulte el paso 4 del procedimiento Crear políticas de IAM. Una vez creada la política, cierre la pestaña y vuelva a la pestaña original para seleccionar la política que va a utilizar para el límite de permisos.
9. Seleccione Siguiente.
10. (Opcional) En la página Revisar y crear, en Etiquetas, seleccione Agregar una etiqueta nueva para agregar metadatos al usuario mediante la asociación de etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
11. Revise todas las opciones que ha seleccionado hasta ahora. Cuando esté listo para continuar, seleccione Crear usuario.
12. En la página Recuperar contraseña, obtendrá la contraseña que se le asignó al usuario:

- Seleccione Mostrar junto a la contraseña para ver la contraseña del usuario y poder registrarla de forma manual.
- Seleccione Descargar .csv para descargar las credenciales de inicio de sesión del usuario como un archivo .csv que puede guardar en una ubicación segura.

13. Seleccione Instrucciones de inicio de sesión por correo electrónico. Su cliente de correo local se abrirá con un borrador que usted puede personalizar y enviar al usuario. La plantilla de correo electrónico contiene los detalles siguientes por cada usuario:

    - Nombre de usuario
    - URL de la página de inicio de sesión de la cuenta. Utilice el ejemplo siguiente y realice la sustitución con el número de ID o de alias de cuenta correcto:

    ```
    https://AWS-account-ID or alias.signin.aws.amazon.com/console
    ```

    > **Important**
    >
    > La contraseña del usuario no está incluida en el correo electrónico. Debe proporcionar la contraseña al usuario de una manera que cumpla con las directrices de seguridad de la organización.

14. Si el usuario también necesita claves de acceso, consulte Administración de las claves de acceso de los usuarios de IAM.

Creación de usuarios de IAM (AWS CLI)

Puede utilizar la AWS CLI para crear un usuario de IAM.

Para crear un usuario de IAM (AWS CLI)

1. Crear un usuario.

    - aws iam create-user

2. (Opcional) Dar al usuario acceso a la AWS Management Console. Esto requiere una contraseña. También debe dar a los usuarios la URL de la página de inicio de sesión de su cuenta.

    - aws iam create-login-profile

3. (Opcional) Dar al usuario acceso mediante programación. Esto requiere claves de acceso.

   - aws iam create-access-key
   - Tools for Windows PowerShell: New-IAMAccessKey
   - API de IAM: CreateAccessKey

   > **Important**
   >
   > Esta es la única oportunidad que tiene para ver o descargar las claves de acceso secretas, y debe proporcionar dicha información a los usuarios para que puedan utilizar la API de AWS. Guarde el nuevo ID de clave de acceso del usuario y la clave de acceso secreta en un lugar seguro. No volverá a tener acceso a la clave de acceso secreta después de este paso.

4. Añadir el usuario a uno o varios grupos. Los grupos que especifique deben tener políticas asociadas que concedan los permisos pertinentes para el usuario.

   - aws iam add-user-to-group

5. (Opcional) Asociar una política al usuario que defina los permisos del usuario. Nota: le recomendamos que administre los permisos de usuario añadiendo el usuario a un grupo y asociando una política al grupo en lugar de asociarla directamente a un usuario.

   - aws iam attach-user-policy

6. (Opcional) Añadir los atributos personalizados al usuario asociando etiquetas. Para obtener más información, consulte Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS).

7. (Opcional) Dar al usuario permiso para administrar sus propias credenciales de seguridad. Para obtener más información, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

Creación de usuarios de IAM (API de AWS)

Puede utilizar la API de AWS para crear un usuario de IAM.

Para crear un usuario de IAM desde la (API de AWS)

1. Crear un usuario.

   - CreateUser

2. (Opcional) Dar al usuario acceso a la AWS Management Console. Esto requiere una contraseña. También debe dar a los usuarios la URL de la página de inicio de sesión de su cuenta.

   - CreateLoginProfile

3. (Opcional) Dar al usuario acceso mediante programación. Esto requiere claves de acceso.

   - CreateAccessKey

   > **Important**
   >
   > Esta es la única oportunidad que tiene para ver o descargar las claves de acceso secretas, y debe proporcionar dicha información a los usuarios para que puedan utilizar la API de AWS. Guarde el nuevo ID de clave de acceso del usuario y la clave de acceso secreta en un lugar seguro. No volverá a tener acceso a la clave de acceso secreta después de este paso.

4. Añadir el usuario a uno o varios grupos. Los grupos que especifique deben tener políticas asociadas que concedan los permisos pertinentes para el usuario.

   - AddUserToGroup

5. (Opcional) Asociar una política al usuario que defina los permisos del usuario. Nota: le recomendamos que administre los permisos de usuario añadiendo el usuario a un grupo y asociando una política al grupo en lugar de asociarla directamente a un usuario.

   - AttachUserPolicy

6. (Opcional) Añadir los atributos personalizados al usuario asociando etiquetas. Para obtener más información, consulte Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS).

7. (Opcional) Dar al usuario permiso para administrar sus propias credenciales de seguridad. Para obtener más información, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

Controlar el acceso de los usuarios de IAM a la AWS Management Console

Los usuarios de IAM con permiso que inicien sesión en su cuenta de Cuenta de AWS a través de la AWS Management Console pueden acceder a sus recursos de AWS. En la siguiente lista se enumeran las formas en las que puede conceder a los usuarios de IAM acceso a los recursos de su cuenta de Cuenta de AWS a través de la AWS Management Console. También se muestra cómo los

usuarios de IAM pueden obtener acceso a otras características de la cuenta de AWS a través del sitio web de AWS.

> **Note**
>
> No se cobra por utilizar IAM.

Con la AWS Management Console

Cree una contraseña para cada usuario de IAM que necesite obtener acceso a la AWS Management Console. Los usuarios obtienen acceso a la consola mediante la página de inicio de sesión de la cuenta de Cuenta de AWS habilitada para IAM. Para obtener más información acerca de cómo acceder a la página de inicio de sesión, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In. Para obtener información sobre cómo crear contraseñas, consulte Administración de las contraseñas de usuarios en AWS.

Puede desactivar el acceso de un usuario de IAM a la AWS Management Console eliminando su contraseña. Esto les impide iniciar sesión en AWS Management Console utilizando sus credenciales de inicio de sesión. No cambia sus permisos ni les impide acceder a la consola utilizando un rol asumido. Si el usuario dispone de claves de acceso activas, estas seguirán funcionando y permitirán el acceso a través de la AWS CLI, Tools for Windows PowerShell, API de AWS o Console Mobile Application de AWS.

Sus recursos de AWS, como instancias de Amazon EC2, buckets de Amazon S3, etc.

Aunque sus usuarios de IAM tengan contraseñas, también necesitan tener permiso para obtener acceso a sus recursos de AWS. Cuando se crea un usuario de IAM, ese usuario no tiene permisos de manera predeterminada. Para dar a sus usuarios de IAM los permisos que necesitan, debe asociarles políticas. Si tiene muchos usuarios de IAM que realizan las mismas tareas con los mismos recursos, puede asignarlos a un grupo. A continuación, asigne los permisos a dicho grupo. Para obtener información sobre cómo crear usuarios de IAM y grupos, consulte Identidades de IAM (usuarios, grupos de usuarios y roles). Para obtener información sobre cómo utilizar políticas para establecer permisos, consulte Recursos de AWS para administración de acceso.

Foros de debate de AWS

Todo el mundo puede leer las publicaciones de los foros de debate de AWS. Los usuarios que quieran publicar preguntas o comentarios en el foro de debate de AWS pueden hacerlo utilizando

su nombre de usuario. La primera vez que un usuario publica en el foro de discusión de AWS, se le pide que introduzca un alias y una dirección de correo electrónico. Solo ese usuario puede utilizar ese alias en los foros de discusión de AWS.

Información de uso y facturación de su cuenta de Cuenta de AWS

Puede conceder a los usuarios acceso a la información de facturación y de uso de su cuenta de Cuenta de AWS. Para obtener más información, consulte Control del acceso de los usuarios a su información de facturación en la Guía del usuario de AWS Billing.

Información de su perfil de Cuenta de AWS

Los usuarios no pueden obtener acceso a la información de su perfil de cuenta de Cuenta de AWS.

Credenciales de seguridad de su cuenta de Cuenta de AWS

Los usuarios no pueden obtener acceso a las credenciales de seguridad de su cuenta de Cuenta de AWS.

> **Note**
>
> Las políticas de IAM controlan el acceso independientemente de la interfaz. Por ejemplo, puede proporcionar a un usuario una contraseña para acceder a la AWS Management Console. Las políticas para ese usuario (o cualquier grupo al que pertenezca) controlarían lo que el usuario puede hacer en la AWS Management Console. O bien, podría proporcionar al usuario claves de acceso de AWS para realizar llamadas de API a AWS. Las políticas controlarían las acciones que el usuario podría llamar a través de una biblioteca o cliente que utiliza esas claves de acceso para la autenticación.

Cómo inician sesión los usuarios de IAM en AWS

Para iniciar sesión en AWS Management Console como usuario de IAM, debe proporcionar el ID de la cuenta o alias de la cuenta además de su nombre de usuario y contraseña. Cuando su administrador creó su usuario de IAM en la consola, deberían haberle enviado sus credenciales de inicio de sesión, incluido su nombre de usuario y la URL a la página de inicio de sesión de su cuenta que incluye tu ID de cuenta o alias de cuenta.

```
https://My_AWS_Account_ID.signin.aws.amazon.com/console/
```

> **Sugerencia**
>
> Para crear un marcador en la página de inicio de sesión de la cuenta en el navegador web, debe escribir manualmente la URL de dicho inicio de sesión de su cuenta en la entrada de marcador. No utilice la función de marcador del navegador web, porque las redirecciones podrían ocultar la URL de inicio de sesión.

También puede iniciar sesión en el siguiente punto de enlace de inicio de sesión general y escribir manualmente el ID de la cuenta o alias de la cuenta:

```
https://console.aws.amazon.com/
```

Para su comodidad, en la página de inicio de sesión de AWS se utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. La próxima vez que el usuario vaya a cualquier página en AWS Management Console, la consola utiliza la cookie para redirigir al usuario a la página de inicio de sesión de la cuenta.

Solo tiene acceso a los recursos de AWS que su administrador especifica en la política que está asociada a su identidad de usuario de IAM. Para trabajar en la consola, usted debe disponer de permisos para llevar a cabo las acciones propias de la consola, como enumerar y crear recursos de AWS. Para obtener más información, consulte Recursos de AWS para administración de acceso y Ejemplos de políticas basadas en identidad de IAM.

> **Note**
>
> Si su organización ya cuenta con un sistema de identidad, puede ser conveniente crear una opción de inicio de sesión único (SSO). SSO proporciona a los usuarios acceso a la AWS Management Console para su cuenta, aunque no dispongan de una identidad de usuario de IAM. El SSO también elimina la necesidad de que los usuarios inicien sesión por separado en el sitio web de la organización y en AWS. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

Registro de los detalles de inicio de sesión en CloudTrail

Si habilita CloudTrail para registrar los eventos de inicio de sesión en los registros, sea consciente del modo en que CloudTrail elige dónde registrar los eventos.

- Si los usuarios inician sesión directamente en una consola, se les redirige a un punto de enlace de inicio de sesión global o regional, en función de si la consola de servicios seleccionada admite las regiones. Por ejemplo, la página de inicio de la consola admite las regiones, por lo que si inicia sesión en la URL siguiente:

  ```
  https://alias.signin.aws.amazon.com/console
  ```

  se le redirigirá a un punto de enlace de inicio de sesión regional, por ejemplo `https://us-east-2.signin.aws.amazon.com`, lo que se traduce en una entrada de log de CloudTrail regional en el registro de la región del usuario:

  Por otra parte, la consola de Amazon S3 no admite las regiones, por lo que si inicia sesión con la URL siguiente

  ```
  https://alias.signin.aws.amazon.com/console/s3
  ```

  AWS le redirige al punto de enlace de inicio de sesión global en `https://signin.aws.amazon.com`, lo que se traduce en una entrada de registro de CloudTrail global.
- Puede solicitar manualmente un determinado punto de enlace de inicio de sesión regional iniciando sesión en la página de inicio principal de la consola habilitada para regiones, mediante una sintaxis de URL como la siguiente:

  ```
  https://alias.signin.aws.amazon.com/console?region=ap-southeast-1
  ```

  AWS le redirige al punto de enlace de inicio de sesión regional `ap-southeast-1`, lo que se traduce en un evento de registro de CloudTrail regional.

Para obtener más información acerca de CloudTrail e IAM, consulte Registro de eventos de IAM con CloudTrail .

Si los usuarios necesitan acceso programático para trabajar con su cuenta, puede crear un par de claves de acceso (un ID de clave de acceso y una clave de acceso secreta) para cada usuario. Sin embargo, hay alternativas más seguras que hay que tener en cuenta antes de crear claves de acceso para los usuarios. Para más información, consulte Consideraciones y alternativas para las claves de acceso a largo plazo en la Guía de la Referencia general de AWS.

Uso de dispositivos MFA con la página de inicio de sesión de IAM

Los usuarios que están configurados con dispositivos de autenticación multifactor (MFA) deben utilizar sus dispositivos MFA para iniciar sesión en la AWS Management Console. Después de que el usuario escriba sus credenciales de inicio de sesión, AWS comprueba la cuenta del usuario para ver si se necesita MFA. En las siguientes secciones, se proporciona información sobre cómo los usuarios completan el inicio de sesión cuando se necesita un código MFA.

Temas

Inicio de sesión con varios dispositivos MFA habilitados

Si un usuario inicia sesión en la AWS Management Console como usuario raíz de una Cuenta de AWS o usuario de IAM con varios dispositivos MFA habilitados para esa cuenta, solo tendrá que utilizar un dispositivo MFA para iniciar sesión. Después de que el usuario se autentique con su contraseña, selecciona qué tipo de dispositivo MFA desea utilizar para finalizar la autenticación. A continuación, se pide al usuario que se autentique con el tipo de dispositivo que ha seleccionado.

Inicio de sesión con una clave de seguridad FIDO

Si el código MFA es obligatorio para el usuario, aparece una segunda página de inicio de sesión. El usuario tiene que tocar la clave de seguridad FIDO.

> **Note**
>
> Los usuarios de Google Chrome no deben elegir ninguna de las opciones disponibles en la ventana emergente que pide verificar su identidad con amazon.com. Solo tiene que tocar la clave de seguridad.

A diferencia de otros dispositivos MFA, las claves de seguridad FIDO no se desincronizan. Los administradores pueden desactivar una clave de seguridad FIDO si se pierde o se rompe. Para obtener más información, consulte Desactivación de dispositivos MFA (consola).

Para obtener información sobre los navegadores compatibles con WebAuthn y los dispositivos de AWS compatibles con FIDO, consulte Configuraciones admitidas para usar las claves de seguridad FIDO.

Inicio de sesión con un dispositivo MFA virtual

Si el código MFA es obligatorio para el usuario, aparece una segunda página de inicio de sesión. En el cuadro MFA code (Código MFA), el usuario escribe el código numérico proporcionado por la aplicación MFA.

Si el código MFA es correcto, el usuario obtiene acceso a la AWS Management Console. Si el código es incorrecto, el usuario puede volver a intentarlo con otro código.

Es posible que un dispositivo MFA virtual no se sincronice. Si después de varios intentos incorrectos, el usuario no puede iniciar sesión en la AWS Management Console, se le solicitará que sincronice el dispositivo MFA virtual. El usuario puede seguir las instrucciones en pantalla para sincronizar el dispositivo MFA virtual. Para obtener información sobre cómo sincronizar un dispositivo en nombre de un usuario en la Cuenta de AWS, consulte Resincronización de dispositivos MFA físicos y virtuales.

Inicio de sesión con un token TOTP físico

Si el código MFA es obligatorio para el usuario, aparece una segunda página de inicio de sesión. En el cuadro MFA code (Código MFA), el usuario debe ingresar el código numérico proporcionado por un token TOTP de hardware.

Si el código MFA es correcto, el usuario obtiene acceso a la AWS Management Console. Si el código es incorrecto, el usuario puede volver a intentarlo con otro código.

Un token TOTP físico puede perder la sincronización. Si después de varios intentos incorrectos, el usuario no puede iniciar sesión en la AWS Management Console, se le solicitará que sincronice el dispositivo de token MFA. El usuario puede seguir las instrucciones en pantalla para sincronizar el dispositivo de token MFA. Para obtener información sobre cómo sincronizar un dispositivo en nombre de un usuario en la Cuenta de AWS, consulte Resincronización de dispositivos MFA físicos y virtuales.

Administración de usuarios de IAM

> **Note**
>
> Como práctica recomendada, le recomendamos que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales. Si sigue las prácticas recomendadas, no estará administrando usuarios ni grupos de IAM. En cambio, sus usuarios y grupos se administran fuera de AWS y pueden acceder a los recursos de AWS como una identidad federada. Una identidad federada es un usuario del directorio de usuarios de su empresa, un proveedor de identidades web, AWS Directory Service, el directorio de Identity Center o cualquier usuario que acceda a los servicios de AWS con credenciales proporcionadas a través de una fuente de identidades. Las identidades federadas utilizan los grupos definidos por su proveedor de identidades. Si utiliza AWS IAM Identity Center, consulte Manage identities in IAM Identity Center (Administración de identidades en IAM Identity Center) en la Guía del usuario de AWS IAM Identity Center para obtener información sobre la creación de usuarios y grupos en IAM Identity Center.

Amazon Web Services ofrece varias herramientas para administrar los usuarios de IAM en su Cuenta de AWS. Puede enumerar los usuarios de IAM de su cuenta o de un grupo de usuario o enumerar todos los grupos de usuario a los que pertenece un usuario. Puede cambiar el nombre o cambiar la ruta de un usuario de IAM. Si va a utilizar identidades federadas en lugar de usuarios de IAM, puede eliminar un usuario de IAM de su cuenta de AWS o desactivarlo.

Para obtener más información sobre cómo agregar, modificar o eliminar las políticas administradas para un usuario de IAM, consulte Cambio de los permisos de un usuario de IAM. Para obtener información acerca de la administración de las políticas insertadas para usuarios de IAM, consulte Adición y eliminación de permisos de identidad de IAM, Edición de políticas de IAM y Eliminación de políticas de IAM. Como práctica recomendada, utilice políticas administradas en lugar de políticas en línea. Las políticas administradas de AWS conceden permisos para muchos casos de uso comunes. Tenga presente que es posible que las políticas administradas de AWS no concedan permisos de privilegios mínimos para sus casos de uso concretos, ya que están disponibles para que las utilicen todos los clientes de AWS. En consecuencia, se recomienda reducir aún más los permisos definiendo políticas administradas por el cliente específicas para sus casos de uso. Para obtener más información, consulte Políticas administradas de AWS. Para obtener más información acerca de las políticas administradas AWS que están diseñadas para funciones de trabajo específicas, consulte Managed Policies de AWS para funciones de trabajo.

Para obtener información acerca de validar las políticas de IAM, consulte Validación de políticas de IAM.

> **Tip**
>
> El Analizador de acceso de IAM puede analizar los servicios y acciones que utilizan sus roles de IAM y, a continuación, generar una política detallada que puede utilizar. Después de probar cada política generada, puede implementarla en el entorno de producción. Eso garantiza que solo se concedan los permisos necesarios a las cargas de trabajo. Para obtener más información acerca de la generación de políticas, consulte Generación de políticas de IAM Access Analyzer.

Para obtener información sobre cómo administrar las contraseñas de usuario de IAM, consulte Administración de las contraseñas de los usuarios de IAM.

Temas

- Ver acceso de usuario
- Enumeración de usuarios de IAM
- Cambio de nombre de un usuario de IAM
- Eliminación de un usuario de IAM
- Desactivación de un usuario de IAM

Ver acceso de usuario

Antes de eliminar un usuario debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Enumeración de usuarios de IAM

Puede enumerar los usuarios de IAM de su Cuenta de AWS o de un determinado grupo de usuarios de IAM y enumerar todos los grupos de usuarios a los que pertenece un usuario. Para obtener información sobre los permisos que necesita para poder enumerar los usuarios, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Para enumerar todos los usuarios de la cuenta

- AWS Management Console: en el panel de navegación, seleccione Users (Usuarios). La consola muestra los usuarios de su Cuenta de AWS.
- AWS CLI: aws iam list-users
- API de AWS: ListUsers

Para obtener una lista de los usuarios en un grupo específico de usuarios

- AWS Management Console: en el panel de navegación, elija Grupos de usuarios elija el nombre del grupo de usuarios y, a continuación, elija la pestaña Usuarios.
- AWS CLI: aws iam get-group
- API de AWS: GetGroup

Para listar todos los grupos de usuarios en los que se encuentra un usuario

- AWS Management Console: en el panel de navegación, elija Users (Usuarios), elija el nombre del usuario y, a continuación, elija la pestaña Groups (Grupos).
- AWS CLI: aws iam list-groups-for-user
- API de AWS: ListGroupsForUser

Cambio de nombre de un usuario de IAM

Para cambiar el nombre de un usuario o ruta, debe utilizar la AWS CLI, Tools for Windows PowerShell o API de AWS. No hay ninguna otra opción en la consola para cambiar el nombre de un usuario. Para obtener información sobre los permisos que necesita para poder cambiar el nombre de un usuario, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Al cambiar la ruta de acceso o el nombre de un usuario, ocurrirá lo siguiente:

- Cualquier política asociada al usuario permanecerá con el usuario con el nuevo nombre.
- El usuario permanecerá en los mismos grupos de usuario con el nuevo nombre.
- El ID único del usuario seguirá siendo el mismo. Para obtener más información sobre los ID únicos, consulte Identificadores únicos.
- Cualquier política de recurso o rol que haga referencia al usuario como principal (se concede acceso al usuario) se actualizará automáticamente para utilizar el nuevo nombre o ruta de acceso.

Por ejemplo, las políticas basadas en colas de Amazon SQS o las políticas basadas en recursos de Amazon S3 se actualizarán automáticamente para utilizar el nuevo nombre y ruta de acceso.

IAM no actualizará automáticamente políticas que hagan referencia al usuario como recurso para utilizar el nuevo nombre o ruta de acceso; debe hacerlo manualmente. Por ejemplo, imagine que el usuario `Richard` tiene una política asociada a él que le permite administrar sus credenciales de seguridad. Si un administrador cambia el nombre de `Richard` a `Rich`, el administrador también debe actualizar dicha política para cambiar el recurso de:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Richard
```

a este:

```
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/Rich
```

Lo mismo sucede si un administrador cambia la ruta de acceso; el administrador debe actualizar la política para reflejar la nueva ruta de acceso para el usuario.

Para cambiar el nombre de un usuario

- AWS CLI: aws iam update-user
- API de AWS: UpdateUser

Eliminación de un usuario de IAM

Puede eliminar un usuario de IAM de su Cuenta de AWS si alguien deja de trabajar en su empresa. Si el usuario está ausente temporalmente, puede desactivar el acceso del usuario en lugar de eliminarlo de la cuenta como se describe en Desactivación de un usuario de IAM.

Temas

- Si eliminas un usuario IAM (Consola)
- Eliminación de un usuario de IAM (AWS CLI)

Si eliminas un usuario IAM (Consola)

Al utilizar la AWS Management Console para eliminar un usuario de IAM, IAM eliminará automáticamente la siguiente información:

- El usuario
- Cualquier suscripción a un grupo de usuarios, es decir el usuario se eliminará de cualquier grupo de IAM al que el usuario haya pertenecido
- Cualquier contraseña asociada al usuario
- Cualquier clave de acceso que pertenezca al usuario
- Todas las políticas insertadas en el usuario (las políticas que se aplican a un usuario a través de los permisos de grupo de usuarios no se verán afectadas)

> **Note**
>
> IAM elimina las políticas administradas adjuntas al usuario cuando se elimina el usuario, pero no elimina las políticas administradas.

- Cualquier dispositivo MFA asociado

Para eliminar un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Usuarios y, a continuación, seleccione la casilla de verificación junto al nombre del rol que desee eliminar.
3. En la parte superior de la página, elija Delete (Eliminar).
4. En el cuadro de diálogo de confirmación, ingrese el nombre de usuario en el campo de entrada de texto para confirmar la eliminación del usuario. Elija Delete (Eliminar).

Eliminación de un usuario de IAM (AWS CLI)

A diferencia de la AWS Management Console, a la hora de eliminar un usuario con la AWS CLI debe eliminar los elementos adjuntos al usuario manualmente. Este procedimiento ilustra el proceso.

Para eliminar un usuario de su cuenta (AWS CLI)

1. Elimine la contraseña de usuario, si el usuario dispone de una.

   ```
   aws iam delete-login-profile
   ```
2. Elimine las claves de acceso de usuario, si el usuario dispone de ellas.

`aws iam list-access-keys` (para generar una lista de las claves de acceso del usuario) y `aws iam delete-access-key`

3. Elimine el certificado de firma del usuario. Tenga en cuenta que al eliminar una credencial de seguridad, ya nunca más podrá recuperarla.

   `aws iam list-signing-certificates` (para generar una lista de los certificados de firma del usuario) y `aws iam delete-signing-certificate`

4. Elimine la clave pública SSH del usuario, si el usuario dispone de ella.

   `aws iam list-ssh-public-keys` (para generar una lista de las claves públicas SSH del usuario) y `aws iam delete-ssh-public-key`

5. Elimine las credenciales de Git del usuario.

   `aws iam list-service-specific-credentials` (para generar una lista de las credenciales de Git del usuario) y `aws iam delete-service-specific-credential`

6. Desactive el dispositivo Multi-Factor Authentication (MFA) del usuario, si este dispone de uno.

   `aws iam list-mfa-devices` (para generar una lista de los dispositivos MFA del usuario), `aws iam deactivate-mfa-device` (para desactivar el dispositivo) y `aws iam delete-virtual-mfa-device` (para eliminar de forma permanente un dispositivo de MFA virtual)

7. Elimine las políticas insertadas del usuario.

   `aws iam list-user-policies` (para generar una lista de las políticas insertadas para el usuario) y `aws iam delete-user-policy` (para eliminar la política)

8. Desasocie cualquier política administrada que esté asociada al usuario.

   `aws iam list-attached-user-policies` (para genear una lista de las políticas administradas adjuntas al usuario) y `aws iam detach-user-policy` (para desasociar la política)

9. Elimine el usuario de cualquier grupo de usuarios en el que se encuentre.

   `aws iam list-groups-for-user` (para generar una lista de los grupos de usuarios a los que pertenece el usuario) y `aws iam remove-user-from-group`

10. Elimine el usuario.

    `aws iam delete-user`

Desactivación de un usuario de IAM

Puede que tenga que desactivar un usuario de IAM mientras esté temporalmente fuera de su empresa. Puede dejar sus credenciales de usuario de IAM en su lugar y seguir bloqueando su acceso a AWS.

Para desactivar un usuario, cree y asocie una política para denegar el acceso del usuario a AWS. Puede restaurar el acceso del usuario más adelante.

A continuación se muestran dos ejemplos de políticas de denegación que puede asociar a un usuario para denegar su acceso.

La siguiente política no incluye un límite de tiempo. Debe eliminar la política para restaurar el acceso del usuario.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*"
      }
   ]
}
```

La siguiente política incluye una condición que inicia la política el 24 de diciembre de 2024 a las 23:59 h (UTC) y la termina el 28 de febrero de 2025 a las 23:59 h (UTC).

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
          "DateGreaterThan": {"aws:CurrentTime": "2024-12-24T23:59:59Z"},
          "DateLessThan": {"aws:CurrentTime": "2025-02-28T23:59:59Z"}
          }
       }
   ]
```

```
}
```

Cambio de los permisos de un usuario de IAM

Puede modificar los permisos de un usuario de IAM de una Cuenta de AWS cambiando su pertenencia a grupos, copiando los permisos de un usuario existente, asociando políticas directamente al usuario o definiendo un límite de permisos. Un límite de permisos controla los permisos que puede tener un usuario como máximo. Los límites de permisos son una característica avanzada de AWS.

Para obtener información sobre los permisos que necesita para poder modificar los permisos de un usuario, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Temas

- Ver acceso de usuario
- Generar una política basada en la actividad de acceso de un usuario
- Adición de permisos a un usuario (consola)
- Cambio de los permisos de un usuario (consola)
- Eliminación de una política de permisos de un usuario (consola)
- Eliminación del límite de permisos de un usuario (consola)
- Adición y eliminación de permisos de un usuario (AWS CLI o API de AWS)

Ver acceso de usuario

Antes de cambiar los permisos para un usuario, debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Generar una política basada en la actividad de acceso de un usuario

A veces puede conceder permisos a una entidad de IAM (usuario o rol) de más allá de lo que requieren. Para ayudarle a refinar los permisos que concede, puede generar una política de IAM que esté basada en la actividad de acceso de una entidad. El analizador de acceso de IAM revisa los registros de AWS CloudTrail y genera una plantilla de política que contiene los permisos que ha

utilizado la entidad en el intervalo de fechas especificado. Puede utilizar la plantilla para crear una política administrada con permisos detallados y, a continuación, adjuntarla a la entidad de IAM. De esta forma, solo concede los permisos que el usuario o rol necesita para interactuar con los recursos de AWS para su caso de uso específico. Para obtener más información, consulte Generar políticas basadas en la actividad de acceso.

Adición de permisos a un usuario (consola)

IAM ofrece de tres formas de agregar políticas de permisos a un usuario:

- Agregar un usuario a un grupo: convertir a un usuario en miembro de un grupo. Las políticas del grupo se asocian al usuario.
- Copiar los permisos de un usuario existente: copiar todas las suscripciones a grupos, las políticas administradas asociadas, las políticas insertadas, así como los límites de permisos existentes para el usuario de origen.
- Adjuntar políticas directamente al usuario: adjunte una política administrada directamente al usuario. Para facilitar la administración de permisos, adjunte sus políticas a un grupo y, a continuación, haga a los usuarios miembros de los grupos apropiados.

> **Important**
>
> Si el usuario tiene un límite de permisos, no se pueden añadir permisos al usuario por encima de los que autoriza el límite de permisos.

Agregar permisos añadiendo al usuario a un grupo

Cuando se añade un usuario a un grupo, el cambio afecta inmediatamente al usuario.

Para agregar permisos a un usuario añadiendo el usuario a un grupo

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Consulte las suscripciones de grupos actuales de los usuarios en la columna Groups (Grupos) de la consola. Si es necesario, agregue la columna a la tabla de usuarios realizando los siguientes pasos:

1. Encima de la tabla, en el extremo derecho, elija el símbolo de configuración ().
2. En el cuadro de diálogo Manage Columns (Administrar columnas), seleccione la columna Groups (Grupos). Opcionalmente, también puede quitar la marca de la casilla de verificación de los encabezados de columna que no quiera que aparezcan en la tabla de los usuarios.
3. Seleccione Close (Cerrar) para volver a la lista de usuarios.

La columna Groups (Grupos) le indica a qué grupos pertenece el usuario. La columna incluye los nombres de un máximo de dos grupos. Si el usuario es miembro de tres o más grupos, se muestran los dos primeros (ordenados alfabéticamente) y el número de grupos adicionales a los que pertenece. Por ejemplo, si el usuario pertenece a los grupos siguientes: Grupo A, Grupo B, Grupo C y Grupo D, el campo contendrá el valor Group A, Group B + 2 more (Grupo A, Grupo B + 2 más). Para ver el número total de grupos a los que pertenece el usuario, puede añadir la columna Group count (Recuento de grupos) a la tabla de los usuarios.

4. Elija el nombre del usuario cuyos permisos quiere modificar.
5. Seleccione la pestaña Permissions (Permisos) y, a continuación, seleccione Add permissions (Añadir permisos). Elija Add user to group.
6. Seleccione la casilla de verificación de todos los grupos a los que desee que el usuario pertenezca. La lista muestra el nombre de cada grupo y las políticas que el usuario recibe si pasa a ser miembro de ese grupo.
7. (Opcional) Además de seleccionar entre los grupos existentes, puede elegir Create group (Crear grupo) para definir un grupo nuevo:

   a. En la pestaña nueva, en Nombre del grupo de usuarios, escriba un nombre para el grupo nuevo.

   > **Note**
   >
   > El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS. Los nombres de grupo pueden ser una combinación de un máximo de 128 letras, dígitos y los siguientes caracteres: más (+), igual (=), coma (,), punto (.), arroba (@) y guion (-). Los nombres deben ser únicos dentro de una cuenta. No distinguen entre

mayúsculas y minúsculas. Por ejemplo, no puede crear dos grupos llamados TESTGROUP y testgroup.

b. Seleccione una o varias casillas de verificación para las políticas administradas que desea asociar al grupo. También puede crear una política administrada nueva eligiendo Create policy (Crear política). Si lo hace, vuelva a esta ventana o pestaña del navegador cuando haya acabado de crear la política nueva; elija Refresh (Actualizar) y, a continuación, elija la nueva política para asociarla a su grupo. Para obtener más información, consulte Crear políticas de IAM.

c. Elija Crear grupo de usuarios.

d. Vuelva a la pestaña original y actualice la lista de grupos. A continuación, seleccione la casilla del grupo nuevo.

8. Elija Siguiente para ver la lista de suscripciones a grupos que se agregarán al usuario. A continuación, elija Add permissions (Añadir permisos).

Agregar permisos copiándolos de otro usuario

Cuando se copian permisos, el cambio afecta inmediatamente al usuario.

Para añadir permisos a un usuario copiando los permisos de otro usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Users (Usuarios) en el panel de navegación, seleccione el nombre del usuario cuyos permisos quiera modificar y, a continuación, elija la pestaña Permissions (Permisos).
3. Elija Add permissions (Añadir permisos) y, a continuación, elija Copy permissions from existing user (Copiar permisos de un usuario existente). La lista muestra los usuarios disponibles junto con sus suscripciones a grupos y las políticas que tienen asociadas. Si la lista completa de grupos o políticas no cabe en una línea, puede elegir el enlace de and ***n*** more (y n más). De esta forma, se abre una pestaña del navegador nueva y podrá ver la lista completa de políticas (pestaña Permissions (Permisos)) y grupos (pestaña Groups (Grupos)).
4. Seleccione el botón de opción que está junto al usuario cuyos permisos quiere copiar.
5. Elija Siguiente para ver la lista de cambios que se realizarán en el usuario. A continuación, elija Add permissions (Añadir permisos).

Agregar permisos asociando políticas directamente al usuario

Cuando se asocian políticas, el cambio afecta inmediatamente al usuario.

Para agregar permisos a un usuario asociando directamente políticas administradas

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Users (Usuarios) en el panel de navegación, seleccione el nombre del usuario cuyos permisos quiera modificar y, a continuación, elija la pestaña Permissions (Permisos).
3. Elija Agregar permisos y luego, elija Asociar políticas directamente.
4. Seleccione una o varias casillas para las políticas administradas que desea asociar al usuario. También puede crear una política administrada nueva eligiendo Create policy (Crear política). Si lo hace, vuelva a la ventana o pestaña de este navegador cuando haya acabado de crear la nueva política. Elija Refresh (Actualizar) y, a continuación, seleccione la casilla de verificación de la política nueva para asociarla a su usuario. Para obtener más información, consulte Crear políticas de IAM.
5. Elija Siguiente para ver la lista de políticas que se asociarán al usuario. A continuación, elija Add permissions (Añadir permisos).

Configuración del límite de permisos de un usuario

Cuando se configura un límite de permisos, el cambio afecta inmediatamente al usuario.

Para configurar el límite de permisos de un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuyo límite de permisos desea modificar.
4. Elija la pestaña Permissions (Permisos). Si es necesario, abra la sección Límite de permisos y luego, elija Configurar límite de permisos.
5. Seleccione la política que desea utilizar para el límite de permisos.
6. Elija Set boundary (Configurar límite).

Cambio de los permisos de un usuario (consola)

IAM le permite cambiar los permisos asociados a un usuario de las siguientes maneras:

- Editar una política de permisos: editar la política insertada del usuario, la política insertada del grupo del usuario o una política administrada que esté asociada al usuario directamente o a través de un grupo. Si el usuario tiene un límite de permisos, no se le pueden conceder permisos por encima de los que autoriza la política utilizada como límite de permisos del usuario.
- Cambiar el límite de permisos: cambiar la política utilizada como límite de permisos para el usuario. Esto puede ampliar o reducir los permisos máximos que puede tener un usuario.

Edición de una política de permisos asociada a un usuario

Cuando se modifican los permisos, el cambio afecta inmediatamente al usuario.

Para editar las políticas administradas asociadas a un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuya política de permisos desea modificar.
4. Elija la pestaña Permissions (Permisos). Si es necesario, abra la sección Permissions policies (Políticas de permisos).
5. Elija el nombre de la política que desea editar para ver sus detalles. Elija la pestaña Uso de políticas para ver otras entidades que podrían verse afectadas si se edita la política.
6. Elija la pestaña Permissions (Permisos) y revise los permisos que concede la política. A continuación, elija Edit policy (Editar política).
7. Edite la política y resuelva las recomendaciones de validación de políticas. Para obtener más información, consulte Edición de políticas de IAM.
8. Elija Review policy (Revisar política), revise el resumen de la política y, a continuación, elija Save changes (Guardar cambios).

Cambio del límite de permisos de un usuario

Cuando se modifica un límite de permisos, el cambio afecta inmediatamente al usuario.

Para cambiar la política que se utiliza para establecer el límite de permisos para un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuyo límite de permisos desea modificar.
4. Elija la pestaña Permissions (Permisos). Si es necesario, abra la sección Permissions boundary (Límite de permisos) y, a continuación, elija Change boundary (Cambiar límite).
5. Seleccione la política que desea utilizar para el límite de permisos.
6. Elija Set boundary (Configurar límite).

Eliminación de una política de permisos de un usuario (consola)

Cuando se elimina una política de un usuario, el cambio afecta inmediatamente al usuario.

Para eliminar los permisos de los usuarios de IAM

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuyo límite de permisos desea eliminar.
4. Elija la pestaña Permissions (Permisos).
5. Si desea eliminar permisos mediante la remoción de una política existente, observe el Tipo para comprender cómo el usuario obtiene la política antes de elegir Eliminar para eliminar la política:

   - Si la política se aplica por la suscripción a un grupo, cuando elija Eliminar eliminará al usuario del grupo. Recuerde que es posible que tenga varias políticas asociadas a un único grupo. Si elimina al usuario de ese grupo, el usuario perderá el acceso a todas las políticas que recibió al pertenecer a dicho grupo.
   - Si la política es una política administrada asociada directamente al usuario, cuando elija Eliminar separará la política del usuario. Esto no afecta a la política en sí o a cualquier otra entidad a la que la política pueda estar asociada.
   - Si la política es una política integrada insertada, al elegir X eliminará la política de IAM. Las políticas insertadas que están directamente asociadas a un usuario existen únicamente en dicho usuario.

Eliminación del límite de permisos de un usuario (consola)

Cuando se elimina un límite de permisos, el cambio afecta inmediatamente al usuario.

Para eliminar el límite de permisos de un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuyo límite de permisos desea eliminar.
4. Elija la pestaña Permissions (Permisos). Si es necesario, abra la sección Permissions boundary (Límite de permisos) y, a continuación, elija Remove boundary (Eliminar límite).
5. Elija Eliminar límite para confirmar que desea eliminar el límite de permisos.

Adición y eliminación de permisos de un usuario (AWS CLI o API de AWS)

Para añadir o eliminar permisos de forma programada, debe añadir o eliminar las suscripciones a grupos, asociar o separar las políticas administradas o eliminar las políticas insertadas. Para obtener más información, consulte los siguientes temas:

- Agregar y eliminar usuarios de un grupo de usuarios de IAM
- Adición y eliminación de permisos de identidad de IAM

Administración de las contraseñas de usuarios en AWS

Puede administrar las contraseñas de los usuarios de IAM de su cuenta. Los usuarios de IAM necesitan contraseñas para obtener acceso a la AWS Management Console. Los usuarios no necesitan contraseñas para obtener acceso a los recursos de AWS mediante programación, utilizando la AWS CLI, Tools for Windows PowerShell, los SDK de AWS o las API. Para esos entornos, tiene la opción de asignar claves de acceso a los usuarios de IAM. Sin embargo, existen otras alternativas más seguras a las claves de acceso que le recomendamos que tenga en cuenta en primer lugar. Para obtener más información, consulte Credenciales de seguridad de AWS.

Contenido

- Configuración de una política de contraseñas de la cuenta para usuarios de IAM
- Administración de las contraseñas de los usuarios de IAM

- Autorización para que los usuarios de IAM cambien sus contraseñas
- Cómo un usuario de IAM cambia su propia contraseña

Configuración de una política de contraseñas de la cuenta para usuarios de IAM

Puede establecer una política de contraseñas personalizada en la Cuenta de AWS para especificar los requisitos de complejidad y los periodos de rotación obligatorios de las contraseñas de los usuarios de IAM. Si no establece una política de contraseñas personalizada, las contraseñas de los usuarios de IAM deberán cumplir con la política de contraseñas predeterminada de AWS. Para obtener más información, consulte Opciones de la política de contraseñas personalizada.

Temas

- Reglas para configurar una política de contraseñas
- Permisos necesarios para establecer una política de contraseñas
- Política de contraseñas predeterminada
- Opciones de la política de contraseñas personalizada
- Configuración de una política de contraseñas (Consola)
- Configuración de una política de contraseñas (AWS CLI)
- Configuración de una política de contraseñas (API de AWS)

Reglas para configurar una política de contraseñas

La política de contraseñas de IAM no se aplica a la contraseña de Usuario raíz de la cuenta de AWS ni a las claves de acceso de los usuarios de IAM. Si una contraseña vence, el usuario de IAM no podrá iniciar sesión en la AWS Management Console, pero podrá seguir utilizando sus claves de acceso.

Al crear o cambiar una política de contraseñas, la mayoría de los ajustes de la política de contraseñas se aplican la siguiente vez que los usuarios cambien sus contraseñas. Sin embargo, algunos de los ajustes se aplican de forma inmediata. Por ejemplo:

- Cuando cambian los requisitos de longitud mínima y de tipos de caracteres, esta configuración se aplica la próxima vez que los usuarios cambian la contraseña. Los usuarios no están obligados a cambiar sus contraseñas, aunque las contraseñas existentes no cumplan la política de contraseñas actualizada.

- Si configura el periodo de vencimiento de una contraseña, este se aplica de forma inmediata. Por ejemplo, supongamos que se establece un periodo de vencimiento de la contraseña de 90 días. En ese caso, la contraseña vence para todos los usuarios de IAM cuya contraseña existente tiene más de 90 días. Esos usuarios deberán cambiar su contraseña la próxima vez que inicien sesión.

No puede crear una “política de bloqueo” para bloquear a los usuarios el acceso a la cuenta después de un número específico de intentos fallidos de inicio de sesión. Para mejorar la seguridad, se recomienda combinar una política de contraseñas seguras con la autenticación multifactor (MFA). Para obtener más información acerca de MFA, consulte Uso de autenticación multifactor (MFA) en AWS.

Permisos necesarios para establecer una política de contraseñas

Debe configurar permisos para permitir que una entidad (usuario o rol) de IAM vea o edite la política de contraseñas de cuenta. Puede incluir las siguientes acciones de política de contraseñas en una política de IAM:

- `iam:GetAccountPasswordPolicy`: permite a la entidad ver la política de contraseñas de su cuenta
- `iam:DeleteAccountPasswordPolicy`: permite a la entidad eliminar la política de contraseñas personalizada para su cuenta y volver a la política de contraseñas predeterminada
- `iam:UpdateAccountPasswordPolicy`: permite a la entidad crear o cambiar la política de contraseñas personalizada de su cuenta

La siguiente política permite el acceso total para ver y editar la política de contraseñas de la cuenta. Para obtener información sobre cómo crear una política de IAM mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessPasswordPolicy",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:DeleteAccountPasswordPolicy",
                "iam:UpdateAccountPasswordPolicy"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Para obtener información acerca de los permisos necesarios para que un usuario de IAM cambie su propia contraseña, consulte Autorización para que los usuarios de IAM cambien sus contraseñas.

Política de contraseñas predeterminada

Si un administrador no establece una política de contraseñas personalizada, las contraseñas de los usuarios de IAM deben cumplir con la política de contraseñas predeterminada de AWS.

La política de contraseñas predeterminada aplica las siguientes condiciones:

- tener una longitud mínima de contraseña de 8 caracteres y una longitud máxima de 128 caracteres
- Un mínimo de tres de los siguientes tipos de caracteres: mayúsculas, minúsculas, números y caracteres no alfanuméricos (`! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`)
- No ser idéntica al nombre de la Cuenta de AWS ni a la dirección de correo electrónico
- La contraseña no caduca nunca

Opciones de la política de contraseñas personalizada

Cuando configure una política de contraseñas personalizada para su cuenta, podrá especificar las siguientes condiciones:

- Establecer la longitud mínima de la contraseña: usted puede especificar un mínimo de 6 caracteres y un máximo de 128 caracteres.
- Establecer la seguridad de la contraseña: usted puede seleccionar cualquiera de las siguientes casillas de verificación para definir la seguridad de las contraseñas de los usuarios de IAM:
  - Exija al menos una letra mayúscula del alfabeto latino (A-Z).
  - Exija al menos una letra minúscula del alfabeto latino (a-z).
  - Exija al menos un número
  - Exija al menos un carácter no alfanumérico `! @ # $ % ^ & * ( ) _ + - = [ ] { } | '`.
- Activar el vencimiento de la contraseña: puede seleccionar y especificar un mínimo de 1 día y un máximo de 1095 días de validez de las contraseñas de los usuarios de IAM una vez establecidas.

Por ejemplo, si especifica un vencimiento de 90 días, afecta inmediatamente a todos los usuarios. Los usuarios con contraseñas de más de 90 días, cuando inician sesión en la consola después del cambio, deben establecer una nueva contraseña. Los usuarios con contraseñas de entre 75 y 89 días de antigüedad reciben un aviso en la AWS Management Console sobre el vencimiento de su contraseña. Los usuarios de IAM pueden cambiar su contraseña en cualquier momento si tienen permiso para hacerlo. Cuando establecen una contraseña nueva, el periodo de vencimiento para esa contraseña vuelve a comenzar. Un usuario de IAM solo puede tener una contraseña válida a la vez.

- La caducidad de la contraseña requiere un restablecimiento por parte del administrador: seleccione esta opción para evitar que los usuarios de IAM utilicen la AWS Management Console para actualizar sus propias contraseñas después de que la contraseña caduque. Antes de seleccionar esta opción, confirme que su Cuenta de AWS tenga más de un usuario con permisos administrativos para restablecer las contraseñas de los usuarios de IAM. Los administradores con el permiso `iam:UpdateLoginProfile` pueden restablecer las contraseñas de usuario de IAM. Los usuarios de IAM con el permiso `iam:ChangePassword` y las claves de acceso activas pueden restablecer su propia contraseña de la consola de usuario de IAM mediante programación. Si desactiva esta casilla de verificación, los usuarios de IAM con contraseñas vencidas aún deberán establecer una nueva contraseña antes de poder acceder a la AWS Management Console.
- Permitir que los usuarios cambien su propia contraseña: puede permitir que todos los usuarios de IAM de su cuenta cambien su propia contraseña. Esto permite a los usuarios acceder a la acción `iam:ChangePassword` solo para su usuario y a la acción `iam:GetAccountPasswordPolicy`. Esta opción no adjunta una política de permisos a cada usuario. Más bien, IAM aplica los permisos en la cuenta para todos los usuarios. También puede permitir que solo algunos usuarios administren sus propias contraseñas. Para ello, desactive esta casilla de verificación. Para obtener más información acerca del uso de políticas para limitar quién puede administrar contraseñas, consulte Autorización para que los usuarios de IAM cambien sus contraseñas.
- Impedir la reutilización de las contraseñas: puede impedir que los usuarios de IAM reutilicen una cantidad específica de contraseñas anteriores. Puede especificar un número mínimo de 1 y un número máximo de 24 contraseñas anteriores que no se pueden repetir.

Configuración de una política de contraseñas (Consola)

Puede utilizar la AWS Management Console para crear, cambiar o eliminar una política de contraseñas personalizada.

Para crear una política de contraseñas personalizada (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Configuración de cuenta.
3. En la sección Password policy (Política de contraseñas), elija Edit (Editar).
4. Elija Custom (Personalizado) para usar una política de contraseñas personalizada.
5. Seleccione las opciones que desee aplicar a su política de contraseñas y elija Save changes (Guardar cambios).
6. Para confirmar que desea establecer la política de contraseñas personalizada, seleccione Set custom (Establecer la política personalizada).

Para cambiar una política de contraseñas personalizada (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Configuración de cuenta.
3. En la sección Password policy (Política de contraseñas), elija Edit (Editar).
4. Seleccione las opciones que desee aplicar a su política de contraseñas y elija Save changes (Guardar cambios).
5. Para confirmar que desea establecer la política de contraseñas personalizada, seleccione Set custom (Establecer la política personalizada).

Para eliminar una política de contraseñas personalizada (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Configuración de cuenta.
3. En la sección Password policy (Política de contraseñas), elija Edit (Editar).
4. Elija IAM default (Predeterminado de IAM) para eliminar la política de contraseñas personalizada y elija Save changes (Guardar cambios).
5. Para confirmar que desea establecer la política de contraseñas predeterminada de IAM, seleccione Set default (Establecer la política predeterminada).

Configuración de una política de contraseñas (AWS CLI)

Puede utilizar la AWS Command Line Interface para establecer una política de contraseñas.

Para administrar la política personalizada de contraseñas de cuentas desde la AWS CLI

Ejecute los comandos siguientes:

- Para crear o cambiar la política de contraseñas personalizada: `aws iam update-account-password-policy`
- Para ver la política de contraseñas: `aws iam get-account-password-policy`
- Para eliminar la política de contraseñas personalizada: `aws iam delete-account-password-policy`

Configuración de una política de contraseñas (API de AWS)

Puede utilizar las operaciones de la API de AWS para establecer una política de contraseñas.

Para administrar la política personalizada de contraseñas de cuentas desde la API de AWS

Llame a las siguientes operaciones:

- Para crear o cambiar la política de contraseñas personalizada: `UpdateAccountPasswordPolicy`
- Para ver la política de contraseñas: `GetAccountPasswordPolicy`
- Para eliminar la política de contraseñas personalizada: `DeleteAccountPasswordPolicy`

Administración de las contraseñas de los usuarios de IAM

Los usuarios de IAM que utilizan la AWS Management Console para trabajar con los recursos de AWS deben tener una contraseña para poder iniciar sesión. Puede crear, cambiar o eliminar la contraseña de un usuario de IAM en su cuenta de AWS.

Una vez que haya asignado una contraseña a un usuario, el usuario puede iniciar sesión en la AWS Management Console con la URL de inicio de sesión de su cuenta, que tiene este aspecto:

```
https://12-digit-AWS-account-ID or alias.signin.aws.amazon.com/console
```

Para obtener más información acerca de cómo los usuarios de IAM inician sesión en la AWS Management Console, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In.

Aunque los usuarios tengan sus propias contraseñas, deben seguir teniendo permisos para obtener acceso a los recursos de AWS. De forma predeterminada, un usuario no tiene permisos. Para conceder a los usuarios los permisos que necesitan, debe asignarles políticas o a los grupos a los que pertenezcan. Para obtener información sobre cómo crear usuarios y grupos, consulte Identidades de IAM (usuarios, grupos de usuarios y roles). Para obtener información sobre cómo utilizar políticas para establecer permisos, consulte Cambio de los permisos de un usuario de IAM.

Puede conceder a los usuarios permiso para cambiar sus propias contraseñas. Para obtener más información, consulte Autorización para que los usuarios de IAM cambien sus contraseñas. Para obtener más información acerca de cómo los usuarios acceden a la página de inicio de sesión de su cuenta, consulte Cómo iniciar sesión en AWS en la Guía del usuario de AWS Sign-In.

Temas

- Creación, cambio o eliminación de la contraseña de un usuario de IAM (consola)
- Creación, cambio o eliminación de la contraseña de un usuario de IAM (AWS CLI)
- Creación, cambio o eliminación de la contraseña de un usuario de IAM (API de AWS)

Creación, cambio o eliminación de la contraseña de un usuario de IAM (consola)

También puede utilizar la AWS Management Console para administrar contraseñas de los usuarios de IAM.

Cuando los usuarios dejen su organización o ya no necesiten acceso a AWS, es importante encontrar las credenciales que utilizaron y garantizar que no ya no estén operativas. Lo ideal es eliminar las credenciales si ya no son necesarias. Siempre puede volver a crearlas más tarde, en caso de que surja la necesidad. Al menos, debe cambiar las credenciales para que los antiguos usuarios ya no tengan acceso.

Para agregar una contraseña de un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuya contraseña desea crear.

4. Elija la pestaña Credenciales de seguridad y luego, en Inicio de sesión en la consola, elija Habilitar el acceso a la consola.
5. En Manage console access (Administrar el acceso a la consola), en Console access (Acceso a la consola) elija la opción Enable (Habilitar) si aún no está seleccionada. Si el acceso a la consola está deshabilitado, no es necesario introducir ninguna contraseña.
6. En Establecer contraseña, seleccione si IAM debe generar una contraseña o crear una contraseña personalizada:

   - Para que IAM genere una contraseña, elija Contraseña generada automáticamente.
   - Para crear una contraseña personalizada, elija Custom password (Contraseña personalizada) y escriba la contraseña.

   > **Note**
   >
   > La contraseña que cree debe cumplir con la política de contraseñas de la cuenta.

7. Para exigir al usuario que cree una contraseña nueva cuando inicie sesión, elija El usuario debe crear una contraseña nueva la próxima vez que inicie sesión. A continuación, elija Apply (Aplicar).

   > **Important**
   >
   > Si selecciona la opción El usuario debe crear una contraseña nueva la próxima vez que inicie sesión, asegúrese de que el usuario tenga el permiso para cambiar la contraseña. Para obtener más información, consulte Autorización para que los usuarios de IAM cambien sus contraseñas.

8. Si elige la opción de generar una contraseña, elija Mostrar en el cuadro de diálogo Contraseña nueva. Esto le permite ver la contraseña para que pueda compartirla con el usuario.

   > **Important**
   >
   > Por motivos de seguridad, no puede obtener acceso a la contraseña después de completar este paso, pero puede crear una nueva contraseña en cualquier momento.

Para cambiar la contraseña de un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuya contraseña desea cambiar.
4. Elija la pestaña Credenciales de seguridad y luego, en Inicio de sesión en la consola, elija Administrar el acceso a la consola.
5. En Manage console access (Administrar el acceso a la consola), en Console access (Acceso a la consola) elija la opción Enable (Habilitar) si aún no está seleccionada. Si el acceso a la consola está deshabilitado, no es necesario introducir ninguna contraseña.
6. En Establecer contraseña, seleccione si IAM debe generar una contraseña o crear una contraseña personalizada:
   - Para que IAM genere una contraseña, elija Contraseña generada automáticamente.
   - Para crear una contraseña personalizada, elija Custom password (Contraseña personalizada) y escriba la contraseña.

   > **Note**
   >
   > La contraseña que cree debe cumplir con la política de contraseñas de la cuenta, si hubiera alguna establecida.

7. Para exigir al usuario que cree una contraseña nueva cuando inicie sesión, elija El usuario debe crear una contraseña nueva la próxima vez que inicie sesión. A continuación, elija Apply (Aplicar).

   > **Important**
   >
   > Si selecciona la opción El usuario debe crear una contraseña nueva la próxima vez que inicie sesión, asegúrese de que el usuario tenga el permiso para cambiar la contraseña. Para obtener más información, consulte Autorización para que los usuarios de IAM cambien sus contraseñas.

8. Si elige la opción de generar una contraseña, elija Mostrar en el cuadro de diálogo Contraseña nueva. Esto le permite ver la contraseña para que pueda compartirla con el usuario.

> **Important**
>
> Por motivos de seguridad, no puede obtener acceso a la contraseña después de completar este paso, pero puede crear una nueva contraseña en cualquier momento.

Para eliminar (desactivar) la contraseña de un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Elija el nombre del usuario cuya contraseña desea eliminar.
4. Elija la pestaña Credenciales de seguridad y luego, en Inicio de sesión en la consola, elija Administrar el acceso a la consola.
5. En Console access (Acceso a la consola), elija Disable (Deshabilitar) y, a continuación, elija Apply (Aplicar).

> **Important**
>
> Puede desactivar el acceso de un usuario de IAM a la AWS Management Console eliminando su contraseña. Esto les impide iniciar sesión en la AWS Management Console utilizando sus credenciales de inicio de sesión. No cambia sus permisos ni les impide acceder a la consola utilizando un rol asumido. Si el usuario dispone de claves de acceso activas, estas seguirán funcionando y permitirán el acceso a través de la AWS CLI, Tools for Windows PowerShell, API de AWS o Console Mobile Application de AWS.

Creación, cambio o eliminación de la contraseña de un usuario de IAM (AWS CLI)

También puede utilizar la API de la AWS CLI para administrar contraseñas de los usuarios de IAM.

Para crear una contraseña (AWS CLI)

1. (Opcional) Para determinar si un usuario tiene una contraseña, ejecute este comando: aws iam get-login-profile
2. Para crear una contraseña, ejecute este comando: aws iam create-login-profile

Para cambiar la contraseña de un usuario (AWS CLI)

1. (Opcional) Para determinar si un usuario tiene una contraseña, ejecute este comando: aws iam get-login-profile
2. Para cambiar una contraseña, ejecute este comando: aws iam update-login-profile

Para eliminar (deshabilitar) la contraseña de un usuario (AWS CLI)

1. (Opcional) Para determinar si un usuario tiene una contraseña, ejecute este comando: aws iam get-login-profile
2. (Opcional) Para determinar cuándo se utilizó una contraseña por última vez, ejecute este comando: aws iam get-user
3. Para eliminar una contraseña, ejecute este comando: aws iam delete-login-profile

> **Important**
>
> Al eliminar la contraseña de un usuario, este ya no puede iniciar sesión en la AWS Management Console. Si el usuario dispone de claves de acceso activas, estas seguirán funcionando y permitirán el acceso a través de AWS CLI, Tools for Windows PowerShell, o llamadas de función de API de AWS. Cuando se utiliza la AWS CLI, Herramientas para Windows PowerShell o la API de AWS para eliminar un usuario de una Cuenta de AWS, primero se debe eliminar la contraseña mediante esta operación. Para obtener más información, consulte Eliminación de un usuario de IAM (AWS CLI).

Creación, cambio o eliminación de la contraseña de un usuario de IAM (API de AWS)

También puede utilizar la API de la AWS para administrar contraseñas de los usuarios de IAM.

Para crear una contraseña (API de AWS)

1. (Opcional) Para determinar si un usuario tiene una contraseña, llame a esta operación: GetLoginProfile
2. Para crear una contraseña, llame a esta operación: CreateLoginProfile

Para cambiar la contraseña de un usuario (API de AWS)

1. (Opcional) Para determinar si un usuario tiene una contraseña, llame a esta operación: GetLoginProfile
2. Para cambiar una contraseña, llame a esta operación: UpdateLoginProfile

Para eliminar (deshabilitar) la contraseña de un usuario (API de AWS)

1. (Opcional) Para determinar si un usuario tiene una contraseña, ejecute este comando: GetLoginProfile
2. (Opcional) Para determinar cuándo se utilizó una contraseña por última vez, ejecute este comando: GetUser
3. Para eliminar una contraseña, ejecute este comando: DeleteLoginProfile

> **Important**
>
> Al eliminar la contraseña de un usuario, este ya no puede iniciar sesión en la AWS Management Console. Si el usuario dispone de claves de acceso activas, estas seguirán funcionando y permitirán el acceso a través de AWS CLI, Tools for Windows PowerShell, o llamadas de función de API de AWS. Cuando se utiliza la AWS CLI, Herramientas para Windows PowerShell o la API de AWS para eliminar un usuario de una Cuenta de AWS, primero se debe eliminar la contraseña mediante esta operación. Para obtener más información, consulte Eliminación de un usuario de IAM (AWS CLI).

Autorización para que los usuarios de IAM cambien sus contraseñas

> **Note**
>
> Los usuarios con identidades federadas utilizarán el proceso definido por su proveedor de identidades para cambiar sus contraseñas. Como práctica recomendada, exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales.

Puede permitir a los usuarios de IAM que cambien sus propias contraseñas para iniciar sesión en la AWS Management Console. Puede hacerlo de una de las dos formas siguientes:

- Permitir a todos los usuarios de IAM de la cuenta cambiar sus propias contraseñas.
- Permitir solo a usuarios de IAM determinados cambiar sus propias contraseñas. En este caso, desactive la opción para que todos los usuarios cambien sus propias contraseñas y utilice una política de IAM para otorgar permisos solo a algunos usuarios. Este enfoque permite a esos usuarios cambiar sus propias contraseñas y, de manera opcional, otras credenciales, como sus propias claves de acceso.

> **Important**
>
> Se recomienda establecer una política de contraseñas personalizada que requiera que los usuarios de IAM creen contraseñas seguras.

Para permitir que todos los usuarios de IAM cambien sus contraseñas

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, haga clic en Account settings (Configuración de la cuenta).
3. En la sección Password policy (Política de contraseñas), elija Edit (Editar).
4. Elija Custom (Personalizado) para usar una política de contraseñas personalizada.
5. Seleccione Allow users to change their own password (Permitir que los usuarios cambien su propia contraseña) y, a continuación, elija Save changes (Guardar cambios). Esto permite a todos los usuarios de la cuenta acceder a la acción `iam:ChangePassword` solo para su usuario y a la acción `iam:GetAccountPasswordPolicy`.
6. Proporcione a los usuarios las siguientes instrucciones para cambiar sus contraseñas: Cómo un usuario de IAM cambia su propia contraseña.

Para obtener información sobre los comandos de la AWS CLI, Tools for Windows PowerShell y la API que puede utilizar para cambiar la política de contraseñas de la cuenta (que incluye permitir a todos los usuarios cambiar sus propias contraseñas), consulte Configuración de una política de contraseñas (AWS CLI).

Para permitir a usuarios de IAM determinados cambiar sus propias contraseñas

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, haga clic en Account settings (Configuración de la cuenta).
3. En la sección Password policy (Política de contraseñas), asegúrese de que la opción Allow users to change their own password (Permitir que los usuarios cambien su propia contraseña) no esté seleccionada. Si la casilla de verificación está marcada, todos los usuarios podrán cambiar sus contraseñas. (Lea el procedimiento previo.)
4. Cree usuarios que puedan cambiar sus propias contraseñas, si no existen todavía. Para obtener más información, consulte Creación de un usuario de IAM en su Cuenta de AWS.
5. (Opcional) Cree un grupo de IAM para los usuarios que deberían poder cambiar sus contraseñas y, a continuación, agregue los usuarios del paso anterior al grupo. Para obtener más información, consulte Administración de grupos de usuarios de IAM.
6. Asigne la siguiente política al grupo. Para obtener más información, consulte Administración de políticas de IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:GetAccountPasswordPolicy",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:ChangePassword",
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    }
  ]
}
```

Esta política otorga acceso a la acción cambiar contraseña, que permite a los usuarios cambiar únicamente sus propias contraseñas desde la consola, la AWS CLI, Tools for Windows PowerShell o la API. También otorga acceso a la acción GetAccountPasswordPolicy, que permite al usuario ver la política de contraseñas actual. Este permiso es necesario para que el usuario pueda ver la política de contraseñas de cuentas en la página Change Password

(Cambiar contraseña). El usuario debe poder leer la política de contraseñas actual para asegurarse de que la contraseña cambiada cumpla los requisitos de la política.

7. Proporcione a los usuarios las siguientes instrucciones para cambiar sus contraseñas: Cómo un usuario de IAM cambia su propia contraseña.

Para obtener más información

Para obtener más información acerca de cómo administrar credenciales, consulte los siguientes temas:

- Autorización para que los usuarios de IAM cambien sus contraseñas
- Administración de las contraseñas de usuarios en AWS
- Configuración de una política de contraseñas de la cuenta para usuarios de IAM
- Administración de políticas de IAM
- Cómo un usuario de IAM cambia su propia contraseña

Cómo un usuario de IAM cambia su propia contraseña

Si dispone de permiso para cambiar su propia contraseña de usuario de IAM, puede utilizar una página especial en la AWS Management Console para hacerlo. También puede utilizar la AWS CLI o la API de AWS.

Temas

- Permisos necesarios
- Cómo cambian los usuarios de IAM su propia contraseña (consola)
- Cómo cambian los usuarios de IAM su propia contraseña (AWS CLI o API de AWS)

Permisos necesarios

Para cambiar la contraseña de su propio usuario de IAM, debe contar con los permisos de la siguiente política: AWS: permite a los usuarios de IAM cambiar su propia contraseña de consola en la página Credenciales de seguridad.

Cómo cambian los usuarios de IAM su propia contraseña (consola)

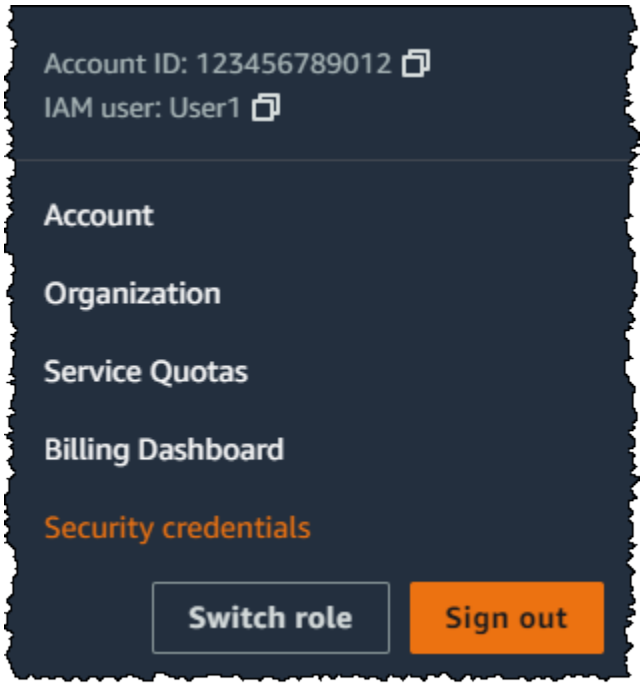
En el siguiente procedimiento se describe cómo un usuario de IAM puede utilizar la AWS Management Console para cambiar su propia contraseña.

Para cambiar la contraseña de su propio usuario de IAM (consola)

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

   > **Note**
   >
   > Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

   Para obtener el ID de la Cuenta de AWS, contacte con su administrador.
2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).



3. En la pestaña Credenciales de AWS IAM, elija Actualizar contraseña.
4. En Current password (Contraseña actual), escriba la contraseña actual. Escriba una contraseña nueva New password (Nueva contraseña) y Confirm new password (Confirmar nueva contraseña). A continuación, elija Actualizar contraseña.

> **Note**
>
> La nueva contraseña debe cumplir los requisitos de la política de contraseñas de cuentas. Para obtener más información, consulte Configuración de una política de contraseñas de la cuenta para usuarios de IAM.

Cómo cambian los usuarios de IAM su propia contraseña (AWS CLI o API de AWS)

En el siguiente procedimiento se describe cómo un usuario de IAM puede utilizar la AWS CLI o la API de AWS para cambiar su propia contraseña.

Para cambiar su propia contraseña de IAM, utilice lo siguiente:

- AWS CLI: `aws iam change-password`
- API de AWS: `ChangePassword`

Administración de las claves de acceso de los usuarios de IAM

Follow us on Twitter

> **Important**
>
> Como práctica recomendada, utilice credenciales de seguridad temporales (por ejemplo, roles de IAM) en lugar de crear credenciales a largo plazo como claves de acceso. Antes de crear claves de acceso, revise las alternativas a las claves de acceso a largo plazo.

Las claves de acceso son credenciales a largo plazo para un usuario de IAM o el Usuario raíz de la cuenta de AWS. Puede utilizar las claves de acceso para firmar solicitudes mediante programación a la AWS CLI o a la API de AWS (directamente o mediante el SDK de AWS). Para obtener más información, consulte Firma de solicitudes API de AWS.

Las claves de acceso se componen de dos partes: un ID de clave de acceso (por ejemplo, `AKIAIOSFODNN7EXAMPLE`) y una clave de acceso secreta (por ejemplo, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Debe utilizar el ID de clave de acceso y la clave de acceso secreta juntos, como un nombre de usuario y contraseña, para autenticar sus solicitudes.

Cuando cree un par de claves de acceso, guarde el ID de clave de acceso y la clave de acceso secreta en un lugar seguro. La clave de acceso secreta solo está disponible en el momento de su creación. Si pierde su clave de acceso secreta, debe eliminar la clave de acceso y crear una nueva. Para obtener más información, consulte Restablecimiento de claves de acceso o contraseñas perdidas u olvidadas para AWS.

Puede tener un máximo de dos claves de acceso por usuario.

> **Important**
>
> Administre las claves de acceso de forma segura. No proporcione sus claves de acceso a terceros no autorizados, ni siquiera para que le ayuden a buscar sus identificadores de cuenta. Si lo hace, podría conceder a otra persona acceso permanente a su cuenta.

En las siguientes secciones, se detallan las tareas de administración asociadas a las claves de acceso.

Temas

- Permisos requeridos para administrar claves de acceso
- Administración de claves de acceso (consola)
- Administración de las claves de acceso (AWS CLI)
- Administración de las claves de acceso (API de AWS)
- Actualización de las claves de acceso
- Protección de las claves de acceso
- Auditoría de las claves de acceso

Permisos requeridos para administrar claves de acceso

> **Note**
>
> `iam:TagUser` es un permiso opcional para agregar y editar las descripciones de la clave de acceso. Para obtener más información, consulte Etiquetado de usuarios de IAM

Para crear claves de acceso para su usuario de IAM, debe contar con los permisos de la siguiente política:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Para actualizar claves de acceso para su propio usuario de IAM, debe contar con los permisos de la siguiente política:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:GetAccessKeyLastUsed",
                "iam:GetUser",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:TagUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

Administración de claves de acceso (consola)

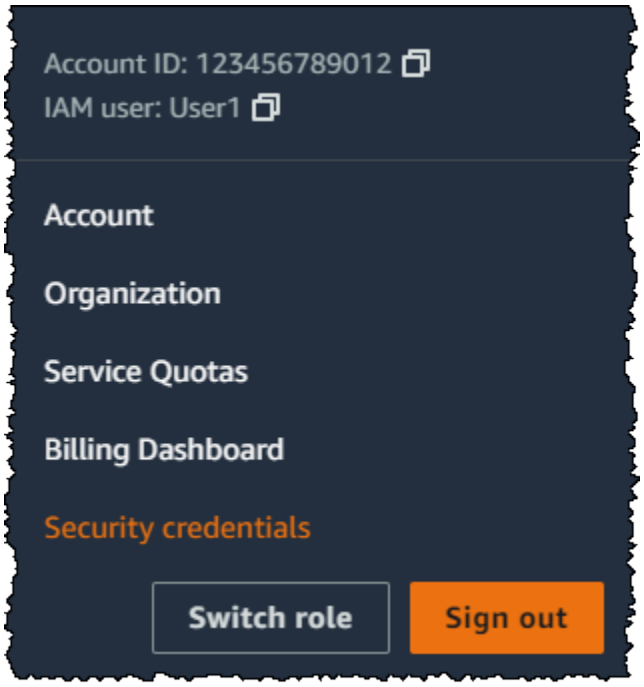
Puede utilizar la AWS Management Console para administrar las claves de acceso de un usuario de IAM.

Para crear, modificar o eliminar sus propias claves de acceso (consola)

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

   > **Note**
   >
   > Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

   Para obtener el ID de la Cuenta de AWS, contacte con su administrador.
2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).



Realice una de las acciones siguientes:

Para crear una clave de acceso

1. En la sección Claves de acceso, haga clic en Crear clave de acceso. Si ya dispone de dos claves de acceso, este botón estará desactivado y deberá eliminar una clave de acceso antes de poder crear una nueva.
2. En la página Access key best practices & alternatives (Prácticas recomendadas y alternativas para la clave de acceso), elija su caso de uso para conocer las opciones adicionales que pueden ayudarle a evitar la creación de una clave de acceso de larga duración. Si determina que su caso de uso aún requiere una clave de acceso, elija Other (Otro) y, a continuación, Next (Siguiente).
3. (Opcional) Establezca un valor de etiqueta de descripción para la clave de acceso. Esto agrega un par clave-valor de etiqueta a su usuario de IAM. Esto puede ayudarlo a identificar y actualizar claves de acceso más adelante. La clave de la etiqueta se establece en el ID de la clave de acceso. El valor de la etiqueta se establece en la descripción de la clave de acceso que especifique. Cuando haya terminado, seleccione Create access key (Crear clave de acceso).
4. En la página Retrieve access keys (Recuperar claves de acceso), elija Show (Mostrar) para revelar el valor de la clave de acceso secreta de su usuario o Download .csv file (Descargar archivo .csv). Esta es su única oportunidad de guardar su clave de acceso secreta. Una vez guardada la clave de acceso secreta en un lugar seguro, seleccione Done (Listo).

Para desactivar una clave de acceso

- En la sección Access keys (Claves de acceso), busque la clave que desea desactivar, seleccione Actions (Acciones) y, a continuación, seleccione Deactivate (Desactivar). Cuando se le pida confirmación, elija Deactivate (Desactivar). Una clave de acceso desactivada sigue contando para el límite de dos claves de acceso.

Para activar una clave de acceso

- En la sección Access keys (Claves de acceso), busque la clave que desea activar, seleccione Actions (Acciones) y, a continuación, Activate (Activar).

Para eliminar una clave de acceso cuando ya no la necesite

- En la sección Access keys (Claves de acceso), busque la clave que desea eliminar, seleccione Actions (Acciones) y, a continuación, Eliminar. Siga las instrucciones del cuadro de diálogo para, en primer lugar, Deactivate (Desactivar) y, a continuación, confirmar la eliminación. Le recomendamos que compruebe que la clave de acceso ya no se utilice antes de eliminarla definitivamente.

Para crear, modificar o eliminar claves de acceso de otro usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Seleccione el nombre del usuario cuyas claves de acceso que desee administrar y, a continuación, elija la pestaña Security credentials (Credenciales de seguridad).
4. En la sección Access Keys (Claves de acceso), ejecute una de las acciones siguientes:
   - Para crear una clave de acceso, elija Create access key (Crear clave de acceso). Si el botón se encuentra desactivado, deberá borrar una de las claves existentes antes de poder crear una nueva. En la página Access key best practices & alternatives (Prácticas recomendadas y alternativas para la clave de acceso), revise las prácticas recomendadas y las alternativas. Elija su caso de uso para conocer las opciones adicionales que pueden ayudarle a evitar la creación de una clave de acceso a largo plazo. Si determina que su caso de uso aún requiere una clave de acceso, elija Other (Otro) y, a continuación, Next (Siguiente). En la página Retrieve access key page (Recuperar clave de acceso), elija Show (Mostrar) para revelar el valor de la clave de acceso secreta de su usuario. Para guardar el ID de la clave de acceso y la clave de acceso secreta en un archivo `.csv` en una ubicación segura de su ordenador, seleccione el botón Download .csv file (Descargar archivo .csv). Cuando se crea una clave de acceso para un usuario, el par de claves se activa de forma predeterminada y el usuario puede utilizarlo inmediatamente.
   - Para desactivar una clave de acceso activa, seleccione Actions (Acciones) y, a continuación, Deactivate (Desactivar).
   - Para activar una clave de acceso inactiva, seleccione Actions (Acciones) y, a continuación, Activate (Activar).
   - Para eliminar una clave de acceso, seleccione Actions (Acciones) y, a continuación, Delete (Eliminar). Siga las instrucciones del diálogo para primero Deactivate (Desactivar) y, a

continuación, confirmar la eliminación. AWS recomienda que, antes de hacerlo, desactive la clave y compruebe que ya no se usa. Cuando utilice la AWS Management Console, deberá desactivar la clave antes de eliminarla.

Para obtener una lista de las claves de acceso de un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Elija el nombre del usuario que desee y, a continuación, elija la pestaña Security credentials (Credenciales de seguridad). En la sección Access keys (Claves de acceso), verá las claves de acceso del usuario y el estado de cada una.

> **Note**
>
> Solo se ve el ID de clave de acceso del usuario. La clave de acceso secreta solo se puede recuperar cuando se crea la clave.

Para generar una lista de los ID de clave de acceso de varios usuarios de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Si es necesario, añada la columna Access key ID (ID de clave de acceso) a la tabla de usuarios ejecutando los siguientes pasos:
   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ( ).
   b. En Manage columns (Administrar columnas), seleccione Access key ID (ID de clave de acceso).
   c. Seleccione Close (Cerrar) para volver a la lista de usuarios.
4. La columna Access key ID (ID de clave de acceso) muestra los ID de clave de acceso seguidos de su estado; por ejemplo, 234782070278420732307623740 23 (Active) (Activo) o 220937402396702370248434203 27 (Inactive) (Inactivo).

Puede utilizar esta información para ver y copiar las claves de acceso de los usuarios que tengan una o dos claves de acceso. La columna muestra None (Ninguna) cuando los usuarios no tienen clave de acceso.

> **Note**
>
> Solo se ve el ID de clave de acceso y el estado del usuario. La clave de acceso secreta solo se puede recuperar cuando se crea la clave.

Para saber qué usuario de IAM posee una clave de acceso específica (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. En el campo de búsqueda, escriba o pegue el ID de clave de acceso del usuario que desea encontrar.
4. Si es necesario, añada la columna Access key ID (ID de clave de acceso) a la tabla de usuarios ejecutando los siguientes pasos:
   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ( ).
   b. En Manage columns (Administrar columnas), seleccione Access key ID (ID de clave de acceso).
   c. Elija Close (Cerrar) para volver a la lista de usuarios y confirmar que el usuario filtrado posee la clave de acceso especificada.

Administración de las claves de acceso (AWS CLI)

Para administrar las claves de acceso de un usuario de IAM con la AWS CLI, ejecute los siguientes comandos.

- Para crear una clave de acceso: `aws iam create-access-key`
- Para desactivar o activar una clave de acceso: `aws iam update-access-key`
- Para generar una lista de las claves de acceso de un usuario: `aws iam list-access-keys`

- Para determinar cuándo se utilizó por última vez una clave de acceso: `aws iam get-access-key-last-used`
- Para eliminar una clave de acceso: `aws iam delete-access-key`

Administración de las claves de acceso (API de AWS)

Para administrar las claves de acceso de un usuario de IAM con la API de AWS, ejecute las siguientes operaciones.

- Para crear una clave de acceso: `CreateAccessKey`
- Para desactivar o activar una clave de acceso: `UpdateAccessKey`
- Para generar una lista de las claves de acceso de un usuario: `ListAccessKeys`
- Para determinar cuándo se utilizó por última vez una clave de acceso: `GetAccessKeyLastUsed`
- Para eliminar una clave de acceso: `DeleteAccessKey`

Actualización de las claves de acceso

Como práctica recomendada de seguridad, se recomienda actualizar las claves de acceso de usuario de IAM cuando sea necesario; por ejemplo, cuando un empleado deje la empresa. Los usuarios de IAM pueden actualizar sus propias claves de acceso si se les han concedido los permisos necesarios.

Para obtener más información sobre cómo conceder a sus usuarios de IAM permisos para actualizar sus propias claves de acceso, consulte AWS: permite a los usuarios de IAM administrar su propia contraseña, sus claves de acceso y sus claves públicas SSH en la página Credenciales de seguridad. También puede aplicar una política de contraseñas a su cuenta para solicitarles a todos los usuarios de IAM que actualicen sus contraseñas periódicamente, e informarles cuán seguido deben hacerlo. Para obtener más información, consulte Configuración de una política de contraseñas de la cuenta para usuarios de IAM.

Temas

- Actualización de las claves de acceso de usuario de IAM (consola)
- Actualización de las claves de acceso (AWS CLI)
- Actualización de las claves de acceso (API de AWS)

Actualización de las claves de acceso de usuario de IAM (consola)

Puede actualizar las claves de acceso desde la AWS Management Console.

Para actualizar las claves de acceso de un usuario de IAM sin interrumpir sus aplicaciones (consola)

1. Aunque la primera clave de acceso sigue activa, cree otra clave de acceso.
   a. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
   b. En el panel de navegación, seleccione Usuarios.
   c. Elija el nombre del usuario que desee y, a continuación, elija la pestaña Security credentials (Credenciales de seguridad).
   d. En la sección Claves de acceso, haga clic en Crear clave de acceso. En la página Access key best practices & alternatives (Prácticas recomendadas y alternativas para la clave de acceso), seleccione Other (Otros) y, a continuación, Next (Siguiente).
   e. (Opcional) Establezca un valor de etiqueta de descripción para la clave de acceso para agregar un par clave-valor de etiqueta a este usuario de IAM. Esto puede ayudarlo a identificar y actualizar claves de acceso más adelante. La clave de la etiqueta se establece en el ID de la clave de acceso. El valor de la etiqueta se establece en la descripción de la clave de acceso que especifique. Cuando haya terminado, seleccione Create access key (Crear clave de acceso).
   f. En la página Retrieve access keys (Recuperar claves de acceso), elija Show (Mostrar) para revelar el valor de la clave de acceso secreta de su usuario o Download .csv file (Descargar archivo .csv). Esta es su única oportunidad de guardar su clave de acceso secreta. Una vez guardada la clave de acceso secreta en un lugar seguro, seleccione Done (Listo).

      Cuando se crea una clave de acceso para un usuario, el par de claves se activa de forma predeterminada y el usuario puede utilizarlo inmediatamente. En este punto, el usuario tiene dos claves de acceso activas.
2. Actualice todas las aplicaciones y herramientas para utilizar la nueva clave de acceso.
3. Para determinar si la primera clave de acceso todavía está en uso, consulte la información Last used (Último uso) de la clave de acceso más antigua. Un enfoque consiste en esperar varios días y después comprobar si se ha usado la clave de acceso antigua antes de continuar.
4. Aunque la información de Last used (Último uso) indique que la clave antigua nunca se ha utilizado, le recomendamos que no elimine inmediatamente la primera clave de acceso. En

su lugar, elija Actions (Acciones) y, a continuación, seleccione Deactivate (Desactivar) para desactivar la primera clave de acceso.

5. Utilice únicamente la clave de acceso nueva para confirmar que sus aplicaciones funcionan. Todas las aplicaciones y herramientas que sigan utilizando la clave de acceso original dejarán de funcionar en este momento, ya que ya no podrán obtener acceso a los recursos de AWS. Si encuentra una aplicación o herramienta de este tipo, puede reactivar la primera clave de acceso. A continuación, vuelva a Step 3 y actualice esta aplicación para utilizar la nueva clave.
6. Después de esperar un tiempo para asegurarse de que todas las aplicaciones y herramientas se hayan actualizado, podrá eliminar la primera clave de acceso:

   a. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
   b. En el panel de navegación, seleccione Usuarios.
   c. Elija el nombre del usuario que desee y, a continuación, elija la pestaña Security credentials (Credenciales de seguridad).
   d. En la sección Access keys (Claves de acceso) de la clave de acceso que desea eliminar, seleccione Actions (Acciones) y, a continuación, Delete (Eliminar). Siga las instrucciones del cuadro de diálogo para, en primer lugar, Deactivate (Desactivar) y, a continuación, confirmar la eliminación.

Para determinar qué claves de acceso deben actualizarse o eliminarse (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Si es necesario, añada la columna Access key age (Antigüedad de la clave de acceso) a la tabla de usuarios ejecutando los siguientes pasos:

   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ().
   b. En Manage columns (Administrar columnas), seleccione Access key age (Antigüedad de la clave de acceso).
   c. Seleccione Close (Cerrar) para volver a la lista de usuarios.
4. La columna Access key age (Antigüedad de la clave de acceso) muestra el número de días que han transcurrido desde la creación de la clave de acceso activa más antigua. Puede utilizar esta

información para encontrar usuarios con claves de acceso que deban actualizarse o eliminarse. La columna muestra None (Ninguna) cuando los usuarios no tienen clave de acceso.

Actualización de las claves de acceso (AWS CLI)

Puede actualizar las claves de acceso desde la AWS Command Line Interface.

Para actualizar las claves de acceso sin interrumpir sus aplicaciones (AWS CLI)

1. Aunque la primera clave de acceso sigue activa, cree otra clave de acceso que, de forma predeterminada, está activa. Ejecute el siguiente comando:

   - `aws iam create-access-key`

   En este punto, el usuario tiene dos claves de acceso activas.
2. Actualice todas las aplicaciones y herramientas para utilizar la nueva clave de acceso.
3. Determine si la primera clave de acceso todavía está en uso utilizando este comando:

   - `aws iam get-access-key-last-used`

   Un enfoque consiste en esperar varios días y después comprobar si se ha usado la clave de acceso antigua antes de continuar.
4. Aunque el paso Step 3 indique que la clave antigua no se usa, le recomendamos que no elimine inmediatamente la primera clave de acceso. En su lugar, cambie el estado de la primera clave de acceso a `Inactive` utilizando este comando:

   - `aws iam update-access-key`
5. Utilice únicamente la clave de acceso nueva para confirmar que sus aplicaciones funcionan. Todas las aplicaciones y herramientas que sigan utilizando la clave de acceso original dejarán de funcionar en este momento, ya que ya no podrán obtener acceso a los recursos de AWS. Si se encuentra con una de estas aplicaciones o herramientas, puede cambiar de nuevo su estado a `Active` para volver a activar la primera clave de acceso. A continuación, vuelva al paso Step 2 y actualice esta aplicación para utilizar la nueva clave.
6. Después de esperar un tiempo para asegurarse de que todas las aplicaciones y herramientas se hayan actualizado, podrá eliminar la primera clave de acceso con este comando:

   - `aws iam delete-access-key`

Actualización de las claves de acceso (API de AWS)

Puede actualizar las claves de acceso con la API de AWS.

Para actualizar claves de acceso sin interrumpir sus aplicaciones (API de AWS)

1. Aunque la primera clave de acceso sigue activa, cree otra clave de acceso que, de forma predeterminada, está activa. Llame a la operación siguiente:

   - `CreateAccessKey`

   En este punto, el usuario tiene dos claves de acceso activas.
2. Actualice todas las aplicaciones y herramientas para utilizar la nueva clave de acceso.
3. Determine si la primera clave de acceso todavía está en uso llamando a esta operación:

   - `GetAccessKeyLastUsed`

   Un enfoque consiste en esperar varios días y después comprobar si se ha usado la clave de acceso antigua antes de continuar.
4. Aunque el paso Step 3 indique que la clave antigua no se usa, le recomendamos que no elimine inmediatamente la primera clave de acceso. En su lugar, cambie el estado de la primera clave de acceso a `Inactive` llamando a esta operación:

   - `UpdateAccessKey`
5. Utilice únicamente la clave de acceso nueva para confirmar que sus aplicaciones funcionan. Todas las aplicaciones y herramientas que sigan utilizando la clave de acceso original dejarán de funcionar en este momento, ya que ya no podrán obtener acceso a los recursos de AWS. Si se encuentra con una de estas aplicaciones o herramientas, puede cambiar de nuevo su estado a `Active` para volver a activar la primera clave de acceso. A continuación, vuelva al paso Step 2 y actualice esta aplicación para utilizar la nueva clave.
6. Después de esperar un tiempo para asegurarse de que todas las aplicaciones y herramientas se hayan actualizado, podrá eliminar la primera clave de acceso llamando a esta operación:

   - `DeleteAccessKey`

Protección de las claves de acceso

Cualquier persona que tenga su clave de acceso disfrutará del mismo nivel de acceso a los recursos de AWS que usted. Por lo tanto, AWS adopta importantes medidas para proteger las claves de acceso y, en consonancia con nuestro modelo de responsabilidad compartida, también usted debería adoptarlas.

Amplíe las siguientes secciones para obtener orientación que lo ayude a proteger sus claves de acceso.

> **Note**
>
> Puede que su organización tenga políticas y requisitos de seguridad distintos de los descritos en este tema. Las sugerencias proporcionadas aquí pretenden ser directrices generales.

Eliminar (o no generar) claves de acceso Usuario raíz de la cuenta de AWS

Una de las mejores formas de proteger su cuenta es no tener una clave de acceso para su Usuario raíz de la cuenta de AWS. A menos que necesite tener una clave de acceso de usuario raíz (lo que es poco frecuente), es mejor no generarla. En su lugar, cree un usuario administrativo en AWS IAM Identity Center para las tareas administrativas diarias. Para obtener más información sobre cómo crear un usuario administrativo en IAM Identity Center, consulte Introducción en la Guía del usuario de IAM Identity Center.

Si ya tiene una clave de acceso de usuario raíz para su cuenta, le recomendamos hacer lo siguiente: buscar lugares en las aplicaciones donde utiliza dicha clave actualmente (si procede) y sustituir la clave de acceso de usuario raíz por una clave de acceso de usuario de IAM. Luego deshabilite y elimine la clave de acceso de usuario raíz. Para obtener más información sobre cómo actualizar claves de acceso, consulte Actualización de las claves de acceso.

Utilice credenciales de seguridad temporales (roles de IAM) en lugar de claves de acceso a largo plazo

En muchos casos, no necesita claves de acceso a largo plazo que nunca caducan (como sucede con un usuario de IAM). En su lugar, puede crear roles de IAM y generar credenciales de seguridad temporales. Las credenciales de seguridad temporales se componen de un ID de clave de acceso y una clave de acceso secreta, pero, además, incluyen un token de seguridad que indica cuándo caducan las credenciales.

Las claves de acceso a largo plazo, como las asociadas a los usuarios de IAM y al usuario raíz, siguen siendo válidas hasta que se revocan manualmente. No obstante, las credenciales de seguridad temporales obtenidas a través de roles de IAM y otras características de AWS Security Token Service caducan tras un breve periodo de tiempo. Utilice las credenciales de seguridad temporales para ayudar a reducir el riesgo en caso de se vean expuestas accidentalmente.

Utilice un rol de IAM y credenciales de seguridad temporales en las siguientes situaciones:

- Si tiene una aplicación o scripts de AWS CLI que se ejecutan en una instancia EC2. No utilice claves de acceso directamente en su aplicación. No transfiera una clave de acceso a la aplicación, no la integre en la aplicación y no deje que la aplicación lea claves de cualquier origen. En cambio, defina un rol de IAM que tenga los permisos adecuados para su aplicación y lance la instancia Amazon Elastic Compute Cloud (Amazon EC2) con roles para EC2. Al hacerlo, se asocia un rol de IAM a la instancia de Amazon EC2. Esta práctica también habilita a la aplicación para obtener credenciales de seguridad temporales que, a su vez, puede usar para realizar llamadas mediante programación a AWS. AWS SDK y AWS Command Line Interface (AWS CLI) pueden obtener credenciales temporales del rol automáticamente.
- Debe conceder acceso entre cuentas. Utilice un rol de IAM para establecer la confianza entre cuentas y, a continuación, conceder a los usuarios de una cuenta permisos limitados para acceder a la cuenta de confianza. Para obtener más información, consulte Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.
- Tiene una aplicación móvil. No integre una clave de acceso en la aplicación, ni siquiera en el almacenamiento cifrado. En su lugar, utilice Amazon Cognito para administrar identidades de los usuarios en su aplicación. Este servicio permite autenticar a los usuarios mediante Login with Amazon, Facebook, Google o cualquier proveedor de identidad compatible con OpenID Connect (OIDC). A continuación, puede utilizar el proveedor de credenciales de Amazon Cognito a fin de administrar las credenciales que la aplicación utiliza para realizar solicitudes a AWS.
- Desea utilizar la federación en AWS y su organización admite SAML 2.0. Si trabaja para una organización que tiene un proveedor de identidad compatible con SAML 2.0, configure el proveedor para que use SAML. Puede utilizar SAML para intercambiar información de autenticación con AWS y recuperar un conjunto de credenciales de seguridad temporales. Para obtener más información, consulte Federación SAML 2.0.
- Desea utilizar la federación en AWS y su organización tiene un almacén de identidades local. Si los usuarios pueden autenticarse dentro de su organización, puede escribir una aplicación que emita credenciales de seguridad temporales para obtener acceso a los recursos de AWS. Para

obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

> **Note**
>
> ¿Está utilizando una instancia de Amazon EC2 con una aplicación que requiere acceso a los recursos de AWS? Si es así, utilice roles de IAM para EC2.

Administración correcta de las claves de acceso de usuario de IAM

Si debe crear claves de acceso para el acceso mediante programación a AWS, créelas para los usuarios de IAM y conceda a los usuarios solo los permisos que necesitan.

Tenga en cuenta estas precauciones para ayudar a proteger las claves de acceso de usuario de IAM:

- No integre las claves de acceso directamente en el código. Los AWS SDK de y las Herramientas de línea de comandos de AWS le permiten colocar las claves de acceso en ubicaciones conocidas para que no tenga que mantenerlas en código.

  Ponga las claves de acceso en una de las siguientes ubicaciones:

  - El archivo de credenciales de AWS. Los SDK de AWS y la AWS CLI utilizan automáticamente las credenciales que se guardan en el archivo de credenciales de AWS.

    Para obtener información acerca de cómo utilizar el archivo de credenciales de AWS, consulte la documentación del SDK. Los ejemplos incluyen Conjunto de AWS credenciales y región en la Guía de desarrollo de AWS SDK for Java y Archivos de configuración y credenciales en la Guía del usuario de AWS Command Line Interface.

    Para almacenar las credenciales para AWS SDK for .NET y AWS Tools for Windows PowerShell, recomendamos utilizar la tienda del SDK. Para obtener más información, consulte Uso de la tienda del SDK en la Guía de desarrollo de AWS SDK for .NET.
  - Variables de entorno. En un sistema multitenencia, opte por las variables de entorno de usuario, en lugar de las variables de entorno de sistema.

    Para obtener más información acerca de cómo utilizar las variables de entorno para almacenar credenciales, consulte Variables de entorno en la Guía del usuario de AWS Command Line Interface.

- Utilice claves de acceso distintas para las diferentes aplicaciones. Hacer esto le permitirá aislar los permisos y revocar las claves de acceso para aplicaciones individuales si se ven expuesta. Tener claves de acceso separadas para diferentes aplicaciones también genera entradas distintas en los archivos de registro de AWS CloudTrail. Esta configuración hace más sencillo determinar qué aplicación realizó acciones concretas.
- Actualice las claves de acceso cuando sea necesario. Si existe el riesgo de que la clave de acceso se vea comprometida, actualícela y elimine la anterior. Para obtener más información, consulte Actualización de las claves de acceso
- Elimine las claves de acceso no utilizadas. Si un usuario deja la organización, elimine el usuario de IAM correspondiente, de tal forma que ya no pueda obtener acceso a los recursos. Para saber cuándo se utilizó por última vez una clave de acceso, utilice la API `GetAccessKeyLastUsed` (comando de AWS CLI: `aws iam get-access-key-last-used`).
- Utilice las credenciales temporales y configure la autenticación multifactor para las operaciones de API más confidenciales. Con las políticas de IAM, puede especificar qué operaciones de API puede llamar un usuario. En algunos casos, es posible que quiera la seguridad adicional de exigir a los usuarios que se autentiquen con MFA de AWS antes de permitirles llevar a cabo acciones especialmente confidenciales. Por ejemplo, es posible que tenga una política que permita a un usuario realizar las acciones de Amazon EC2 `RunInstances`, `DescribeInstances` y de `StopInstances`. Sin embargo, es posible que quiera restringir una acción destructiva, como `TerminateInstances` y asegurarse de que los usuarios solo pueden realizar esta acción si se autentican mediante un dispositivo MFA de AWS. Para obtener más información, consulte Configuración del acceso a una API protegido por MFA.

Acceder a la aplicación móvil usando claves de acceso de AWS

Puede acceder a un conjunto limitado de servicios y características de AWS mediante la aplicación móvil de AWS. La aplicación móvil le ayuda a dar soporte a la respuesta frente a incidentes mientras está en movimiento. Para obtener más información y descargar la aplicación, consulte Aplicación móvil de la consola de AWS.

Puede iniciar sesión en la aplicación móvil con la contraseña de la consola o las claves de acceso. Como práctica recomendada, no utilice las clave de acceso de usuario raíz. En su lugar, le recomendamos encarecidamente que, además de utilizar una contraseña o un bloqueo biométrico en su dispositivo móvil, cree un usuario de IAM específicamente para administrar los recursos de AWS mediante la aplicación móvil. Si pierde su dispositivo móvil, puede eliminar el acceso del usuario de IAM.

Para iniciar sesión con las teclas de acceso (aplicación móvil)

1. Abra la aplicación en su dispositivo móvil.
2. Si es la primera vez que agrega una identidad al dispositivo, elija Add an identity (Agregar una identidad) y, a continuación, elija Access keys (Teclas de acceso).

   Si ya ha iniciado sesión con otra identidad, elija el icono de menú y elija Switch identity (Cambiar identidad). A continuación, elija Sign in as a different identity (Iniciar sesión con una identidad diferente) y, a continuación, Access keys (Teclas de acceso).
3. En la página Access keys (Claves de acceso) introduzca su información:

   - ID de clave de acceso: introduzca el ID de clave de acceso.
   - Clave de acceso secreta: introduzca la clave de acceso secreta.
   - Nombre de identidad: introduzca el nombre de la identidad que aparecerá en la aplicación móvil. No es necesario que coincida con su nombre de usuario de IAM.
   - PIN de identidad: cree un número de identificación personal (PIN) que utilizará en los futuros inicios de sesión.

   > **Note**
   >
   > Si habilita la biometría para la aplicación móvil de AWS, se le pedirá que utilice su huella digital o reconocimiento facial para la verificación en lugar del PIN. Si la biometría falla, es posible que se le pida el PIN en su lugar.
4. Elija Verify and add keys (Verificar y agregue claves).

   Ahora puede acceder a un conjunto selecto de sus recursos mediante la aplicación móvil.

Información relacionada

En las siguientes secciones, se proporciona información sobre cómo configurar los AWS SDK y la AWS CLI para utilizar claves de acceso:

- Conjunto de AWS credenciales y región en la Guía para desarrolladores de AWS SDK for Java
- Uso de la tienda del SDK en la Guía para desarrolladores de AWS SDK for .NET.
- Proporcione credenciales al SDK en la Guía para desarrolladores de AWS SDK for PHP.
- Configuración en la documentación de Boto 3 (AWS SDK para Python).

- Using AWS Credentials en la Guía del usuario de AWS Tools for Windows PowerShell
- Archivos de configuración y credenciales en la Guía del usuario de AWS Command Line Interface.
- Conceder acceso mediante un rol de IAM en la Guía AWS SDK for .NETpara desarrolladores
- Configure los roles de IAM para Amazon EC2 en el AWS SDK for Java 2.x

Auditoría de las claves de acceso

Puede revisar las claves de acceso de AWS en su código para determinar si las claves proceden de una cuenta de su propiedad. Puede transferir un ID de clave de acceso mediante el comando de la AWS CLI `aws sts get-access-key-info` o la operación de la API de AWS `GetAccessKeyInfo`.

Las operaciones de AWS CLI y de la API de AWS devuelven el ID de la cuenta de Cuenta de AWS a la que pertenece la clave de acceso. Los ID de clave de acceso que comienzan por `AKIA` son credenciales a largo plazo para un usuario de IAM o un Usuario raíz de la cuenta de AWS. Los ID de clave de acceso que comienzan por `ASIA` son credenciales temporales que se crean mediante operaciones de AWS STS. Si la cuenta de la respuesta le pertenece, puede iniciar sesión como usuario raíz y revisar las claves de acceso de usuario raíz. A continuación, puede extraer un informe de credenciales para saber qué usuario de IAM es el propietario de las claves. Para saber quién solicitó las credenciales temporales para una clave de acceso `ASIA`, consulte los eventos de AWS STS en los registros de CloudTrail.

Por motivos de seguridad, puede revisar los registros de AWS CloudTrail para saber quién realizó una acción en AWS. Puede utilizar la clave de condición de `sts:SourceIdentity` en la política de confianza de rol para exigir a los usuarios que especifiquen una identidad cuando asuman un rol. Por ejemplo, puede requerir que los usuarios de IAM especifiquen su propio nombre de usuario como su identidad de origen. Esto puede ayudarle a determinar qué usuario realizó una acción específica en AWS. Para obtener más información, consulte `sts:SourceIdentity`.

Esta operación no indica el estado de la clave de acceso. La clave podría estar activa, inactiva o eliminada. Es posible que las claves activas no tengan permisos para realizar una operación. Proporcionar una clave de acceso eliminada podría devolver un error que indicara que la clave no existe.

Restablecimiento de claves de acceso o contraseñas perdidas u olvidadas para AWS

> **Important**
>
> ¿Tiene problemas para iniciar sesión en AWS? Asegúrese de que está en la página de inicio de sesión de AWS correcta para su tipo de usuario. Si es el Usuario raíz de la cuenta de AWS (propietario de la cuenta), puede iniciar sesión en AWS con las credenciales que configuró cuando creó la Cuenta de AWS. Si es usuario de IAM, el administrador de su cuenta puede proporcionarle las credenciales que puede utilizar para iniciar sesión en AWS. Si necesita solicitar soporte técnico, no utilice el enlace de comentarios de esta página, ya que el formulario lo recibe el equipo de documentación de AWS, no AWS Support. En lugar de ello, en la página Contacte con nosotros, elija Todavía no es posible iniciar sesión en la cuenta de AWS y, a continuación, elija una de las opciones de asistencia disponibles.

En la página principal de inicio de sesión, debe especificar su dirección de correo electrónico para iniciar sesión como usuario raíz o bien escribir su ID de cuenta para iniciar sesión como usuario de IAM. Solo puede proporcionar su contraseña en la página de inicio de sesión que coincide con su tipo de usuario. Para obtener más información, consulte Inicio de sesión en la AWS Management Console.

Si se encuentra en la página de inicio de sesión correcta y pierde u olvida sus contraseñas o claves de acceso, no podrá recuperarlas desde IAM. En lugar de ello, puede restablecerlas utilizando los siguientes métodos:

- Contraseña de Usuario raíz de la cuenta de AWS: si olvida su contraseña de usuario raíz, puede restablecer la contraseña desde la AWS Management Console. Para obtener información detallada, consulte the section called “Restablecimiento de una contraseña de usuario raíz perdida u olvidada” más adelante en este tema.
- Claves de acceso de la cuenta de Cuenta de AWS. Si olvida las claves de acceso de su cuenta, puede crear nuevas claves de acceso sin deshabilitar las existentes. Si no va a utilizar las claves existentes, puede eliminarlas. Para más detalles, consulte Creación de claves de acceso para el usuario raíz y Eliminación de claves de acceso para el usuario raíz.
- Contraseña de usuario de IAM - Si es un usuario de IAM y olvida la contraseña, debe pedir al administrador que restablezca la contraseña. Para obtener información sobre cómo un

administrador puede administrar su contraseña, consulte Administración de las contraseñas de los usuarios de IAM.

- Claves de acceso de usuario de IAM - Si es un usuario de IAM y olvida sus claves de acceso, necesitará nuevas claves de acceso. Si tiene permiso para crear sus propias claves de acceso, puede encontrar instrucciones sobre cómo crear una en Administración de claves de acceso (consola). Si no dispone de los permisos necesarios, debe pedir al administrador que cree nuevas claves de acceso. Si continúa utilizando sus claves antiguas, pida a su administrador que no las elimine. Para obtener información sobre cómo un administrador puede administrar sus claves de acceso, consulte Administración de las claves de acceso de los usuarios de IAM.

Uso de autenticación multifactor (MFA) en AWS

Follow us on Twitter

Para más seguridad, le recomendamos que configure la autenticación multifactor (MFA) para ayudar a proteger sus recursos de AWS. Puede habilitar MFA para el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Cuando habilita MFA para el usuario raíz, esto solo afecta a las credenciales del usuario raíz. Los usuarios de IAM de la cuenta son identidades diferenciadas que tienen sus propias credenciales, y cada identidad tiene su propia configuración de MFA. Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Para más información sobre los tipos de MFA admitidos, consulte ¿Qué es MFA?. Con múltiples dispositivos de MFA, solo se necesita un dispositivo de MFA para iniciar sesión en la AWS Management Console o crear una sesión a través de la AWS CLI como ese usuario.

> **Note**
>
> Se recomienda exigir a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. ¿Ha considerado la posibilidad de usar AWS IAM Identity Center? Puede usar IAM Identity Center para administrar de forma centralizada el acceso a múltiples Cuentas de AWS y proporcionar a los usuarios un acceso protegido por MFA y de inicio de sesión único a todas sus cuentas asignadas desde un solo lugar. Con IAM Identity Center, puede crear y administrar identidades de usuario en IAM Identity Center o conectarse fácilmente a su proveedor de identidades existente compatible con SAML 2.0. Para obtener más información, consulte ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center.

¿Qué es MFA?

MFA aporta seguridad adicional, ya que exige a los usuarios que proporcionen una autenticación exclusiva obtenida de un mecanismo de MFA admitido por AWS, además de las credenciales de inicio de sesión habituales cuando se accede a sitios web o servicios de AWS. AWS es compatible con los siguientes tipos de MFA.

Seguridad FIDO

Las claves de seguridad de hardware certificadas por FIDO las proporcionan proveedores externos.

FIDO Alliance mantiene una lista de todos los productos certificados por FIDO que son compatibles con las especificaciones de FIDO. Los estándares de autenticación FIDO se basan en la criptografía de clave pública, que permite una autenticación sólida y resistente a la suplantación de identidad que es más segura que las contraseñas. Las claves de seguridad de FIDO admiten múltiples cuentas raíz y usuarios de IAM que utilicen una única clave de seguridad. Para más información acerca de la habilitación de las claves de seguridad con FIDO, consulte Habilitación de una clave de seguridad FIDO (consola).

Dispositivos MFA virtuales

Aplicación de autenticador virtual que se ejecuta en un teléfono u otro dispositivo y emula un dispositivo físico.

Las aplicaciones de autenticación virtual aplican el algoritmo de contraseña temporal de un solo uso (TOTP) y admiten varios tokens en un mismo dispositivo. El usuario debe escribir un código válido del dispositivo en otra página web durante el inicio de sesión. Cada dispositivo MFA virtual asignado a un usuario debe ser único. Un usuario no puede escribir un código desde el dispositivo MFA virtual de otro usuario para la autenticación. Dado que pueden ejecutarse en dispositivos móviles no seguros, es posible que la MFA virtual no ofrezca el mismo nivel de seguridad que las claves de seguridad FIDO.

Recomendamos que utilice un dispositivo MFA virtual mientras espera la aprobación de compra de hardware o mientras espera a que llegue su hardware. Para ver una lista de algunas aplicaciones compatibles que puede utilizar como dispositivo MFA virtual, consulte Autenticación multifactor. Para obtener instrucciones acerca de cómo configurar un dispositivo MFA virtual con AWS, consulte Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola).

Token TOTP de hardware

Un dispositivo de hardware que genera un código numérico de seis dígitos basado en el algoritmo de contraseña temporal de un solo uso (TOTP).

El usuario debe escribir un código válido del dispositivo en otra página web durante el inicio de sesión. Cada dispositivo MFA asignado a un usuario debe ser único. Un usuario no puede escribir un código desde el dispositivo de otro usuario para la autenticación. Para obtener más información sobre los dispositivos MFA físicos admitidos, consulte Autenticación multifactor. Para obtener instrucciones sobre cómo configurar un token TOTP de hardware con AWS, consulte Habilitar un token TOTP físico (consola).

Le recomendamos que utilice las llaves de seguridad FIDO como alternativa a los dispositivos TOTP de hardware. Las llaves de seguridad FIDO no necesitan baterías, son resistentes a la suplantación de identidad y son compatibles con varios usuarios raíz o de IAM en un solo dispositivo, lo que mejora la seguridad.

> **Note**
>
> MFA basada en mensaje de texto SMS: AWS finalizó la compatibilidad con la habilitación de la autenticación multifactor (MFA) por SMS. Recomendamos a los clientes que tienen usuarios de IAM que utilizan MFA basada en mensajes de texto SMS que cambien a uno de los siguientes métodos alternativos: clave de seguridad FIDO, dispositivo MFA virtual (basado en software) o dispositivo MFA de hardware. Puede identificar a los usuarios de su cuenta con un dispositivo MFA de SMS asignado. Para ello, vaya a la consola de IAM, elija Users (Usuarios) en el panel de navegación y busque los usuarios con SMS en la columna MFA de la tabla.

Temas

- Habilitación de dispositivos MFA para usuarios en AWS
- Comprobación del estado de MFA
- Resincronización de dispositivos MFA físicos y virtuales
- Desactivación de dispositivos MFA
- ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?
- Configuración del acceso a una API protegido por MFA

- Código de muestra: solicitud de credenciales con autenticación multifactor

Habilitación de dispositivos MFA para usuarios en AWS

Los pasos para configurar un dispositivo MFA dependen del tipo de dispositivo MFA que se utilice.

Temas

- Pasos generales para habilitar dispositivos MFA
- Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola)
- Habilitación de una clave de seguridad FIDO (consola)
- Habilitar un token TOTP físico (consola)
- Activación y administración de dispositivos de MFA virtuales (API de AWS CLI o de AWS)

Pasos generales para habilitar dispositivos MFA

En el siguiente procedimiento de información general se describe cómo configurar y utilizar MFA y proporciona enlaces a información relacionada.

> **Nota**
>
> Para obtener más información, también puede ver el siguiente video en inglés: Cómo configurar la autenticación multifactor (MFA) de AWS y las alertas de presupuesto de AWS.

1. Obtener un dispositivo MFA, como uno de los siguientes. Puede habilitar hasta ocho dispositivos MFA por Usuario raíz de la cuenta de AWS o usuario de IAM de cualquier combinación de los siguientes tipos.
   - Un dispositivo MFA virtual, que es una aplicación de software que cumple con RFC 6238, un algoritmo TOTP (contraseña de un solo uso basada en el tiempo) basado en estándares. Puede instalar la aplicación en un teléfono u otro dispositivo. Para ver una lista de algunas aplicaciones compatibles que puede utilizar como dispositivo MFA virtual, consulte Autenticación multifactor.
   - Una clave de seguridad FIDO con una configuración de AWS compatible. FIDO Alliance mantiene una lista de todos los productos certificados por FIDO que son compatibles con las especificaciones de FIDO.
   - Un dispositivo MFA basado en hardware de un proveedor externo, como un dispositivo simbólico. Estos tokens se utilizan exclusivamente con Cuentas de AWS. Para obtener más

información, consulte Habilitar un token TOTP físico (consola). Solo se pueden utilizar tokens que posean sus propias semillas de token que estén compartidas de manera secura con AWS. Las semillas de token son claves secretas que se generan en el momento de la producción de los tokens. Los tokens comprados en otras fuentes no funcionarán con IAM. Para garantizar la compatibilidad, debe comprar su dispositivo MFA de hardware en uno de los siguientes enlaces: token OTP o tarjeta de pantalla OTP.

2. Habilitar el dispositivo MFA.
   - Tokens TOTP virtuales o de hardware: puede utilizar comandos de la AWS CLI u operaciones de la API de AWS para habilitar un dispositivo MFA virtual para un usuario de IAM. No es posible habilitar un dispositivo MFA para el Usuario raíz de la cuenta de AWS con la AWS CLI, la API de AWS, las herramientas para Windows PowerShell ni con ninguna otra herramienta de línea de comandos. Sin embargo, puede utilizar la AWS Management Console para habilitar un dispositivo MFA para el usuario raíz.
   - Claves de seguridad FIDO: los usuarios raíz y los usuarios de IAM con claves de seguridad FIDO pueden habilitarlas únicamente desde la AWS Management Console, no desde la AWS CLI ni la API de AWS.

   Para obtener información sobre la activación de cada tipo de dispositivo MFA, consulte las siguientes páginas:
   - Dispositivo de MFA virtual: Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola)
   - Clave de seguridad FIDO: Habilitación de una clave de seguridad FIDO (consola)
   - Token TOTP de hardware: Habilitar un token TOTP físico (consola)
3. Habilitar varios dispositivos MFA (recomendado)
   - Le recomendamos que habilite varios dispositivos MFA para el Usuario raíz de la cuenta de AWS y los usuarios de IAM en las Cuentas de AWS. Esto le permite subir el nivel de seguridad de sus Cuentas de AWS y simplificar la administración de acceso a usuarios con privilegios elevados, como el Usuario raíz de la cuenta de AWS.
   - Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Con varios dispositivos MFA, solo necesita un dispositivo MFA para iniciar sesión en la AWS Management Console o crear una sesión a través de AWS CLI como ese usuario. Un usuario de IAM debe autenticarse con un dispositivo de MFA existente para habilitar o deshabilitar un dispositivo de MFA adicional.

- En caso de pérdida, robo o inaccesibilidad de un dispositivo MFA, puede utilizar uno de los dispositivos MFA restantes para acceder a la Cuenta de AWS sin seguir el procedimiento de recuperación de Cuenta de AWS. En caso de pérdida o robo de un dispositivo MFA, este debe disociarse del principal de IAM al que está asociado.
- El uso de varios MFA permite que sus empleados que se encuentren en ubicaciones geográficamente dispersas o que trabajen de forma remota utilicen MFA basado en hardware para acceder a AWS sin necesidad de coordinar un intercambio físico de un único dispositivo de hardware entre empleados.
- El uso de dispositivos MFA adicionales para los directores de IAM permite utilizar uno o más MFA para el uso diario y, al mismo tiempo, mantener los dispositivos MFA físicos en una ubicación física segura, como una bóveda o una caja fuerte para realizar copias de seguridad y redundancia.

4. Utilizar el dispositivo MFA al iniciar sesión para obtener acceso a los recursos de AWS. Tenga en cuenta lo siguiente:
   - Claves de seguridad FIDO: para acceder a un sitio web de AWS, ingrese las credenciales y, a continuación, toque la clave de seguridad FIDO cuando se le solicite.
   - Dispositivos MFA virtuales y tokens TOTP de hardware: para acceder a un sitio web de AWS, necesita un código MFA del dispositivo además de su nombre de usuario y contraseña.

     Para obtener acceso a operaciones de la API protegidas por MFA, necesita lo siguiente:
     - Un código MFA
     - El identificador del dispositivo MFA (el número de serie de un dispositivo físico o el ARN de un dispositivo virtual definido en AWS)
     - El ID de clave de acceso y la clave de acceso secreta normales

> **Notas**
>
> - No se puede pasar la información MFA de una clave de seguridad FIDO a las operaciones de la API de AWS STS para solicitar credenciales temporales.
> - No puede utilizar los comandos de la AWS CLI ni operaciones de la API de AWS para activar FIDO security keys (Claves de seguridad FIDO).
> - No puede utilizar el mismo nombre para más de un dispositivo MFA de raíz o de IAM.

Para obtener más información, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola)

Puede utilizar un teléfono u otro dispositivo como dispositivo de autenticación multifactor (MFA) virtual. Para ello, instale una aplicación móvil que cumpla con RFC 6238, un algoritmo TOTP (contraseña de un solo uso basada en el tiempo) basado en estándares. Estas aplicaciones generan un código de autenticación de seis dígitos. Dado que pueden ejecutarse en dispositivos móviles no seguros, es posible que la MFA virtual no ofrezca el mismo nivel de seguridad que las claves de seguridad FIDO. Recomendamos que utilice un dispositivo MFA virtual mientras espera la aprobación de compra de hardware o mientras espera a que llegue su hardware.

La mayoría de aplicaciones de MFA virtual admiten la creación de varios dispositivos virtuales, lo que le permite utilizar la misma aplicación para varias Cuentas de AWS o usuarios. Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Con varios dispositivos MFA, solo necesita un dispositivo MFA para iniciar sesión en la AWS Management Console o crear una sesión a través de AWS CLI como ese usuario. Le recomendamos que registre varios dispositivos MFA. Para las aplicaciones de autenticación, también recomendamos activar la copia de seguridad en la nube o la característica de sincronización en esas aplicaciones para evitar perder el acceso a la cuenta si pierde o se rompe el dispositivo que contiene las aplicaciones de autenticación.

Para obtener una lista de las aplicaciones MFA virtuales que puede utilizar, consulte Multi-Factor Authentication. AWS requiere una aplicación MFA virtual que genere una OTP de seis dígitos.

Temas

- Permisos necesarios
- Habilite un dispositivo MFA virtual para un usuario de IAM (Consola)
- Reemplazar un dispositivo MFA virtual

Permisos necesarios

Para administrar dispositivos MFA virtuales para su usuario de IAM, debe contar con los permisos de la siguiente política: AWS: permite a los usuarios de IAM autenticados por MFA administrar su propio dispositivo MFA en la página Credenciales de seguridad.

Habilite un dispositivo MFA virtual para un usuario de IAM (Consola)

Puede utilizar IAM en la AWS Management Console para habilitar y administrar un dispositivo MFA virtual para un usuario de IAM en su cuenta. Puede asociar etiquetas a los recursos de IAM, incluidos los dispositivos MFA virtuales, a fin de identificar, organizar y controlar el acceso a ellos. Puede etiquetar dispositivos MFA virtuales solo cuando utiliza la AWS CLI o la API de AWS. Para habilitar y administrar un dispositivo MFA utilizando la AWS CLI o la API de AWS, consulte Activación y administración de dispositivos de MFA virtuales (API de AWS CLI o de AWS). Para más información acerca del etiquetado de recursos de IAM, consulte Etiquetado de recursos de IAM.

> **Note**
>
> Debe tener acceso físico al hardware que alojará el dispositivo MFA virtual del usuario para poder configurar la MFA. Por ejemplo, puede configurar MFA para un usuario que utilice un dispositivo MFA virtual que se ejecute en un smartphone. En ese caso, debe tener el smartphone disponible para completar el asistente. Por este motivo, puede interesarle que los usuarios puedan configurar y administrar sus propios dispositivos MFA virtuales. En ese caso, debe conceder a los usuarios los permisos necesarios para realizar las acciones de IAM necesarias. Para obtener más información y consultar un ejemplo de una política de IAM que concede dichos permisos, consulte el Tutorial de IAM: permitir a los usuarios administrar sus credenciales y configuración de MFA y AWS: permite a los usuarios de IAM autenticados por MFA administrar su propio dispositivo MFA en la página Credenciales de seguridad en la política de ejemplo.

Para habilitar un dispositivo MFA virtual para un usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. En la lista Users (Usuarios), elija el nombre de usuario de IAM.
4. Elija la pestaña Security credentials (Credenciales de seguridad). En Multi-factor authentication (MFA) (Autenticación multifactor [MFA]), seleccione Assign MFA device (Asignar dispositivo MFA).
5. En el asistente, escriba un Nombre de dispositivo, elija Aplicación del autenticador y luego, Siguiente.

IAM generará y mostrará la información de configuración del dispositivo MFA virtual, incluido un gráfico de código QR. El gráfico es una representación de la "clave de configuración secreta" que se puede introducir manualmente en dispositivos que no admiten códigos QR.

6. Abra su aplicación de MFA virtual. Para ver una lista de las aplicaciones que puede utilizar para alojar dispositivos MFA virtuales, consulte Multi-Factor Authentication.

   Si la aplicación de MFA virtual admite varios dispositivos o cuentas de MFA, elija la opción para crear un nuevo dispositivo o cuenta de MFA virtual.

7. Determine si la aplicación MFA admite códigos QR y, a continuación, lleve a cabo alguna de las siguientes operaciones:

   - Desde el asistente, elija Show QR code (Mostrar código QR) y, a continuación, utilice la aplicación para escanear el código QR. Por ejemplo, puede elegir el icono de la cámara o una opción similar a Scan code (Escanear código) y, a continuación, utilizar la cámara del dispositivo para escanear el código.
   - En el asistente, elija Show secret key (Mostrar clave secreta) y, a continuación, escriba la clave secreta en su aplicación MFA.

   Cuando haya terminado, el dispositivo MFA virtual comenzará a generar contraseñas de uso único.

8. En la página Configurar el dispositivo, en el cuadro Código MFA 1, escriba la contraseña de uso único que aparece actualmente en el dispositivo MFA virtual. Espere hasta 30 segundos a que el dispositivo genere una nueva contraseña de uso único. A continuación, escriba la otra contraseña de uso único en el cuadro MFA code 2 (Código MFA 2). Elija Add MFA (Agregar MFA).

   > **Important**
   >
   > Envíe su solicitud inmediatamente después de generar los códigos. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas de un solo uso basadas en el tiempo (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

Ahora el dispositivo MFA virtual ya está listo para utilizarlo con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Reemplazar un dispositivo MFA virtual

Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Si el usuario pierde un dispositivo o debe reemplazarlo por cualquier motivo, primero debe desactivar el antiguo dispositivo. Después, puede añadir el nuevo dispositivo para el usuario.

- Para desactivar el dispositivo que tenga asociado actualmente a otro usuario de IAM, consulte Desactivación de dispositivos MFA.
- Para agregar un dispositivo MFA virtual de sustitución para otro usuario de IAM, siga los pasos que se indican en el procedimiento Habilite un dispositivo MFA virtual para un usuario de IAM (Consola) anterior.
- Para agregar un dispositivo MFA virtual de reemplazo para Usuario raíz de la cuenta de AWS, siga los pasos que se indican en el procedimiento Habilitación de un dispositivo MFA virtual para su Usuario raíz de la cuenta de AWS (consola).

Habilitación de una clave de seguridad FIDO (consola)

Las claves de seguridad FIDO son un tipo de dispositivo de autenticación multifactor (MFA) que puede utilizar para proteger sus recursos de AWS. Conecte su clave de seguridad FIDO a un puerto USB de su ordenador y actívela según las instrucciones siguientes. Después de habilitarla, pulse en ella cuando se le solicite para completar con seguridad el proceso de inicio de sesión. Si ya utiliza una clave de seguridad FIDO con otros servicios y tiene una configuración compatible con AWS (por ejemplo, YubiKey 5 Series de Yubico), también puede utilizarla con AWS. De lo contrario, deberá adquirir una clave de seguridad FIDO si desea utilizar WebAuthn para MFA en AWS. Además, las llaves de seguridad FIDO con compatibles con varios usuarios raíz o de IAM en el mismo dispositivo, lo que mejora su utilidad para la seguridad de las cuentas. Para obtener información sobre las especificaciones y opciones de compra de ambos tipos de dispositivo, consulte Autenticación multifactor. Para obtener información sobre las especificaciones y opciones de compra, consulte Autenticación multifactor.

FIDO2 es un estándar de autenticación abierto y una extensión de FIDO U2F, que ofrece el mismo alto nivel de seguridad basado en la criptografía de clave pública. FIDO2 se compone de la especificación de autenticación web del W3C (WebAuthn API) y del Protocolo de cliente a

autenticador (CTAP) de FIDO Alliance, un protocolo de capa de aplicación. El CTAP permite la comunicación entre el cliente o la plataforma, como un navegador o un sistema operativo, con un autenticador externo. Cuando se habilita un autenticador certificado FIDO en AWS, la clave de seguridad FIDO crea un nuevo par de claves para utilizarlo solo con AWS. En primer lugar, introduzca sus credenciales. Cuando se le solicite, toque la clave de seguridad FIDO, que responde al reto de autenticación emitido por AWS. Para obtener más información sobre el estándar FIDO2, consulte FIDO2 Project (Proyecto FIDO2).

Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el usuario raíz de la Cuenta de AWS y los usuarios de IAM. Con varios dispositivos MFA, solo necesita un dispositivo MFA para iniciar sesión en la AWS Management Console o crear una sesión a través de AWS CLI como ese usuario. Le recomendamos que registre varios dispositivos MFA. Por ejemplo, puede registrar un autenticador integrado y también una clave de seguridad que guarde en un lugar físico seguro. Si no puede utilizar el autenticador integrado, puede utilizar su clave de seguridad registrada. Para las aplicaciones de autenticación, también recomendamos activar la copia de seguridad en la nube o la característica de sincronización en esas aplicaciones para evitar perder el acceso a la cuenta si pierde o se rompe el dispositivo que contiene las aplicaciones de autenticación.

> **Note**
>
> Se recomienda exigir a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. Sus usuarios pueden federarse en AWS con un proveedor de identidades donde se autentican con sus credenciales corporativas y configuraciones MFA. Para administrar el acceso a AWS y a las aplicaciones empresariales, le recomendamos que utilice IAM Identity Center. Para más información, consulte la Guía del usuario de IAM Identity Center.

Temas

Permisos necesarios

Para administrar una clave de seguridad FIDO para su propio usuario de IAM mientras protege las acciones confidenciales relacionadas con MFA, debe tener los permisos de la siguiente política:

> **Note**
>
> Los valores de ARN son valores estáticos y no son un indicador de qué protocolo se utilizó para registrar el autenticador. El protocolo U2F está obsoleto, por lo que todas las nuevas implementaciones utilizan WebAuthn.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
```

```
        }
    ]
}
```

Habilitación de una clave de seguridad FIDO para su propio usuario de IAM (consola)

Solo puede habilitar una clave de seguridad FIDO para su propio usuario de IAM desde la AWS Management Console, no desde la AWS CLI ni la API de AWS.

> **Note**
>
> Antes de habilitar una clave de seguridad FIDO, debe tener acceso físico al dispositivo.

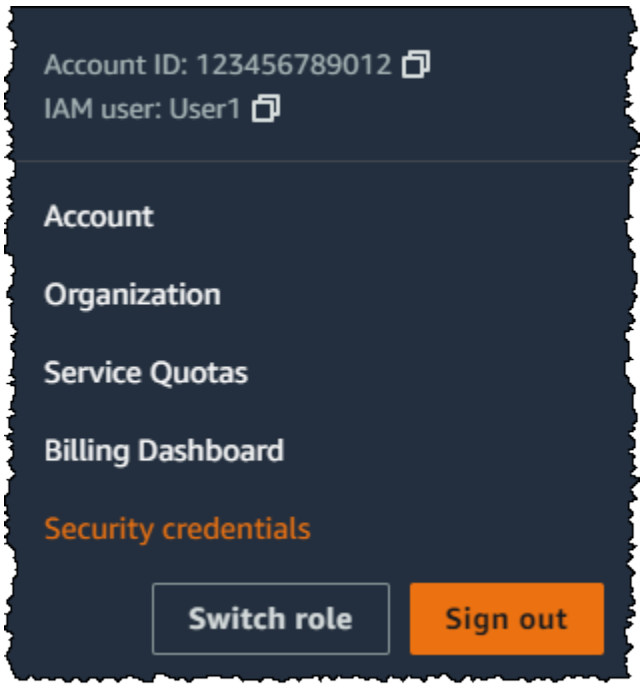
> **Note**
>
> No debe elegir ninguna de las opciones disponibles en la ventana emergente de Google Chrome que le pide verificar su identidad con amazon.com. Solo tiene que tocar la clave de seguridad.

Para habilitar una clave de seguridad FIDO para su propio usuario de IAM (consola)

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

   > **Note**
   >
   > Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

   Para obtener el ID de la Cuenta de AWS, contacte con su administrador.
2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).



3. En la pestaña Credenciales de AWS IAM, en la sección Autenticación multifactor (MFA), elija Asignar dispositivo MFA.
4. En el asistente, escriba un Device name (Nombre del dispositivo), seleccione Security Key (Clave de seguridad) y, a continuación, Next (Siguiente).
5. Inserte la clave de seguridad FIDO en el puerto USB de su ordenador.

6. Pulse la clave de seguridad FIDO.

La clave de seguridad FIDO está lista para utilizarse con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Habilitación de una clave de seguridad FIDO para otro usuario de IAM (consola)

Solo puede habilitar una clave de seguridad FIDO para otro usuario de IAM desde la AWS Management Console, no desde la AWS CLI ni la API de AWS.

Para habilitar una clave de seguridad FIDO para otro usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Elija el nombre del usuario para el que desea activar la MFA.
4. Elija la pestaña Credenciales de seguridad. En Multi-factor authentication (MFA) (Autenticación multifactor [MFA]), seleccione Assign MFA device (Asignar dispositivo MFA).
5. En el asistente, escriba un Device name (Nombre del dispositivo), seleccione Security Key (Clave de seguridad) y, a continuación, Next (Siguiente).
6. Inserte la clave de seguridad FIDO en el puerto USB de su ordenador.

7. Pulse la clave de seguridad FIDO.

La clave de seguridad FIDO está lista para utilizarse con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Reemplazar una clave de seguridad FIDO

Puede tener hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA actualmente compatibles asignados a un usuario a la vez con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Si el usuario pierde un autenticador FIDO o necesita sustituirlo por cualquier motivo, antes debe desactivar el autenticador FIDO antiguo. Después, puede añadir un nuevo dispositivo MFA para el usuario.

- Para desactivar el dispositivo que tenga asociado actualmente a otro usuario de IAM, consulte Desactivación de dispositivos MFA.
- Para agregar una nueva clave de seguridad FIDO para un usuario de IAM, consulte Habilitación de una clave de seguridad FIDO para su propio usuario de IAM (consola).

Si no tiene acceso a una nueva clave de seguridad FIDO, puede habilitar un nuevo dispositivo MFA virtual o token TOTP de hardware. Consulte una de las siguientes opciones para obtener instrucciones:

- Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola)
- Habilitar un token TOTP físico (consola)

Configuraciones admitidas para usar las claves de seguridad FIDO

Puede utilizar las claves de seguridad de FIDO2 como un método de autenticación multifactor (MFA) con IAM mediante el uso de las configuraciones admitidas actualmente. Entre ellos, se encuentran los dispositivos FIDO2 admitidos por IAM y los navegadores que con compatibles con FIDO2. Antes de registrar su dispositivo FIDO2, compruebe que esté utilizando la última versión del navegador y del sistema operativo (SO). Las funciones pueden comportarse de forma diferente en los distintos navegadores, autenticadores y clientes del sistema operativo. Si el registro del dispositivo falla en un navegador, puede intentar registrarse en otro navegador.

Dispositivos FIDO2 compatibles con AWS

IAM es compatible con dispositivos de seguridad FIDO2 que se conectan a sus dispositivos a través de USB, Bluetooth o NFC. No admitimos autenticadores de plataforma como TouchID, FaceID o Windows Hello.

> **Note**
>
> AWS requiere acceso al puerto USB físico en el equipo para verificar su dispositivo FIDO2. Las claves de seguridad de FIDO2 no funcionan con máquinas virtuales, conexiones remotas o en el modo incógnito de un navegador.

FIDO Alliance mantiene una lista de todos los productos FIDO2 que son compatibles con las especificaciones de FIDO.

Navegadores compatibles con FIDO2

La disponibilidad de los dispositivos de seguridad FIDO2 que se ejecutan en un navegador web depende de la combinación del navegador y el sistema operativo. Los siguientes navegadores admiten el uso de llaves de seguridad FIDO2:

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Sí	Sí	Sí	Sí	No
Safari	Sí	No	No	Sí	No
Periferia	Sí	Sí	No	Sí	No
Firefox	Sí	Sí	No	Sí	No

> **Note**
>
> La mayoría de las versiones de Firefox que actualmente admiten FIDO2 no lo hacen de forma predeterminada. Para obtener instrucciones sobre la habilitación de FIDO2 en Firefox, consulte Solución de problemas con claves de seguridad FIDO.

Para más información sobre la compatibilidad del navegador con un dispositivo certificado para FIDO2, como YubiKey, consulte Compatibilidad de los sistemas operativos y los navegadores web con FIDO2 y U2F.

Complementos de navegador

AWS es compatible solo con los navegadores que admiten FIDO2 de forma nativa. AWS no es compatible con el uso de complementos para agregar compatibilidad con los navegadores de FIDO2. Algunos complementos de navegador son incompatibles con el estándar FIDO2 y pueden causar resultados inesperados con las claves de seguridad FIDO2.

Para obtener información sobre la desactivación de complementos del navegador y otras sugerencias para la solución de problemas, consulte No puedo habilitar mi clave de seguridad FIDO.

Certificaciones de dispositivos

Recopilamos y asignamos certificaciones relacionadas con el dispositivo, como la validación FIPS y el nivel de certificación FIDO, solo durante el registro de una clave de seguridad de FIDO. La certificación de su dispositivo se obtiene del Servicio de metadatos (MDS) de FIDO Alliance. Si el estado de certificación o el nivel de su clave de seguridad FIDO cambian, eso no se reflejará automáticamente en las etiquetas del dispositivo. Para actualizar la información de certificación de un dispositivo, vuelva a registrar el dispositivo para obtener la información de certificación actualizada.

AWS proporciona los siguientes tipos de certificación como claves de condición durante el registro del dispositivo, obtenidos del FIDO MDS: niveles de certificación FIPS-140-2, FIPS-140-3 y FIDO. Puede especificar el registro de autenticadores específicos en sus políticas de IAM, según el tipo y el nivel de certificación que prefiera. Para obtener más información, consulte las políticas a continuación.

Ejemplos de políticas para certificaciones de dispositivos

Los siguientes casos de uso muestran ejemplos de políticas que le permiten registrar dispositivos MFA con certificaciones FIPS.

Temas

- Caso de uso 1: permitir el registro únicamente de dispositivos que tengan certificaciones FIPS-140-2 L2
- Caso de uso 2: permitir el registro de dispositivos que tengan las certificaciones FIPS-140-2 L2 y FIDO L1
- Caso de uso 3: permitir el registro de dispositivos que tengan las certificaciones FIPS-140-2 L2 o FIPS-140-3 L2
- Caso de uso 4: permitir el registro de dispositivos que cuentan con la certificación FIPS-140-2 L2 y que admiten otros tipos de MFA, como los autenticadores virtuales y el TOTP físico

Caso de uso 1: permitir el registro únicamente de dispositivos que tengan certificaciones FIPS-140-2 L2

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
}
```

Caso de uso 2: permitir el registro de dispositivos que tengan las certificaciones FIPS-140-2 L2 y FIDO L1

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
```

```
                    "iam:FIDO-FIPS-140-2-certification": "L2",
                    "iam:FIDO-certification": "L1"
                }
            }
        }
    ]
}
```

Caso de uso 3: permitir el registro de dispositivos que tengan las certificaciones FIPS-140-2 L2 o FIPS-140-3 L2

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L2"
                }
```

```
            }
        }
    ]
}
```

Caso de uso 4: permitir el registro de dispositivos que cuentan con la certificación FIPS-140-2 L2 y que admiten otros tipos de MFA, como los autenticadores virtuales y el TOTP físico

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey": "Activate",
                    "iam:FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "iam:RegisterSecurityKey": "true"
                }
            }
        }
    ]
```

```
}
```

AWS CLI y API de AWS

AWS admite el uso de claves de seguridad FIDO2 solo en la AWS Management Console. El uso de claves de seguridad FIDO2 para la MFA no es compatible con la AWS CLI ni la API de AWS, y tampoco se pueden utilizar para acceder a las operaciones de la API protegidas por MFA.

Recursos adicionales de

- Para obtener más información sobre el uso de llaves de seguridad FIDO2 en AWS, consulte Habilitación de una clave de seguridad FIDO (consola).
- Para obtener ayuda con la resolución de problemas de claves de seguridad de FIDO2 en AWS, consulte Solución de problemas con claves de seguridad FIDO.
- Para obtener información general del sector sobre la compatibilidad con FIDO2, consulte Proyecto FIDO2.

Habilitar un token TOTP físico (consola)

Un token TOTP de hardware genera un código numérico de seis dígitos basado en un algoritmo de contraseña temporal de un solo uso (TOTP). El usuario debe escribir un código válido del dispositivo cuando se le solicite durante el proceso de inicio de sesión. Todos los dispositivos MFA asignados a un usuario deben ser únicos; un usuario no puede escribir el código del dispositivo de otro usuario para autenticarlo. Los dispositivos MFA no se pueden compartir entre cuentas o usuarios.

Los tokens TOTP de hardware y las claves de seguridad FIDO son dispositivos físicos que se compran. Los dispositivos MFA de hardware generan códigos TOTP para la autenticación al iniciar sesión en AWS. Se basan en baterías, las cuales pueden necesitar ser reemplazadas y resincronizadas con AWS en el tiempo. Las claves de seguridad FIDO, que utilizan criptografía de clave pública, no requieren baterías y ofrecen un proceso de autenticación perfecto. Recomendamos utilizar las claves de seguridad FIDO para evitar la suplantación de identidad, ya que ofrecen una alternativa más segura a los dispositivos TOTP. Además, las llaves de seguridad FIDO con compatibles con varios usuarios raíz o de IAM en el mismo dispositivo, lo que mejora su utilidad para la seguridad de las cuentas. Para obtener información sobre las especificaciones y opciones de compra de ambos tipos de dispositivo, consulte Autenticación multifactor.

Se puede habilitar un token TOTP de hardware para un usuario de IAM desde la AWS Management Console, la línea de comandos o la API de IAM. Para habilitar un dispositivo MFA para su Usuario

raíz de la cuenta de AWS, consulte Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola).

Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Con varios dispositivos MFA, solo necesita un dispositivo MFA para iniciar sesión en la AWS Management Console o crear una sesión a través de AWS CLI como ese usuario.

> **Important**
>
> Se recomienda habilitar varios dispositivos MFA para que los usuarios puedan seguir accediendo a la cuenta en caso de pérdida o inaccesibilidad del dispositivo MFA.

> **Note**
>
> Si desea habilitar el dispositivo MFA desde la línea de comandos, utilice `aws iam enable-mfa-device`. Para habilitar el dispositivo MFA con la API de IAM, utilice la operación `EnableMFADevice`.

Temas

- Permisos necesarios
- Habilitar un token TOTP de hardware para su propio usuario de IAM (consola)
- Habilitar un token TOTP de hardware para otro usuario de IAM (consola)
- Reemplazar un dispositivo MFA físico

Permisos necesarios

Para administrar un token TOTP de hardware para su propio usuario de IAM mientras protege acciones sensibles relacionadas con MFA, debe tener los permisos de la siguiente política:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowManageOwnUserMFA",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Habilitar un token TOTP de hardware para su propio usuario de IAM (consola)

Puede habilitar su propio token TOTP de hardware desde la AWS Management Console.

> **Note**
>
> Para poder habilitar un dispositivo MFA físico, debe tener acceso físico al dispositivo.

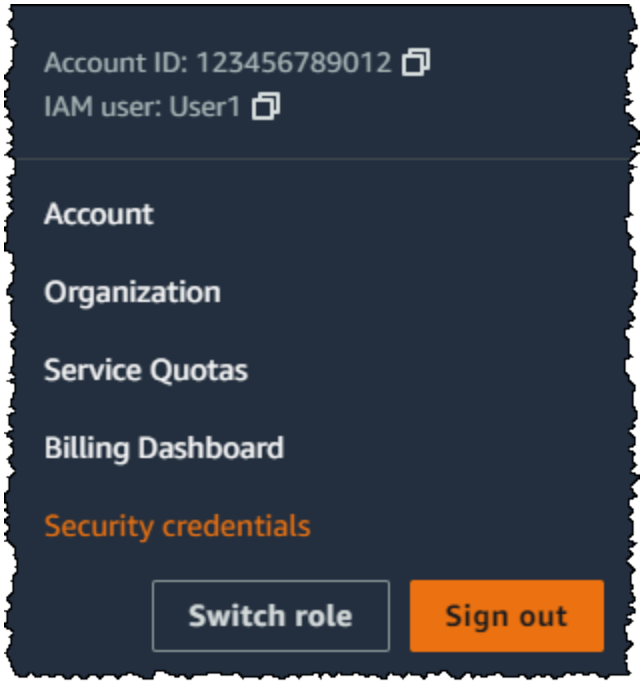
Para habilitar un token TOTP de hardware para su propio usuario de IAM (consola)

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

> **Note**
>
> Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

Para obtener el ID de la Cuenta de AWS, contacte con su administrador.

2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).



3. En la pestaña Credenciales de AWS IAM, en la sección Autenticación multifactor (MFA), elija Asignar dispositivo MFA.
4. En el asistente, escriba un Device name (Nombre del dispositivo), elija Hardware TOTP token (Token TOTP de hardware) y, a continuación, elija Next (Siguiente).
5. Escriba el número de serie del dispositivo. Por lo general, se encuentra en la parte posterior del dispositivo.

6. En el cuadro Código MFA 1, escriba el número de seis dígitos que se encuentra en el dispositivo MFA. Es posible que tenga que pulsar el botón de la parte anterior del dispositivo para mostrar el número.

7. Espere 30 segundos a que el dispositivo actualice el código y, a continuación, escriba el número de seis dígitos siguiente en el cuadro Código MFA 2. Es posible que tenga que volver a pulsar el botón de la parte anterior del dispositivo para mostrar el otro número.
8. Elija Agregar MFA.

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos de autenticación. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

El dispositivo ya está listo para utilizarlo con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Habilitar un token TOTP de hardware para otro usuario de IAM (consola)

Puede habilitar un token TOTP de hardware para otro usuario de IAM desde la AWS Management Console.

Para habilitar un token TOTP de hardware para otro usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Elija el nombre del usuario para el que desea activar la MFA.
4. Elija la pestaña Credenciales de seguridad. En autenticación multifactor (MFA), seleccione Asignar dispositivo MFA.

5. En el asistente, escriba un Nombre del dispositivo, elija Token TOTP de hardware y, a continuación, elija Siguiente.
6. Escriba el número de serie del dispositivo. Por lo general, se encuentra en la parte posterior del dispositivo.
7. En el cuadro Código MFA 1, escriba el número de seis dígitos que se encuentra en el dispositivo MFA. Es posible que tenga que pulsar el botón de la parte anterior del dispositivo para mostrar el número.

8. Espere 30 segundos a que el dispositivo actualice el código y, a continuación, escriba el número de seis dígitos siguiente en el cuadro Código MFA 2. Es posible que tenga que volver a pulsar el botón de la parte anterior del dispositivo para mostrar el otro número.
9. Elija Agregar MFA.

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos de autenticación. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede volver a sincronizar el dispositivo.

El dispositivo ya está listo para utilizarlo con AWS. Para obtener más información sobre el uso de MFA con la AWS Management Console, consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Reemplazar un dispositivo MFA físico

Puede tener hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente asignados a un usuario a la vez con el Usuario raíz de la cuenta de AWS y los usuarios de IAM. Si el usuario pierde un dispositivo o debe reemplazarlo por cualquier motivo, primero debe desactivar el antiguo dispositivo. Después, puede añadir el nuevo dispositivo para el usuario.

- Para desactivar el dispositivo que tenga asociado actualmente a un usuario, consulte Desactivación de dispositivos MFA.

- Para agregar un token TOTP de hardware de reemplazo para un usuario de IAM, siga los pasos del procedimiento Habilitar un token TOTP de hardware para otro usuario de IAM (consola) anterior en este tema.
- Para agregar un token TOTP de hardware de reemplazo para el Usuario raíz de la cuenta de AWS, siga los pasos del procedimiento Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola) anterior de este tema.

Activación y administración de dispositivos de MFA virtuales (API de AWS CLI o de AWS)

Puede utilizar comandos de la AWS CLI u operaciones de la API de AWS para habilitar un dispositivo MFA virtual para un usuario de IAM. No es posible habilitar un dispositivo MFA para el Usuario raíz de la cuenta de AWS con la AWS CLI, la API de AWS, las herramientas para Windows PowerShell ni con ninguna otra herramienta de línea de comandos. Sin embargo, puede utilizar la AWS Management Console para habilitar un dispositivo MFA para el usuario raíz.

Al habilitar un dispositivo MFA desde la AWS Management Console, esta ejecuta varias operaciones automáticamente. En cambio, si crea un dispositivo virtual utilizando la AWS CLI, las Tools for Windows PowerShell o la API de AWS, entonces deberá realizar los pasos manualmente y en el orden correcto. Por ejemplo, para crear un dispositivo MFA virtual, debe crear el objeto de IAM y extraer el código como una cadena o un código QR gráfico. A continuación, debe sincronizar el dispositivo y asociarlo con un usuario de IAM. Consulte la sección Examples de New-IAMVirtualMFADevice para obtener más detalles. Para un dispositivo físico, puede omitir el paso de creación e ir directamente a sincronizar el dispositivo y asociarlo con el usuario.

Puede asociar etiquetas a los recursos de IAM, incluidos los dispositivos MFA virtuales, a fin de identificar, organizar y controlar el acceso a ellos. Puede etiquetar dispositivos MFA virtuales solo cuando utiliza la AWS CLI o la API de AWS.

Un usuario de IAM que utilice el SDK o la CLI puede habilitar un dispositivo MFA adicional llamando a `EnableMFADevice` o desactivar un dispositivo MFA existente mediante una llamada a `DeactivateMFADevice`. Para hacerlo correctamente, primero deben llamar a `GetSessionToken` y enviar los códigos MFA con un dispositivo MFA existente. Esta llamada devuelve credenciales de seguridad temporales que luego se pueden usar para firmar las operaciones de API que requieren la autenticación MFA. Para ver un ejemplo de solicitud y respuesta, consulte `GetSessionToken`: credenciales temporales para usuarios de entornos que no son de confianza.

Para crear la entidad del dispositivo virtual en IAM para representar un dispositivo de MFA virtual

Estos comandos proporcionan un ARN para el dispositivo que se utiliza en lugar de un número de serie en muchos de los siguientes comandos.

- AWS CLI: `aws iam create-virtual-mfa-device`
- API de AWS: `CreateVirtualMFADevice`

Para habilitar un dispositivo MFA para su uso con AWS

Estos comandos sincronizan el dispositivo con AWS y lo asocian con un usuario. Si el dispositivo es virtual, utilice el ARN del dispositivo virtual como número de serie.

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos de autenticación. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas de un solo uso basadas en el tiempo (TOTP) caducan tras un corto periodo de tiempo. Si esto ocurre, puede resincronizar el dispositivo utilizando los comandos que se describen a continuación.

- AWS CLI: `aws iam enable-mfa-device`
- API de AWS: `EnableMFADevice`

Para desactivar un dispositivo

Utilice estos comandos para desvincular el dispositivo del usuario y desactivarlo. Si el dispositivo es virtual, utilice el ARN del dispositivo virtual como número de serie. Asimismo, debe eliminar la entidad de dispositivo virtual por separado.

- AWS CLI: `aws iam deactivate-mfa-device`
- API de AWS: `DeactivateMFADevice`

Para crear una lista de entidades de dispositivo de MFA virtual

Utilice estos comandos para una lista de entidades de dispositivo de MFA virtual.

- AWS CLI: `aws iam list-virtual-mfa-devices`

- API de AWS: `ListVirtualMFADevices`

Cómo etiquetar un dispositivo MFA virtual

Utilice estos comandos para etiquetar un dispositivo MFA virtual.

- AWS CLI: `aws iam tag-mfa-device`
- API de AWS: `TagMFADevice`

Enumerar etiquetas para un dispositivo MFA virtual

Utilice estos comandos para enumerar las etiquetas asociadas a un dispositivo MFA virtual.

- AWS CLI: `aws iam list-mfa-device-tags`
- API de AWS: `ListMFADeviceTags`

Quitar etiquetas de un dispositivo MFA virtual

Utilice estos comandos para quitar las etiquetas asociadas a un dispositivo MFA virtual.

- AWS CLI: `aws iam untag-mfa-device`
- API de AWS: `UntagMFADevice`

Para resincronizar un dispositivo de MFA

Utilice estos comandos si el dispositivo genera códigos que AWS no acepta. Si el dispositivo es virtual, utilice el ARN del dispositivo virtual como número de serie.

- AWS CLI: `aws iam resync-mfa-device`
- API de AWS: `ResyncMFADevice`

Para eliminar una entidad de dispositivo MFA virtual en IAM

En cuanto el dispositivo se desvincule del usuario, puede eliminar la entidad de dispositivo.

- AWS CLI: `aws iam delete-virtual-mfa-device`
- API de AWS: `DeleteVirtualMFADevice`

Para recuperar un dispositivo MFA virtual perdido o que no funciona

A veces, el dispositivo de un usuario que aloja la aplicación de MFA virtual se pierde, se reemplaza o no funciona. Cuando esto sucede, el usuario no puede recuperarlo por sí mismo. Los usuarios deben ponerse en contacto con un administrador para desactivar el dispositivo. Para obtener más información, consulte ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?.

Comprobación del estado de MFA

Utilice la consola de IAM para comprobar si un Usuario raíz de la cuenta de AWS o un usuario de IAM tiene un dispositivo MFA válido habilitado.

Para comprobar el estado de un usuario raíz MFA

1. Inicie sesión en la AWS Management Console con sus credenciales de usuario raíz y, a continuación, abrir la consola IAM en https://console.aws.amazon.com/iam/.
2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).
3. Compruebe en Multi-factor Authentication (MFA) (Autenticación multifactor [MFA]) para ver si la MFA está habilitada o desactivada. Si la MFA no se ha activado, aparecerá un símbolo de alerta ( ).

Si desea activar MFA para la cuenta, consulte una de las siguientes opciones:

- Habilitación de un dispositivo MFA virtual para su Usuario raíz de la cuenta de AWS (consola)
- Habilitación de una clave de seguridad FIDO para el usuario raíz de la Cuenta de AWS (consola)
- Habilitación de un token TOTP de hardware para el usuario raíz de la Cuenta de AWS (consola)

Para verificar el estado de MFA de los usuarios de IAM

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Si es necesario, agregue la columna MFA a la tabla de usuarios realizando los siguientes pasos:
   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ( ).
   b. En Manage Columns (Administrar columnas), seleccione MFA.

c. (Opcional) Desactive la casilla de verificación de los encabezados de columna que no quiera que aparezcan en la tabla de los usuarios.

d. Seleccione Close (Cerrar) para volver a la lista de usuarios.

4. La columna MFA le informa sobre el dispositivo MFA que está activado. Si ningún dispositivo MFA está activado para el usuario, la consola muestra None (Ninguno). Si el usuario tiene un dispositivo MFA activado, la columna MFA muestra el tipo de dispositivo que está activado con un valor Virtual, FIDO Security Key (Clave de seguridad FIDO), Hardware o SMS.

> **Note**
>
> AWS finalizó el soporte para habilitar la autenticación multifactor (MFA) por SMS. Recomendamos a los clientes que tienen usuarios de IAM que utilizan MFA basado en mensajes de texto SMS que cambien a uno de los siguientes métodos alternativos: dispositivo MFA virtual (basado en software), clave de seguridad FIDO o dispositivo MFA de hardware. Puede identificar a los usuarios de su cuenta con un dispositivo MFA de SMS asignado. Para ello, vaya a la consola de IAM, elija Users (Usuarios) en el panel de navegación y busque los usuarios con SMS en la columna MFA de la tabla.

5. Para ver información adicional sobre el dispositivo MFA de un usuario, seleccione el nombre del usuario cuyo estado de MFA quiere comprobar. A continuación, elija la pestaña Security credentials (Credenciales de seguridad).

6. Si no hay ningún dispositivo MFA activo para el usuario, la consola muestra No hay dispositivos MFA. Asigne un dispositivo MFA para mejorar la seguridad de su entorno de AWS en la sección Autenticación multifactor (MFA). Si el usuario tiene dispositivos MFA activados, la sección Multi-factor authentication (MFA) (Autenticación multifactor [MFA]) muestra detalles sobre los dispositivos:

   - El nombre del dispositivo
   - El tipo de dispositivo
   - El identificador del dispositivo, como el número de serie de un dispositivo físico o el ARN en AWS de un dispositivo virtual
   - Cuándo se creó el dispositivo

Para eliminar o volver a sincronizar un dispositivo, seleccione el botón de radio situado junto al dispositivo y, a continuación, Remove (Eliminar) o Resync (Volver a sincronizar).

Para obtener más información sobre la habilitación de MFA, consulte los siguientes temas:

- Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola)
- Habilitación de una clave de seguridad FIDO (consola)
- Habilitar un token TOTP físico (consola)

Resincronización de dispositivos MFA físicos y virtuales

Puede utilizar AWS para volver a sincronizar sus dispositivos MFA (autenticación multifactor [MFA]) físicos y virtuales. Si el dispositivo no se sincroniza al intentar utilizarlo, el intento de inicio de sesión del usuario dará un error e IAM le pedirá que vuelva a sincronizar el dispositivo.

> **Note**
>
> Las claves de seguridad FIDO no pierden la sincronización. Si una llave de seguridad FIDO se pierde o avería, puede desactivarla. Para obtener instrucciones sobre cómo desactivar cualquier tipo de dispositivo MFA, consulte Para desactivar un dispositivo MFA de otro usuario de IAM (consola).

Como administrador de AWS, puede volver a sincronizar sus dispositivos MFA físicos y virtuales de los usuarios de IAM si pierden la sincronización.

Si el dispositivo MFA de Usuario raíz de la cuenta de AWS no funciona, puede volver a sincronizarlo con la consola de IAM y completar o no el proceso de inicio de sesión. Si no puedes resincronizar correctamente el dispositivo, es posible que tengas que desasociarlo y volver a asociarlo. Para obtener más información acerca de cómo hacerlo, consulte Desactivación de dispositivos MFA y Habilitación de dispositivos MFA para usuarios en AWS.

Temas

- Permisos necesarios
- Resincronización de dispositivos MFA físicos y virtuales (consola de IAM)
- Resincronización de dispositivos MFA físicos y virtuales (AWS CLI)
- Resincronización de dispositivos MFA físicos y virtuales (API de AWS)

Permisos necesarios

Para resincronizar dispositivos MFA virtuales o de hardware para su propio usuario de IAM, debe tener los permisos de la siguiente política. Esta política no permite crear ni desactivar un dispositivo.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToViewAndManageTheirOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "BlockAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

Resincronización de dispositivos MFA físicos y virtuales (consola de IAM)

Puede utilizar la consola de IAM para volver a sincronizar dispositivos de MFA físicos y virtuales.

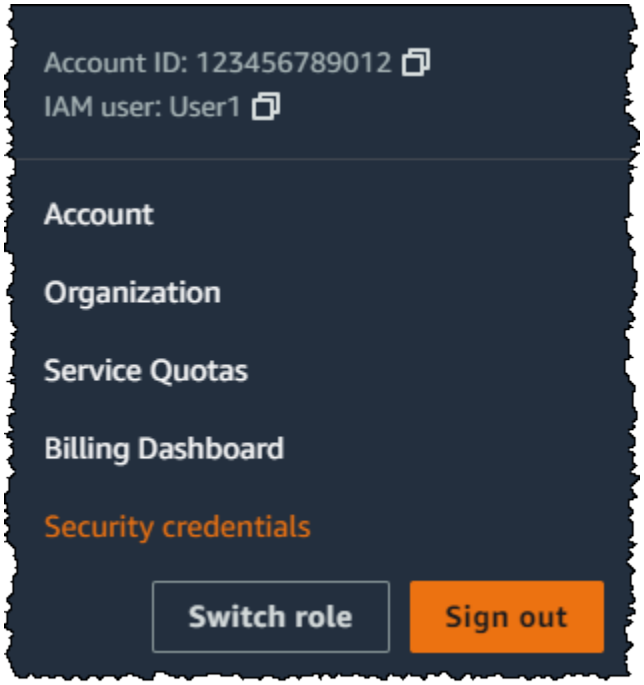
Para volver a sincronizar un dispositivo MFA físico o virtual para su propio usuario de IAM (consola)

1. Utilice el ID de su cuenta de AWS o el alias de su cuenta, el nombre de usuario de IAM y la contraseña para iniciar sesión en la consola de IAM.

   > **Note**
   >
   > Para su comodidad, la página de inicio de sesión AWS utiliza una cookie del navegador para recordar su nombre de usuario de IAM y la información de su cuenta. Si ha iniciado sesión anteriormente como un usuario diferente, elija Iniciar sesión en otra cuenta cerca del final de la página para volver a la página principal de inicio de sesión. Desde allí, puede escribir su ID de cuenta AWS o su alias de cuenta, de modo que se lo redirija a la página de inicio de sesión del usuario de IAM y tenga acceso a su cuenta.

   Para obtener el ID de la Cuenta de AWS, contacte con su administrador.

2. En la esquina superior derecha de la barra de navegación, elija su nombre de usuario y, a continuación, Security credentials (Credenciales de seguridad).



3. En la pestaña Credenciales de AWS IAM, en la sección Autenticación multifactor (MFA), elija el botón de opción junto al dispositivo MFA y elija Volver a sincronizar.
4. Escriba los dos códigos generados de forma secuencial desde el dispositivo en MFA code 1 (Código MFA 1) y MFA code 2 (Código MFA 2). Luego, seleccione Resync (Volver a sincronizar).

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos. Si genera los códigos y, a continuación, espera demasiado tiempo para enviar la solicitud, la solicitud parece funcionar, pero el dispositivo permanece sin sincronizar. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo.

Para volver a sincronizar un dispositivo MFA físico o virtual para otro usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios) y, a continuación, elija el nombre del usuario cuyo dispositivo MFA debe volver a sincronizarse.
3. Seleccione la pestaña de credenciales de seguridad. En la sección Autenticación multifactor (MFA), elija el botón de opción junto al dispositivo MFA y seleccione Volver a sincronizar.
4. Escriba los dos códigos generados de forma secuencial desde el dispositivo en MFA code 1 (Código MFA 1) y MFA code 2 (Código MFA 2). Luego, seleccione Resync (Volver a sincronizar).

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos. Si genera los códigos y, a continuación, espera demasiado tiempo para enviar la solicitud, la solicitud parece funcionar, pero el dispositivo permanece sin sincronizar. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo.

Para volver a sincronizar su MFA de usuario raíz antes de iniciar sesión (consola)

1. En la página Inicio de sesión en Amazon Web Services con un dispositivo de autenticación, seleccione ¿Tiene problemas con su dispositivo de autenticación? Click here (Haga clic aquí).

   > **Note**
   >
   > Puede ver texto diferente, como, por ejemplo, Iniciar sesión mediante MFA y Solución de problemas con el dispositivo de autenticación. Sin embargo, se proporcionan las mismas características.

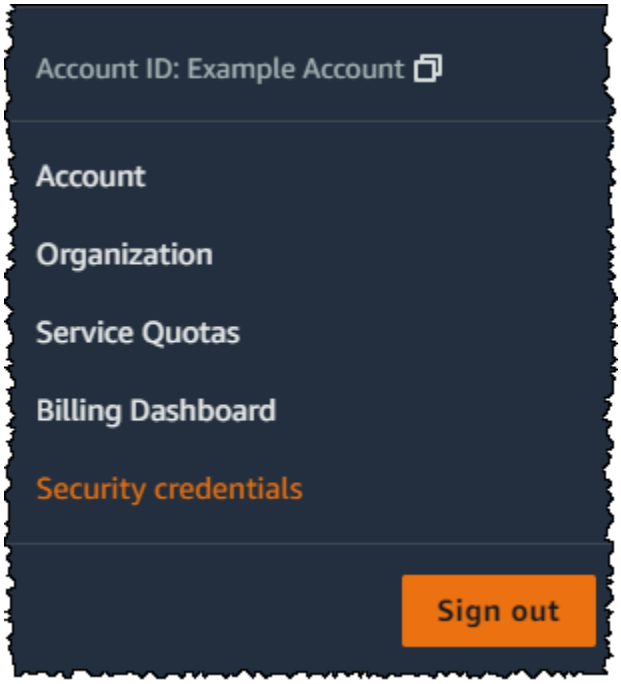
2. En la sección Re-Sync With Our Servers (Volver a sincronizar con nuestros servidores), escriba los dos códigos generados de forma secuencial desde el dispositivo en MFA code 1 (Código MFA 1) y MFA code 2 (Código MFA 2). A continuación, seleccione Re-sync authentication device (Volver a sincronizar dispositivo de autenticación).
3. Si es necesario, escriba la contraseña de nuevo y elija Sign in (Iniciar sesión). Finalmente, complete el inicio de sesión con su dispositivo MFA.

Para volver a sincronizar su dispositivo MFA de su usuario raíz tras iniciar sesión (consola)

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   > **Note**
   >
   > Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En la parte derecha de la barra de navegación, elija su nombre de cuenta y, a continuación, Security credentials (Credenciales de seguridad). Si es necesario, elija Continue to Security Credentials (Seguir en Credenciales de seguridad).



3. Expanda la sección Multi-factor authentication (MFA) (Autenticación multifactor [MFA]) en la página.
4. Seleccione el botón de opción situado junto al dispositivo y haga clic en Resync (Volver a sincronizar).
5. En el cuadro de diálogo Resync MFA device (Volver a sincronizar el dispositivo MFA), escriba los dos códigos generados de forma secuencial desde el dispositivo en MFA code 1 (Código MFA 1) y MFA code 2 (Código MFA 2). Luego, seleccione Resync (Volver a sincronizar).

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos. Si genera los códigos y después espera demasiado tiempo a enviar la solicitud, el dispositivo MFA se asociará correctamente al usuario, pero no estará sincronizado. Esto ocurre porque las contraseñas temporales de un solo uso (TOTP) caducan tras un corto periodo de tiempo.

Resincronización de dispositivos MFA físicos y virtuales (AWS CLI)

Puede volver a sincronizar dispositivos MFA físicos y virtuales con la AWS CLI.

Para volver a sincronizar un dispositivo MFA físico o virtual para un usuario de IAM (AWS CLI)

En una línea de comandos, emita el comando aws iam resync-mfa-device:

- Dispositivo MFA virtual: especifique el nombre de recurso de Amazon (ARN) del dispositivo como el número de serie.

```
aws iam resync-mfa-device --user-name Richard --serial-number
 arn:aws:iam::123456789012:mfa/RichardsMFA --authentication-code1 123456 --
authentication-code2 987654
```

- Dispositivo MFA físico: especifique el número de serie del dispositivo físico como el número de serie. El formato es específico del proveedor. Por ejemplo, puede comprar un token Gemalto de Amazon. Su número de serie suele ser de cuatro letras seguidas de cuatro números.

```
aws iam resync-mfa-device --user-name Richard --serial-number ABCD12345678 --
authentication-code1 123456 --authentication-code2 987654
```

> **Important**
>
> Envíe su solicitud inmediatamente después de generar los códigos. Si genera los códigos y después espera demasiado tiempo para enviar la solicitud, esta dará un error porque los códigos caducan tras un breve intervalo de tiempo.

Resincronización de dispositivos MFA físicos y virtuales (API de AWS)

IAM tiene una llamada a la API que realiza la sincronización. En este caso, le recomendamos que dé permiso a los usuarios de dispositivos MFA físicos y virtuales para obtener acceso a esta llamada a la API. A continuación, debe crear una herramienta basada en esa llamada a la API que permita a sus usuarios volver a sincronizar sus dispositivos siempre que sea necesario.

Para volver a sincronizar un dispositivo MFA físico o virtual para un usuario de IAM (API de AWS)

- Envíe la solicitud ResyncMFADevice.

Desactivación de dispositivos MFA

Si tiene problemas para iniciar sesión con un dispositivo de autenticación multifactor (MFA) como un usuario de IAM, póngase en contacto con su administrador para obtener ayuda.

Como administrador, puede desactivar el dispositivo para otro usuario de IAM. Esto permite al usuario iniciar sesión sin utilizar MFA. Puede hacerlo como una solución temporal, mientras se

sustituye el dispositivo MFA o si no está disponible temporalmente. Sin embargo, le recomendamos que habilite un nuevo dispositivo para el usuario tan pronto como sea posible. Para obtener información sobre cómo habilitar un nuevo dispositivo MFA, consulte the section called “Habilitación de dispositivos MFA”.

> **Note**
>
> Si utiliza la API o la AWS CLI para eliminar un usuario de su Cuenta de AWS, debe desactivar o eliminar el dispositivo MFA del usuario. Puede realizar este cambio como parte del proceso de eliminación del usuario. Para obtener más información sobre cómo eliminar los usuarios, consulte Administración de usuarios de IAM.

Temas

- Desactivación de dispositivos MFA (consola)
- Desactivación de dispositivos MFA (AWS CLI)
- Desactivación de dispositivos MFA (API de AWS)

Desactivación de dispositivos MFA (consola)

Para desactivar un dispositivo MFA de otro usuario de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. Para desactivar el dispositivo MFA para un usuario, elija el nombre del usuario cuyo MFA desea eliminar.
4. Seleccione la pestaña de credenciales de seguridad.
5. En Autenticación multifactor (MFA), elija el botón de opción junto al dispositivo MFA, seleccione Eliminar y luego, Eliminar.

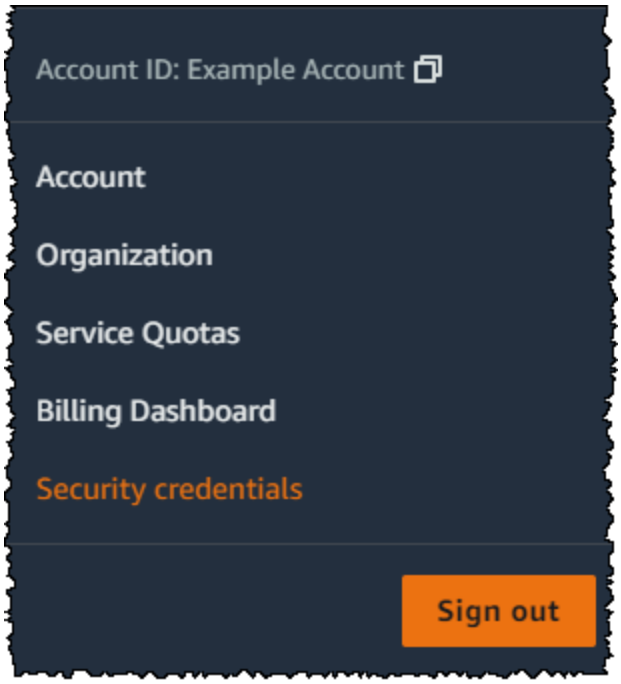
   El dispositivo se elimina de AWS. No puede utilizarse para iniciar sesión ni autenticar solicitudes hasta que se vuelva a activar y asociar a una cuenta de usuario de AWS o Usuario raíz de la cuenta de AWS.

Para deshabilitar el dispositivo de MFA para su usuario Usuario raíz de la cuenta de AWS (consola)

1. Inicie sesión en la consola de IAM como el propietario de la cuenta; para ello, elija Root user (Usuario raíz) e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.

   > **Note**
   >
   > Como usuario raíz, no puede iniciar sesión en la página Iniciar sesión como usuario de IAM. Si aparece la página Iniciar sesión como usuario de IAM, elija Iniciar sesión con el correo electrónico de usuario raíz en la parte inferior de la página. Para obtener ayuda para iniciar sesión como usuario raíz, consulte Inicio de sesión a la AWS Management Console como usuario raíz en la Guía del usuario de AWS Sign-In.

2. En la parte derecha de la barra de navegación, elija su nombre de cuenta y, a continuación, Security credentials (Credenciales de seguridad). Si es necesario, elija Continue to Security Credentials (Seguir en Credenciales de seguridad).



3. En la sección Multi-factor authentication (MFA) (Autenticación multifactor [MFA]), seleccione el botón de opción situado junto al dispositivo MFA que desea desactivar y, a continuación, Remove (Eliminar).
4. Elija Eliminar.

Se ha desactivado el dispositivo MFA para la Cuenta de AWS. Busque en la bandeja del correo electrónico asociado con su Cuenta de AWS un mensaje de confirmación de Amazon Web Services. El correo electrónico le informa de que su autenticación multifactor (MFA) de Amazon Web Services se ha desactivado. El mensaje vendrá de `@amazon.com` o `@aws.amazon.com`.

Desactivación de dispositivos MFA (AWS CLI)

Para desactivar un dispositivo MFA para un usuario de IAM (AWS CLI)

- Ejecute este comando: `aws iam deactivate-mfa-device`

Desactivación de dispositivos MFA (API de AWS)

Para desactivar un dispositivo MFA para un usuario de IAM (API de AWS)

- Llame a esta operación: `DeactivateMFADevice`

¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?

Si el dispositivo MFA virtual o el token TOTP de hardware parece funcionar correctamente, pero no puede utilizarlo para obtener acceso a sus recursos de AWS, es posible que no esté sincronizado con AWS. Para obtener más información acerca de cómo sincronizar un dispositivo de MFA virtual o físico, consulte Resincronización de dispositivos MFA físicos y virtuales. Las claves de seguridad FIDO no pierden la sincronización.

Si el dispositivo de autenticación multifactor (MFA) del Usuario raíz de la cuenta de AWS se pierde, se daña o no funciona, puede recuperar el acceso a su cuenta. Los usuarios de IAM deben ponerse en contacto con un administrador para desactivar el dispositivo.

> **Important**
>
> Le recomendamos que habilite varios dispositivos MFA para sus usuarios de IAM a fin de garantizar el acceso continuo a su cuenta en caso de pérdida o inaccesibilidad del dispositivo MFA. Puede registrar hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA admitidos actualmente con el usuario raíz de la Cuenta de AWS y los usuarios de IAM.

Recuperación de un dispositivo MFA de usuario raíz

Si el dispositivo de autenticación multifactor (MFA) del Usuario raíz de la cuenta de AWS se pierde, se daña o no funciona, puede iniciar sesión con otro dispositivo MFA registrado con el mismo Usuario raíz de la cuenta de AWS. Si el usuario raíz solo tiene un dispositivo MFA activado, puede usar métodos alternativos de autenticación. Esto significa que, si no puede iniciar sesión con su dispositivo MFA, puede iniciar la sesión verificando su identidad mediante el correo electrónico y el número de teléfono de contacto principal registrados en su cuenta.

Antes de utilizar factores alternativos de autenticación para iniciar sesión como usuario raíz, debe poder acceder al correo electrónico y al número de teléfono de contacto principal que están asociados a su cuenta. Si necesita actualizar el número de teléfono de contacto principal, puede iniciar sesión como usuario de IAM con acceso de Administrator (Administrador) en lugar del usuario raíz. Para obtener instrucciones adicionales sobre cómo actualizar la información de contacto de la cuenta, consulte Editar la información de contacto en la Guía del usuario de AWS Billing. Si no tiene acceso a un correo electrónico ni a un número de teléfono de contacto principal, debe contactar con AWS Support.

> **Important**
>
> Se recomienda mantener actualizados la dirección de correo electrónico y el número de teléfono de contacto vinculados al usuario raíz para recuperar la cuenta correctamente. Para obtener más información, consulte Actualizar su contacto principal para Cuenta de AWS en la Guía de referencia de AWS Account Management.

Para iniciar sesión con otros factores de autenticación como Usuario raíz de la cuenta de AWS

1. Inicie sesión en AWS Management Console como propietario de cuenta eligiendo Usuario raíz e ingrese el email de su Cuenta de AWS. En la siguiente página, escriba su contraseña.
2. En la página Se requiere verificación adicional, seleccione un método de MFA con el que autenticarse y elija Siguiente.

   > **Note**
   >
   > Es posible que aparezca un texto alternativo, como Sign in using MFA (Iniciar sesión con MFA), Troubleshoot your authentication device (Solucionar problemas de su dispositivo de autenticación) o Troubleshoot MFA (Solucionar problemas de MFA), pero la funcionalidad es la misma. Si no puede utilizar factores alternativos de autenticación

para verificar la dirección de correo electrónico y el número de teléfono de contacto principal de su cuenta, contacte con AWS Support para desactivar su dispositivo MFA.

3. Según el tipo de MFA que utilice, verá una página diferente, pero la opción Solución de problemas de MFA funciona igual. En la página Se requiere verificación adicional o en la página Autenticación multifactor, elija Solución problemas de MFA.
4. Si se le solicita, escriba la contraseña de nuevo y elija Sign in (Inicio de sesión).
5. En la página Solución de problemas con el dispositivo de autenticación, en la sección Iniciar sesión con otros factores de autenticación, elija Iniciar sesión con otros factores.
6. En la página Iniciar sesión con otros factores de autenticación, autentique su cuenta verificando la dirección de correo electrónico y seleccione Enviar correo electrónico de verificación.
7. Busque en la bandeja del correo electrónico asociado a su Cuenta de AWS un mensaje de Amazon Web Services (no-reply-aws@amazon.com). Siga las indicaciones del correo electrónico.

   Si no ve el correo electrónico en su cuenta, revise la carpeta de spam o vuelva a su navegador y seleccione Resend the email (Reenviar el correo electrónico).
8. Después de verificar su dirección de correo electrónico, podrá continuar el proceso de autenticación de la cuenta. Para verificar su número de teléfono de contacto principal, elija Call me now (Llamarme ahora).
9. Responda a la llamada de AWS y, cuando se le solicite, introduzca el número de 6 dígitos del sitio web de AWS en el teclado de su teléfono.

   Si no recibe la llamada de AWS, seleccione Sign in (Iniciar sesión) para iniciar sesión de nuevo en la consola y volver a comenzar. O consulte Lost or unusable Multi-Factor Authentication (MFA) device (Dispositivo de autenticación multifactor [MFA] perdido o inutilizado) para contactar con el servicio de asistencia técnica para obtener ayuda.
10. Después de verificar su número de teléfono, podrá iniciar sesión en su cuenta. Para ello, elija Sign in to the console (Iniciar sesión en la consola).
11. El siguiente paso varía en función del tipo de MFA que utilice:

    - Si utiliza un dispositivo MFA virtual, elimine la cuenta del dispositivo. A continuación, diríjase a la página Credenciales de seguridad de AWS y elimine la entidad de dispositivo MFA virtual antigua antes de crear una nueva.
    - Para obtener una clave de seguridad FIDO, diríjase a la página Credenciales de seguridad de AWS y desactive la clave de seguridad FIDO antigua antes de habilitar una nueva.

- En el caso de un token TOTP de hardware, contacte con el proveedor externo para que le ayude a reparar o sustituir el dispositivo. Puede seguir iniciando sesión a través de factores de autenticación alternativos hasta que reciba su nuevo dispositivo. Una vez que tenga su nuevo dispositivo MFA físico, visite la página Credenciales de seguridad de AWS y elimine la entidad del dispositivo MFA físico antigua antes de crear una nueva.

> **Note**
>
> No es necesario reemplazar un dispositivo MFA perdido o robado con el mismo tipo de dispositivo. Por ejemplo, si se rompe la clave de seguridad FIDO y pide una nueva, puede utilizar MFA virtual o un token TOTP de hardware hasta que reciba una nueva clave de seguridad FIDO.

> **Important**
>
> Si su dispositivo MFA ha desaparecido o se lo han robado, después de iniciar sesión con factores de autenticación alternativos y establecer el dispositivo MFA de reemplazo, cambie la contraseña de usuario raíz en caso de que un atacante haya robado el dispositivo de autenticación y también pueda tener su contraseña actual. Para obtener más información, consulteCambiar la contraseña delUsuario raíz de la cuenta de AWS en la AWS Account Management Guía de referencia.

Recuperación de un dispositivo MFA de usuario de IAM

Si es usuario de IAM y su dispositivo se pierde o deja de funcionar, no puede recuperarlo usted mismo. Debe ponerse en contacto con un administrador para desactivar el dispositivo. Después, puede habilitar un nuevo dispositivo.

Para obtener ayuda relacionada con un dispositivo de MFA asociado a un usuario de IAM

1. Póngase en contacto con el administrador de AWS o cualquier otra persona que le diera el nombre de usuario y la contraseña de usuario de IAM. El administrador debe desactivar el dispositivo de MFA tal y como se describe en Desactivación de dispositivos MFA para que pueda iniciar sesión.
2. El siguiente paso varía en función del tipo de MFA que utilice:

- Si utiliza un dispositivo MFA virtual, elimine la cuenta del dispositivo. A continuación, active el dispositivo virtual tal y como se describe en Habilitación de un dispositivo de autenticación multifactor (MFA) virtual (consola).
- Si se trata de una clave de seguridad FIDO, contacte con el proveedor externo para que le ayude a sustituir el dispositivo. Cuando reciba la nueva clave de seguridad FIDO, habilítela como se describe en Habilitación de una clave de seguridad FIDO (consola).
- En el caso de un token TOTP de hardware, contacte con el proveedor externo para que le ayude a reparar o sustituir el dispositivo. Una vez que tenga el nuevo dispositivo de MFA físico, habilítelo tal y como se describe en Habilitar un token TOTP físico (consola).

> **Note**
>
> No es necesario reemplazar un dispositivo MFA perdido o robado con el mismo tipo de dispositivo. Puede tener hasta ocho dispositivos MFA de cualquier combinación. Por ejemplo, si se rompe la clave de seguridad FIDO y pide una nueva, puede utilizar MFA virtual o un token TOTP de hardware hasta que reciba una nueva clave de seguridad FIDO.

3. Si su dispositivo MFA ha sido robado o se ha extraviado, cambie su contraseña para que la persona que tenga el dispositivo de autenticación no tenga también su contraseña actual. Para obtener más información, consultar Administración de las contraseñas de los usuarios de IAM

Configuración del acceso a una API protegido por MFA

Con las políticas de IAM, puede especificar qué operaciones de API puede llamar un usuario. En algunos casos, es posible que quiera añadir más seguridad y exigir a los usuarios que se autentiquen mediante la autenticación multifactor (MFA) de AWS para permitirles llevar a cabo acciones especialmente confidenciales.

Por ejemplo, es posible que tenga una política que permita a un usuario realizar las acciones de Amazon EC2 `RunInstances`, `DescribeInstances` y de `StopInstances`. Sin embargo, es posible que quiera restringir una acción destructiva, como `TerminateInstances` y asegurarse de que los usuarios solo pueden realizar esta acción si se autentican mediante un dispositivo MFA de AWS.

Temas

- Información general
- Situación: protección de MFA para la delegación entre cuentas
- Situación: protección de MFA para el acceso a operaciones de API de la cuenta actual
- Situación: protección de MFA para recursos que tienen políticas basadas en recursos

Información general

Para agregar la protección de MFA a las operaciones de API es preciso realizar estas tareas:

1. El administrador configura un dispositivo de MFA de AWS para cada usuario que necesite realizar solicitudes de API que requieran una autenticación MFA. Este proceso se describe en Habilitación de dispositivos MFA para usuarios en AWS.
2. El administrador crea políticas para los usuarios que contienen un elemento `Condition` que comprueba si el usuario se ha autenticado con un dispositivo MFA de AWS.
3. El usuario llama a una de las operaciones de la API de AWS STS que admiten los parámetros de MFA AssumeRole o GetSessionToken, según la situación de la protección de MFA, tal y como se explica más adelante. Dentro de la llamada, el usuario incluye el identificador del dispositivo que está asociado al usuario. El usuario también incluye la contraseña de un solo uso basada en el tiempo (TOTP) que el dispositivo genera. En cualquier caso, el usuario obtiene credenciales de seguridad temporales que puede utilizar para realizar solicitudes adicionales a AWS.

> **Note**
>
> La protección de MFA para las operaciones de API de un servicio solo está disponible si el servicio es compatible con las credenciales de seguridad temporales. Para obtener una lista de estos servicios, consulte Uso de credenciales de seguridad temporales para acceder a AWS.

Si se produce un error en la autorización, AWS devuelve un mensaje de error de acceso denegado (al igual que con cualquier otro acceso no autorizado). Con las políticas de API protegidas mediante MFA, AWS deniega el acceso a las operaciones de API especificadas en las políticas si el usuario intenta llamar a una operación de API sin una autenticación MFA válida. La operación también se deniega si la marca temporal de la solicitud de la operación API no entra en el rango especificado en la política. Debe volver a autenticarse al usuario en MFA solicitando nuevas credenciales de seguridad temporales con un código de MFA y el número de serie del dispositivo.

Políticas de IAM con condiciones de MFA

Las políticas con condiciones de MFA se pueden asociar a los elementos siguientes:

- Un usuario o grupo de IAM
- Un recurso, como un bucket de Amazon S3, una cola de Amazon SQS o un tema de Amazon SNS
- La política de confianza de un rol de IAM que un usuario puede asumir

Puede utilizar una condición de MFA de una política para comprobar las propiedades siguientes:

- Existencia: para simplemente verificar que el usuario se autenticó realmente con MFA, compruebe que la clave `aws:MultiFactorAuthPresent` sea `True` en una condición `Bool`. La clave solo está presente cuando el usuario se autentica con credenciales a corto plazo. Las credenciales a largo plazo, como, por ejemplo, las claves de acceso, no contienen esta clave.
- Duración: si quiere conceder acceso únicamente dentro de un periodo determinado de tiempo después de la autenticación MFA, utilice un tipo de condición numérica para comparar la edad de la clave `aws:MultiFactorAuthAge` con un valor (por ejemplo, 3600 segundos). Tenga en cuenta que la clave `aws:MultiFactorAuthAge` no está presente si no se ha utilizado la MFA.

El siguiente ejemplo muestra la política de confianza de un rol de IAM que contiene una condición MFA para probar la existencia de la autenticación MFA. Con esta política, los usuarios de la cuenta de Cuenta de AWS especificada en el elemento `Principal` (sustituir `ACCOUNT-B-ID` por un ID de cuenta de Cuenta de AWS válido) pueden asumir la función a la que esta política está asociada. Sin embargo, tales usuarios solo puede asumir la función si el usuario está autenticado con MFA.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

Para obtener más información sobre los tipos de condición para MFA, consulte Claves de contexto de condición globales de AWS, Operadores de condición numérica y Operador de condición para comprobar la existencia de claves de condición .

Elegir entre GetSessionToken y AssumeRole

AWS STS proporciona dos operaciones de la API que permiten a los usuarios transmitir información de MFA: `GetSessionToken` y `AssumeRole`. La operación de API a la que el usuario llama para obtener credenciales de seguridad temporales depende de la situación a la que la API se aplique.

Utilice **`GetSessionToken`** en las situaciones siguientes:

- Llamadas a operaciones de API que tienen acceso a los recursos en la misma cuenta de Cuenta de AWS que el usuario de IAM que realiza la solicitud. Tenga en cuenta que las credenciales temporales de una solicitud `GetSessionToken` pueden obtener acceso a IAM y a las operaciones de la API de AWS STS y solo si contienen información de MFA en la solicitud de credenciales. Dado que las credenciales temporales devueltas por `GetSessionToken` contienen información de MFA, puede buscar la presencia de MFA en llamadas a las operaciones de API realizadas por las credenciales.
- El acceso a recursos que están protegidos con políticas basadas en recursos que contienen una condición de MFA.

La finalidad de la operación `GetSessionToken` es autenticar al usuario mediante MFA. No se pueden utilizar políticas para controlar las operaciones de autenticación.

Utilice **`AssumeRole`** en las situaciones siguientes:

- Llamadas a operaciones de API que obtienen acceso a los recursos que están en la misma cuenta de Cuenta de AWS o en otra. Las llamadas a la API pueden incluir cualquier API de IAM o AWS STS. Tenga en cuenta que, para proteger el acceso, aplicar MFA en el momento en que el usuario asume el rol. Las credenciales temporales devueltas por `AssumeRole` no contienen información de MFA en el contexto, por lo que no puede buscar la presencia de MFA en operaciones de API individuales. Por ello debe utilizar `GetSessionToken` para restringir el acceso a los recursos protegidos por políticas basadas en recursos.

Más adelante, se proporciona información detallada sobre cómo implementar estas situaciones.

Información importante sobre el acceso mediante API protegido por MFA

Es importante comprender los siguientes aspectos de la protección de operaciones de API con MFA:

- La protección con MFA solo está disponible con credenciales de seguridad temporales, las cuales deben obtenerse con `AssumeRole` o `GetSessionToken`.

- No puede utilizar el acceso a API protegido por MFA con las credenciales de usuario Usuario raíz de la cuenta de AWS.
- No puede utilizar el acceso a API protegido por MFA con las llaves de seguridad U2F.
- A los usuarios federados no se les puede asignar un dispositivo MFA para utilizarlo con servicios de AWS, por lo que no pueden obtener acceso a recursos de AWS controlados por MFA. (véase el punto siguiente).
- Las demás operaciones de API de AWS STS que devuelven credenciales temporales no admiten MFA. En `AssumeRoleWithWebIdentity` y `AssumeRoleWithSAML`, un proveedor externo autentica al usuario y AWS no puede determinar si ese proveedor exige una MFA. En `GetFederationToken`, MFA no está obligatoriamente asociado a un usuario concreto.
- Del mismo modo, las credenciales a largo plazo (claves de acceso de usuario de IAM y claves de acceso de usuario raíz) no se pueden utilizar con el acceso mediante API protegido por MFA, ya que no caducan.
- También se puede llamar a `AssumeRole` y `GetSessionToken` sin información de MFA. En este caso, el intermediario obtiene credenciales de seguridad temporales, pero la información de la sesión para dichas credenciales temporales no indica si el usuario se autenticó con MFA.
- Para establecer una protección de MFA para las operaciones de API, agregue condiciones de MFA a las políticas. Una política debe incluir la clave de condición `aws:MultiFactorAuthPresent` para aplicar el uso de MFA. Para la delegación entre cuentas, la política de confianza de la función debe incluir la clave de condición.
- Cuando se permite que otra cuenta de Cuenta de AWS obtenga acceso a los recursos de su cuenta, la seguridad de los recursos dependerá de la configuración de la cuenta de confianza; es decir, de la otra cuenta (no de la suya). Esto es válido aunque se exija la autenticación multifactor. Cualquier identidad de la cuenta de confianza que tenga permiso para crear dispositivos MFA virtuales puede crear una notificación de MFA que respete la parte de la política de confianza de su rol. Antes de permitir que los miembros de otra cuenta obtengan acceso a sus recursos de AWS que requieren autenticación multifactor, debe asegurarse de que el propietario de la cuenta de confianza aplique las prácticas recomendadas de seguridad. Por ejemplo, la cuenta de confianza debe restringir exclusivamente a identidades concretas y de confianza el acceso a las operaciones de API confidenciales, tales como las operaciones de API de administración de dispositivos MFA.
- Si una política contiene una condición de MFA, se deniega una solicitud si los usuarios de esta no se han autenticado con MFA o si proporcionan un identificador de dispositivo MFA no válido o una TOTP no válida.

Situación: protección de MFA para la delegación entre cuentas

En este caso, quiere delegar el acceso de usuarios de IAM a otra cuenta, pero solo si dichos usuarios se autentican con un dispositivo MFA de AWS. Para obtener más información acerca de la delegación entre cuentas, consulte Términos y conceptos de roles.

Supongamos que tiene una cuenta A (la cuenta que confía y es propietaria del recurso al que debe accederse) con la usuaria de IAM Anaya, que tiene permiso de administrador. Alice quiere conceder acceso al usuario Richard de la cuenta B (la cuenta de confianza), pero antes quiere asegurarse de que Richard se autentica con MFA antes de asumir el rol.

1. En la cuenta que confía A, Anaya crea un rol de IAM denominado `CrossAccountRole` y establece la entidad principal de la política de confianza del rol en el ID de la cuenta B. La política de confianza concede permiso a la acción AWS STS `AssumeRole`. Anaya también añade una condición de MFA a la política de confianza, como la del siguiente ejemplo.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "ACCOUNT-B-ID"},
    "Action": "sts:AssumeRole",
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }
}
```

2. Anaya añade una política de permisos al rol que especifica lo que le permite hacer la función. La política de permisos de un rol con protección de MFA no es diferente de cualquier otra política de permisos de rol. En el siguiente ejemplo, se muestra la política que Anaya agrega al rol; permite al usuario que la asume realizar cualquier acción de Amazon DynamoDB en la tabla `Books` de la cuenta A. Esta política también permite la acción `dynamodb:ListTables`, que es necesaria para llevar a cabo acciones en la consola.

> **Note**
>
> La política de permisos no incluye una condición de MFA. Es importante comprender que la autenticación MFA solo se usa para determinar si un usuario puede asumir el rol. Una vez que el usuario haya asumido el rol, no se realizarán más verificaciones de MFA.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID:table/Books"
        },
        {
            "Sid": "ListTables",
            "Effect": "Allow",
            "Action": "dynamodb:ListTables",
            "Resource": "*"
        }
    ]
}
```

3. En la cuenta de confianza B, el administrador se asegura de que el usuario de IAM Richard se haya configurado con un dispositivo MFA de AWS y de que conozca el ID del dispositivo. El ID de dispositivo es el número de serie si se trata de un dispositivo MFA físico, o bien el ARN si se trata de un dispositivo MFA virtual.
4. En la cuenta B, el administrador asocia la siguiente política al usuario Richard (o a un grupo del que sea miembro) que le permita llamar a la acción `AssumeRole`. El recurso se establece en el ARN del rol que Anaya creó en el paso 1. Observe que esta política no contiene una condición de MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["sts:AssumeRole"],
    "Resource": ["arn:aws:iam::ACCOUNT-A-ID:role/CrossAccountRole"]
  }]
}
```

5. En la cuenta B, Richard (o una aplicación que Richard esté ejecutando) llama a `AssumeRole`. La llamada a la API contiene el ARN del rol que se asumirá (`arn:aws:iam::ACCOUNT-A-`

ID:role/CrossAccountRole), el ID del dispositivo de MFA, y la TOTP actual que Richard obtiene de su dispositivo.

Cuando Richard llama a AssumeRole, AWS determina si tiene credenciales válidas, incluido la exigencia de MFA. En caso afirmativo, Richard asume el rol efectivamente y puede realizar cualquier acción de DynamoDB de la tabla denominada Books de la cuenta A con las credenciales temporales del rol.

Si desea ver un ejemplo de un programa que llame a AssumeRole, consulte Llamada a AssumeRole con autenticación MFA.

Situación: protección de MFA para el acceso a operaciones de API de la cuenta actual

En este caso, debe asegurarse de que un usuario de su cuenta de Cuenta de AWS pueda obtener acceso a operaciones de la API confidenciales solamente cuando el usuario se haya autenticado con un dispositivo MFA de AWS.

Supongamos que tiene una cuenta A que contiene un grupo de desarrolladores que necesitan trabajar con instancias EC2. Los desarrolladores normales pueden trabajar con las instancias, pero no se les conceden permisos para las acciones ec2:StopInstances o ec2:TerminateInstances. Usted quiere limitar estas acciones privilegiadas "destructivas" solo a unos cuantos usuarios de confianza, por lo que añade protección de MFA a la política que permite estas acciones de Amazon EC2 de gran importancia.

En este escenario, uno de los usuarios de confianza es Sofía. La usuaria Anaya es administradora de la cuenta A.

1. Anaya se asegura de que Sofía esté configurada con un dispositivo MFA de AWS y de que Sofía conozca el ID del dispositivo. El ID de dispositivo es el número de serie si se trata de un dispositivo MFA físico, o bien el ARN si se trata de un dispositivo MFA virtual.
2. Anaya crea un grupo denominado EC2-Admins y añade a la usuario Sofía al grupo.
3. Anaya asocia la siguiente política al grupo EC2-Admins. Esta política concede a los usuarios permiso para llamar a las acciones de Amazon EC2 StopInstances y TerminateInstances solo si el usuario se ha autenticado con MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [{
```

```
    "Effect": "Allow",
    "Action": [
      "ec2:StopInstances",
      "ec2:TerminateInstances"
    ],
    "Resource": ["*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

4.
> **Note**
>
> Para que esta política surta efecto, antes los usuarios deben cerrar la sesión e iniciarla de nuevo.

   Si el usuario Sofía tiene que detener o terminar una instancia de Amazon EC2, ella (o una aplicación que esté ejecutando) llamará a `GetSessionToken`. Esta operación de API transmite el ID del dispositivo MFA y la TOTP actual que Sofía obtiene de su dispositivo.

5. La usuaria Sofía (o una aplicación que Sofía esté usando) utiliza las credenciales temporales que ofrece `GetSessionToken` para llamar a la acción de Amazon EC2 `StopInstances` o la acción de `TerminateInstances`.

   Si desea ver un ejemplo de un programa que llame a `GetSessionToken`, consulte Llamada a GetSessionToken con autenticación MFA, más adelante en el documento.

Situación: protección de MFA para recursos que tienen políticas basadas en recursos

En este caso, es usted propietario de un bucket de S3, una cola de SQS o un tema de SNS. Quiere asegurarse de que todos los usuarios de todas las cuentas de Cuenta de AWS que tengan acceso al recurso se autentiquen mediante un dispositivo MFA de AWS.

Esta situación ilustra una forma de proporcionar una protección MFA entre cuentas en la que no se exija al usuario que asuma primero un rol. En este caso, el usuario puede obtener acceso al recursos si se cumplen tres condiciones. el usuario debe estar autenticado por MFA, debe poder obtener credenciales de seguridad temporales de `GetSessionToken` y debe pertenecer a una cuenta que sea de confianza para la política del recurso.

Supongamos que está en la cuenta A y que crea un bucket de S3. Quiere conceder acceso a este bucket a usuarios que están en varias cuentas de Cuentas de AWS, pero solo si dichos usuarios están autenticados con MFA.

En este escenario, la usuaria Anaya es administradora de la cuenta A. El usuario Nikhil es un usuario de IAM de la cuenta C.

1. En la cuenta A, Anaya crea un bucket denominado `Account-A-bucket`.
2. Anaya añade la política de bucket al bucket. La política permite a todos los usuarios de la cuenta A, la cuenta B o la cuenta C las acciones de Amazon S3 `PutObject` y de `DeleteObject` en el bucket. La política contiene una condición de MFA.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"AWS": [
      "ACCOUNT-A-ID",
      "ACCOUNT-B-ID",
      "ACCOUNT-C-ID"
    ]},
    "Action": [
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::ACCOUNT-A-BUCKET-NAME/*"],
    "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
  }]
}
```

> **Note**
>
> Amazon S3 tiene una función de eliminación de MFA para el acceso de cuenta raíz (solo). Puede habilitar la eliminación de MFA de Amazon S3 cuando establezca el estado de control de versiones del bucket. La eliminación de MFA de Amazon S3 no se puede aplicar a un usuario de IAM y se administra de forma independiente del acceso mediante API protegido por MFA. Un usuario de IAM con permisos para eliminar un bucket no puede eliminar un bucket si la eliminación de MFA de Amazon S3 está habilitada. Para

obtener más información acerca de la eliminación de MFA de Amazon S3, consulte Eliminación de MFA.

3. En la cuenta de confianza C, un administrador se asegura de que el usuario Nikhil se haya configurado con un dispositivo MFA de AWS y de que conozca el ID del dispositivo. El ID de dispositivo es el número de serie si se trata de un dispositivo MFA físico, o bien el ARN si se trata de un dispositivo MFA virtual.
4. En la cuenta C, Nikhil (o una aplicación que esté ejecutando) llama a `GetSessionToken`. La llamada contiene el ID o el ARN del dispositivo de MFA y la TOTP actual que Nikhil obtiene de su dispositivo.
5. Nikhil (o una aplicación que esté utilizando) utiliza las credenciales temporales devueltas por `GetSessionToken` para llamar a la acción `PutObject` Amazon S3 para cargar un archivo en `Account-A-bucket`.

   Si desea ver un ejemplo de un programa que llame a `GetSessionToken`, consulte Llamada a GetSessionToken con autenticación MFA, más adelante en el documento.

> **Note**
>
> Las credenciales temporales que `AssumeRole` devuelve no funcionarán en este caso. Aunque el usuario puede proporcionar información de MFA para asumir un rol, las credenciales temporales devueltas por `AssumeRole` no contienen la información de MFA. Esa información se requiere para cumplir la condición de MFA de la política.

Código de muestra: solicitud de credenciales con autenticación multifactor

En los ejemplos siguientes se muestra cómo llamar a las operaciones `GetSessionToken` y `AssumeRole` y transmitir los parámetros de autenticación MFA. No se requieren permisos para llamar a `GetSessionToken`, pero debe tener una política que le permita llamar a `AssumeRole`. Las credenciales devueltas se utilizan para enumerar todos los buckets de S3 de la cuenta.

Llamada a GetSessionToken con autenticación MFA

Los siguientes ejemplos muestran cómo llamar a `GetSessionToken` y transmitir la información de autenticación MFA. Las credenciales de seguridad temporales devueltas por la operación `GetSessionToken` se utilizarán para enumerar todos los buckets de S3 de la cuenta.

La política adjunta al usuario que ejecuta este código (o a un grupo al que pertenezca el usuario) proporciona los permisos para las credenciales temporales devueltas. En el código de este ejemplo, la política debe conceder al usuario permiso para solicitar la operación de Amazon S3 `ListBuckets`.

En los siguientes ejemplos de código se muestra cómo obtener un token de sesión con AWS STS y utilizarlo para hacer una acción de servicio que requiere un token MFA.

CLI

AWS CLI

Cómo obtener un conjunto de credenciales a corto plazo para una identidad de IAM

El siguiente comando `get-session-token` recupera un conjunto de credenciales a corto plazo para la identidad de IAM que realiza la llamada. Las credenciales resultantes se pueden utilizar para las solicitudes donde la política requiere la autenticación multifactor (MFA). Las credenciales caducan 15 minutos después de haberse generado.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Salida:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Para obtener más información, consulte Solicitud de credenciales de seguridad temporales en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetSessionToken en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga un token de sesión pasando un token MFA y utilícelo para enumerar los buckets de Amazon S3 de la cuenta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
```

```
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Para obtener detalles sobre la API, consulte GetSessionToken en la Referencia de la API de AWS SDK para Python (Boto3).

Llamada a AssumeRole con autenticación MFA

Los siguientes ejemplos muestran cómo llamar a `AssumeRole` y transmitir la información de autenticación MFA. Las credenciales de seguridad temporales devueltas por `AssumeRole` se utilizan para enumerar todos los buckets de Amazon S3 de la cuenta.

Para obtener más información acerca de esta situación, consulte Situación: protección de MFA para la delegación entre cuentas.

Los siguientes ejemplos de código muestran cómo asumir un rol con AWS STS.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;
```

```
namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");

            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
```

```
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
```

```
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
##############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
```

```
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }
```

```
    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo asumir un rol

El siguiente comando `assume-role` recupera un conjunto de credenciales a corto plazo para el rol de IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Salida:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

El resultado del comando contiene una clave de acceso, una clave secreta y un token de sesión que puede utilizar para autenticarse con AWS.

Para el uso de la CLI de AWS, puede configurar un perfil con nombre asociado a un rol. Cuando utilice el perfil, la CLI de AWS llamará a assume-role y administrará las credenciales por usted. Para obtener más información, consulte Uso de un rol de IAM en la CLI de AWS en la Guía del usuario de la CLI de AWS.

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
```

```
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
```

```
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree el cliente.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asuma un rol de IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
```

```
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Asuma un rol de IAM que requiera un token MFA y utilice credenciales temporales para enumerar los buckets de Amazon S3 para la cuenta.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.
```

```
   The assumed role must grant permission to list the buckets in the other
account.

   :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                           grants access to list the other account's buckets.
   :param session_name: The name of the STS session.
   :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                             device, this is an ARN.
   :param mfa_totp: A time-based, one-time password issued by the MFA device.
   :param sts_client: A Boto3 STS instance that has permission to assume the
role.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn,
       RoleSessionName=session_name,
       SerialNumber=mfa_serial_number,
       TokenCode=mfa_totp,
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   s3_resource = boto3.resource(
       "s3",
       aws_access_key_id=temp_credentials["AccessKeyId"],
       aws_secret_access_key=temp_credentials["SecretAccessKey"],
       aws_session_token=temp_credentials["SessionToken"],
   )

   print(f"Listing buckets for the assumed role's account:")
   for bucket in s3_resource.buckets.all():
       print(bucket.name)
```

- Para obtener detalles sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                async throws -> STSClientTypes.Credentials {
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        throw error
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Swift.

Búsqueda de credenciales AWS no utilizadas

Para aumentar la seguridad de su cuenta de Cuenta de AWS, elimine las credenciales de usuario de IAM (es decir, contraseñas y claves de acceso) que no sean necesarias. Por ejemplo, cuando los usuarios dejen su organización o ya no necesiten obtener acceso a AWS, busque las credenciales que utilizaron y asegúrese de que ya no sean operativas. Lo ideal es eliminar las credenciales si ya no son necesarias. Siempre puede volver a crearlas más tarde, en caso de que surja la necesidad. Como mínimo, debe cambiar la contraseña o desactivar las claves de acceso para que los antiguos usuarios ya no puedan obtener acceso.

Por supuesto, la definición de sin utilizar puede variar y normalmente significa una credencial que no se ha utilizado en un periodo de tiempo especificado.

Búsqueda de contraseñas no utilizadas

Puede utilizar la AWS Management Console para ver la información de uso de la contraseña por parte de sus usuarios. Si tiene una gran cantidad de usuarios, puede utilizar la consola para descargar un informe de credenciales que le indique cuándo cada usuario utilizó por última vez su contraseña de la consola. También puede obtener acceso a la información desde la AWS CLI o la API de IAM.

Para encontrar contraseñas no utilizadas (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).
3. Si es necesario, añada la columna Console last sign-in (Último inicio de sesión de la consola) a la tabla de usuarios:

   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ( ).

   b. En Seleccionar columnas visibles, seleccione Último inicio de sesión de la consola.

   c. Elija Confirmar para volver a la lista de usuarios.

4. La columna Console last sign-in (Último inicio de sesión de la consola) muestra la fecha de la última vez que el usuario inició sesión en AWS a través de la consola. Puede utilizar esta información para encontrar usuarios que tengan contraseñas y que no hayan iniciado sesión en un periodo de tiempo superior al especificado. La columna muestra Never (Nunca) para los usuarios que tengan contraseñas y que nunca hayan iniciado sesión. None (Ninguna) indica los usuarios sin contraseñas. Las contraseñas que no se hayan utilizado recientemente probablemente deban eliminarse.

> **Important**
>
> Debido a un problema de servicio, los datos de la última vez que se utilizó la contraseña no incluyen el uso de la contraseña desde el 3 de mayo de 2018 22:50 PDT al 23 de mayo de 2018 14:08 PDT. Esto afecta a las fechas del último inicio de sesión mostradas en la consola de IAM y las fechas de la última vez que se utilizó la contraseña en el Informe de credenciales de IAM y devueltas mediante la Operación de la API GetUser. Si los usuarios han iniciado sesión durante el tiempo afectado, la fecha de la última vez que se utilizó la contraseña que se devuelve es la fecha en que el usuario inició sesión antes del 3 de mayo de 2018. Para los usuarios que iniciaron sesión después del 23 de mayo de 2018 14:08 PDT, la fecha devuelta de la última vez que se utilizó la contraseña es precisa.
> Si utiliza la información de la última vez que se utilizó la contraseña para identificar las credenciales de no utilizados para su eliminación, como, por ejemplo, eliminar los usuarios que no iniciaron sesión en AWS en los últimos 90 días, recomendamos ajustar la ventana de evaluación para incluir fechas después del 23 de mayo de 2018. De forma alternativa, si los usuarios utilizan claves de acceso para acceder a AWS mediante programación puede hacer referencia a la información de la última vez que se utilizó la clave de acceso ya que es precisa para todas las fechas.

Para encontrar contraseñas no utilizadas descargando el informe de credenciales (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En panel de navegación, elija Credential report (Informe de credenciales).
3. Seleccione Download Report (Descargar informe) para descargar un archivo de valores separados por comas (CSV) denominado `status_reports_<date>T<time>.csv`. La quinta columna es la columna `password_last_used` con las fechas o uno de los títulos siguientes:

- N/A - Usuarios que no tienen una contraseña asignada en absoluto.
- no_information (sin_información) - Usuarios que no han utilizado la contraseña desde que IAM comenzó a hacer un seguimiento del tiempo de la contraseña, el 20 de octubre de 2014.

Para encontrar contraseñas no utilizadas (AWS CLI)

Ejecute el siguiente comando para encontrar contraseñas no utilizadas:

- `aws iam list-users` devuelve una lista de usuarios, cada uno con un valor `PasswordLastUsed`. Si el valor no aparece significa que el usuario no tiene contraseña o no la ha utilizado desde que IAM comenzó a hacer un seguimiento de la antigüedad de las contraseñas, el 20 de octubre de 2014.

Para encontrar contraseñas no utilizadas (API de AWS)

Llame a la siguiente operación para encontrar contraseñas no utilizadas:

- `ListUsers` devuelve una colección de usuarios; cada uno con un valor <PasswordLastUsed>. Si el valor no aparece significa que el usuario no tiene contraseña o no la ha utilizado desde que IAM comenzó a hacer un seguimiento de la antigüedad de las contraseñas, el 20 de octubre de 2014.

Para obtener más información sobre los comandos de descarga del informe de credenciales, consulte Obtención de informes de credenciales (AWS CLI).

Búsqueda de claves de acceso no utilizadas

Puede utilizar la AWS Management Console para ver la información de uso de la clave de acceso de sus usuarios. Si tiene una gran cantidad de usuarios, puede utilizar la consola para descargar un informe de credenciales para saber cuándo cada usuario utilizó por última vez sus claves de acceso de la consola. También puede obtener acceso a la información desde la AWS CLI o la API de IAM.

Para encontrar claves de acceso no utilizadas (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios).

3. Si es necesario, añada la columna Access key last used (Último uso de la clave de acceso) a la tabla de usuarios:

   a. Encima de la tabla, en el extremo derecho, elija el icono de configuración ( ).

   b. En Seleccionar columnas visibles, seleccione Último uso de la clave de acceso.

   c. Elija Confirmar para volver a la lista de usuarios.

4. La columna Access key last used (Último uso de la clave de acceso) muestra el número de días que ha pasado desde que el usuario obtuvo acceso por última vez a AWS mediante programación. Puede utilizar esta información para encontrar usuarios con claves de acceso que no se han usado por más días que un periodo de tiempo especificado. La columna muestra – cuando los usuarios no tienen claves de acceso. Las claves de acceso que no se han utilizado recientemente probablemente deban eliminarse.

Para encontrar claves de acceso no utilizadas descargando el informe de credenciales (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En panel de navegación, elija Credential Report (Informe de credenciales).
3. Seleccione Download Report (Descargar informe) para descargar un archivo de valores separados por comas (CSV) denominado `status_reports_<date>T<time>.csv`. Las columnas 11 a 13 contienen la información sobre la fecha de última utilización, la región y el servicio de la clave de acceso 1. Las columnas 16 a 18 contienen la misma información sobre la clave de acceso 2. El valor es N/A si el usuario no tiene una clave de acceso o no ha utilizado la clave de acceso desde que IAM comenzó a realizar el seguimiento de la antigüedad de la clave de acceso, el 22 de abril de 2015.

Para encontrar claves de acceso no utilizadas (AWS CLI)

Ejecute los siguientes comandos para encontrar claves de acceso no utilizadas:

- `aws iam list-access-keys` devuelve información sobre las claves de acceso de un usuario, incluido el `AccessKeyID`.
- `aws iam get-access-key-last-used` toma un ID de clave de acceso y devuelve una salida que incluye la `LastUsedDate`, la `Region` en la que se utilizó la clave de acceso por última vez y el `ServiceName` del último servicio solicitado. Si `LastUsedDate` no existe, la clave de acceso no

se ha utilizado desde que IAM comenzó a realizar el seguimiento de la clave de acceso, el 22 de abril de 2015.

Para encontrar claves de acceso no utilizadas (API de AWS)

Llame a las siguientes operaciones para encontrar claves de acceso no utilizadas:

- `ListAccessKeys` devuelve una lista de valores `AccessKeyID` de las claves de acceso que están asociadas al usuario especificado.
- `GetAccessKeyLastUsed` toma un ID de clave de acceso y devuelve una colección de valores. Se incluyen la `LastUsedDate`, la `Region` en la que se utilizó por última vez la clave de acceso y el `ServiceName` del último servicio solicitado. Si el valor no aparece, el usuario no tiene una clave de acceso o no la ha utilizado desde que IAM comenzó a realizar el seguimiento de la antigüedad de la clave de acceso, el 22 de abril de 2015.

Para obtener más información sobre los comandos de descarga del informe de credenciales, consulte Obtención de informes de credenciales (AWS CLI).

Obtención de informes de credenciales para su cuenta de Cuenta de AWS

Puede generar y descargar un informe de credenciales que contenga una lista de todos los usuarios de su cuenta y el estado de sus credenciales, tales como contraseñas, claves de acceso y dispositivos MFA. Puede obtener un informe de credenciales de la AWS Management Console, los SDK de AWS y las herramientas de línea de comandos o la API de IAM.

Puede utilizar los informes de credenciales para fines de auditoría y conformidad. Puede utilizar el informe para auditar los efectos de los requisitos del ciclo de vida de la credencial, como las actualizaciones de contraseñas y claves de acceso. Puede proporcionar el informe a un auditor externo o conceder permisos a un auditor, para que pueda descargar el informe directamente.

Puede generar un informe de credenciales cada cuatro horas. Al solicitar un informe, IAM comprueba primero si se ha generado algún informe para la cuenta de Cuenta de AWS en las últimas cuatro horas. En caso afirmativo, se descarga el informe más reciente. Si el informe más reciente de la cuenta es de hace más de cuatro horas, o si no hay informes anteriores de la cuenta, IAM genera y descarga un nuevo informe.

Temas

- Permisos de necesarios

- Descripción del formato del informe
- Obtención de informes de credenciales (consola)
- Obtención de informes de credenciales (AWS CLI)
- Obtención de informes de credenciales (API de AWS)

Permisos de necesarios

Se necesitan los siguientes permisos para crear y descargar informes:

- Para crear un informe de credenciales: `iam:GenerateCredentialReport`
- Para descargar el informe: `iam:GetCredentialReport`

Descripción del formato del informe

Los informes de credenciales tienen el formato de ficheros CSV (valores separados por comas). Puede abrir archivos CSV con software de hojas de cálculo comunes para analizar, o bien puede crear una aplicación que consuma los archivos CSV mediante programación y realice análisis personalizados.

El archivo CSV contiene las siguientes columnas:

user

Es el nombre fácil de recordar del usuario.

arn

Es el nombre de recurso de Amazon (ARN) del usuario. Para obtener más información sobre los ARN, consulte ARN de IAM.

user_creation_time

Es la fecha y la hora en que se creó el usuario con el formato de fecha y hora ISO 8601.

password_enabled

Cuando el usuario tiene una contraseña, el valor es `TRUE`. En caso contrario, es `FALSE`.El valor de Usuario raíz de la cuenta de AWS es siempre `not_supported`.

password_last_used

La fecha y la hora en que se utilizó por última vez la contraseña de Usuario raíz de la cuenta de AWS o de un usuario para iniciar sesión en un sitio web de AWS, con el formato fecha-hora ISO 8601. Los sitios web de AWS que capturan el momento en que un usuario inició sesión por última vez son la AWS Management Console, los foros de debate de AWS y AWS Marketplace. Cuando una contraseña se utiliza más de una vez en un periodo de 5 minutos, solo se registra el primer uso en este campo.

- El valor de este campo es `no_information` en los siguientes casos:
  - La contraseña del usuario no se ha utilizado nunca.
  - No hay datos de inicio de sesión asociados con la contraseña, como, por ejemplo, cuando la contraseña del usuario no se ha utilizado después de que IAM empezara a realizar el seguimiento de esta información a partir del 20 de octubre de 2014.
- El valor en este campo es `N/A` (no aplicable) cuando el usuario no tiene ninguna contraseña.

Important

Debido a un problema de servicio, los datos de la última vez que se utilizó la contraseña no incluyen el uso de la contraseña desde el 3 de mayo de 2018 22:50 PDT al 23 de mayo de 2018 14:08 PDT. Esto afecta a las fechas del último inicio de sesión mostradas en la consola de IAM y las fechas de la última vez que se utilizó la contraseña en el Informe de credenciales de IAM y devueltas mediante la Operación de la API GetUser. Si los usuarios han iniciado sesión durante el tiempo afectado, la fecha de la última vez que se utilizó la contraseña que se devuelve es la fecha en que el usuario inició sesión antes del 3 de mayo de 2018. Para los usuarios que iniciaron sesión después del 23 de mayo de 2018 14:08 PDT, la fecha devuelta de la última vez que se utilizó la contraseña es precisa.
Si utiliza la información de la última vez que se utilizó la contraseña para identificar las credenciales de no utilizados para su eliminación, como, por ejemplo, eliminar los usuarios que no iniciaron sesión en AWS en los últimos 90 días, recomendamos ajustar la ventana de evaluación para incluir fechas después del 23 de mayo de 2018. De forma alternativa, si los usuarios utilizan claves de acceso para acceder a AWS mediante programación puede hacer referencia a la información de la última vez que se utilizó la clave de acceso ya que es precisa para todas las fechas.

password_last_changed

Es la fecha y la hora en que se definió la contraseña del usuario por última vez con el formato de fecha y hora ISO 8601. Si el usuario no tiene ninguna contraseña, el valor en este campo es `N/A` (no aplicable). El valor de la cuenta de Cuenta de AWS (raíz) es siempre `not_supported`.

password_next_rotation

Si la cuenta tiene una política de contraseñas que requiere la rotación de contraseñas, este campo contiene la fecha y la hora en formato de fecha y hora ISO 8601, cuando el usuario debe definir una contraseña nueva. El valor de la cuenta de Cuenta de AWS (raíz) es siempre `not_supported`.

mfa_active

Si se activa una autenticación multifactor (MFA) para el usuario, el valor será `TRUE`. De lo contrario es `FALSE`.

access_key_1_active

Si el usuario tiene una clave de acceso y el estado de la clave de acceso es `Active`, el valor será `TRUE`. De lo contrario es `FALSE`.

access_key_1_last_rotated

Es la fecha y la hora en que se creó o modificó por última vez la clave de acceso del usuario con el formato de fecha y hora ISO 8601. Si el usuario no tiene ninguna clave de acceso activa, el valor en este campo será `N/A` (no aplicable).

access_key_1_last_used_date

Es la fecha y hora en que se utilizó la clave de acceso del usuario por última vez para iniciar sesión en una solicitud de la API de AWS con el formato de fecha y hora ISO 8601. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo.

El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna clave de acceso.
- La clave de acceso no se ha utilizado nunca.
- La clave de acceso no se ha utilizado después de que IAM empezara a realizar el seguimiento de esta información a partir del 22 de abril de 2015.

access_key_1_last_used_region

Es la región de AWS en la que se ha utilizado por última vez la clave de acceso. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo.

El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna clave de acceso.
- La clave de acceso no se ha utilizado nunca.
- La clave de acceso se utilizó por última vez antes de que IAM empezara a realizar el seguimiento de esta información el 22 de abril de 2015.
- El último servicio utilizado no es específico de una región, como Amazon S3.

access_key_1_last_used_service

Es el servicio de AWS al que se ha accedido más recientemente con la clave de acceso. El valor de este campo utiliza el espacio de nombres del servicio, por ejemplo, `s3` para Amazon S3 y `ec2` para Amazon EC2. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo.

El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna clave de acceso.
- La clave de acceso no se ha utilizado nunca.
- La clave de acceso se utilizó por última vez antes de que IAM empezara a realizar el seguimiento de esta información el 22 de abril de 2015.

access_key_2_active

Si el usuario tiene una segunda clave de acceso y el estado de la segunda clave de acceso es `Active`, el valor será `TRUE`. De lo contrario es `FALSE`.

> **Note**
>
> Los usuarios pueden tener un máximo de dos claves de acceso para facilitar la rotación mediante la actualización de la clave en primer lugar y de la eliminación de la clave anterior en segundo lugar. Para obtener más información acerca de la actualización de las claves de acceso, consulte Actualización de las claves de acceso.

access_key_2_last_rotated

Fecha y hora, en formato de fecha y hora ISO 8601, en las que se creó o modificó por última vez la segunda clave de acceso del usuario. Si el usuario no tiene ninguna segunda clave de acceso activa, el valor en este campo será `N/A` (no aplicable).

access_key_2_last_used_date

Es la fecha y hora en que se utilizó la segunda clave de acceso del usuario por última vez para iniciar sesión en una solicitud de la API de AWS con el formato de fecha y hora ISO 8601. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo.

El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna segunda clave de acceso.
- La segunda clave de acceso del usuario no se ha utilizado nunca.
- La segunda clave de acceso del usuario se utilizó por última vez antes de que IAM empezara a realizar el seguimiento de esta información el 22 de abril de 2015.

access_key_2_last_used_region

Es la región de AWS en la que se ha utilizado por última vez la segunda clave de acceso. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo. El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna segunda clave de acceso.
- La segunda clave de acceso del usuario no se ha utilizado nunca.
- La segunda clave de acceso del usuario se utilizó por última vez antes de que IAM empezara a realizar el seguimiento de esta información el 22 de abril de 2015.
- El último servicio utilizado no es específico de una región, como Amazon S3.

access_key_2_last_used_service

Es el servicio de AWS al que se ha accedido más recientemente con la segunda clave de acceso del usuario. El valor de este campo utiliza el espacio de nombres del servicio, por ejemplo, `s3` para Amazon S3 y `ec2` para Amazon EC2. Si una clave de acceso se utiliza más de una vez en un periodo de 15 minutos, solo se registra el primer uso en este campo. El valor de este campo es `N/A` (no aplicable) en los siguientes casos:

- El usuario no tiene ninguna segunda clave de acceso.

- La segunda clave de acceso del usuario no se ha utilizado nunca.
- La segunda clave de acceso del usuario se utilizó por última vez antes de que IAM empezara a realizar el seguimiento de esta información el 22 de abril de 2015.

cert_1_active

Si el usuario dispone de un certificado de firma X.509 y el estado del certificado es `Active`, este valor es `TRUE`. De lo contrario es `FALSE`.

cert_1_last_rotated

Es la fecha y la hora en que se creó o modificó por última vez el certificado de firma del usuario con el formato de fecha y hora ISO 8601. Si el usuario no tiene ningún certificado de firma activo, el valor en este campo será `N/A` (no aplicable).

cert_2_active

Si el usuario dispone de un segundo certificado de firma X.509 y el estado del certificado es `Active`, este valor es `TRUE`. De lo contrario es `FALSE`.

> **Note**
>
> Los usuarios pueden tener hasta dos certificados de firma X.509, para facilitar la rotación de los certificados.

cert_2_last_rotated

Es la fecha y la hora en que se creó o modificó por última vez el segundo certificado de firma del usuario con el formato de fecha y hora ISO 8601. Si el usuario no tiene ningún segundo certificado de firma activo, el valor en este campo será `N/A` (no aplicable).

Obtención de informes de credenciales (consola)

Puede utilizar la AWS Management Console para descargar un informe de credenciales como archivo CSV (valores separados por comas).

Para descargar un informe de credenciales mediante la (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En panel de navegación, elija Credential report (Informe de credenciales).
3. Elija Download Report (Descargar informe).

Obtención de informes de credenciales (AWS CLI)

Para descargar un informe de credenciales (AWS CLI)

1. Genere un informe de credenciales. AWS almacena un único informe. Si existe un informe, cuando se genera un nuevo informe de credenciales, se sobrescribe el anterior. `aws iam generate-credential-report`
2. Ver el último informe que se generó: `aws iam get-credential-report`

Obtención de informes de credenciales (API de AWS)

Para descargar un informe de credenciales (API de AWS)

1. Genere un informe de credenciales. AWS almacena un único informe. Si existe un informe, cuando se genera un nuevo informe de credenciales, se sobrescribe el anterior. `GenerateCredentialReport`
2. Ver el último informe que se generó: `GetCredentialReport`

Uso de IAM con CodeCommit: credenciales de Git, claves SSH y claves de acceso de AWS

CodeCommit es un servicio de control de versiones administrado que aloja repositorios privados Git en la nube de AWS. Para utilizar CodeCommit, debe configurar su cliente de Git para comunicarse con los repositorios de CodeCommit. Como parte de esta configuración, debe proporcionar credenciales de IAM que CodeCommit puede utilizar para autenticarle. IAM admite CodeCommit con tres tipos de credenciales:

- Credenciales de Git, un nombre de usuario y contraseña generados por IAM que puede utilizar para comunicarse con los repositorios de CodeCommit a través de HTTPS.
- Claves SSH, un par de claves pública y privada generadas a nivel local que puede asociar a su usuario de IAM para comunicarse con los repositorios de CodeCommit a través de SSH.
- Claves de acceso de AWS, que puede utilizar con el auxiliar de credenciales incluido con AWS CLI para comunicarse con los repositorios de CodeCommit a través de HTTPS.

> **Note**
>
> No puede utilizar las claves SSH ni las credenciales de Git para obtener acceso a los repositorios de otra cuenta de AWS. Para obtener más información sobre cómo configurar el acceso a los repositorios de CodeCommit para los usuarios y grupos de IAM en otra Cuenta de AWS, consulte Configurar el acceso entre cuentas a un repositorio de AWS CodeCommit mediante roles en la Guía del usuario de AWS CodeCommit.

Consulte las siguientes secciones para obtener más información sobre cada opción.

Uso de las credenciales de Git y HTTPS con CodeCommit (recomendado)

Con las credenciales de Git, puede generar un nombre de usuario y contraseña estáticos para su usuario de IAM y, a continuación, utilizar dichas credenciales para conexiones HTTPS. También puede utilizar estas credenciales con cualquier herramienta de terceros o entorno de desarrollo integrado (IDE) que admita credenciales de Git estáticas.

Dado que estas credenciales son universales para todos los sistemas operativos admitidos y compatibles con la mayoría de los sistemas de administración de credenciales, entornos de desarrollo y otras herramientas de desarrollo de software, este es el método recomendado. Puede restablecer la contraseña para las credenciales de Git en cualquier momento. También puede desactivarlas o eliminarlas si ya no las necesita.

> **Note**
>
> No puede elegir su propio nombre de usuario o contraseña para las credenciales de Git. IAM genera estas credenciales para que pueda asegurarse de que cumplen los estándares de seguridad de AWS y protegen los repositorios de CodeCommit. Puede descargar las credenciales solo una vez, en el momento en que se generan. Asegúrese de guardarlas en un lugar seguro. Si es necesario, puede restablecer la contraseña en cualquier momento, pero si lo hace, anulará las conexiones configuradas con la antigua contraseña. Debe volver a configurar las conexiones para utilizar la nueva contraseña antes de poder conectarse.

Consulte los siguientes temas para obtener más información:

- Para crear un usuario de IAM, consulte Creación de un usuario de IAM en su Cuenta de AWS.

- Para generar y utilizar las credenciales de Git, con CodeCommit, consulte Para usuarios HTTPS mediante credenciales de Git en la Guía del usuario de AWS CodeCommit.

> **Note**
>
> Cambiar el nombre de un usuario de IAM después de generar las credenciales de Git no cambia el nombre de usuario de las credenciales de Git. El nombre de usuario y la contraseña siguen siendo los mismos y siguen siendo válidos.

Para actualizar las credenciales específicas del servicio

1. Cree un segundo conjunto de credenciales específicas de servicios, además del conjunto que se utiliza actualmente.
2. Actualice todas sus aplicaciones para utilizar el nuevo conjunto de credenciales y compruebe que las aplicaciones están funcionando.
3. Cambie el estado de las credenciales originales a "Inactive".
4. Asegúrese de que todas las aplicaciones siguen funcionando.
5. Elimine las credenciales específicas de servicios que no estén activas.

Uso de las claves SSH y SSH con CodeCommit

Con conexiones SSH, debe crear archivos de claves públicas y privadas en su equipo local que Git y CodeCommit utilizarán para la autenticación SSH. Puede asociar la clave pública con el usuario de IAM y almacenar la clave privada en el equipo local. Consulte los siguientes temas para obtener más información:

- Para crear un usuario de IAM, consulte Creación de un usuario de IAM en su Cuenta de AWS.
- Para crear una clave pública SSH y asociarla a un usuario IAM, consulte Para conexiones SSH en Linux, macOS o Unix o consulte Para conexiones SSH en Windows en la Guía del usuario de AWS CodeCommit.

> **Note**
>
> La clave pública debe estar codificada en formato ssh-rsa o PEM. La longitud mínima de bits de la clave pública es de 2 048 bits, y la longitud máxima es de 16 384 bits. Este valor es

independiente del tamaño del archivo que se cargue. Por ejemplo, puede generar una clave de 2 048 bits y el archivo PEM resultante tiene una longitud de 1 679 bytes. Si proporciona la clave pública en otro formato o tamaño, aparecerá un mensaje de error que indica que el formato de clave no es válido.

Uso de HTTPS con el auxiliar de credenciales de AWS CLI y CodeCommit

Además de las conexiones HTTPS con las credenciales de Git, puede permitir a Git utilizar una versión con firma criptográfica de sus credenciales de usuario de IAM o del rol de la instancia de Amazon EC2 cada vez que Git requiera autenticación en AWS para interactuar con los repositorios de CodeCommit. Este es el único método de conexión para repositorios de CodeCommit que no exige un usuario de IAM. Este también es el único método que funciona con acceso federado y credenciales temporales. Consulte los siguientes temas para obtener más información:

- Para obtener más información sobre el acceso federado, consulte Federación y proveedores de identidades y Proporcionar acceso a usuarios autenticados externamente (identidad federada).
- Para obtener más información sobre las credenciales temporales, consulte Credenciales de seguridad temporales en IAM y Acceso temporal a los repositorios de CodeCommit.

El auxiliar de credenciales de AWS CLI no es compatible con otros sistemas auxiliares de credenciales, como Keychain Access o Windows Credential Management. Existen otras consideraciones de configuración cuando establece conexiones HTTPS con el auxiliar de credenciales. Para más información, consulte Para conexiones HTTPS en Linux, macOS o Unix con el Ayudante de credenciales de AWS CLI o Conexiones HTTPS en Windows con el Ayudante de credenciales de AWS CLI en la Guía del usuario de AWS CodeCommit.

Utilizar IAM con Amazon Keyspaces (for Apache Cassandra)

El servicio Amazon Keyspaces (for Apache Cassandra) es un servicio de bases de datos administrado, de alta disponibilidad y escalable compatible con Apache Cassandra. Puede acceder a Amazon Keyspaces a través de la AWS Management Console o mediante programación. Para acceder a Amazon Keyspaces mediante programación con credenciales específicas del servicio, puede utilizar `cqlsh` o los controladores Cassandra de código abierto. Las credenciales específicas del servicio incluyen un nombre de usuario y una contraseña como los que utiliza Cassandra para la autenticación y la administración del acceso. Puede tener un máximo de dos conjuntos de credenciales específicas del servicio para cada servicio compatible por usuario.

Para acceder a Amazon Keyspaces mediante programación con claves de acceso de AWS, puede utilizar el SDK de AWS, la AWS Command Line Interface (AWS CLI) o los controladores de código abierto de Cassandra con el plugin SigV4. Para obtener más información, consulte Connecting programmatically to Amazon Keyspaces (Conexión mediante programación a Amazon Keyspaces) en la Amazon Keyspaces (for Apache Cassandra) Developer Guide (Guía para desarrolladores de Amazon Keyspaces [para Apache Cassandra]).

> **Note**
>
> Si tiene previsto interactuar con Amazon Keyspaces únicamente a través de la consola, no es necesario que genere credenciales específicas del servicio. Para obtener más información, consulte Accessing Amazon Keyspaces using the console (Acceso a Amazon Keyspaces mediante la consola) en la Amazon Keyspaces (for Apache Cassandra) Developer Guide (Guía para desarrolladores de Amazon Keyspaces [para Apache Cassandra]).

Para obtener más información sobre los permisos necesarios para acceder a Amazon Keyspaces, consulte ejemplos de políticas basadas en identidad de Amazon Keyspaces (for Apache Cassandra) en la Guía para desarrolladores de Amazon Keyspaces (for Apache Cassandra).

Generar credenciales de Amazon Keyspaces (consola)

Puede utilizar la AWS Management Console para generar credenciales de Amazon Keyspaces (for Apache Cassandra) para los usuarios de IAM.

Para generar credenciales específicas del servicio de Amazon Keyspaces (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Users (Usuarios) y, a continuación, el nombre del usuario que requiere las credenciales.
3. En la pestaña Credenciales de seguridad debajo de Credenciales para el servicio Amazon Keyspaces (for Apache Cassandra), seleccione Generar credenciales.
4. Las credenciales específicas del servicio ya están disponibles. Esta es la única vez que se puede ver o descargar la contraseña. No puede recuperarla más adelante. Sin embargo, puede restablecerla en cualquier momento. Guarde el usuario y la contraseña en una ubicación segura, ya que los necesitará más adelante.

Generación de credenciales de Amazon Keyspaces (AWS CLI)

Puede utilizar la AWS CLI para generar credenciales de Amazon Keyspaces (for Apache Cassandra) para los usuarios de IAM.

Para generar credenciales específicas del servicio de Amazon Keyspaces (AWS CLI)

- Para ello, utilice el siguiente comando:
  - aws iam create-service-specific-credential

Generación de credenciales de Amazon Keyspaces (API de AWS)

Puede utilizar la API de AWS para generar credenciales de Amazon Keyspaces (for Apache Cassandra) para los usuarios de IAM.

Para generar credenciales específicas del servicio de Amazon Keyspaces (API de AWS)

- Complete la siguiente operación:
  - CreateServiceSpecificCredential

Administración de certificados de servidor en IAM

Para habilitar las conexiones HTTPS en su sitio web o aplicación en AWS, necesita un certificado de servidor SSL/TLS. Si se trata de una región compatible con AWS Certificate Manager (ACM), le recomendamos que utilice ACM para aprovisionar, administrar e implementar los certificados de servidor. En las regiones no compatibles, debe utilizar IAM como Certificate Manager. Para saber qué regiones admite , consulte puntos finales y cuotas de AWS Certificate Manager en la Referencia general de AWS.

ACM es la herramienta preferida para aprovisionar, administrar e implementar los certificados de servidor. Con ACM puede solicitar un certificado o implementar un certificado de ACM existente o un certificado externo en los recursos de AWS. Los certificados que proporciona ACM son gratuitos y se renuevan automáticamente. En las regiones compatibles, puede utilizar ACM para administrar los certificados de servidor desde la consola o mediante programación. Para obtener más información sobre ACM, consulte la Guía del usuario de AWS Certificate Manager. Para obtener más información sobre cómo solicitar un certificado de ACM, consulte Solicitar un certificado público o Solicitar un certificado privado en la Guía del usuario de AWS Certificate Manager. Para obtener más

información sobre la importación de certificados de terceros en ACM, consulte Importar certificados en la Guía del usuario de AWS Certificate Manager.

Utilice IAM como Certificate Manager solo cuando tenga que admitir conexiones HTTPS en una región que no es compatible con ACM. IAM cifra de forma segura sus claves privadas y almacena la versión cifrada en el almacenamiento de certificados SSL de IAM. IAM admite la implementación de certificados de servidor en todas las regiones, pero debe obtener su certificado de un proveedor externo para utilizarlo con AWS. No puede cargar un certificado de ACM en IAM. Además, no puede administrar los certificados desde la consola de IAM.

Para obtener más información acerca de cómo cargar certificados de terceros en IAM, consulte los siguientes temas.

Contenido

Carga de un certificado de servidor (API de AWS)

Para cargar un certificado de servidor en IAM, debe proporcionar el certificado y la clave privada correspondiente. Si el certificado no está autofirmado, también debe proporcionar una cadena de certificados. (No necesita una cadena de certificados para cargar un certificado autofirmado). Antes de cargar un certificado, asegúrese de que dispone de todos estos elementos y de que cumplen los siguientes criterios:

- El certificado debe ser válido en el momento de la carga. No puede cargar un certificado antes de que empiece su periodo de validez (la fecha `NotBefore` del certificado) o después de que expire (la fecha `NotAfter` del certificado).
- La clave privada no debe estar cifrada. No puede cargar una clave privada que esté protegida por una contraseña o frase de contraseña. Para ayudar a descifrar una clave privada cifrada, consulte Solución de problemas.

- El certificado, la clave privada y la cadena de certificados deben tener todos codificación PEM. Para ayudar a convertir estos elementos al formato PEM, consulte Solución de problemas.

Para utilizar la API de IAM para cargar un certificado, envíe una solicitud UploadServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS Command Line Interface (AWS CLI). El ejemplo supone lo siguiente:

- El certificado codificado en PEM se guarda en un archivo llamado `Certificate.pem`.
- La cadena de certificados codificados en PEM se guarda en un archivo llamado `CertificateChain.pem`.
- La clave privada codificada en PEM sin cifrar se guarda en un archivo llamado `PrivateKey.pem`.
- (Opcional) Debe etiquetar el certificado del servidor con un par de valor de clave. Por ejemplo, puede agregar la clave de etiqueta `Department` y el valor de la etiqueta `Engineering` para ayudarlo a identificar y organizar los certificados.

Para utilizar los siguientes comandos de ejemplo, sustituya estos nombres de archivo por los suyos. Reemplace *`ExampleCertificate`* por un nombre para el certificado cargado. Si desea etiquetar el certificado, reemplace el par clave-valor de la etiqueta *`ExampleKey`* y *`ExampleValue`* con sus propios valores. Escriba el comando en una línea continua. El siguiente ejemplo incluye saltos de línea y espacios adicionales para facilitar su lectura.

```
aws iam upload-server-certificate --server-certificate-name ExampleCertificate
                                  --certificate-body file://Certificate.pem
                                  --certificate-chain file://CertificateChain.pem
                                  --private-key file://PrivateKey.pem
                                  --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Si el comando anterior se ejecuta correctamente, devolverá los metadatos del certificado cargado, incluido su Amazon Resource Name (ARN), su nombre descriptivo, su identificador (ID), su fecha de vencimiento, etiquetas y otra información.

> **Note**
>
> Si va a cargar un certificado de servidor para utilizarlo con Amazon CloudFront, debe especificar una ruta mediante la opción `--path`. La ruta debe empezar con `/cloudfront` y debe incluir una barra inclinada al final (por ejemplo, `/cloudfront/test/`).

Para utilizar las AWS Tools for Windows PowerShell para cargar un certificado, utilice Publish-IAMServerCertificate.

Recuperación de un certificado de servidor (API de AWS)

Para utilizar la API de IAM para recuperar un certificado, envíe una solicitud GetServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI. Sustituya *`ExampleCertificate`* por el nombre del certificado a recuperar.

```
aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

Si el comando anterior se ejecuta correctamente, devolverá el certificado, la cadena de certificados (si se ha cargado alguna) y los metadatos del certificado.

> **Note**
>
> No puede descargar o recuperar una clave privada de IAM después de cargarla.

Para utilizar las AWS Tools for Windows PowerShell para recuperar un certificado, utilice Get-IAMServerCertificate.

Lista de los certificados de servidor (API de AWS)

Para utilizar la API de IAM para realizar una lista de los certificados de servidor cargados, envíe una solicitud ListServerCertificates. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI.

```
aws iam list-server-certificates
```

Si el comando anterior se ejecuta correctamente, devolverá una lista que contiene los metadatos de cada certificado.

Para utilizar las AWS Tools for Windows PowerShell para realizar una lista de los certificados de servidor cargados, utilice Get-IAMServerCertificates.

Etiquetado y desetiquetado de certificados de servidor (API de AWS)

Puede asociar etiquetas a sus recursos de IAM para organizar y controlar el acceso a ellos. Para utilizar la API de IAM para etiquetar un certificado de servidor existente, envíe una solicitud TagServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI.

```
aws iam tag-server-certificate --server-certificate-name ExampleCertificate
                               --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Si el comando anterior se ejecuta correctamente, no devolverá ningún resultado.

Para utilizar la API de IAM para quitar la etiqueta de un certificado de servidor, envíe una solicitud UntagServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI.

```
aws iam untag-server-certificate --server-certificate-name ExampleCertificate
                               --tag-keys ExampleKeyName
```

Si el comando anterior se ejecuta correctamente, no devolverá ningún resultado.

Cambio de nombre de un certificado de servidor o actualización de su ruta (API de AWS)

Para utilizar la API de IAM para cambiar el nombre de un certificado de servidor o para actualizar su ruta, envíe una solicitud UpdateServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI.

Para utilizar el siguiente comando de ejemplo, sustituya los nombres antiguo y nuevo del certificado y la ruta del certificado y escriba el comando en una línea continua. El siguiente ejemplo incluye saltos de línea y espacios adicionales para facilitar su lectura.

```
aws iam update-server-certificate --server-certificate-name ExampleCertificate
                                  --new-server-certificate-name CloudFrontCertificate
                                  --new-path /cloudfront/
```

Si este comando anterior se ejecuta correctamente, no devolverá ningún resultado.

Para utilizar las AWS Tools for Windows PowerShell para cambiar el nombre de un certificado de servidor o para actualizar su ruta, utilice Update-IAMServerCertificate.

Eliminación de un certificado de servidor (API de AWS)

Para utilizar la API de IAM para eliminar un certificado de servidor, envíe una solicitud DeleteServerCertificate. El siguiente ejemplo muestra cómo hacerlo con la AWS CLI.

Para utilizar el siguiente comando de ejemplo, sustituya *ExampleCertificate* por el nombre del certificado a eliminar.

```
aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

Si este comando anterior se ejecuta correctamente, no devolverá ningún resultado.

Para utilizar las AWS Tools for Windows PowerShell para eliminar un certificado de servidor, utilice Remove-IAMServerCertificate.

Solución de problemas

Antes de cargar un certificado en IAM, debe asegurarse de que el certificado, la clave privada y la cadena de certificados estén codificados en PEM. También debe asegurarse de que la clave privada no esté cifrada. Vea los siguientes ejemplos.

Example Ejemplo de certificado codificado por PEM

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Example Ejemplo de clave privada sin cifrar y codificada en PEM

```
-----BEGIN RSA PRIVATE KEY-----
Base64-encoded private key
-----END RSA PRIVATE KEY-----
```

Example Ejemplo de cadena de certificados codificada en PEM

Una cadena de certificados contiene uno o más certificados. Puede utilizar un editor de texto, el comando copy de Windows o el comando cat de Linux para concatenar archivos de certificado en

una cadena. Cuando incluye varios certificados, cada certificado debe certificar el certificado anterior. Puede hacerlo concatenando los certificados, incluyendo el certificado de CA raíz al final.

El siguiente ejemplo contiene tres certificados, pero una cadena de certificados podría contener más o menos certificados.

```
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
Base64-encoded certificate
-----END CERTIFICATE-----
```

Si estos elementos no tienen el formato adecuado para cargarlos en IAM, puede utilizar OpenSSL para convertirlos al formato adecuado.

Para convertir un certificado o cadena de certificados de DER a PEM

Utilice el comando OpenSSL x509, como en el siguiente ejemplo. En el siguiente comando de ejemplo, sustituya *Certificate.der* por el nombre del archivo que contiene el certificado con codificación DER. Sustituya *Certificate.pem* por el nombre preferido del archivo de salida que va a contener el certificado con codificación PEM.

```
openssl x509 -inform DER -in Certificate.der -outform PEM -out Certificate.pem
```

Para convertir una clave privada de DER a PEM

Utilice el comando OpenSSL rsa, como en el siguiente ejemplo. En el siguiente comando de ejemplo, sustituya *PrivateKey.der* por el nombre del archivo que contiene la clave privada con codificación DER. Sustituya *PrivateKey.pem* por el nombre preferido del archivo de salida que va a contener la clave privada con codificación PEM.

```
openssl rsa -inform DER -in PrivateKey.der -outform PEM -out PrivateKey.pem
```

Para descifrar una clave privada cifrada (eliminar la contraseña o frase de contraseña)

Utilice el comando OpenSSL rsa, como en el siguiente ejemplo. Para utilizar el siguiente comando de ejemplo, sustituya *EncryptedPrivateKey.pem* por el nombre del archivo que contiene la clave privada cifrada. Sustituya *PrivateKey.pem* por el nombre preferido del archivo de salida que va a contener la clave privada no cifrada con codificación PEM.

```
openssl rsa -in EncryptedPrivateKey.pem -out PrivateKey.pem
```

Para convertir un paquete de certificados de PKCS #12 (PFX) a PEM

Utilice el comando OpenSSL pkcs12, como en el siguiente ejemplo. En el siguiente comando de ejemplo, sustituya *CertificateBundle.p12* por el nombre del archivo que contiene el paquete de certificados con codificación PKCS#12. Sustituya *CertificateBundle.pem* por el nombre preferido del archivo de salida que va a contener el paquete de certificado con codificación PEM.

```
openssl pkcs12 -in CertificateBundle.p12 -out CertificateBundle.pem -nodes
```

Para convertir un paquete de certificados de PKCS#7 a PEM

Utilice el comando OpenSSL pkcs7, como en el siguiente ejemplo. En el siguiente comando de ejemplo, sustituya *CertificateBundle.p7b* por el nombre del archivo que contiene el paquete de certificados con codificación PKCS#7. Sustituya *CertificateBundle.pem* por el nombre preferido del archivo de salida que va a contener el paquete de certificado con codificación PEM.

```
openssl pkcs7 -in CertificateBundle.p7b -print_certs -out CertificateBundle.pem
```

Grupos de usuarios de IAM

Un grupo de usuarios de IAM es un conjunto de usuarios de IAM. Los grupos de usuarios le permiten especificar permisos para varios usuarios, lo que puede facilitar la administración de los permisos para dichos usuarios. Por ejemplo, podría tener un grupo de usuarios denominado Admins (Administradores) y proporcionar a dicho grupo de los tipos de permisos que los administradores suelen necesitar. Cualquier usuario de ese grupo tiene automáticamente permisos del grupo

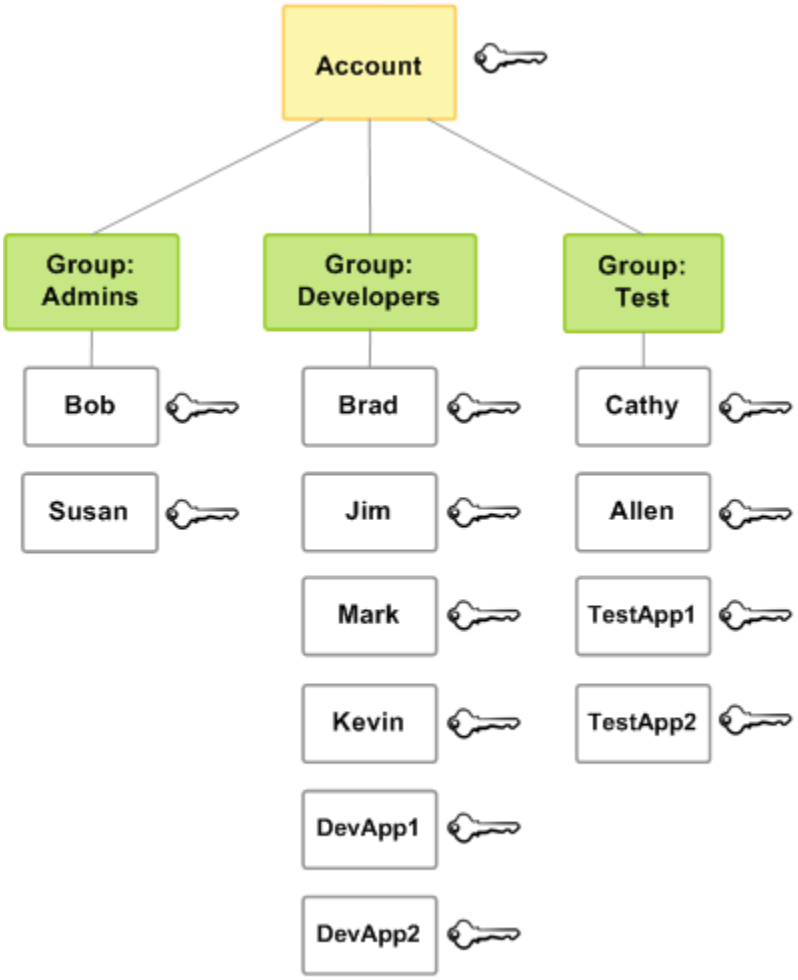
Admins (Administradores). Si un nuevo usuario se une a su organización y necesita privilegios de administrador, puede asignar los permisos adecuados al agregar el usuario al grupo de usuarios Admins (Administradores). Si una persona cambia de trabajo en su organización, en lugar de editar los permisos de ese usuario puede eliminarlo de los antiguos grupos de usuarios y agregarlo a los nuevos grupos de usuarios correspondientes.

Puede asociar una política basada en identidad a un grupo de usuarios para que todos los usuarios del grupo reciban los permisos de la política. No puede identificar un grupo de usuarios como `Principal` en una política (como una política basada en recursos) porque los grupos están relacionados con los permisos, no con la autenticación, y las entidades principales son entidades de IAM autenticadas. Para obtener más información acerca de los tipos de políticas, consulte Políticas basadas en identidad y políticas basadas en recursos.

A continuación, algunas características importantes de los grupos de usuarios:

- Un grupo de usuarios puede incluir muchos usuarios y un usuario puede pertenecer a varios grupos de usuarios.
- Los grupos de usuarios no pueden anidarse; solo pueden incluir usuarios y no otros grupos de usuarios.
- No hay ningún grupo de usuarios predeterminado que incluya automáticamente todos los usuarios de la Cuenta de AWS. Si desea tener un grupo de usuarios de este tipo, debe crearlo y asignarle cada nuevo usuario.
- El número y tamaño de recursos de IAM de un Cuenta de AWS, como el número de grupos y el número de grupos de los que un usuario puede ser miembro, son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

En el siguiente diagrama se muestra un ejemplo sencillo de un pequeño negocio. El propietario del negocio crea un grupo de usuarios `Admins` para que los usuarios creen otros usuarios y los administren a medida que crece el negocio. La `Admins` crea un grupo de usuarios `Developers` y un grupo de usuarios `Test`. Cada uno de estos grupos de usuarios se compone de usuarios (personas y aplicaciones) que interactúan con AWS (Jim, Brad, DevApp1, etc.). Cada usuario tiene un conjunto individual de credenciales de seguridad. En este ejemplo, cada usuario pertenece a un solo grupo de usuarios. Sin embargo, los usuarios pueden pertenecer a varios grupos de usuarios.



Creación de un grupo de usuarios de IAM

> **Note**
>
> Como práctica recomendada, le recomendamos que exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales. Si sigue las prácticas recomendadas, no estará administrando usuarios ni grupos de IAM. En cambio, sus usuarios y grupos se administran fuera de AWS y pueden acceder a los recursos de AWS como una identidad federada. Una identidad federada es un usuario del directorio de usuarios de su empresa, un proveedor de identidades web, AWS Directory Service, el directorio de Identity Center o cualquier usuario que acceda a los servicios de AWS con credenciales proporcionadas a través de una fuente de identidades. Las identidades federadas utilizan los grupos definidos por su proveedor de identidades. Si utiliza AWS IAM Identity Center, consulte Manage identities in IAM Identity Center (Administración

de identidades en IAM Identity Center) en la Guía del usuario de AWS IAM Identity Center para obtener información sobre la creación de usuarios y grupos en IAM Identity Center.

Para configurar un grupo de usuarios, debe crear el grupo. A continuación, debe conceder al grupo permisos en función del tipo de trabajo que espera que realicen los usuarios del grupo. Finalmente, añada usuarios al grupo.

Para obtener información sobre los permisos que necesita para poder crear un grupo de usuarios, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Para crear un grupo de usuarios de IAM y asociar políticas (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios y, a continuación, elija Crear nuevo grupo.
3. En Nombre de grupo de usuarios, escriba el nombre del grupo.

   > **Note**
   >
   > El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS. Los nombres de grupo pueden ser una combinación de un máximo de 128 letras, dígitos y los siguientes caracteres: más (+), igual (=), coma (,), punto (.), arroba (@), guion bajo (_) y guion (-). Los nombres deben ser únicos dentro de una cuenta. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear funciones denominados **`ADMINS`** ni **`admins`**.

4. En la lista de usuarios, seleccione la casilla de verificación de cada usuario que desee agregar al grupo.
5. En la lista de políticas, seleccione la casilla de verificación de cada política que desea aplicar a todos los miembros del grupo.
6. Elija Create group.

Para crear grupos de usuarios de IAM (AWS CLI o API de AWS)

Utilice una de las siguientes:

- AWS CLI: aws iam create-group

- AWS API: CreateGroup

Administración de grupos de usuarios de IAM

Amazon Web Services ofrece varias herramientas para administrar los grupos de usuarios de IAM. Para obtener información sobre los permisos que necesita para poder agregar usuarios a un grupo de usuarios y eliminarlos, consulte Permisos obligatorios para obtener acceso a recursos de IAM.

Temas

- Enumeración de grupos usuarios de IAM
- Agregar y eliminar usuarios de un grupo de usuarios de IAM
- Asociación de una política a un grupo de usuarios de IAM
- Cambio del nombre de un grupo de usuarios de IAM
- Eliminación de un grupo de usuarios de IAM

Enumeración de grupos usuarios de IAM

Puede enumerar todos los grupos de usuarios de su cuenta, enumerar los usuarios de un grupo de usuarios y enumerar los grupos de usuarios a los que pertenece un usuario. Si utiliza la API de AWS CLI o AWS, puede enumerar todos los grupos de usuarios de un determinado prefijo de ruta de acceso.

Para enumerar todos los grupos de usuarios de su cuenta

Realice uno de los siguientes procedimientos:

- AWS Management Console: En el panel de navegación, elija Grupos de usuarios.
- AWS CLI: aws iam list-groups
- AWS API: ListGroups

Para obtener una lista de los usuarios en un grupo específico de usuarios

Realice uno de los siguientes procedimientos:

- AWS Management Console: En el panel de navegación, elija Grupos de usuarios elija el nombre del grupo y, a continuación, elija la pestaña Usuarios.
- AWS CLI: aws iam get-group

- API de AWS: GetGroup

Para listar todos los grupos de usuarios en los que se encuentra un usuario

Realice uno de los siguientes procedimientos:

- AWS Management Console: en el panel de navegación, elija Users (Usuarios), elija el nombre del usuario y, a continuación, elija la pestaña Groups (Grupos).
- AWS CLI: aws iam list-groups-for-user
- API de AWS: ListGroupsForUser

Agregar y eliminar usuarios de un grupo de usuarios de IAM

Utilice los grupos de usuarios para aplicar las mismas políticas de permisos en varios usuarios a la vez. Puede entonces agregar usuarios o eliminar usuarios de un grupo de usuarios de IAM. Esto resulta útil, ya que los empleados van y vienen de su organización.

Ver acceso a políticas

Antes de cambiar los permisos para una política, debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Agregar o eliminar un usuario en un grupo de usuarios (consola)

Puede utilizar la AWS Management Console para agregar o eliminar un usuario de un grupo de usuarios.

Para agregar un usuario a un grupo de usuarios de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios y, a continuación, elija el nombre del grupo.
3. Elija la pestaña Usuarios y, a continuación, elija Agregar usuarios. Seleccione la casilla de verificación junto a los usuarios que desee agregar.
4. Elija Agregar usuarios.

Para eliminar un usuario de un grupo de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios y, a continuación, elija el nombre del grupo.
3. Elija la pestaña Users. Seleccione la casilla de verificación junto a los usuarios que desee eliminar y, a continuación, elija Eliminar usuarios.

Agregar o eliminar un usuario en un grupo de usuarios (AWS CLI)

Puede utilizar la AWS CLI para agregar o eliminar un usuario de un grupo de usuarios.

Para agregar un usuario a un grupo de usuarios de IAM (AWS CLI)

- Para ello, utilice el siguiente comando:
  - aws iam add-user-to-group

Para eliminar un usuario de un grupo de usuarios de IAM (AWS CLI)

- Para ello, utilice el siguiente comando:
  - aws iam remove-user-from-group

Agregar o eliminar un usuario en un grupo de usuarios (API de AWS)

Puede utilizar la API de AWS para agregar o eliminar un usuario en un grupo de usuarios.

Para agregar un usuario a un grupo de IAM (API de AWS)

- Complete la siguiente operación:
  - AddUserToGroup

Para eliminar un usuario de un grupo de usuarios de IAM (API de AWS)

- Complete la siguiente operación:
  - RemoveUserFromGroup

Asociación de una política a un grupo de usuarios de IAM

Puede asociar una política administrada de AWS es decir, una política escrita previamente proporcionada por AWS a un grupo de usuarios, tal como se explica en los pasos siguientes. Para asociar una política administrada por el cliente es decir, una política con los permisos personalizados que ha creado primero debe crear la política. Para obtener más información sobre la creación de políticas administradas por el cliente, consulte Crear políticas de IAM.

Para obtener más información sobre permisos y políticas, consulte Recursos de AWS para administración de acceso.

Para adjuntar una política a un grupo de usuarios (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios y, a continuación, elija el nombre del grupo.
3. Elija la pestaña Permisos.
4. Elija Agregar permisos y luego Adjuntar políticas.
5. Las políticas actuales adjuntas al grupo de usuarios se muestran en la lista de Políticas de permisos actuales. En la lista de Otras políticas de permisos, seleccione la casilla de verificación junto al nombre de las políticas que desea adjuntar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas por tipo y nombre de política.
6. Seleccione la política que desee adjuntar a su grupo de usuarios de IAM y elija Adjuntar políticas.

Para adjuntar una política a un grupo de usuarios (AWS CLI o API de AWS)

Haga una de estas dos operaciones:

- AWS CLI: aws iam attach-group-policy
- AWS API: AttachGroupPolicy

Cambio del nombre de un grupo de usuarios de IAM

Cuando cambia la ruta o el nombre de un grupo de usuarios, ocurre lo siguiente:

- Todas las políticas asociadas al grupo de usuarios permanecen en el grupo con el nuevo nombre.

- El grupo de usuarios conserva todos sus usuarios bajo el nuevo nombre.
- El ID único del grupo de usuarios sigue siendo el mismo. Para obtener más información sobre los ID únicos, consulte Identificadores únicos.

IAM no actualiza automáticamente las políticas que hacen referencia al grupo como recurso para utilizar el nuevo nombre. Por lo tanto, debe tener cuidado al cambiar el nombre de un grupo de usuarios. Antes de hacerlo, deberá comprobar manualmente todas sus políticas para encontrar aquellas en las que se menciona el nombre del grupo de usuarios. Por ejemplo, supongamos que Bob es el administrador de la parte de pruebas de la organización. Bob tiene una política asociada a su entidad de usuario de IAM que le permite agregar y quitar usuarios del grupo de usuarios Test. Si un administrador cambia el nombre del grupo de usuarios (o cambia la ruta del grupo), también debe actualizar la política asociada para que Bob utilice el nuevo nombre o ruta. De lo contrario, Bob no podrá agregar ni quitar usuarios del grupo de usuarios.

Para las buscar políticas que hacen referencia a un grupo de usuarios como recurso:

1. En el panel de navegación de la consola de IAM, elija Políticas.
2. Ordena por la columna Tipo para buscar sus políticas personalizadas Gestión por el cliente.
3. Elija el nombre de la política para editarlo.
4. Seleccione la pestaña Permisos y, a continuación, Resumen.
5. Elija IAM en la lista de servicios, en caso de que exista.
6. Busque el nombre del grupo de usuarios en la columna Recurso.
7. Seleccione Editar para cambiar el nombre del grupo de usuarios en la política.

Para cambiar el nombre de un grupo de usuarios de IAM

Realice uno de los siguientes procedimientos:

- AWS Management Console: En el panel de navegación, elija Grupos de usuarios y, a continuación, elija el nombre del grupo. Elija Edit (Editar). Escriba el nuevo nombre del grupo de usuarios y, a continuación, elija Guardar los cambios.
- AWS CLI: aws iam update-group
- AWS API: UpdateGroup

Eliminación de un grupo de usuarios de IAM

Si elimina un grupo de usuarios en la AWS Management Console, la consola eliminará automáticamente todos los miembros del grupo, desasociará todas las políticas administradas y eliminará todas las políticas insertadas. Sin embargo, dado que IAM no elimina automáticamente las políticas que hacen referencia al grupo de usuarios como recurso; debe tener cuidado al eliminar un grupo de usuarios. Antes de hacerlo, deberá verificar manualmente todas sus políticas para encontrar aquellas en las que se menciona el grupo por el nombre. Por ejemplo, John, el director del equipo de pruebas, tiene una política asociada a su entidad de usuario de IAM que le permite agregar y eliminar usuarios del grupo de usuarios de pruebas. Si un administrador elimina el grupo, también deberá eliminar la política asociada a John. De lo contrario, si el administrador vuelve a crear el grupo eliminado y le da el mismo nombre, los permisos de John seguirán vigentes, aunque haya abandonado el equipo de pruebas.

Para buscar políticas que hacen referencia a un grupo de usuarios como recurso

1. En el panel de navegación de la consola de IAM, elija Políticas.
2. Ordena por la columna Tipo para buscar sus políticas personalizadas Gestión por el cliente.
3. Elija el nombre de política de la política que desea eliminar.
4. Seleccione la pestaña Permisos y, a continuación, Resumen.
5. Elija IAM en la lista de servicios, en caso de que exista.
6. Busque el nombre del grupo de usuarios en la columna Recurso.
7. Seleccione Eliminar para eliminar la política.
8. Escriba el nombre de la política para confirmar su eliminación y seleccione Eliminar.

En cambio, si utiliza la AWS CLI, Tools for Windows PowerShell o la API de AWS para eliminar un grupo, primero debe eliminar los usuarios del grupo de usuarios. A continuación, elimine cualquier política insertada en el grupo de usuarios. A continuación, desasocie cualquier política administrada que esté asociada al grupo. Solo entonces podrá eliminar el propio grupo de usuarios.

Eliminación de un grupo de usuarios de IAM (consola)

Puede eliminar un grupo de usuarios de IAM de la AWS Management Console.

Para eliminar un grupo de usuarios de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija User groups (Grupos de usuarios).
3. En la lista de grupos de usuarios, seleccione la casilla de verificación junto a los nombres del grupo de usuarios que desea eliminar. Puede utilizar el cuadro de búsqueda para filtrar la lista de grupos de usuarios por tipo, permisos y nombre de grupo de usuarios.
4. Elija Eliminar (Delete).
5. En el cuadro de confirmación, si desea eliminar un solo grupo de usuarios, escriba el nombre del grupo de usuarios y elija Eliminar. Si desea eliminar varios grupos de usuarios, escriba el número de grupos de usuarios que desea eliminar seguido de **`user groups`** y elija Eliminar. Por ejemplo, si elimina tres grupos de usuarios, escriba **`3 user groups`**.

Eliminar un grupo de usuarios de IAM (AWS CLI)

Puede eliminar un grupo de usuarios de IAM de la AWS CLI.

Para eliminar un grupo de usuarios de IAM (AWS CLI)

1. Elimine todos los usuarios del grupo de usuarios.
   - aws iam get-group (para obtener la lista de usuarios del grupo de usuarios) y aws iam remove-user-from-group (para eliminar un usuario del grupo de usuarios)
2. Elimine todas las políticas insertadas en el grupo de usuarios.
   - aws iam list-group-policies (para obtener una lista de las políticas insertadas del grupo de usuarios) y aws iam delete-group-policy (para eliminar las políticas insertadas del grupo de usuarios)
3. Desasocie todas las políticas administradas asociadas al grupo de usuarios.
   - aws iam list-attached-group-policies (para obtener una lista de las políticas administradas asociadas al grupo de usuarios) y aws iam detach-group-policy (para desasociar una política administrada del grupo de usuarios)
4. Elimine el grupo de usuarios.
   - aws iam delete-group

Eliminación de un grupo de usuarios de IAM (API de AWS)

Puede utilizar la API de AWS para eliminar un grupo de usuarios de IAM.

Para eliminar un grupo de usuarios de IAM (API de AWS)

1. Elimine todos los usuarios del grupo de usuarios.
   - GetGroup (para obtener la lista de usuarios del grupo de usuarios) y RemoveUserFromGroup (para eliminar un usuario del grupo de usuarios)
2. Elimine todas las políticas insertadas en el grupo de usuarios.
   - ListGroupPolicies (para obtener una lista de las políticas insertadas del grupo de usuarios) y DeleteGroupPolicy (para eliminar las políticas insertadas del grupo de usuarios)
3. Desasocie todas las políticas administradas asociadas al grupo de usuarios.
   - ListAttachedGroupPolicies (para obtener una lista de las políticas administradas asociadas al grupo de usuarios) y DetachGroupPolicy (para desasociar una política administrada del grupo de usuarios)
4. Elimine el grupo de usuarios.
   - DeleteGroup

Roles de IAM

Un rol de IAM es una identidad de IAM que puede crear en su cuenta y que tiene permisos específicos. Un rol de IAM es similar a un usuario de IAM en que se trata de una identidad de AWS con políticas de permisos que determinan lo que la identidad puede hacer y lo que no en AWS. No obstante, en lugar de asociarse exclusivamente a una persona, la intención es que cualquier usuario pueda asumir un rol que necesite. Además, un rol no tiene asociadas credenciales a largo plazo estándar, como una contraseña o claves de acceso. En su lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol.

Puede utilizar roles para delegar el acceso a usuarios, aplicaciones o servicios que normalmente no tendrían acceso a los recursos de AWS. Por ejemplo, es posible que desee conceder a los usuarios de la cuenta de AWS el acceso a los recursos que no suelen tener, o conceder a los usuarios de una Cuenta de AWS el acceso a los recursos de otra cuenta. O es posible que quiera permitir que una aplicación móvil utilice los recursos de AWS, pero no desea integrar las claves de AWS

dentro la aplicación (donde puede ser difícil actualizarlas y donde es posible que los usuarios las extraigan). En ocasiones, es posible que quiera conceder acceso a AWS a los usuarios que ya tienen identidades definidas fuera de AWS, como en su directorio corporativo. O bien, es posible que quiera conceder acceso a su cuenta a terceros para que puedan realizar una auditoría en los recursos.

En estas situaciones, puede delegar el acceso a los recursos de AWS con un rol de IAM. En esta sección se presentan los roles y las distintas formas de utilizarlos, cuándo y cómo elegir entre enfoques y cómo crear, administrar, cambiar (o asumir) y eliminar roles.

> **Note**
>
> Cuando crea su Cuenta de AWS por primera vez, no se crea ningún rol de forma predeterminada. A medida que agregue servicios a su cuenta, es posible que agreguen roles vinculados a servicios para respaldar sus casos de uso.
> Un rol vinculado al servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.
> Antes de eliminar los roles vinculados a servicios, debe borrar sus recursos relacionados. De esta forma, se protegen los recursos de , ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.
> Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado a servicios en cuestión.

Temas

Términos y conceptos de roles

A continuación se muestran algunos términos básicos para ayudarle a comenzar con el uso de los roles.

Rol

Una identidad de IAM que se puede crear en una cuenta y que tiene permisos específicos. Un rol de IAM tiene algunas similitudes con un usuario de IAM. Los roles y los usuarios son identidades de AWS con políticas de permisos que determinan lo que la identidad puede y no puede hacer en AWS. No obstante, en lugar de asociarse exclusivamente a una persona, la intención es que cualquier usuario pueda asumir un rol que necesite. Además, un rol no tiene asociadas credenciales a largo plazo estándar, como una contraseña o claves de acceso. En su lugar, cuando se asume un rol, este proporciona credenciales de seguridad temporales para la sesión de rol.

Las siguientes opciones pueden utilizar los roles:

- Un usuario de IAM de la misma Cuenta de AWS que el rol
- Un usuario de IAM de una Cuenta de AWS distinta de la del rol
- Un servicio web que ofrece AWS como Amazon Elastic Compute Cloud (Amazon EC2)
- Un usuario externo autenticado por un servicio de proveedor de identidad (IdP) externo que sea compatible con SAML 2.0 u OpenID Connect, o un agente de identidades personalizado.

Rol de servicio de AWS

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

Rol de servicio de AWS para una instancia de EC2

Un tipo especial de función de servicio que una aplicación que se ejecuta en una instancia de Amazon EC2 puede asumir para realizar acciones en una cuenta. Este rol se asigna a la instancia EC2 cuando se lanzó. Las aplicaciones que se ejecutan en dicha instancia pueden recuperar las credenciales de seguridad temporales y llevar a cabo las acciones que permite el rol. Para obtener más información sobre el uso de un rol de servicio para una instancia EC2, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2.

Rol vinculado a servicio de AWS

Un rol vinculado al servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

> **Note**
>
> Si ya está utilizando un servicio cuando comienza a admitir roles vinculados a servicios, es posible que reciba un mensaje de correo electrónico anunciándole la adición de un nuevo rol en su cuenta. En este caso, el servicio crea automáticamente el rol vinculado a sí mismo en su cuenta. No es necesario realizar ninguna acción para admitir este rol y no debe eliminarlo manualmente. Para obtener más información, consulte Un nuevo rol ha aparecido en la cuenta de AWS.

Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado a servicios en cuestión. Para obtener más información, consulte Uso de roles vinculados a servicios.

Encadenamiento de roles

El encadenamiento de roles se produce cuando se utiliza un rol para asumir un segundo rol a través de la AWS CLI o la API. Por ejemplo, `RoleA` tiene permiso para asumir `RoleB`. Puede permitir a User1 que asuma el `RoleA` al utilizar las credenciales de usuario a largo plazo en la operación de la API AssumeRole. Esto devuelve las credenciales a corto plazo de `RoleA`. Para utilizar el encadenamiento de roles, puede emplear las credenciales a corto plazo de `RoleA` para permitir a User1 que asuma el `RoleB`.

Cuando asume un rol, puede pasar una etiqueta de sesión y establecer la etiqueta como transitiva. Las etiquetas de sesión transitivas se pasan a todas las sesiones posteriores de una cadena de roles. Para obtener más información sobre las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

El encadenamiento de roles limita la duración de la sesión de rol de la API de AWS CLI o AWS a un máximo de una hora. Cuando utilice la operación API AssumeRole para asumir un rol,

puede especificar la duración de la sesión de su rol con el parámetro `DurationSeconds`. Puede especificar un valor de parámetro de hasta 43200 segundos (12 horas), en función del valor de la duración máxima de la sesión del rol. Sin embargo, si se asume un rol mediante el encadenamiento de roles y se proporciona un valor para el parámetro `DurationSeconds` superior a una hora, la operación produce un error.

AWS no trata el uso de roles para conceder permisos a las aplicaciones que se ejecutan en instancias EC2 como encadenamiento de roles.

Delegación

Es la concesión de permisos a alguien para que obtenga acceso a los recursos que estén bajo su control. La delegación implica establecer una relación de confianza entre dos cuentas. La primera es la cuenta que posee el recurso (la cuenta que confía). El segundo es la cuenta que contiene los usuarios que necesitan acceder al recurso (la cuenta de confianza). Las cuentas de confianza y que confía pueden ser cualquiera de las siguientes:

- La misma cuenta.
- Dos cuentas distintas que están bajo el control de la organización.
- Dos cuentas que son propiedad de diferentes organizaciones.

Para delegar el permiso para obtener acceso a un recurso, cree un rol de IAM en la cuenta que confía. Este rol debe tener dos políticas asociadas. La política de permisos concede al usuario del rol los permisos necesarios para realizar las tareas previstas en el recurso. La política de confianza especifica los miembros de la cuenta de confianza que pueden asumir el rol.

Al crear una política de confianza, no puede especificar un comodín (*) como parte de un ARN en la entidad principal. La política de confianza se asocia al rol de la cuenta que confía, y supone la mitad de los permisos. La otra mitad es una política de permisos asociada al usuario de la cuenta de confianza que permite a dicho usuario cambiar al rol o asumirlo. Un usuario que asume un rol renuncia temporalmente a sus permisos y, en su lugar, asume los permisos del rol. Si el usuario se desconecta o deja de utilizar el rol, los permisos originales del usuario se restablecerán. Un parámetro adicional denominado ID externo garantiza el uso seguro de roles entre cuentas no controladas por la misma organización.

Federación

La creación de una relación de confianza entre un proveedor de identidad externo y AWS. Los usuarios pueden iniciar sesión en un proveedor OIDC, tales como Inicio de sesión con Amazon, Facebook, Google o cualquier proveedor de identidad (IdP) que sea compatible

con OpenID Connect (OIDC). Los usuarios también pueden iniciar sesión en un sistema de identidad empresarial que sea compatible con Security Assertion Markup Language (SAML) 2.0, como Microsoft Active Directory Federation Services. Cuando se utiliza OIDC y SAML 2.0 para configurar una relación de confianza entre estos proveedores de identidad externos y AWS, el usuario se asigna a un rol de IAM. El usuario también recibe credenciales temporales que le permiten tener acceso a los recursos de AWS.

Usuario federado

En lugar de crear un usuario de IAM, puede utilizar identidades existentes de AWS Directory Service, del directorio de usuarios de la empresa o de un proveedor OIDC. A estas identidades se les llama usuarios federados. AWS asigna una función a un usuario federado cuando se solicita acceso a través de un proveedor de identidad. Para obtener más información acerca de los usuarios federados, consulte Usuarios federados y roles.

Política de confianza

Un documento de política JSON en el que se definen las entidades principales en las que confía para asumir el rol. Una política de confianza de rol es una política basada en recursos requerida que se adjunta a un rol en IAM. Las entidades principales que puede especificar en la política de confianza incluyen usuarios, roles, cuentas y servicios.

Política de permisos

Un documento de permisos en formato JSON en el que define qué acciones y recursos puede utilizar el rol. El documento se redacta según las reglas del lenguaje de la política de IAM.

Límite de permisos

Una característica avanzada que le permite utilizar políticas para limitar los permisos máximos que una política basada en identidad puede conceder a un rol. No se puede aplicar un límite de permisos a un rol vinculado a un servicio. Para obtener más información, consulte Límites de permisos para las entidades de IAM.

Entidad principal

Una entidad de AWS que puede realizar acciones y obtener acceso a los recursos. Una entidad principal puede ser un Usuario raíz de la cuenta de AWS, un usuario de IAM o un rol. Puede conceder permisos para obtener acceso a un recurso de una de las dos formas siguientes:

- Puede asociar una política de permisos a un usuario (directa o indirectamente a través de un grupo) o a un rol.

- Para dichos servicios que admiten políticas basadas en recursos, puede identificar la entidad principal del elemento `Principal` de una política asociada al recurso.

Si hace referencia a una Cuenta de AWS como entidad principal, por lo general esto significa cualquier entidad principal definida en dicha cuenta.

> **Note**
>
> No puede usar un comodín (*) para hacer coincidir parte de un nombre de una entidad principal o ARN en la política de confianza de un rol. Para obtener más información, consulte Elemento de la política de JSON de AWS: Principal.

Rol para acceso entre cuentas

Un rol que concede acceso a los recursos de una cuenta a una entidad principal de confianza de otra cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, algunos servicios de AWS permiten asociar una política directamente a un recurso (en lugar de utilizar un rol como proxy). Estas se denominan políticas basadas en recursos y puede utilizarlas para conceder a las entidades principales de otra Cuenta de AWS acceso al recurso. Algunos de estos recursos incluyen buckets de Amazon Simple Storage Service (S3), bóvedas de S3 Glacier, temas de Amazon Simple Notification Service (SNS) y colas de Amazon Simple Queue Service (Amazon SQS). Para saber qué servicios admiten políticas basadas en recursos, consulte Servicios de AWS que funcionan con IAM. Para obtener más información sobre las políticas basadas en recursos, consulte Acceso a recursos entre cuentas en IAM.

Situaciones habituales con los roles: usuarios, aplicaciones y servicios

Al igual que sucede con la mayoría de las características de AWS, hay dos formas de utilizar un rol: de forma interactiva en la consola de IAM o con programas, con la AWS CLI, Tools for Windows PowerShell o la API.

- Los usuarios de IAM de su cuenta que usan la consola de IAM pueden cambiar a un rol para utilizar temporalmente los permisos del rol en la consola. Los usuarios renuncian a sus permisos originales y adoptan los permisos asignados al rol. Cuando el usuario deja de utilizar el rol, sus permisos originales se restablecen.
- Una aplicación o un servicio que AWS ofrece (como Amazon EC2) puede asumir un rol solicitando credenciales de seguridad temporales para un rol con el que realizar solicitudes programadas

a AWS. Un rol se utiliza de este modo para no tener que compartir ni mantener credenciales de seguridad a largo plazo (por ejemplo, mediante la creación de un usuario de IAM) para todas las entidades que requieran acceso a un recurso.

> **Note**
>
> Esta guía utiliza las frases cambiar a un rol y asumir un rol indistintamente.

La forma más sencilla de utilizar roles consiste en conceder a los usuarios de IAM permisos para cambiar a los roles que usted crea dentro de su propia Cuenta de AWS o en otra. De esta forma, pueden cambiar de roles con facilidad utilizando la consola de IAM para utilizar permisos que usted no quiere que tengan normalmente y salir de los roles para renunciar a los permisos. Esto es útil para evitar el acceso accidental a recursos confidenciales o su modificación.

Para informarse de los usos de roles más complejos, como la concesión de acceso a aplicaciones y servicios, o a usuarios externos federados, puede llamar a la API `AssumeRole`. Esta llamada a la API devuelve un conjunto de credenciales temporales que la aplicación puede utilizar en las llamadas a la API posteriores. Las acciones que se intenten efectuar con las credenciales temporales solo tienen los permisos que el rol asociado les concede. Una aplicación no tiene que "salir" del rol de la misma forma que lo hace un usuario en la consola; en su lugar, la aplicación simplemente deja de utilizar las credenciales temporales y vuelve a realizar llamadas con las credenciales originales.

Los usuarios federados inician sesión con las credenciales de un proveedor de identidades (IdP). A continuación, AWS proporciona credenciales temporales al proveedor de identidades de confianza para que las transmita al usuario y que este las incluya en las solicitudes de recursos de AWS posteriores. Estas credenciales proporcionan los permisos concedidos al rol asignado.

En esta sección se proporciona información general sobre las situaciones siguientes:

- Proporcionar a un usuario de IAM de una Cuenta de AWS propia acceso a recursos de otra cuenta propia
- Proporcionar acceso a cargas de trabajo ajenas a AWS
- Proporcionar acceso a usuarios de IAM de Cuentas de AWS que le pertenezcan a terceros
- Proporcionar acceso a servicios ofrecidos por AWS a recursos de AWS
- Proporcionar acceso a usuarios autenticados externamente (identidad federada)

Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS propia

Puede conceder a los usuarios de IAM permiso para cambiar de roles en su Cuenta de AWS o a los roles definidos en otras Cuentas de AWS propias.

> **Note**
>
> Si desea conceder acceso a una cuenta de la que no es propietario o no tiene control, consulte Proporcionar acceso a las Cuentas de AWS que le pertenezcan a terceros más adelante en este tema.

Imagine que dispone de instancias de Amazon EC2 que son de vital importancia para su organización. En lugar de conceder directamente permiso a los usuarios para finalizar las instancias, puede crear un rol con estos privilegios. A continuación, permita que los administradores cambien de rol cuando necesiten terminar una instancia. Al hacer esto, se añaden las siguientes capas de protección a las instancias:

- Debe conceder explícitamente permiso a los usuarios para asumir el rol.
- Los usuarios deben cambiar de rol de forma activa con la AWS Management Console o asumir el rol utilizando la AWS CLI o la API de AWS.
- Puede añadir una protección Multi-Factor Authentication (MFA) al rol para que únicamente los usuarios que inicien sesión con un dispositivo MFA puedan asumir el rol. Para obtener información sobre cómo configurar un rol para que los usuarios que lo asuman deban primero autenticarse mediante la autenticación multifactor (MFA), consulte Configuración del acceso a una API protegido por MFA.

Recomendamos utilizar este enfoque para aplicar el principio de privilegio mínimo. Esto significa limitar el uso de permisos elevados únicamente cuando sean necesarios para realizar tareas específicas. Con los roles puede ayudar a impedir cambios accidentales en entornos confidenciales, especialmente si los combina con un proceso de auditoría con el fin de garantizar que los roles solo se utilizan cuando sean necesarios.

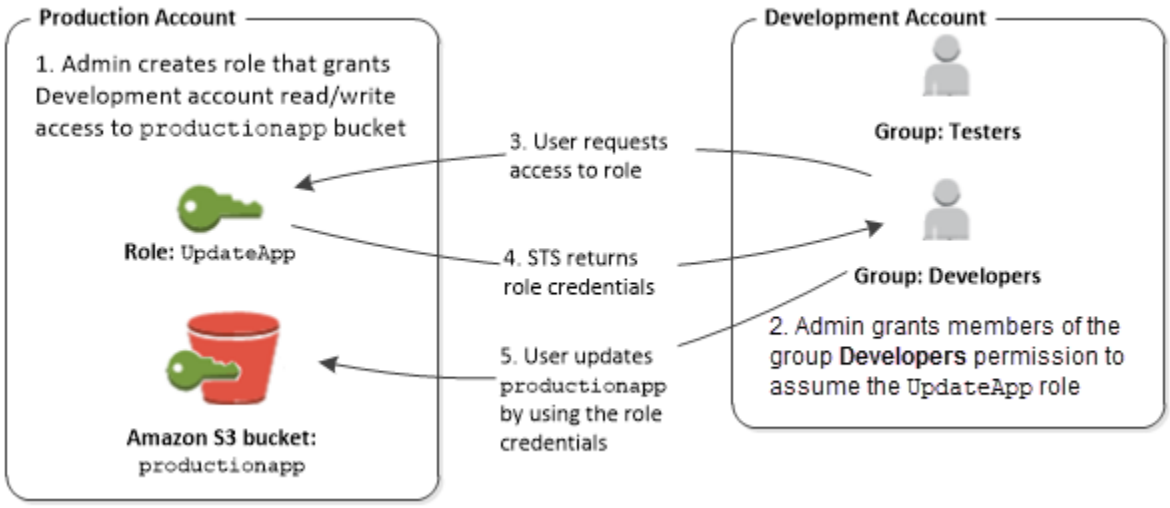
Al crear un rol con este fin, debe especificar las cuentas mediante el ID cuyos usuarios necesitan obtener acceso en el elemento `Principal` de la política de confianza del rol. Puede conceder permisos a usuarios específicos de estas otras cuentas para cambiar de rol. Consulte ¿Qué es

IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Un usuario de una cuenta puede cambiar de rol en la misma cuenta o en una cuenta diferente. Al utilizar el rol, el usuario solo puede realizar las acciones y obtener acceso únicamente a los recursos permitidos por el rol; sus permisos originales de usuario se suspenden. Si el usuario deja de utilizar el rol, sus permisos originales se restablecen.

Situación de ejemplo en la que se usan cuentas de desarrollo y producción separadas

Imagine que la organización tiene varias Cuentas de AWS para aislar su entorno de desarrollo del de producción. Los usuarios de la cuenta de desarrollo podrían necesitar ocasionalmente acceder a los recursos en la cuenta de producción. Por ejemplo, es posible que necesite el acceso entre cuentas al promocionar una actualización desde el entorno de desarrollo al entorno de producción. Aunque podría crear identidades distintas (y contraseñas) para los usuarios que trabajan en las dos cuentas, la administración de credenciales de varias cuentas dificulta la administración de las identidades. En la siguiente figura, todos los usuarios se administran en la cuenta de desarrollo, pero algunos desarrolladores exigen acceso limitado a la cuenta de producción. La cuenta de desarrollo tiene dos grupos: Evaluadores y Desarrolladores, y cada grupo tiene su propia política.



1. En la cuenta de producción un administrador utiliza IAM para crear el rol `UpdateApp` en dicha cuenta. En el rol, el administrador define una política de confianza que especifica la cuenta de desarrollo como `Principal`, lo que significa que los usuarios autorizados de la cuenta de desarrollo pueden utilizar el rol `UpdateApp`. El administrador también define una política de permisos para el rol que especifica los permisos de lectura y escritura del bucket de Amazon S3 denominado `productionapp`.

El administrador comparte entonces la información pertinente con cualquier usuario que necesite asumir el rol. Dicha información es el número de cuenta y el nombre del rol (para usuarios de la consola de AWS) o el Nombre de recurso de Amazon (ARN) (para acceso de API de AWS o AWS CLI). El ARN del rol podría parecerse a `arn:aws:iam::123456789012:role/UpdateApp`, donde el rol se denomina `UpdateApp` y el rol se creó en el número de cuenta 123456789012.

> **Note**
>
> El administrador puede configurar de forma opcional el rol para que los usuarios que lo asuman deban primero autenticarse mediante la opción Multi-Factor Authentication (MFA). Para obtener más información, consulte Configuración del acceso a una API protegido por MFA.

2. En la cuenta de desarrollo un administrador concede a los miembros del grupo Desarrolladores permiso para cambiar de rol. Esto se realiza mediante la concesión de permiso al grupo Desarrolladores para llamar a la API `AssumeRole` de AWS Security Token Service (AWS STS) para el rol `UpdateApp`. Cualquier usuario de IAM que pertenezca al grupo Desarrolladores de la cuenta de desarrollo ahora puede cambiar al rol `UpdateApp` en la cuenta de producción. Otros usuarios que no están en el grupo Desarrolladores no tienen permiso para cambiar de rol y, por lo tanto, no pueden obtener acceso al bucket de S3 en la cuenta de producción.
3. El usuario solicita cambios de rol:
   - Consola de AWS: el usuario selecciona el nombre de la cuenta en la barra de navegación y elige Switch Role (Cambiar rol). El usuario especifica el ID de la cuenta (o alias) y el nombre del rol. O bien, el usuario puede hacer clic en un enlace enviado por correo electrónico enviado por el administrador. Este enlace redirigirá al usuario a la página Switch Role (Cambiar rol) con los detalles ya completados.
   - API de AWS/AWS CLI: un usuario del grupo Desarrolladores de la cuenta de desarrollo llama a la función `AssumeRole` para obtener las credenciales para el rol `UpdateApp`. El usuario especifica el ARN del rol `UpdateApp` como parte de la llamada. Si un usuario del grupo Evaluadores realiza la misma solicitud, esta no podrá llevarse a cabo, ya que los evaluadores no tienen permiso para llamar a `AssumeRole` para solicitar el ARN del rol `UpdateApp`.
4. AWS STS devuelve credenciales temporales:
   - Consola de AWS: AWS STS verifica la solicitud con la política de confianza del rol para garantizar que la solicitud procede de una entidad de confianza (es decir, la cuenta de

desarrollo). Tras realizar la verificación, AWS STS devuelve credenciales de seguridad temporales a la consola de AWS.

- API/CLI: AWS STS verifica la solicitud con la política de confianza del rol para garantizar que la solicitud procede de una entidad de confianza (es decir, la cuenta de desarrollo). Tras realizar la verificación, AWS STS devuelve credenciales de seguridad temporales a la aplicación.

5. Las credenciales temporales permiten el acceso a los recursos de AWS:

- Consola de AWS: la consola de AWS utiliza las credenciales temporales en nombre del usuario para todas las acciones de consola posteriores, en este caso, para leer y escribir en el bucket `productionapp`. La consola no puede obtener acceso a cualquier otro recurso en la cuenta de producción. Si el usuario deja de utilizar el rol, los permisos del usuario vuelven a los permisos originales antes de cambiar de rol.
- API/CLI: la aplicación utiliza las credenciales de seguridad temporales para actualizar el bucket de `productionapp`. Con las credenciales de seguridad temporales, la aplicación solo puede leer y escribir en el bucket de `productionapp` y no puede obtener acceso a cualquier otro recurso en la cuenta de producción. La aplicación no tiene que dejar de utilizar el rol, sino que, en cambio, deja de utilizar las credenciales temporales y utiliza las credenciales originales en las llamadas API posteriores.

Más información

Para obtener más información, consulte los siguientes enlaces:

- Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM

Proporcionar acceso a cargas de trabajo ajenas a AWS

Un rol de IAM es un objeto de AWS Identity and Access Management (IAM) al que se le asignan permisos. Cuando se asume ese rol mediante una identidad de IAM o una identidad externa a AWS, proporciona credenciales de seguridad temporales para la sesión de rol. Es posible que tenga cargas de trabajo ejecutándose en un centro de datos u otra infraestructura situada fuera de AWS que necesite acceder a los recursos de AWS. En lugar de crear, distribuir y administrar claves de acceso de larga duración, puede utilizar AWS Identity and Access Management Roles Anywhere (IAM Roles Anywhere) para autenticar las cargas de trabajo ajenas a AWS. IAM Roles Anywhere utiliza certificados X.509 de la entidad de certificación (CA) para autenticar identidades y proporcionar acceso seguro a Servicios de AWS con las credenciales temporales proporcionadas por un rol de IAM.

Para utilizar IAM Roles Anywhere, debe configurar una CA mediante AWS Private Certificate Authority, o bien emplear una CA de su propia infraestructura de PKI. Después de configurar una CA, debe crear un objeto en IAM Roles Anywhere denominado anclaje de confianza para establecer la confianza entre IAM Roles Anywhere y la CA para la autenticación. A continuación, puede configurar los roles de IAM existentes o crear otros nuevos que confíen en el servicio de IAM Roles Anywhere. Cuando las cargas de trabajo ajenas a AWS se autentiquen con IAM Roles Anywhere mediante el anclaje de confianza, pueden obtener credenciales temporales para que los roles de IAM accedan a los recursos de AWS.

Para obtener más información sobre cómo configurar IAM Roles Anywhere, consulte Qué es AWS Identity and Access Management Roles Anywhere en la Guía del usuario de IAM Roles Anywhere.

Proporcionar acceso a las Cuentas de AWS que le pertenezcan a terceros

Cuando terceros necesitan obtener acceso a recursos de AWS de su organización, puede utilizar roles para delegarles el acceso. Por ejemplo, puede que un tercero proporcione un servicio de administración de sus recursos de AWS. Con los roles de IAM, puede concederle acceso a sus recursos de AWS a terceros sin tener que compartir sus credenciales de seguridad de AWS. En vez de ello, el tercero puede obtener acceso a sus recursos de AWS asumiendo un rol que usted crea en su Cuenta de AWS. Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Los terceros deben proporcionarle la siguiente información para que pueda crear un rol que puedan asumir:

- El ID de Cuenta de AWS del tercero. Especifique su ID de Cuenta de AWS como entidad principal cuando defina la política de confianza del rol.
- Un ID externo para la asociación exclusiva con el rol. El ID externo puede ser cualquier identificador secreto que usted y el tercero conozcan. Por ejemplo, puede utilizar un ID de factura entre usted y el tercero, pero no utilice nada que pueda adivinarse, como el nombre o el número de teléfono del tercero. Debe especificar este ID cuando defina la política de confianza del rol. El tercero debe proporcionar este ID cuando asuma el rol. Para obtener más información acerca del ID externo, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.
- Los permisos que el tercero necesita para poder trabajar con sus recursos de AWS. Debe especificar estos permisos cuando defina la política de permisos del rol. Esta política define qué acciones pueden ejecutar y a qué recursos pueden obtener acceso.

Después de crear el rol, debe proporcionar el nombre de recurso de Amazon (ARN) del rol al tercero. El ARN del rol es necesario para asumir el rol.

> **Important**
>
> Cuando concede acceso a terceros a sus recursos de AWS, estos pueden obtener acceso a cualquier recurso que especifique en la política. El uso que efectúen de sus recursos se le facturará a usted. Asegúrese de que limita el uso de los recursos de forma adecuada.

Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros

A veces, debe otorgar acceso a terceros a sus recursos de AWS (delegar el acceso). Un aspecto importante de esta situación es el ID externo, una información opcional que puede utilizar en una política de confianza del rol de IAM para señalar quién puede asumir el rol.

> **Important**
>
> AWS no trate el ID externo como un secreto. Después de crear un secreto como un par de claves de acceso o una contraseña en AWS, no puede verlos de nuevo. Cualquier usuario con permiso para ver el rol puede ver el ID externo de dicho rol.

En un entorno de varios inquilinos en el que se admiten varios clientes con cuentas de AWS diferentes, recomendamos utilizar un ID externo por cada Cuenta de AWS. Este ID debe ser una cadena aleatoria generada por el tercero.

Para requerir que el tercero proporcione un ID externo al asumir un rol, actualice la política de confianza del rol con el ID externo de su elección.

Para proporcionar un ID externo cuando asuma un rol, utilice la AWS CLI o la API de AWS para asumir ese rol. Para obtener más información, consulte la operación de la API AssumeRole o la operación de la CLI assume-role.

Por ejemplo, digamos que decide contratar a una empresa externa denominada Example Corp para supervisar su Cuenta de AWS y ayudarlo a optimizar los costos. A fin de realizar un seguimiento de su gasto diario, Example Corp necesita acceder a los recursos de AWS. Example Corp también monitoriza muchas otras cuentas de AWS para otros clientes.

No conceda a Example Corp acceso a un usuario de IAM y sus credenciales a largo plazo en su cuenta de AWS. En su lugar, utilice un rol de IAM y sus credenciales de seguridad temporales. Un rol de IAM proporciona un mecanismo para permitir que un tercero acceda a sus recursos de AWS sin necesidad de compartir credenciales a largo plazo (por ejemplo, una clave de acceso del usuario de IAM).

Puede utilizar un rol de IAM para establecer una relación de confianza entre su Cuenta de AWS y la cuenta de Example Corp. Después de que se establezca esta relación, un miembro de la cuenta de Example Corp puede llamar a la API de AWS Security Token Service AssumeRole para obtener credenciales de seguridad temporales. A continuación, los miembros de Example Corp pueden utilizar las credenciales para obtener acceso a los recursos de AWS en su cuenta.

> **Note**
>
> Para obtener más información acerca de AssumeRole y de otras operaciones de la API de AWS que puede llamar para obtener credenciales de seguridad temporales, consulte Solicitud de credenciales de seguridad temporales.

A continuación presentamos un desglose más detallado de la situación:

1. Contrata a Example Corp para que cree un único identificador de cliente para usted. Te proporcionan este ID de cliente único y su número de Cuenta de AWS. Usted necesita esta información para crear un rol de IAM en el siguiente paso.

   > **Note**
   >
   > Example Corp puede utilizar cualquier valor de cadena que desee para el ExternalId, siempre que sea exclusivo para cada cliente. Puede ser un número de cuenta de cliente o incluso una cadena de caracteres aleatoria, siempre que no haya dos clientes con el mismo valor. No pretende ser un "secreto". Example Corp debe proporcionar el valor ExternalId a cada cliente. Lo fundamental es que el valor lo debe generar Example Corp y no sus clientes, para garantizar que cada ID externo sea único.

2. Puede iniciar sesión en AWS y crear un rol de IAM que otorgue acceso a Example Corp a sus recursos. Como cualquier rol de IAM, el rol tiene dos políticas: una política de permisos y una política de confianza. La política de confianza del rol especifica quién puede asumir el rol. En nuestro caso, la política especifica el número de Cuenta de AWS de Example Corp como el

`Principal`. Esto permite que las identidades de la cuenta asuman el rol. Además, se agrega un elemento `Condition` a la política de confianza. Esta `Condition` prueba la clave de contexto `ExternalId` para garantizar que coincide con el ID de cliente único de Example Corp. Por ejemplo:

```
   "Principal": {"AWS": "Example Corp's Cuenta de AWS ID"},
   "Condition": {"StringEquals": {"sts:ExternalId": "Unique ID Assigned by Example
Corp"}}
```

3. La política de permisos del rol especifica qué permite realizar dicho rol. Por ejemplo, podría especificar que el rol permite administrar únicamente los recursos de Amazon EC2 y Amazon RDS, pero no los recursos de usuarios o grupos de IAM. En nuestro escenario de ejemplo, utiliza la política de permisos para ofrecer a Example Corp acceso de solo lectura a todos los recursos de la cuenta.
4. Después de crear el rol, debe proporcionar el nombre de recurso de Amazon (ARN) del rol a Example Corp.
5. Cuando Example Corp necesita acceder a los recursos de AWS, un miembro de la compañía llama a la API AWS de `sts:AssumeRole`. La llamada incluye el ARN de la función que se ha de asumir y el parámetro ExternalId que se corresponde con el ID de cliente.

Si la solicitud proviene de alguien que utiliza la Cuenta de AWS de Example Corp y si el ARN de rol y el ID externo son correctos, la solicitud se realiza correctamente. A continuación, proporciona credenciales de seguridad temporales que Example Corp puede utilizar para obtener acceso a los recursos de AWS que permite su rol.

En otras palabras, cuando una política de roles incluye un ID externo, cualquiera que desee asumir el rol debe ser principal en el rol y debe incluir el ID externo correcto.

¿Por qué utilizar un ID externo?

En términos abstractos, el ID externo permite al usuario que asume el rol afirmar las circunstancias en las que opera. También ofrece al propietario de la cuenta una forma de permitir asumir el rol únicamente en circunstancias específicas. La función principal del ID externo es abordar y prevenir Problema del suplente confuso.

¿Cuándo debería utilizar un ID externo?

Utilice un ID externo en las siguientes situaciones:

- Es un propietario de la Cuenta de AWS y ha configurado un rol para un tercero que obtiene acceso a otras Cuentas de AWS, además de la suya. Debe pedir a ese tercero un ID externo que incluye cuándo asume su rol. A continuación, busque el ID externo en la política de confianza de su rol. De este modo se garantiza que el tercero puede asumir su rol solo cuando actúa en su nombre.
- Se encuentra en la posición de asumir roles en nombre de diferentes clientes como Example Corp en nuestra situación anterior. Debe asignar un ID externo único a cada cliente e indicarle que lo agregue a la política de confianza de su rol. A continuación, deberá asegurarse de incluir siempre el ID externo correcto en sus solicitudes para asumir roles.

  Es muy probable que ya tenga un identificador único para cada uno de sus clientes y que este ID único sea suficiente para su uso como ID externo. El ID externo no es un valor especial que deba crear de forma explícita o realizar un seguimiento por separado, solo para este fin.

  Siempre debe especificar el ID externo en las llamadas a la API `AssumeRole`. Además, cuando un cliente le ofrezca un ARN de rol, pruebe si puede asumir el rol tanto con como sin el ID externo correcto. Si puede asumir el rol sin el ID externo correcto, no almacene el ARN de rol del cliente en su sistema. Espere hasta que el cliente haya actualizado la política de confianza de rol para solicitar el ID externo correcto. De esta forma ayuda a sus clientes a hacer lo correcto, lo que ayuda a mantenerles a ambos protegidos frente al problema del suplente confuso.

Proporcionar acceso a un servicio de AWS

Muchos servicios de AWS exigen el uso de roles para controlar a qué tiene acceso dicho servicio. Un rol que asume un servicio para realizar acciones en su nombre se denomina rol de servicio. Si un rol tiene un fin especializado para un servicio, puede categorizarse como rol de servicio para instancias EC2 o rol vinculado a servicios. Consulte la AWS documentación de cada servicio para ver si utiliza roles y para aprender a asignar un rol para que lo utilice el servicio.

Para obtener más información sobre cómo crear un rol para delegar el acceso a un servicio ofrecido por AWS, consulte Creación de un rol para delegar permisos a un servicio de AWS.

Problema del suplente confuso

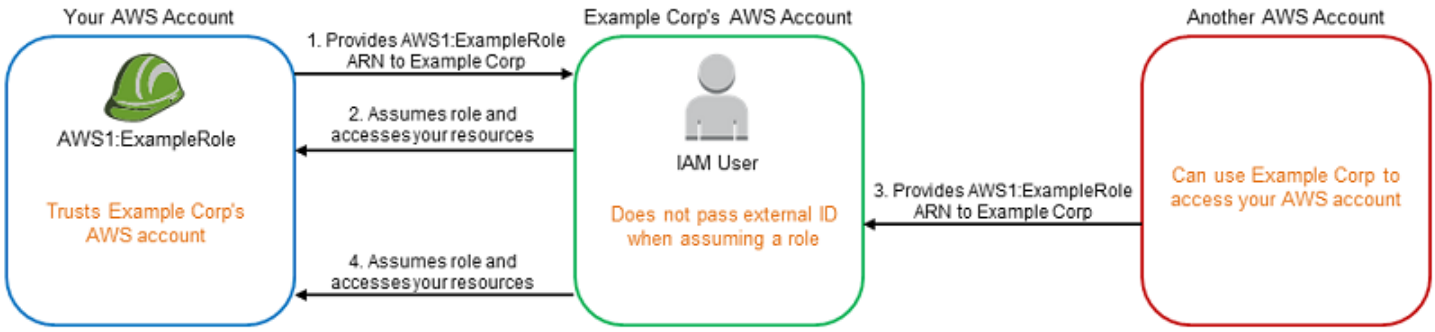
El problema del suplente confuso es un problema de seguridad en el que una entidad que no tiene permiso para realizar una acción puede obligar a una entidad con más privilegios a realizar la acción. Para evitarlo, AWS proporciona herramientas que lo ayudan a proteger su cuenta si proporciona acceso a terceros (conocido como entre cuentas) u otros servicios de AWS (conocido como entre servicios) a los recursos de su cuenta.

A veces, es posible que deba otorgar acceso a terceros a sus recursos de AWS (delegar el acceso). Por ejemplo, digamos que decide contratar a una empresa externa denominada Example Corp para supervisar su Cuenta de AWS y ayudarlo a optimizar los costos. A fin de realizar un seguimiento de su gasto diario, Example Corp necesita acceder a los recursos de AWS. Example Corp también monitoriza muchas otras cuentas de Cuentas de AWS para otros clientes. Puede utilizar un rol de IAM para establecer una relación de confianza entre su Cuenta de AWS y la cuenta de Example Corp. Un aspecto importante de esta situación es el ID externo, una información opcional que puede utilizar en una política de confianza del rol de IAM para señalar quién puede asumir el rol. La función principal del ID externo es abordar y prevenir el problema del suplente confuso.

En AWS, la suplantación entre servicios puede resultar en el problema del suplente confuso. La suplantación entre servicios puede producirse cuando un servicio (el servicio que lleva a cabo las llamadas) llama a otro servicio (el servicio al que se llama). El servicio que lleva a cabo las llamadas se puede manipular para utilizar sus permisos a fin de actuar en función de los recursos de otro cliente de una manera en la que no debe tener permiso para acceder.

Prevención del suplente confuso entre cuentas

En el diagrama siguiente se muestra el problema del suplente confuso entre cuentas.



Este escenario presupone lo siguiente:

- AWS1 es su cuenta de Cuenta de AWS.
- AWS1:ExampleRole es un rol de la cuenta. La política de confianza del rol confía en Example Corp especificando la cuenta de AWS de Example Corp como la que puede asumir el rol.

Y ocurre lo siguiente:

1. Al empezar a utilizar el servicio de Example Corp, usted proporciona el ARN de AWS1:ExampleRole a Example Corp.

2. Example Corp utiliza dicho ARN del rol para obtener credenciales de seguridad temporales para acceder a los recursos de la cuenta de Cuenta de AWS. De esta forma, confía en Example Corp como "suplente" que puede actuar en su nombre.
3. Otro cliente de AWS también ha empezado a utilizar los servicios de Example Corp y también ofrece el ARN de AWS1: ExampleRole para que lo utilice Example Corp. Probablemente el otro cliente ha averiguado o adivinado el AWS1:ExampleRole, que no es un secreto.
4. Cuando el otro cliente solicita a Example Corp obtener acceso a los recursos de AWS en (la que parece ser) su cuenta, Example Corp utiliza AWS1:ExampleRole para obtener acceso a los recursos de la cuenta.

Este es el modo en que el otro cliente podría acceder de nuevo sin autorización a sus recursos. Dado que este otro cliente ha engañado a Example Corp para actuar en los recursos de forma accidental, Example Corp se ha convertido en un "suplente confuso".

Example Corp puede abordar el problema del suplente confuso al solicitarle que incluya la verificación de la condición `ExternalId` en la política de confianza del rol. Example Corp genera un único valor de `ExternalId` para cada cliente y utiliza ese valor en su solicitud para asumir el rol. El valor de `ExternalId` debe ser único entre los clientes de Example Corp y tiene que estar controlado por Example Corp, no por sus clientes. Este es el motivo por el que lo recibe de Example Corp y que no crea el suyo propio. Esto evita que Example Corp sea un suplente confuso y conceda acceso a los recursos de AWS de otra cuenta.

En nuestro caso, imagine que el identificador único de Example Corp para usted es 12345, y su identificador para el otro cliente es 67890. Estos identificadores están simplificados para este ejemplo. Por lo general, estos identificadores son GUID. Suponiendo que estos identificadores son exclusivos para cada cliente de Example Corp, son valores confidenciales que deben utilizarse para el ID externo.

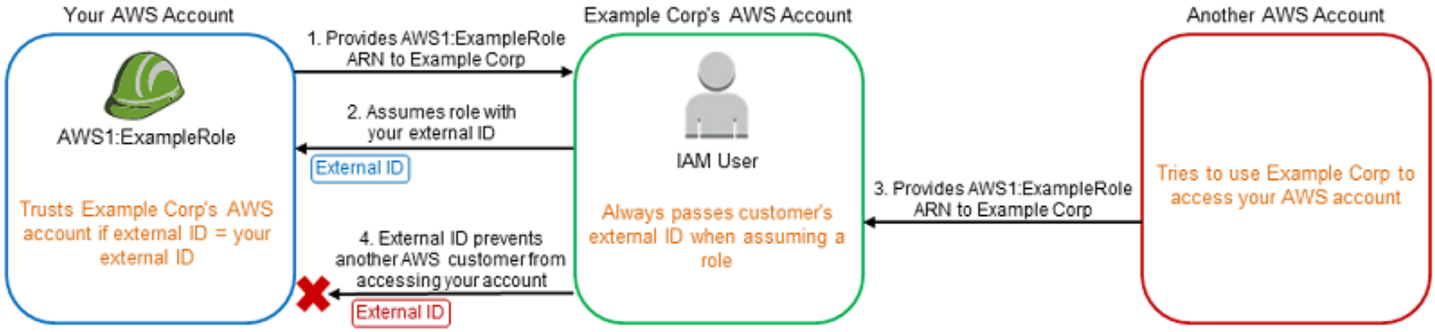
Example Corp le proporciona el valor de ID externo 12345. A continuación, deberá añadir un elemento `Condition` a la política de confianza del rol que exige que el valor `sts:ExternalId` sea 12345, como a continuación:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "AWS": "Example Corp's AWS Account ID"
```

```
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "12345"
      }
    }
  }
}
```

El elemento Condition (Condición) de esta política permite a Example Corp asumir el rol solo cuando la llamada a la API AssumeRole incluye el valor de ID externo 12345. Example Corp se asegura de que cada vez que se asume un rol en nombre de un cliente, siempre incluye el valor de ID externo del cliente en la llamada a AssumeRole. Incluso aunque otro cliente suministre el ARN a Example Corp, no puede controlar el ID externo que Example Corp incluye en su solicitud a AWS. Esto ayuda a impedir que un cliente no autorizado acceda a los recursos.

El siguiente diagrama ilustra este ejemplo.



1. Como anteriormente, al empezar a utilizar el servicio de Example Corp, usted proporciona el ARN de AWS1:ExampleRole a Example Corp.
2. Cuando Example Corp usa ese ARN del rol para asumir el rol AWS1:ExampleRole, Example Corp incluye el ID externo (12345) en la llamada a la API AssumeRole. El ID externo coincide con la política de confianza del rol, por lo que la llamada AssumeRole a la API funciona y Example Corp obtiene las credenciales de seguridad temporales para acceder a los recursos de la cuenta de Cuenta de AWS.
3. Otro cliente de AWS también ha empezado a utilizar los servicios de Example Corp y, como antes, también proporciona el ARN de AWS1: ExampleRole para que lo utilice Example Corp.
4. Sin embargo, ahora, cuando Example Corp intenta asumir el rol AWS1:ExampleRole, proporciona el ID externo asociado con el otro cliente (67890). El otro cliente no puede cambiarlo. Example

Corp lo hace porque la solicitud para utilizar el rol procede de otro cliente, por lo que 67890 indica la circunstancia en la que actúa Example Corp. Dado que ha agregado una condición con su propio ID externo (12345) a la política de confianza de AWS1:ExampleRole, la llamada a la API AssumeRole provoca un error. Así se evita que el otro cliente obtenga un acceso no autorizado a los recursos de su cuenta (indicado por la "X" roja en el diagrama).

El ID externo ayuda a impedir que cualquier otro cliente engañe a Example Corp para que acceda a sus recursos.

Prevención del suplente confuso entre servicios

Le recomendamos que utilice las claves de contexto de condición global `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` o `aws:SourceOrgPaths` en las políticas basadas en recursos a fin de limitar los permisos que un servicio tiene para un recurso específico. Utilice `aws:SourceArn` para asociar solo un recurso al acceso entre servicios. Utilice `aws:SourceAccount` para permitir que cualquier recurso de esa cuenta se asocie al uso entre servicios. Utilice `aws:SourceOrgID` para permitir que cualquier recurso de cuentas dentro de una organización se asocie al uso entre servicios. Utilice `aws:SourceOrgPaths` para asociar cualquier recurso de cuentas dentro de una ruta de AWS Organizations al uso entre servicios. Para obtener más información acerca de las rutas de acceso, consulte Comprender la ruta de la entidad de AWS Organizations.

La forma más granular de protegerse contra el problema del suplente confuso es utilizar la clave de contexto de condición global `aws:SourceArn` con el ARN completo del recurso en sus políticas basadas en recursos. Si no conoce el ARN completo del recurso o si está especificando varios recursos, utilice la clave de condición de contexto global `aws:SourceArn` con comodines (*) para las partes desconocidas del ARN. Por ejemplo, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Si el valor de `aws:SourceArn` no contiene el ID de cuenta, como un ARN de bucket de Amazon S3, debe utilizar `aws:SourceAccount` y `aws:SourceArn` para limitar los permisos.

Para protegerse contra el problema del suplente confuso a gran escala, utilice la clave de contexto de condición global `aws:SourceOrgID` o `aws:SourceOrgPaths`con el identificador de organización o la ruta de organización del recurso en sus políticas basadas en recursos. Las políticas que incluyan la clave `aws:SourceOrgID` o `aws:SourceOrgPaths` incluirán automáticamente las cuentas correctas y no requerirán una actualización manual cuando se agregan, quitan o mueven cuentas en la organización.

En el caso de las políticas de confianza de roles no vinculadas a servicios, todos los servicios de la política de confianza han realizado la acción `iam:PassRole` para comprobar que el rol está en la misma cuenta que el servicio de llamadas. Como resultado, no es necesario utilizar `aws:SourceAccount`, `aws:SourceOrgID` o `aws:SourceOrgPaths` con esas políticas de confianza. El uso de `aws:SourceArn` en una política de confianza permite especificar los recursos para los que se puede asumir una función, como el ARN de una función de Lambda. Algunos Servicios de AWS usan `aws:SourceAccount` y `aws:SourceArn` en políticas de confianza para funciones recién creadas, pero no es necesario usar las claves para las funciones existentes en su cuenta.

> **Note**
>
> Servicios de AWS que se integran con AWS Key Management Service mediante el uso de concesiones de claves de KMS no admiten las claves de condición `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID`, o `aws:SourceOrgPaths`. El uso de estas claves de condición en una política de claves de KMS provocará un comportamiento inesperado si la clave también la utiliza Servicios de AWS mediante concesiones de claves de KMS.

Prevención del suplente confuso entre servicios para AWS Security Token Service

Muchos servicios de AWS requieren que utilice roles para permitir que el servicio obtenga acceso a los recursos de otros servicios en su nombre. Un rol que asume un servicio para realizar acciones en su nombre se denomina rol de servicio. Un rol requiere dos políticas: una política de confianza de rol que especifica la entidad principal que puede asumir el rol y una política de permisos que especifica qué se puede hacer con el rol. Una política de confianza de rol es el único tipo de política basada en recursos de IAM. Otros Servicios de AWS tienen políticas basadas en recursos, como una política de bucket de Amazon S3.

Cuando un servicio asume un rol en su nombre, se debe permitir que la entidad principal del servicio realice la acción `sts:AssumeRole` en la política de confianza de rol. Cuando un servicio llama a `sts:AssumeRole`, AWS STS devuelve un conjunto de credenciales de seguridad temporales que la entidad principal del servicio utiliza para obtener acceso a los recursos permitidos por la política de permisos del rol. Cuando un servicio asume un rol en su cuenta, puede incluir las claves de contexto de condición global `aws:SourceArn` y `aws:SourceAccount`, `aws:SourceOrgID` y `aws:SourceOrgPaths` en la política de confianza de rol para limitar el acceso al rol a solo las solicitudes generadas por los recursos esperados.

Por ejemplo, en AWS Systems Manager Incident Manager, debe elegir un rol para permitir a Incident Manager ejecutar un documento de automatización de Systems Manager en su nombre. El documento de automatización puede incluir planes de respuesta automatizados para incidentes iniciados por alarmas de CloudWatch o eventos de EventBridge. En el siguiente ejemplo de política de confianza de rol, puede utilizar la clave de condición `aws:SourceArn` para restringir el acceso al rol de servicio en función del ARN del registro de incidentes. Solo los registros de incidentes creados a partir del recurso del plan de respuesta `myresponseplan` pueden utilizar este rol.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
}
```

> **Note**
>
> No todas las integraciones de servicios con AWS STS son compatibles con las claves de condición `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` o `aws:SourceOrgPaths`. El uso de estas claves en las políticas de confianza de IAM con integraciones no compatibles puede provocar un comportamiento inesperado.

Proporcionar acceso a usuarios autenticados externamente (identidad federada)

Es posible que los usuarios tengan ya identidades fuera de AWS, por ejemplo, en el directorio corporativo. Si estos usuarios necesitan trabajar con recursos de AWS (o con aplicaciones que necesiten acceso a dichos recursos), estos usuarios también necesitarán tener credenciales de seguridad de AWS. Puede utilizar un rol de IAM para especificar permisos para los usuarios con identidad federada de la organización o para un proveedor de identidad (IdP) externo.

> **Note**
>
> Como práctica recomendada de seguridad, le recomendamos que administre el acceso de los usuarios en IAM Identity Center mediante la federación de identidades en lugar de crear usuarios de IAM. Para obtener más información acerca de situaciones específicas en las que se requiere un usuario de IAM, consulte Cuándo crear un usuario de IAM (en lugar de un rol).

Federación de usuarios de una aplicación móvil o web con Amazon Cognito

Si crea una aplicación móvil o web que tenga acceso a los recursos de AWS, esta aplicación necesita credenciales de seguridad para poder realizar solicitudes mediante programación en AWS. En la mayoría de escenarios de aplicaciones móviles, le recomendamos utilizar Amazon Cognito. Puede utilizar este servicio con AWS Mobile SDK para iOS y AWS Mobile SDK para Android y Fire OS a fin de crear identidades exclusivas para usuarios y autenticarlos para proteger el acceso a sus recursos de AWS. Amazon Cognito es compatible con los mismos proveedores de identidad indicados en la sección siguiente, y también admite identidades autenticadas por desarrollador y acceso no autenticado (invitado). Amazon Cognito también ofrece operaciones de API para sincronizar los datos del usuario de modo que se preserven cuando cambia de un dispositivo a otro. Para obtener más información, consulte Uso de Amazon Cognito para aplicaciones móviles.

Federación de usuarios con proveedores de servicio de identidad pública u OpenID Connect

Siempre que sea posible, utilice Amazon Cognito para escenarios de aplicaciones móviles y web. Amazon Cognito realiza la mayoría del trabajo en segundo plano con los servicios del proveedor de identidad pública. Funciona con los mismos servicios de terceros y también admite inicios de sesión anónimos. Sin embargo, en los escenarios más avanzados, puede trabajar directamente con un servicio de terceros como Login with Amazon, Facebook, Google o cualquier proveedor (IdP) compatible con OpenID Connect (OIDC). Para obtener más información sobre el uso de la federación OIDC con uno de estos servicios, consulte Federación OIDC.

Federación de usuarios con SAML 2.0

Si su organización ya utiliza un paquete de software de proveedor de identidad que admite SAML 2.0 (Security Assertion Markup Language 2.0), puede crear una relación de confianza entre su organización como proveedor de identidad (IdP) y AWS como proveedor del servicio. Entonces podrá utilizar SAML para proporcionar a los usuarios un inicio de sesión único (SSO) federado para el acceso a la AWS Management Console o acceso federado a las llamadas a operaciones de API de

AWS. Por ejemplo, si su compañía utiliza Microsoft Active Directory y Active Directory Federation Services, puede realizar la federación con SAML 2.0. Para obtener más información sobre cómo federar usuarios con SAML 2.0, consulte Federación SAML 2.0.

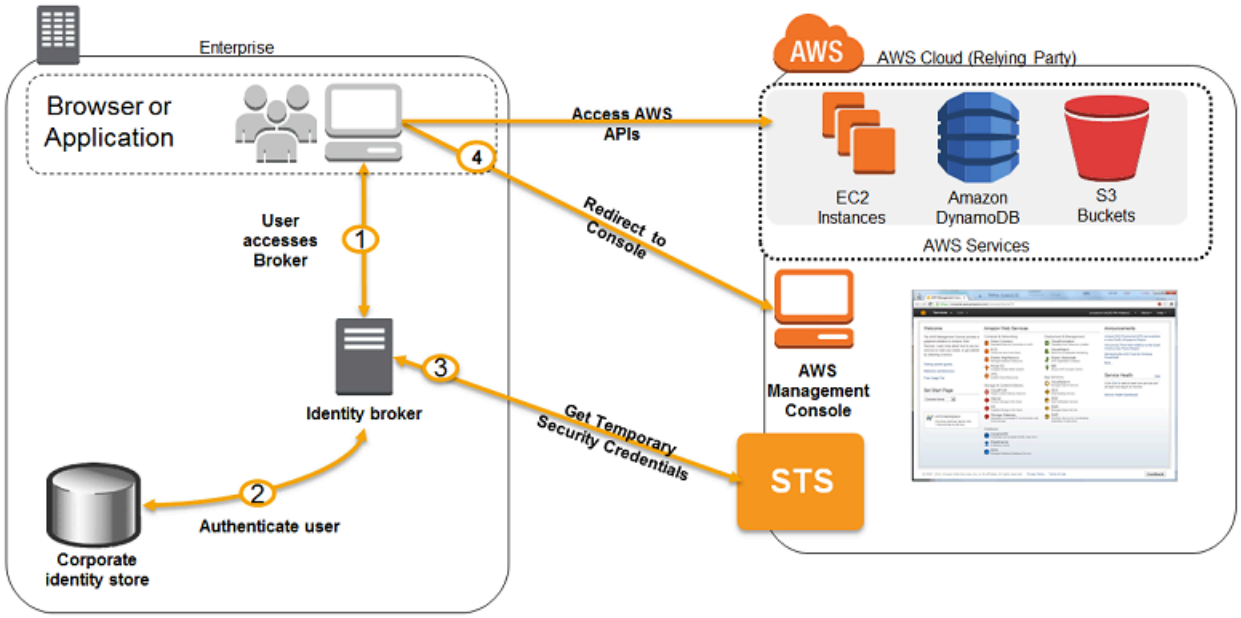
Federación de usuarios mediante la creación de una aplicación personalizada de agente de identidades

Si su almacén de identidades no es compatible con SAML 2.0, puede crear una aplicación personalizada de agente de identidades para llevar a cabo una función similar. La aplicación de agente autentica a los usuarios, solicita credenciales temporales para los usuarios de AWS y les proporciona acceso a los recursos de AWS.

Por ejemplo, Example Corp. tiene muchos empleados que necesitan ejecutar aplicaciones internas que obtengan acceso a los recursos de AWS de la compañía. Los empleados ya tienen identidades en el sistema de autenticación e identidad de la compañía y Example Corp. no quiere crear otro usuario de IAM para cada empleado de la compañía.

Bob es un desarrollador de Example Corp. Para permitir que las aplicaciones internas de la compañía obtengan acceso a los recursos de AWS, Bob desarrolla una aplicación personalizada de agente de identidades. La aplicación verifica que los empleados hayan iniciado sesión en el sistema existente de autenticación e identidad de Example Corp., que podría utilizar LDAP, Active Directory u otro sistema. La aplicación de agente de identidades obtiene credenciales de seguridad temporales para los empleados. Este escenario es similar al anterior (una aplicación móvil que utiliza un sistema personalizado de autenticación), salvo que todas las aplicaciones que necesitan acceso a los recursos de AWS se ejecutan en la red corporativa y la compañía tiene un sistema existente de autenticación.

Para obtener credenciales de seguridad temporales, la aplicación de agente de identidades llama a `AssumeRole` o `GetFederationToken` para obtener credenciales de seguridad temporales, en función del modo en que Bob quiere administrar las políticas para los usuarios y el momento en el que las credenciales temporales caduquen. (Para obtener información sobre las diferencias entre estas operaciones de API, consulte Credenciales de seguridad temporales en IAM y Control de los permisos para credenciales de seguridad temporales). La llamada devuelve credenciales de seguridad temporales que incluyen un token de sesión, una clave de acceso secreta y un ID de clave de acceso de AWS. La aplicación de agente de identidades pone a disposición de la aplicación interna de la compañía estas credenciales de seguridad temporales. La aplicación puede utilizar las credenciales temporales para realizar llamadas a AWS directamente. La aplicación almacena en caché las credenciales hasta que caducan y solicita un nuevo conjunto de credenciales temporales. La siguiente figura ilustra este escenario.



Este escenario tiene los siguientes atributos:

- La aplicación de agente de identidades tiene permisos para obtener acceso a la API de Security Token Service (STS) de IAM para crear credenciales de seguridad temporales.
- La aplicación de agente de identidades puede verificar que los empleados estén autenticados en el sistema existente de autenticación.
- Los usuarios pueden obtener una dirección URL temporal que les proporcione acceso a Management Console de AWS (lo que se denomina inicio de sesión único).

Para obtener información sobre la creación de credenciales de seguridad, consulte Solicitud de credenciales de seguridad temporales. Para obtener más información sobre cómo los usuarios federados obtienen acceso a Management Console de AWS, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 .

Uso de roles vinculados a servicios

Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a un servicio de AWS. Los roles vinculados a servicios son predefinidos por el servicio e incluyen todos los permisos que el servicio requiere para llamar a otros servicios de AWS en su nombre. El servicio vinculado también define cómo crear, modificar y eliminar un rol vinculado a un servicio. Un servicio podría crear el rol automáticamente o eliminarlo. Podría permitirle crear, modificar o eliminar el rol como parte de un asistente o proceso del servicio. También podría exigir que utilice IAM para crear

o eliminar el rol. Independientemente del método, los roles vinculados a servicios simplifican la configuración de un servicio porque ya no tendrá que agregar manualmente los permisos necesarios para que el servicio complete acciones en su nombre.

> **Note**
>
> Recuerde que los roles de servicio son diferentes a los roles vinculados a servicios. Una función del servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM. Un rol vinculado a un servicio es un tipo de rol de servicio que está vinculado a un Servicio de AWS. El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados a servicios aparecen en la Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

El servicio vinculado define los permisos de los roles vinculados con el servicio mismo y, a menos que esté definido de otra manera, solo ese servicio puede asumir los roles. Los permisos definidos incluyen las políticas de confianza y de permisos y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Antes de eliminar las funciones, debe borrar antes sus recursos relacionados. De esta forma, se protegen los recursos de , ya que se evita que se puedan eliminar accidentalmente permisos de acceso a los recursos.

> **Tip**
>
> Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios

Debe configurar permisos para que una entidad de IAM (usuario, grupo o función) permita al usuario o rol crear o editar el rol vinculado al servicio.

> **Note**
>
> El ARN de un rol vinculado a un servicio incluye una entidad principal del servicio, que se indica en las políticas siguientes como *SERVICE-NAME*.amazonaws.com. No intente adivinar la entidad principal del servicio, ya que distingue entre mayúsculas y minúsculas y su formato puede variar para los distintos servicios de AWS. Para ver el elemento principal de un servicio, consulte la documentación correspondiente su rol vinculado a servicio.

Para permitir a una entidad de IAM que cree un rol vinculado a un servicio específico

Agregue la siguiente política a la entidad de IAM que necesite crear el rol vinculado con un servicio.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*",
            "Condition": {"StringLike": {"iam:AWSServiceName": "SERVICE-
NAME.amazonaws.com"}}
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
        }
    ]
}
```

Para permitir a una entidad de IAM crear un rol vinculado a cualquier servicio

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite crear un rol vinculado con un servicio o cualquier función de servicio que incluya las políticas necesarias. Esta instrucción de política no permite la entidad IAM adjunte una política al rol.

```
{
    "Effect": "Allow",
    "Action": "iam:CreateServiceLinkedRole",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir a una entidad IAM editar la descripción de cualquier función de servicio de servicio

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite editar la descripción de un rol vinculado con un servicio o cualquier función de servicio.

```
{
    "Effect": "Allow",
    "Action": "iam:UpdateRoleDescription",
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir a una entidad de IAM eliminar un rol vinculado a un servicio específico

agregue la siguiente instrucción a la política de permisos de la entidad de IAM entidad que necesita eliminar el rol vinculado con el servicio.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/SERVICE-
NAME.amazonaws.com/SERVICE-LINKED-ROLE-NAME-PREFIX*"
}
```

Para permitir a una entidad de IAM eliminar un rol vinculado a cualquier servicio

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita eliminar un rol vinculado con un servicio pero no una función de servicio.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
```

```
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/*"
}
```

Para permitir que una entidad de IAM pase un rol existente al servicio

Algunos servicios de AWS le permiten pasar un rol existente en lugar de crear uno nuevo vinculado al servicio. Para ello, un usuario debe tener permisos para pasar el rol al servicio. Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite pasar el rol. Esta instrucción de la política también permite a la entidad ver la lista de roles entre los que elegir el que se debe pasar. Para obtener más información, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
  "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
  "Effect": "Allow",
  "Action": [ "iam:PassRole" ],
  "Resource": "arn:aws:iam::account-id:role/my-role-for-XYZ"
}
```

Permisos indirectos con funciones vinculadas al servicio

Los permisos concedidos por un rol vinculado a un servicio se pueden transferir indirectamente a otros usuarios y roles. Cuando un rol vinculado a un servicio es utilizado por un servicio AWS, ese rol vinculado a un servicio puede usar sus propios permisos para llamar a otros servicios AWS. Esto significa que los usuarios y los roles con permisos para llamar a un servicio que usa un rol vinculado al servicio pueden tener acceso indirecto a los servicios a los que puede acceder dicho rol vinculado al servicio.

Por ejemplo, al crear una instancia de base de datos de Amazon RDS, se crea automáticamente un rol vinculado a un servicio para RDS si aún no existe ninguno. Este rol vinculado al servicio permite a RDS llamar a Amazon EC2, Amazon SNS, Amazon CloudWatch Logs y Amazon Kinesis en su nombre siempre que edite la instancia de base de datos. Si se permite que los usuarios y los roles de su cuenta modifiquen o creen bases de datos de RDS, es posible que puedan

interactuar indirectamente con los registros de Amazon EC2, Amazon SNS, los registros de Amazon CloudWatch y los recursos de Amazon Kinesis llamando a RDS, ya que RDS utilizaría su función vinculada al servicio para acceder a esos recursos.

Crear un rol vinculado a servicios

El método que se utiliza para crear roles vinculados a servicios depende del servicio. En algunos casos, no es necesario crear manualmente roles vinculados a servicios. Por ejemplo, al finalizar una acción específica (por ejemplo, la creación de un recurso) en el servicio, este puede crear el rol vinculado al servicio en su nombre. Si utilizaba un servicio antes de que comenzara a admitir roles vinculados a servicios, el servicio podría crear automáticamente el rol en la cuenta. Para obtener más información, consulte Un nuevo rol ha aparecido en la cuenta de AWS.

En otros casos, el servicio podría admitir la creación manual de un rol vinculado al servicio mismo mediante su consola, la API o la CLI. Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Para saber si el servicio admite la creación de roles vinculados al servicio mismo, haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Si el servicio no admite la creación de roles, puede utilizar IAM para crear el rol vinculado al servicio.

> **Important**
>
> Los roles vinculados a servicios se contabilizan como roles de IAM en una Cuenta de AWS pero, si ha alcanzado el límite de servicio, igualmente puede crear roles vinculados a servicios en su cuenta. Los roles vinculados a servicios son los únicos que pueden superar el límite.

Creación de un rol vinculado a un servicio (consola)

Antes de crear un rol vinculado a un servicio en IAM, averigüe primero si el servicio vinculado crea automáticamente roles vinculados a servicios, además aprenderá si es posible crearlo desde la consola del servicio, la API o la CLI.

Para crear un rol vinculado a un servicio (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación de la consola de IAM, elija Roles (Roles). A continuación, elija Create role (Crear rol).
3. Elija el tipo de rol Servicio de AWS.
4. Elija el caso de uso para su servicio. Los casos de uso son definidos por el servicio de modo tal que ya incluyen la política de confianza exigida por el servicio mismo. A continuación, haga clic en Next.
5. Elija una o varias políticas de permisos para asociarlas al rol. En función del caso de uso seleccionado, el servicio podría realizar cualquiera de las siguientes acciones:

   - Definir los permisos que utiliza el rol.
   - Permitirle elegir permisos de un conjunto limitado.
   - Permitirle elegir cualquier permiso.
   - Permitirle no seleccionar ninguna política en ese momento, crear las políticas más adelante y, a continuación, asociarlas al rol.

   Seleccione la casilla situada junto a la política que asigna los permisos que desea que tenga el rol y, a continuación, elija Next (Siguiente).

   > **Note**
   >
   > Los permisos que especifique están disponibles para cualquier entidad que utilice el rol. De forma predeterminada, un rol no tiene permisos.

6. En Role Name (Nombre del rol), el servicio define el grado de personalización del nombre del rol. Si el servicio define el nombre del rol, esta opción no es editable. En otros casos, el servicio puede definir un prefijo para el rol y permitirle ingresar un sufijo opcional.

   De ser posible, ingrese un sufijo de nombre de rol para agregarlo al nombre predeterminado. Este sufijo le ayuda a identificar el propósito de este rol. Los nombres de rol deben ser únicos en su cuenta de AWS. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear funciones denominado tanto **`<service-linked-role-name>_SAMPLE`** y **`<service-linked-role-name>_sample`**. Dado que varias entidades pueden hacer referencia al rol, no puede editar el nombre del rol después de crearlo.
7. (Opcional) En Description (Descripción), edite la descripción del nuevo rol vinculado al servicio.
8. No puede asociar etiquetas a roles vinculados a servicios durante la creación. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.

9. Revise el rol y, a continuación, seleccione Create role.

Crear una función vinculada a un servicio (AWS CLI)

Antes de crear un rol vinculado a un servicio en IAM, averigüe primero si el servicio vinculado crea automáticamente roles vinculados a servicios y si no es posible crearlo desde la CLI del servicio. Si la CLI del servicio no es compatible, puede utilizar comandos de IAM para crear un rol vinculado al servicio con la política de confianza y las políticas insertadas que el servicio necesita para asumir el rol.

Para crear un rol vinculado a un servicio (AWS CLI)

Ejecute el siguiente comando:

```
aws iam create-service-linked-role --aws-service-name SERVICE-NAME.amazonaws.com
```

Crear un rol vinculado a un servicio (API de AWS)

Antes de crear un rol vinculado a un servicio en IAM, averigüe primero si el servicio vinculado crea automáticamente roles vinculados a servicios y si no es posible crearlo desde la API del servicio. Si la API del servicio no es compatible, puede utilizar la API de AWS para crear un rol vinculado al servicio con la política de confianza y las políticas insertadas que el servicio necesita para asumir el rol.

Para crear un rol vinculado a un servicio (API de AWS)

Utilice la llamada a la API CreateServiceLinkedRole. En la solicitud, especifique el nombre del servicio de *SERVICE_NAME_URL*`.amazonaws.com`.

Por ejemplo, para crear el rol vinculado al servicio Lex Bots (Robots de Lex), utilice `lex.amazonaws.com`.

Editar un rol vinculado a servicios

El método que utilice para editar roles vinculados a servicios depende del servicio. Algunos servicios le permiten editar los permisos de un rol vinculado a un servicio desde la consola, la API o la CLI. Sin embargo, después de crear un rol vinculado a un servicio, no puede cambiarle el nombre, ya que que varias entidades pueden hacer referencia al rol. Puede editar la descripción de cualquier rol desde la consola, la API o la CLI de IAM.

Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Para saber si el servicio admite la edición de roles vinculados al servicio mismo, haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Editar la descripción de un rol vinculado a un servicio (consola)

Puede utilizar la consola de IAM para editar la descripción de un rol vinculado a un servicio.

Para editar la descripción de un rol vinculado a un servicio (consola)

1. En el panel de navegación de la consola de IAM, elija Roles (Roles).
2. Seleccione el nombre del rol que desea modificar.
3. En el extremo derecho de Role description, seleccione Edit.
4. Ingrese una descripción nueva en el cuadro Save (Guardar).

Edición de la descripción de un rol vinculado a servicio (AWS CLI)

Puede utilizar comandos de IAM desde la AWS CLI para editar la descripción de un rol vinculado a un servicio.

Para cambiar la descripción de un rol vinculado a un servicio (AWS CLI)

1. (Opcional) Para ver la descripción actual de un rol, ejecute los siguientes comandos:

   ```
   aws iam get-role --role-name ROLE-NAME
   ```

   Utilice el nombre del rol, no el ARN, para hacer referencia a los roles con los comandos de CLI. Por ejemplo, si un rol tiene el ARN `arn:aws:iam::123456789012:role/myrole`, debe referirse a él como **`myrole`**.
2. Para actualizar la descripción de un rol vinculado a un servicio, ejecute el siguiente comando:

   ```
   aws iam update-role --role-name ROLE-NAME --description OPTIONAL-DESCRIPTION
   ```

Edición de la descripción de una función vinculada a servicio (API de AWS)

Puede utilizar la API de AWS para editar la descripción de un rol vinculado a un servicio.

Para cambiar la descripción de un rol vinculado a un servicio (API de AWS)

1. (Opcional) Para ver la descripción actual de un rol, llame a la siguiente operación y especifique el nombre del rol:

   API de AWS: GetRole

2. Para actualizar la descripción de un rol, llame a la siguiente operación y especifique el nombre (y opcionalmente, una descripción) del rol:

   API de AWS: UpdateRole

Eliminar un rol vinculado a servicios

El método que se utiliza para crear roles vinculados a servicios depende del servicio. En algunos casos, no es necesario eliminar manualmente roles vinculados a servicios. Por ejemplo, al finalizar una acción específica (por ejemplo, retirar un recurso) en el servicio, este puede eliminar el rol vinculado al servicio en su nombre.

En otros casos, el servicio podría admitir la eliminación manual de un rol vinculado al servicio mismo desde su consola, la API o la AWS CLI.

Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Para saber si el servicio admite la eliminación de roles vinculados al servicio mismo, haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Si el servicio no admite eliminar el rol, puede eliminar el rol vinculado al servicio desde la consola de IAM, la API o la AWS CLI. Si ya no necesita utilizar una característica o servicio que requiere un rol vinculado a un servicio, le recomendamos que elimine dicho rol. De esta forma no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa. Sin embargo, debe limpiar el rol vinculado al servicio antes de eliminarlo.

Limpiar un rol vinculado a servicios

Antes de poder utilizar IAM para eliminar un rol vinculado a un servicio, primero debe confirmar que dicho rol no tiene sesiones activas y eliminar los recursos que utiliza.

Para comprobar si el rol vinculado a un servicio tiene una sesión activa en la consola de IAM

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles (Roles). A continuación, elija el nombre (no la casilla) del rol vinculado al servicio.
3. En la página Summary (Resumen) del rol seleccionado, elija la pestaña Access Advisor (Asesor de acceso).
4. En la pestaña Access Advisor (Asesor de acceso), revise la actividad reciente del rol vinculado a servicios.

> **Note**
>
> Si no está seguro de si el servicio está utilizando el rol vinculado al mismo, puede intentar eliminar el rol para comprobarlo. Si el servicio utiliza el rol, este no podrá eliminarse y podrá ver las regiones en las que se utiliza. Si el rol se está utilizando, debe esperar que la sesión finalice para poder eliminarlo. No se puede revocar la sesión de un rol vinculado a servicios.

Para eliminar los recursos utilizados por un rol vinculado a un servicio

Para obtener información sobre los servicios que admiten el uso de roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes en la columna Service-Linked Role. Para saber si el servicio admite la eliminación de roles vinculados al servicio mismo, haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio. Consulte la documentación del servicio en cuestión para obtener información acerca de cómo eliminar los recursos utilizados por un rol vinculado a dicho servicio.

Eliminar una función vinculada a un servicio (consola)

Puede utilizar la consola de IAM para eliminar un rol vinculado a un servicio.

Para eliminar un rol vinculado a un servicio (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación de la consola de IAM, elija Roles. A continuación, seleccione la casilla junto al nombre del rol que desea eliminar, no el nombre ni la fila.
3. En Acciones de rol en la parte superior de la página, elija Eliminar.
4. En el cuadro de diálogo de confirmación, revise la información de acceso reciente, donde se indica cuándo accedió cada uno de los roles seleccionados a un servicio de AWS por última vez. Esto lo ayuda a confirmar si el rol está actualmente activo. Si desea continuar, seleccione Yes, Delete para enviar la solicitud de eliminación del rol vinculado al servicio.
5. Consulte las notificaciones de la consola de IAM para monitorear el progreso de la eliminación del rol vinculado al servicio. Como el proceso de eliminación del rol vinculado al servicio de IAM es asíncrono, dicha tarea puede realizarse correctamente o fallar después de que envía la solicitud de eliminación.

   - Si la tarea se realiza correctamente, el rol se elimina de la lista y aparece una notificación informando de ello en la parte superior de la página.
   - Si la tarea no se realiza correctamente, puede seleccionar View details (Ver detalles) o View Resources (Ver recursos) desde las notificaciones para obtener información sobre el motivo por el que no se pudo eliminar el rol. Si la eliminación no pudo producirse porque el rol está utilizando los recursos del servicio, la notificación incluye una lista de dichos recursos si el servicio proporciona dicha información. Tras conocer esa información, podrá limpiar los recursos y volver a enviar la solicitud de eliminación.

     > **Note**
     >
     > Es posible que tenga que repetir este proceso varias veces, en función de la información que devuelva el servicio. Por ejemplo, el rol vinculado al servicio podría estar utilizando seis recursos y el servicio podría estar devolviendo información solo acerca de cinco de ellos. Si limpia los cinco recursos y envía la solicitud de eliminación del rol de nuevo, se producirá un error y el servicio informará del recurso restante. Un servicio podría informar de todos los recursos, algunos o ninguno.

   - Si se produce un error en la tarea y la notificación no incluye una lista de los recursos, el servicio no podría no devolver dicha información. Para obtener más información sobre cómo limpiar los recursos del servicio en cuestión, consulte Servicios de AWS que funcionan con IAM. Identifique su servicio en la tabla y haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Eliminar un rol vinculado a un servicio (AWS CLI)

Puede utilizar los comandos de IAM desde la AWS CLI para eliminar un rol vinculado a un servicio.

Para eliminar un rol vinculado a un servicio (AWS CLI)

1. Si conoce el nombre del rol vinculado al servicio que desea eliminar, ingrese el siguiente comando para enumerar el rol de su cuenta:

   ```
   aws iam get-role --role-name role-name
   ```

   Utilice el nombre del rol, no el ARN, para hacer referencia a los roles con los comandos de CLI. Por ejemplo, si un rol tiene el ARN `arn:aws:iam::123456789012:role/myrole`, debe referirse a él como **`myrole`**.

2. Como los roles vinculados a servicios no se puede eliminar si están en uso o tienen recursos asociados, debe enviar una solicitud de eliminación. Esta solicitud puede denegarse si no se cumplen estas condiciones. Debe apuntar el valor `deletion-task-id` de la respuesta para comprobar el estado de la tarea de eliminación. Ingrese el siguiente comando para enviar una solicitud de eliminación de un rol vinculado a un servicio:

   ```
   aws iam delete-service-linked-role --role-name role-name
   ```

3. Ingrese el siguiente comando para comprobar el estado de la tarea de eliminación:

   ```
   aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   El estado de la tarea de eliminación puede ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` o `FAILED`. Si ocurre un error durante la eliminación, la llamada devuelve el motivo del error para que pueda resolver el problema. Si la eliminación no pudo producirse porque el rol está utilizando los recursos del servicio, la notificación incluye una lista de dichos recursos si el servicio proporciona dicha información. Tras conocer esa información, podrá limpiar los recursos y volver a enviar la solicitud de eliminación.

   > **Note**
   >
   > Es posible que tenga que repetir este proceso varias veces, en función de la información que devuelva el servicio. Por ejemplo, el rol vinculado al servicio podría estar utilizando seis recursos y el servicio podría estar devolviendo información solo acerca de cinco

de ellos. Si limpia los cinco recursos y envía la solicitud de eliminación del rol de nuevo, se producirá un error y el servicio informará del recurso restante. Un servicio podría informar de todos los recursos, algunos o ninguno. Para obtener más información sobre cómo limpiar los recursos de un servicio que no está informando de ningún recurso, consulte Servicios de AWS que funcionan con IAM. Identifique su servicio en la tabla y haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Eliminar una función vinculada a un servicio (API de AWS)

Puede utilizar la API de AWS para eliminar una función vinculada a un servicio.

Para eliminar un rol vinculado a un servicio (API de AWS)

1. Para enviar una solicitud de eliminación de un rol vinculado a un servicio, realice una llamada a DeleteServiceLinkedRole. En la solicitud, especifique el nombre del rol.

   Como los roles vinculados a servicios no se puede eliminar si están en uso o tienen recursos asociados, debe enviar una solicitud de eliminación. Esta solicitud puede denegarse si no se cumplen estas condiciones. Debe apuntar el valor `DeletionTaskId` de la respuesta para comprobar el estado de la tarea de eliminación.
2. Para comprobar el estado de la tarea de eliminación, realice una llamada a GetServiceLinkedRoleDeletionStatus. En la solicitud, especifique el valor de `DeletionTaskId`.

   El estado de la tarea de eliminación puede ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED` o `FAILED`. Si ocurre un error durante la eliminación, la llamada devuelve el motivo del error para que pueda resolver el problema. Si la eliminación no pudo producirse porque el rol está utilizando los recursos del servicio, la notificación incluye una lista de dichos recursos si el servicio proporciona dicha información. Tras conocer esa información, podrá limpiar los recursos y volver a enviar la solicitud de eliminación.

   > **Note**
   >
   > Es posible que tenga que repetir este proceso varias veces, en función de la información que devuelva el servicio. Por ejemplo, el rol vinculado al servicio podría estar utilizando seis recursos y el servicio podría estar devolviendo información solo acerca de cinco de ellos. Si limpia los cinco recursos y envía la solicitud de eliminación del rol de nuevo,

se producirá un error y el servicio informará del recurso restante. Un servicio podría informar de todos los recursos, algunos o ninguno. Para obtener más información sobre cómo limpiar los recursos de un servicio que no está informando de ningún recurso, consulte Servicios de AWS que funcionan con IAM. Identifique su servicio en la tabla y haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con los roles vinculados a dicho servicio.

Creación de roles de IAM

Para crear un rol, puede utilizar AWS Management Console, AWS CLI, Tools for Windows PowerShell o la API de IAM.

Si utiliza la AWS Management Console, un asistente le guiará por los pasos de creación de un rol. Los pasos del asistente varían ligeramente en función de si crea un rol para un servicio de AWS, una Cuenta de AWS o un usuario federado.

Temas

- Creación de un rol para delegar permisos a un usuario de IAM
- Creación de un rol para delegar permisos a un servicio de AWS
- Creación de un rol para un proveedor de identidad de terceros (federación)
- Creación de un rol mediante políticas de confianza personalizadas (consola)
- Ejemplos de políticas para delegar el acceso

Creación de un rol para delegar permisos a un usuario de IAM

Puede utilizar roles de IAM para delegar el acceso a sus recursos de AWS. Con roles de IAM puede establecer relaciones de confianza entre la cuenta que confía y otras cuentas de confianza de AWS. La cuenta que confía posee el recurso al que se obtiene acceso y la cuenta de confianza incluye los usuarios que necesitan obtener acceso al recurso. Sin embargo, es posible que otra cuenta sea propietaria de un recurso de su cuenta. Por ejemplo, la cuenta que confía podría permitir a la cuenta de confianza crear recursos, como, por ejemplo, crea objetos en un bucket de Amazon S3. En ese caso, la cuenta que crea el recurso es la propietaria del recurso y controla quién pueden tener acceso a dicho recurso.

Después de crear la relación de confianza, un usuario de IAM o una aplicación de la cuenta de confianza pueden utilizar la operación AWS Security Token Service (AWS STS) `AssumeRole` de la

API. Esta operación proporciona credenciales de seguridad temporales que permiten el acceso a los recursos de AWS de su cuenta.

Usted puede controlar ambas cuentas o un tercero puede controlar la cuenta con los usuarios. Si la otra cuenta con los usuarios se encuentra en una Cuenta de AWS que usted no controla, puede utilizar el atributo `externalId`. El ID externo puede ser cualquier palabra o número acordados entre usted y el administrador de la cuenta de terceros. Esta opción agrega automáticamente una condición a la política de confianza que permite al usuario asumir el rol únicamente si la solicitud incluye el `sts:ExternalID` correcto. Para obtener más información, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.

Para obtener información sobre cómo utilizar los roles para delegar permisos, consulte Términos y conceptos de roles. Para obtener más información sobre el uso de un rol de servicio para permitir que los servicios obtengan acceso a los recursos de su cuenta, consulte Creación de un rol para delegar permisos a un servicio de AWS.

Creación de un rol de IAM (consola)

Puede utilizar la AWS Management Console para crear un rol que un usuario de IAM pueda asumir. Por ejemplo, suponga que su organización tiene varias Cuentas de AWS para aislar un entorno de desarrollo de uno de producción. Para información general sobre cómo crear un rol que permita a usuarios de la cuenta de desarrollo acceder a los recursos de la cuenta de producción, consulte Situación de ejemplo en la que se usan cuentas de desarrollo y producción separadas.

Para crear un rol (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Roles y, a continuación, seleccione Crear rol.
3. Elija el tipo de rol de Cuenta de AWS.
4. Para crear un rol para la cuenta, elija Esta cuenta. Para crear un rol para otra cuenta, elija Otra Cuenta de AWS e ingrese el ID de cuenta al que desea conceder acceso a los recursos.

   El administrador de la cuenta especificada puede conceder permiso para asumir este rol a cualquier usuario de IAM en esa cuenta. Para ello, el administrador asocia una política al usuario o grupo que concede permiso para la acción `sts:AssumeRole`. Esta política debe especificar el ARN del rol como `Resource`.
5. Si concede permisos a los usuarios desde una cuenta que no controla y los usuarios van a asumir este rol mediante programación, seleccione Requerir ID externo. El ID externo puede ser

cualquier palabra o número acordados entre usted y el administrador de la cuenta de terceros. Esta opción agrega automáticamente una condición a la política de confianza que permite al usuario asumir el rol únicamente si la solicitud incluye el `sts:ExternalID` correcto. Para obtener más información, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.

> **Important**
>
> Si elige esta opción, restringe el acceso al rol únicamente a través de la API de AWS CLI, Tools for Windows PowerShell o API de AWS. Esto se debe a que no puede utilizar la consola de AWS para cambiar a un rol que tiene una condición `externalId` en su política de confianza. Sin embargo, puede crear este tipo de acceso mediante programación si escribe un script o una aplicación con el correspondiente SDK. Para obtener más información y un script de muestra, consulte Cómo habilitar el acceso entre cuentas a la AWS Management Console en el blog de seguridad de AWS.

6. Si desea restringir el rol a aquellos usuarios que inicien sesión con autenticación multifactor (MFA), seleccione Requerir MFA. De esta forma se agrega una condición a la política de confianza del rol que comprueba si se produce un inicio de sesión con MFA. Un usuario que desee asumir el rol debe iniciar sesión con una contraseña temporal de uso único desde un dispositivo MFA configurado. Los usuarios sin autenticación MFA no pueden asumir el rol. Para obtener más información acerca de MFA, consulte Uso de autenticación multifactor (MFA) en AWS
7. Seleccione Siguiente.
8. IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta. Seleccione la política que desea utilizar como política de permisos o elija Crear política para abrir una pestaña nueva del navegador y crear una política nueva desde cero. Para obtener más información, consulte Crear políticas de IAM. Después de crear la política, cierre esa pestaña y vuelva a la pestaña original. Seleccione la casilla situada junto a las políticas de permisos que desea conceder a cualquier persona que asuma el rol. Si lo prefiere, puede optar por no seleccionar ninguna política en ese momento y asociar las políticas al rol más adelante. De forma predeterminada, un rol no tiene permisos.
9. (Opcional) Configure un límite de permisos. Esta es una característica avanzada.

   Abra la sección Configurar límite de permisos y elija Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo. Seleccione la política que desea utilizar para el límite de permisos.

10. Seleccione Siguiente.
11. Escriba un nombre para el rol en Nombre de rol. Los nombres de rol deben ser únicos en su Cuenta de AWS. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre del rol distingue entre mayúsculas y minúsculas. Cuando los clientes ven un nombre de rol en la consola, por ejemplo, durante el proceso de inicio de sesión, el nombre del rol no distingue entre mayúsculas y minúsculas. Dado que varias entidades pueden hacer referencia al rol, no se puede editar el nombre del rol una vez que se crea.
12. (Opcional) En Descripción, ingrese una descripción para el nuevo rol.
13. Elija Editar en las secciones Paso 1: seleccionar entidades de confianza o Paso 2: agregar permisos para editar los casos de uso y los permisos del rol. Volverá a las páginas anteriores para realizar las modificaciones.
14. De manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
15. Revise el rol y, a continuación, seleccione Crear rol.

> **Important**
>
> Recuerde que esto es solo la primera mitad de la configuración necesaria. También debe conceder permisos a determinados usuarios de la cuenta de confianza para cambiar al rol en la consola o para asumir el rol mediante programación. Para obtener más información acerca de este paso, consulte Conceder permisos de usuario para cambiar de rol.

Creación de un rol de IAM (AWS CLI)

Para crear un rol desde la AWS CLI se deben seguir varios pasos. Si utiliza la consola para crear un rol, muchos de los pasos se realizan automáticamente, pero con la AWS CLI deberá realizar cada paso usted mismo. Debe crear el rol y, a continuación, asignar una política de permisos al rol. Si lo prefiere, también puede configurar el límite de permisos para el rol.

Para crear un rol para el acceso entre cuentas (AWS CLI)

1. Crear un rol: aws iam create-role
2. Asociar una política de permisos administrada al rol: aws iam attach-role-policy

o

Crear una política de permisos insertada para el rol: aws iam put-role-policy

3. (Opcional) Añadir los atributos personalizados al rol asociando etiquetas: aws iam tag-role

   Para obtener más información, consulte Administrar etiquetas en roles de IAM (AWS CLI o API de AWS).

4. (Opcional) Configurar el límite de permisos para el rol: aws iam put-role-permissions-boundary

   Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

El siguiente ejemplo muestra los dos primeros pasos, que también son los más comunes, para crear un rol entre cuentas en un entorno sencillo. Este ejemplo permite a cualquier usuario de la cuenta `123456789012` asumir el rol y ver el bucket de `example_bucket` Amazon S3. Este ejemplo también supone que se está utilizando un equipo cliente con Windows y que ya se ha configurado la interfaz de línea de comandos con las credenciales de la cuenta y la región. Para obtener más información, consulte Configuración de la interfaz de línea de comandos de AWS.

En este ejemplo, incluya la siguiente política de confianza en el primer comando al crear el rol. Esta política de confianza permite a los usuarios de la cuenta `123456789012` asumir el rol utilizando la operación `AssumeRole`, pero solo si el usuario proporciona la autenticación MFA utilizando los parámetros `SerialNumber` y `TokenCode`. Para obtener más información acerca de MFA, consulte Uso de autenticación multifactor (MFA) en AWS.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
          "Action": "sts:AssumeRole",
          "Condition": { "Bool": { "aws:MultiFactorAuthPresent": "true" } }
      }
  ]
}
```

> **Important**
>
> Si el elemento `Principal` incluye el ARN de un determinado usuario o rol de IAM, dicho ARN se transforma en un ID exclusivo de entidad principal cuando se guarda la política. Esto ayuda a mitigar el riesgo de que alguien aumente sus permisos eliminando o volviendo a crear el rol o usuario. Normalmente, este ID no se muestra en la consola porque también existe una transformación inversa al ARN cuando se muestra la política de confianza. Sin embargo, si se elimina el rol o el usuario, el ID de entidad principal aparece en la consola porque AWS ya no puede volver a asignarlo a un ARN. Por lo tanto, si elimina y vuelve a crear un usuario o rol al que se hace referencia en un elemento `Principal` de la política de confianza, debe editar el rol para sustituir el ARN.

Cuando utilice el segundo comando, debe asociar una política administrada existente al rol. La siguiente política de permisos permite a cualquiera que asuma el rol realizar únicamente la acción `ListBucket` en el bucket de Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": "s3:ListBucket",
          "Resource": "arn:aws:s3:::example_bucket"
      }
  ]
}
```

Para crear este rol `Test-UserAccess-Role`, primero debe guardar la política de confianza anterior con el nombre `trustpolicyforacct123456789012.json` en la carpeta `policies` del disco duro local `C:`. A continuación, guarde el política de permisos anterior como una política administrada por el cliente en su Cuenta de AWS con el nombre `PolicyForRole`. A continuación, puede utilizar los comandos siguientes para crear el rol y asociarle la política administrada.

```
# Create the role and attach the trust policy file that allows users in the specified
 account to assume the role.
$ aws iam create-role --role-name Test-UserAccess-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforacct123456789012.json
```

```
# Attach the permissions policy (in this example a managed policy) to the role to
 specify what it is allowed to do.
$ aws iam attach-role-policy --role-name Test-UserAccess-Role --policy-arn
 arn:aws:iam::123456789012:policy/PolicyForRole
```

> **Important**
>
> Recuerde que esto es solo la primera mitad de la configuración necesaria. También debe conceder permisos a los usuarios individuales de la cuenta de confianza para cambiar al rol. Para obtener más información acerca de este paso, consulte Conceder permisos de usuario para cambiar de rol.

Después de crear el rol y concederle permisos para realizar tareas de AWS u obtener acceso a los recursos de AWS, cualquier usuario de la cuenta `123456789012` puede asumir el rol. Para obtener más información, consulte Cambio a un rol de IAM (AWS CLI).

Creación de un rol de IAM (API de AWS)

Para crear un rol desde la API de AWS se deben seguir varios pasos. Si utiliza la consola para crear un rol, muchos de los pasos se realizan automáticamente, pero con la API deberá realizar cada paso usted mismo. Debe crear el rol y, a continuación, asignar una política de permisos al rol. Si lo prefiere, también puede configurar el límite de permisos para el rol.

Para crear un rol en código (API de AWS)

1. Creación de un rol: CreateRole

   Para la política de confianza del rol, puede especificar una ubicación de archivo.
2. Asociar una política de permisos administrada al rol: AttachRolePolicy

   o

   Crear una política de permisos insertada para el rol: PutRolePolicy

   > **Important**
   >
   > Recuerde que esto es solo la primera mitad de la configuración necesaria. También debe conceder permisos a los usuarios individuales de la cuenta de confianza para

cambiar al rol. Para obtener más información acerca de este paso, consulte Conceder permisos de usuario para cambiar de rol.

3. (Opcional) Añadir los atributos personalizados al usuario asociando etiquetas: TagRole

   Para obtener más información, consulte Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS).

4. (Opcional) Configuración del límite de permisos para el rol: PutRolePermissionsBoundary

   Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

Después de crear el rol y concederle permisos para realizar tareas de AWS u obtener acceso a los recursos de AWS, debe conceder permisos a los usuarios de la cuenta para que puedan asumir el rol. Para obtener más información sobre cómo asumir un rol, consulte Cambio a un rol de IAM (API de AWS).

Creación de un rol de IAM (AWS CloudFormation)

Para obtener información acerca de cómo crear un rol de IAM en AWS CloudFormation, consulte la referencia de recursos y propiedades y los ejemplos en la Guía del usuario de AWS CloudFormation.

Para obtener más información acerca de las plantillas de IAM en AWS CloudFormation, consulte fragmentos de plantilla AWS Identity and Access Management en la Guía del usuario de AWS CloudFormation.

Creación de un rol para delegar permisos a un servicio de AWS

Muchos servicios de AWS requieren que utilice roles para permitir que el servicio obtenga acceso a los recursos de otros servicios en su nombre. Un rol que asume un servicio para realizar acciones en su nombre se denomina rol de servicio. Si un rol tiene un fin específico para un servicio, se identifica como rol de servicio para instancias EC2 (por ejemplo) o como rol vinculado a un servicio. Para ver qué servicios son compatibles con el uso de roles vinculados a servicios, o si un servicio admite algún tipo de credenciales temporales, consulte Servicios de AWS que funcionan con IAM. Para obtener información sobre cómo un servicio determinado utiliza los roles, elija el nombre del servicio en la tabla para ver la documentación correspondiente a dicho servicio.

Al configurar el permiso `PassRole`, debe asegurarse de que un usuario no pase un rol en el que el rol tenga más permisos de los que usted desea que tenga el usuario. Por ejemplo, es posible que a

Alice no se le permita realizar ninguna acción de Amazon S3. Si Alice pudiera transferir un rol a un servicio que permita acciones de Amazon S3, el servicio podría realizar acciones de Amazon S3 en su nombre al ejecutar el trabajo.

Para obtener información sobre cómo los roles le pueden ayudar a delegar permisos, consulte Términos y conceptos de roles.

Permisos del rol de servicio

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear o editar una función de servicio.

> **Note**
>
> El ARN de un rol vinculado a un servicio incluye una entidad principal del servicio, que se indica en las políticas siguientes como *SERVICE-NAME*.amazonaws.com. No intente adivinar la entidad principal del servicio, ya que distingue entre mayúsculas y minúsculas y su formato puede variar para los distintos servicios de AWS. Para ver el elemento principal de un servicio, consulte la documentación correspondiente su rol vinculado a servicio.

Para permitir a una entidad de IAM cree un rol vinculado a un servicio específico

Agregue la siguiente política a la entidad de IAM que necesite crear la función de servicio. Esta política le permite crear un rol de servicio para el servicio especificado y con un nombre específico. A continuación, puede asociar políticas administradas o insertadas al rol.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        }
    ]
```

```
}
```

Cómo permitir a una entidad de IAM crear un rol de servicio

AWS recomienda permitir solo a los administradores crear cualquier rol de servicio. Una persona con permisos para crear un rol y adjuntar cualquier política puede escalar sus propios permisos. En su lugar, cree una política que les permita crear solo los roles que necesitan o haga que un administrador cree el rol de servicio en su nombre.

Para adjuntar una política que permita a un administrador acceder a toda la Cuenta de AWS, utilice la política administrada AdministratorAccess de AWS.

Cómo permitir a una entidad de IAM editar un rol de servicio

Agregue la siguiente política a la entidad de IAM que necesite editar la función de servicio.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EditSpecificServiceRole",
            "Effect": "Allow",
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:DetachRolePolicy",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "iam:ListRolePolicies",
                "iam:PutRolePolicy",
                "iam:UpdateRole",
                "iam:UpdateRoleDescription"
            ],
            "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
        },
        {
            "Sid": "ViewRolesAndPolicies",
            "Effect": "Allow",
            "Action": [
                "iam:GetPolicy",
                "iam:ListRoles"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Para permitir a una entidad de IAM eliminar una función de servicio específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite eliminar la función de servicio especificado.

```
{
    "Effect": "Allow",
    "Action": "iam:DeleteRole",
    "Resource": "arn:aws:iam::*:role/SERVICE-ROLE-NAME"
}
```

Cómo permitir a una entidad de IAM eliminar cualquier rol de servicio

AWS recomienda permitir solo a los administradores eliminar cualquier rol de servicio. En su lugar, cree una política que les permita eliminar solo los roles que necesitan o haga que un administrador elimine el rol de servicio en su nombre.

Para adjuntar una política que permita a un administrador acceder a toda la Cuenta de AWS, utilice la política administrada AdministratorAccess de AWS.

Creación de un rol para un servicio de AWS (consola)

Puede utilizar la AWS Management Console para crear un rol para un servicio. Algunos servicios admiten más de un rol de servicio. Por lo tanto, recomendamos que consulte la documentación de AWS relacionada con su servicio para ver qué caso de uso debe elegir. Puede obtener más información acerca de cómo asignar las políticas de confianza y de permisos necesarias para el rol, para que el servicio pueda asumir el rol en su nombre. Los pasos que puede utilizar para controlar los permisos para el rol pueden variar en función de cómo defina el servicio los casos de uso y de si se crea o no un rol vinculado al servicio.

Cómo crear un rol para un Servicio de AWS (consola de IAM)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione un servicio y, a continuación, el caso de uso. Los casos de uso son definidos por el servicio de modo tal que ya incluyen la política de confianza que el servicio mismo requiere.
5. Seleccione Siguiente.
6. Para las Políticas de permisos, las opciones dependen del caso de uso que haya seleccionado:
   - Si el servicio define los permisos para el rol, no puede seleccionar políticas de permisos.
   - Seleccione entre un conjunto limitado de políticas de permisos.
   - Seleccione una de todas las políticas de permisos.
   - No seleccione ninguna política de permisos en este momento. Después de crear el rol, genere las políticas y luego asócielas al rol.
7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio, pero no para los roles vinculados a servicios.
   a. Abra la sección Configurar límite de permisos y, a continuación, elija Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo.

      IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta.
   b. Seleccione la política que desea utilizar para el límite de permisos.
8. Seleccione Siguiente.
9. Para Nombre del rol, las opciones varían según el servicio:
   - Si el servicio define el nombre del rol, no podrá editarlo.
   - Si el servicio define un prefijo para el nombre del rol, puede ingresar un sufijo opcional.
   - Si el servicio no define el nombre del rol, podrá nombrarlo usted mismo.

> **Important**
>
> Cuando asigne un nombre a un rol, tenga en cuenta lo siguiente:
>
> - Los nombres de rol deben ser únicos dentro de su Cuenta de AWS, y no se pueden hacer únicos mediante mayúsculas y minúsculas.

Por ejemplo, no puede crear roles denominados tanto **PRODROLE** como **prodrole**. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre de rol distingue entre mayúsculas y minúsculas, sin embargo, cuando un nombre de rol les aparece a los clientes en la consola, como por ejemplo durante el proceso de inicio de sesión, el nombre de rol no distingue entre mayúsculas y minúsculas.

- Dado que otras entidades podrían hacer referencia al rol, no es posible editar el nombre del rol una vez creado.

10. (Opcional) En Descripción, ingrese una descripción para el rol.
11. (Opcional) Para editar los casos de uso y los permisos de la función, en las secciones Paso 1: Seleccionar entidades confiables o en Paso 2: Agregar permisos, elija Editar.
12. (Opcional) Para ayudar a identificar, organizar o buscar el rol, agregue etiquetas como pares clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.
13. Revise el rol y, a continuación, elija Crear rol.

Creación de un rol para un servicio (AWS CLI)

Para crear un rol desde la AWS CLI se deben seguir varios pasos. Si utiliza la consola para crear un rol, muchos de los pasos se realizan automáticamente, pero con la AWS CLI deberá realizar cada paso usted mismo. Debe crear el rol y, a continuación, asignar una política de permisos al rol. Si el servicio con el que está trabajando es Amazon EC2 también deberá crear un perfil de instancia y agregarle el rol. Si lo prefiere, también puede configurar el límite de permisos para el rol.

Para crear un rol para un servicio de AWS desde la AWS CLI

1. Los siguientes comandos `create-role` crean un rol llamado Test-Role y le asigna una política de confianza:

   ```
   aws iam create-role --role-name Test-Role --assume-role-policy-document
   file://Test-Role-Trust-Policy.json
   ```

2. Asociar una política de permisos administrada al rol: aws iam attach-role-policy.

   Por ejemplo, el siguiente comando `attach-role-policy` adjunta la política administrada AWS denominada `ReadOnlyAccess` en el rol de IAM denominado `ReadOnlyRole`:

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
ReadOnlyAccess --role-name ReadOnlyRole
```

o

Crear una política de permisos insertada para el rol: aws iam put-role-policy

Para agregar una política de permisos insertada, consulte el siguiente ejemplo:

```
aws iam put-role-policy --role-name Test-Role --policy-name
ExamplePolicy --policy-document file://AdminPolicy.json
```

3. (Opcional) Añadir los atributos personalizados al rol asociando etiquetas: aws iam tag-role

   Para obtener más información, consulte Administrar etiquetas en roles de IAM (AWS CLI o API de AWS).

4. (Opcional) Configurar el límite de permisos para el rol: aws iam put-role-permissions-boundary

   Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

Si va a utilizar el rol con Amazon EC2 o con otro servicio de AWS que utiliza Amazon EC2, debe almacenar el rol en un perfil de instancias. Un perfil de instancias es un contenedor para un rol que se puede asociar a una instancia de Amazon EC2 cuando se lanza. Un perfil de instancia puede contener un único rol de y este límite no se puede aumentar. Si crea el rol con la AWS Management Console, el perfil de instancia se crea con el mismo nombre que el rol. Para obtener más información sobre los perfiles de instancia, consulte Uso de perfiles de instancia. Para obtener información sobre cómo lanzar una instancia EC2 con un rol, consulte Control del acceso a los recursos de Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux.

Para crear un perfil de instancia y almacenar el rol en él (AWS CLI)

1. Crear un perfil de instancia: aws iam create-instance-profile
2. Añadir el rol al perfil de instancia: aws iam add-role-to-instance-profile

En el siguiente ejemplo de conjunto de comandos de la AWS CLI, se muestran los dos primeros pasos para crear un rol y asociar permisos. También muestra los dos pasos para crear un perfil de instancia y añadir el rol al perfil. Esta política de confianza de ejemplo permite al servicio Amazon

EC2 asumir el rol y ver el bucket de `example_bucket` Amazon S3. El ejemplo también supone que se está utilizando un equipo cliente con Windows y que ya se ha configurado la interfaz de línea de comandos con las credenciales y región de la cuenta. Para obtener más información, consulte Configuración de la interfaz de línea de comandos de AWS.

En este ejemplo, incluya la siguiente política de confianza en el primer comando al crear el rol. Esta política de confianza permite que el servicio Amazon EC2 asuma el rol.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": "ec2.amazonaws.com"},
    "Action": "sts:AssumeRole"
  }
}
```

Cuando utilice el segundo comando, debe asociar una política de permisos al rol. La siguiente política de permisos de ejemplo permite al rol realizar únicamente la acción `ListBucket` en el bucket de Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Para crear el rol `Test-Role-for-EC2`, primero debe guardar la política de confianza anterior con el nombre `trustpolicyforec2.json` y la política de permisos anterior con el nombre `permissionspolicyforec2.json` en el directorio `policies` del disco duro local `C:`. A continuación, puede utilizar los siguientes comandos para crear el rol, asociar la política, crear el perfil de instancia y añadir el rol al perfil de instancia.

```
# Create the role and attach the trust policy that allows EC2 to assume this role.
$ aws iam create-role --role-name Test-Role-for-EC2 --assume-role-policy-document
 file://C:\policies\trustpolicyforec2.json
```

```
# Embed the permissions policy (in this example an inline policy) to the role to
 specify what it is allowed to do.
$ aws iam put-role-policy --role-name Test-Role-for-EC2 --policy-name Permissions-
Policy-For-Ec2 --policy-document file://C:\policies\permissionspolicyforec2.json

# Create the instance profile required by EC2 to contain the role
$ aws iam create-instance-profile --instance-profile-name EC2-ListBucket-S3

# Finally, add the role to the instance profile
$ aws iam add-role-to-instance-profile --instance-profile-name EC2-ListBucket-S3 --
role-name Test-Role-for-EC2
```

Al lanzar la instancia EC2, especifique el nombre de perfil de instancia en la página Configurar detalles de la instancia si utiliza la consola de AWS. Si utiliza el comando de la CLI `aws ec2 run-instances`, especifique el parámetro `--iam-instance-profile`.

Creación de un rol para un servicio (API de AWS)

Para crear un rol desde la API de AWS se deben seguir varios pasos. Si utiliza la consola para crear un rol, muchos de los pasos se realizan automáticamente, pero con la API deberá realizar cada paso usted mismo. Debe crear el rol y, a continuación, asignar una política de permisos al rol. Si el servicio con el que está trabajando es Amazon EC2 también deberá crear un perfil de instancia y agregarle el rol. Si lo prefiere, también puede configurar el límite de permisos para el rol.

Para crear un rol para un servicio de AWS (API de AWS)

1. Creación de un rol: CreateRole

   Para la política de confianza del rol, puede especificar una ubicación de archivo.
2. Asociar una política de permisos administrada al rol: AttachRolePolicy

   o

   Crear una política de permisos insertada para el rol: PutRolePolicy
3. (Opcional) Añadir los atributos personalizados al usuario asociando etiquetas: TagRole

   Para obtener más información, consulte Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS).
4. (Opcional) Configuración del límite de permisos para el rol: PutRolePermissionsBoundary

Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

Si va a utilizar el rol con Amazon EC2 o con otro servicio de AWS que utiliza Amazon EC2, debe almacenar el rol en un perfil de instancias. Un perfil de instancia es un contenedor para un rol. Cada perfil de instancia solo puede contener un único rol y dicho límite no se puede superar. Si crea el rol en la AWS Management Console, el perfil de instancia se crea con el mismo nombre que el rol. Para obtener más información sobre los perfiles de instancia, consulte Uso de perfiles de instancia. Para obtener información sobre cómo lanzar una instancia de Amazon EC2 con un rol, consulte Control del acceso a los recursos de Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux.

Para crear un perfil de instancia y almacenar el rol en él (API de AWS)

1. Crear un perfil de instancia: CreateInstanceProfile
2. Añadir el rol al perfil de instancia: AddRoleToInstanceProfile

Creación de un rol para un proveedor de identidad de terceros (federación)

Puede utilizar proveedores de identidad en lugar de crear usuarios de IAM en una Cuenta de AWS. Con un proveedor de identidad (IdP), puede administrar sus identidades de usuario fuera de AWS y conceder permisos a estas identidades de usuarios externos para que tengan acceso a los recursos de AWS de su cuenta. Para obtener más información acerca de la identidad federada y los proveedores de identidad, consulte Federación y proveedores de identidades.

Creación de un rol para usuarios federados (consola)

Los procedimientos que ha de seguir para crear un rol para los usuarios federados dependen de su elección de proveedores de terceros:

- Para conexiones OpenID Connect (OIDC), consulte Creación de un rol para una federación de OpenID Connect (consola).
- Para SAML 2.0, consulte Creación de un rol para una federación SAML 2.0 (consola).

Creación de un rol para acceso federado (AWS CLI)

Los pasos que ha de seguir para crear un rol para los proveedores de identidad compatibles (OIDC o SAML) desde la AWS CLI son idénticos. La diferencia está en el contenido de la política de confianza que crea en los pasos de requisitos previos. Empiece siguiendo los pasos de la sección Requisitos previos para el tipo de proveedor que utilice:

- Para un proveedor OIDC, consulte Requisitos previos para crear un rol para OIDC.
- Para un proveedor SAML, consulte Requisitos previos para crear un rol para SAML.

Para crear un rol desde la AWS CLI se deben seguir varios pasos. Si utiliza la consola para crear un rol, muchos de los pasos se realizan automáticamente, pero con la AWS CLI deberá realizar cada paso usted mismo. Debe crear el rol y, a continuación, asignar una política de permisos al rol. Si lo prefiere, también puede configurar el límite de permisos para el rol.

Para crear un rol para la identidad federada (AWS CLI)

1. Crear un rol: aws iam create-role
2. Asociar una política de permisos al rol: aws iam attach-role-policy

   o

   Crear una política de permisos insertada para el rol: aws iam put-role-policy
3. (Opcional) Añadir los atributos personalizados al rol asociando etiquetas: aws iam tag-role

   Para obtener más información, consulte Administrar etiquetas en roles de IAM (AWS CLI o API de AWS).
4. (Opcional) Configurar el límite de permisos para el rol: aws iam put-role-permissions-boundary

   Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

El siguiente ejemplo muestra los dos primeros pasos, que también son los más comunes, para crear un rol de proveedor de identidad en un entorno sencillo. Este ejemplo permite a cualquier usuario de la cuenta `123456789012` asumir el rol y ver el bucket de `example_bucket` Amazon S3. En el ejemplo, también se presupone que está ejecutando la AWS CLI en un equipo con Windows y que ya ha configurado la AWS CLI con sus credenciales. Para obtener más información, consulte Configuración de la AWS Command Line Interface.

En el ejemplo siguiente, la política de confianza está diseñada para una aplicación móvil si el usuario inicia sesión mediante Amazon Cognito. En este ejemplo, *us-east:12345678-ffff-ffff-ffff-123456* representa el ID del grupo de identidades asignado por Amazon Cognito.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "RoleForCognito",
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
    }
}
```

La siguiente política de permisos permite a cualquiera que asuma el rol realizar únicamente la acción `ListBucket` en el bucket de Amazon S3 `example_bucket`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

Para crear el rol `Test-Cognito-Role`, primero debe guardar la política de confianza anterior con el nombre `trustpolicyforcognitofederation.json` y la política de permisos anterior con el nombre `permspolicyforcognitofederation.json` en la carpeta `policies` del disco duro local `C:`. A continuación, puede utilizar los comandos siguientes para crear el rol y asociarle la política insertada.

```
# Create the role and attach the trust policy that enables users in an account to
 assume the role.
$ aws iam create-role --role-name Test-Cognito-Role --assume-role-policy-document
 file://C:\policies\trustpolicyforcognitofederation.json

# Attach the permissions policy to the role to specify what it is allowed to do.
```

```
aws iam put-role-policy --role-name Test-Cognito-Role --policy-name
 Perms-Policy-For-CognitoFederation --policy-document file://C:\policies
\permspolicyforcognitofederation.json
```

Creación de un rol para acceso federado (API de AWS)

Los pasos que ha de seguir para crear un rol para los proveedores de identidad compatibles (OIDC o SAML) desde la AWS CLI son idénticos. La diferencia está en el contenido de la política de confianza que crea en los pasos de requisitos previos. Empiece siguiendo los pasos de la sección Requisitos previos para el tipo de proveedor que utilice:

- Para un proveedor OIDC, consulte Requisitos previos para crear un rol para OIDC.
- Para un proveedor SAML, consulte Requisitos previos para crear un rol para SAML.

Para crear un rol para la identidad federada (API de AWS)

1. Creación de un rol: CreateRole
2. Asociar una política de permisos al rol: AttachRolePolicy

   o

   Crear una política de permisos insertada para el rol: PutRolePolicy
3. (Opcional) Añadir los atributos personalizados al usuario asociando etiquetas: TagRole

   Para obtener más información, consulte Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS).
4. (Opcional) Configuración del límite de permisos para el rol: PutRolePermissionsBoundary

   Un límite de permisos controla los permisos que puede tener un rol como máximo. Los límites de permisos son una característica avanzada de AWS.

Creación de un rol para una federación de OpenID Connect (consola)

Puede utilizar proveedores de identidad federados de OpenID Connect (OIDC) en lugar de crear usuarios de AWS Identity and Access Management en la Cuenta de AWS. Con un proveedor de identidad (IdP), puede administrar sus identidades de usuario fuera de AWS y conceder permisos a estas identidades de usuarios externos para que tengan acceso a los recursos de AWS de su

cuenta. Para obtener más información acerca de la federación y los IdP, consulte Federación y proveedores de identidades.

Requisitos previos para crear un rol para OIDC

Para poder crear un rol para federación de OIDC, antes debe completar los siguientes pasos de los requisitos previos.

Preparativos para crear un rol para la federación de OIDC

1. Regístrese con uno o más servicios que ofrezcan identidad de OIDC federada. Si está creando una aplicación que necesita obtener acceso a los recursos de AWS, también deberá configurarla con la información del proveedor. Cuando lo haga, el proveedor le proporcionará un ID de aplicación o de público exclusivo de la aplicación. (Cada proveedor utiliza una terminología diferente para este proceso. En esta guía se utiliza el término configurar para el proceso de identificación de su aplicación con el proveedor). Puede configurar varias aplicaciones con cada proveedor o varios proveedores con una sola aplicación. Consulte la información sobre el uso de los proveedores de identidades de la siguiente manera:

   - Login with Amazon Developer Center
   - Añadir inicio de sesión con Facebook a su aplicación o sitio web en el sitio de desarrolladores de Facebook.
   - Uso de OAuth 2.0 para iniciar sesión (OpenID Connect) en el sitio de desarrolladores de Google.

2. Después de recibir la información necesaria del IdP, cree un IdP en IAM. Para obtener más información, consulte Crear un proveedor de identidad de IAM OpenID Connect (OIDC).

   > **Important**
   >
   > Si utiliza un IdP de OIDC de Google, Facebook o Amazon Cognito, no cree un IdP de IAM independiente en el AWS Management Console. Estos proveedores de identidades de OIDC ya están integrados en AWS y están disponibles para su uso. Omita este paso y cree nuevos roles con su IdP en el paso siguiente.

3. Prepare las políticas para el rol que los usuarios autenticados mediante el proveedor de identidades asumirán. Como sucede con cualquier otro rol, un rol para una aplicación móvil incluye dos políticas. Una es la política de confianza que especifica quién puede asumir el rol.

La otra es la política de permisos que especifica las acciones y los recursos de AWS a los que la aplicación móvil puede obtener acceso o se le deniega.

En el caso de IdP web, le recomendamos que utilice Amazon Cognito para administrar las identidades. En este caso, utilice una política de confianza similar a la de este ejemplo.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Principal": {"Federated": "cognito-identity.amazonaws.com"},
        "Action": "sts:AssumeRoleWithWebIdentity",
        "Condition": {
            "StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east-2:12345678-abcd-abcd-abcd-123456"},
            "ForAnyValue:StringLike": {"cognito-identity.amazonaws.com:amr":
 "unauthenticated"}
        }
    }
}
```

Reemplace `us-east-2:12345678-abcd-abcd-abcd-123456` por el ID del grupo de identidades que Amazon Cognito le ha asignado.

Si configura de manera manual un proveedor de identidad OIDC (IdP), al crear la política de confianza debe utilizar tres valores que garanticen que únicamente su aplicación puede asumir el rol:

- En el elemento `Action`, utilice la acción `sts:AssumeRoleWithWebIdentity`.
- En el elemento `Principal`, utilice la cadena `{"Federated":`*`providerUrl/providerArn`*`}`.
  - En el caso de algunos IdP de OIDC conocidos, el *`providerUrl`* es una URL. En los siguientes ejemplos se incluyen métodos que permiten especificar la entidad principal para algunos de estos proveedores de identidad:

    `"Principal":{"Federated":"cognito-identity.amazonaws.com"}`

    `"Principal":{"Federated":"www.amazon.com"}`

    `"Principal":{"Federated":"graph.facebook.com"}`

```
"Principal":{"Federated":"accounts.google.com"}
```

- Para los demás proveedores de OIDC, utilice el nombre de recurso de Amazon (ARN) del proveedor de identidades de OIDC que ha creado en el Step 2, como se muestra en el siguiente ejemplo:

  ```
  "Principal":{"Federated":"arn:aws:iam::123456789012:oidc-provider/
  server.example.com"}
  ```

- En el elemento `Condition`, utilice una condición `StringEquals` para limitar los permisos. Pruebe el ID del grupo de identidades (para Amazon Cognito) o el ID de aplicación (para otros proveedores). El ID del grupo de identidades debe coincidir con el ID de la aplicación que ha recibido al configurarla con el IdP. Esta coincidencia entre los ID asegura que la solicitud provenga de su aplicación.

  > **Note**
  >
  > Una política de confianza de roles que confíe en los grupos de identidades (`cognito-identity.amazonaws.com`) de Amazon Cognito debe contener al menos una clave de condición para limitar el número de entidades principales que puede asumir el rol. Para obtener más información, consulte Políticas de confianza para roles de IAM en la autenticación básica (clásica) en la Guía para desarrolladores de Amazon Cognito.

  Cree un elemento de condición similar a uno de los ejemplos siguientes, en función del IdP que esté utilizando:

  ```
  "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud":
  "us-east:12345678-ffff-ffff-ffff-123456"}}
  ```

  ```
  "Condition": {"StringEquals": {"www.amazon.com:app_id":
  "amzn1.application-oa2-123456"}}
  ```

  ```
  "Condition": {"StringEquals": {"graph.facebook.com:app_id":
  "111222333444555"}}
  ```

  ```
  "Condition": {"StringEquals": {"accounts.google.com:aud":
  "666777888999000pro0"}}
  ```

En el caso de los proveedores OIDC, utilice la dirección URL completa del proveedor de identidad OIDC con la clave de contexto aud, como se muestra en el siguiente ejemplo:

```
"Condition": {"StringEquals": {"server.example.com:aud":
"appid_from_oidc_idp"}}
```

> **Note**
>
> Observe que los valores de la entidad principal de la política de confianza del rol son específicos de un IdP. Un rol para OIDC puede especificar solo una entidad principal. Por lo tanto, si la aplicación móvil permite a los usuarios iniciar sesión desde varios IdP, debe crear un rol independiente para cada IdP que desee admitir. Cree políticas de confianza independientes para cada IdP.

Si un usuario utiliza una aplicación móvil para iniciar sesión desde Login with Amazon, se aplicará el ejemplo siguiente de política de confianza. En el ejemplo, *amzn1.application-oa2-123456* representa el ID de la aplicación que Amazon asignó cuando configuró la aplicación con Login with Amazon.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForLoginWithAmazon",
          "Effect": "Allow",
          "Principal": {"Federated": "www.amazon.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"www.amazon.com:app_id":
 "amzn1.application-oa2-123456"}}
      }]
  }
```

Si un usuario utiliza una aplicación móvil para iniciar sesión desde Facebook, se aplicará el ejemplo siguiente de política de confianza. En este ejemplo, *111222333444555* representa el ID de la aplicación asignado por Facebook.

```
{
```

```
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForFacebook",
          "Effect": "Allow",
          "Principal": {"Federated": "graph.facebook.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"graph.facebook.com:app_id":
 "111222333444555"}}
      }]
  }
```

Si un usuario utiliza una aplicación móvil para iniciar sesión desde Google, se aplicará el ejemplo siguiente de política de confianza. En este ejemplo, *666777888999000* representa el ID de la aplicación asignado por Google.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForGoogle",
          "Effect": "Allow",
          "Principal": {"Federated": "accounts.google.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
          "Condition": {"StringEquals": {"accounts.google.com:aud":
 "666777888999000"}}
      }]
  }
```

Si un usuario utiliza una aplicación móvil para iniciar sesión desde Amazon Cognito, se aplicará el ejemplo siguiente de política de confianza. En este ejemplo, *us-east:12345678-ffff-ffff-ffff-123456* representa el ID del grupo de identidades asignado por Amazon Cognito.

```
{
      "Version": "2012-10-17",
      "Statement": [{
          "Sid": "RoleForCognito",
          "Effect": "Allow",
          "Principal": {"Federated": "cognito-identity.amazonaws.com"},
          "Action": "sts:AssumeRoleWithWebIdentity",
```

```
          "Condition": {"StringEquals": {"cognito-identity.amazonaws.com:aud": "us-
east:12345678-ffff-ffff-ffff-123456"}}
      }]
  }
```

Creación de un rol para OIDC

Después de completar los requisitos previos, puede crear el rol en IAM. En el siguiente procedimiento se describe cómo crear el rol de federación de OIDC en la AWS Management Console. Para crear un rol desde la AWS CLI o la API de AWS, consulte los procedimientos en Creación de un rol para un proveedor de identidad de terceros (federación).

> **Important**
>
> Si utiliza Amazon Cognito, debe utilizar la consola de Amazon Cognito para configurar los roles. De lo contrario, utilice la consola de IAM para crear un rol para la federación de OIDC.

Cómo crear un rol de IAM para una federación de OIDC

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y, a continuación, seleccione Crear rol.
3. Elija el tipo de rol OIDC.
4. En Proveedor de identidades, elija el proveedor de identidades para el rol:
   - Si está creando un rol para un IdP web individual, elija Login with Amazon, Facebook o Google.

     > **Note**
     >
     > Debe crear un rol independiente para cada IdP al que desee admitir.

   - Si está creando un rol de situación avanzada para Amazon Cognito, elija Amazon Cognito.

> **Note**
>
> Solo deberá crear manualmente un rol para utilizarlo con Amazon Cognito si está trabajando en una situación avanzada. En caso contrario, Amazon Cognito puede crear roles de forma automática. Para obtener más información sobre Amazon Cognito, consulte Proveedores de identidades externos de grupos de identidad (identidades federadas) en la Guía para desarrolladores de Amazon Cognito.

- Si quiere crear un rol para GitHub Actions, primero debe agregar el proveedor OIDC de GitHub a IAM. Después de agregar el proveedor OIDC de GitHub a IAM, elija token.actions.githubusercontent.com.

> **Note**
>
> Para obtener información acerca de cómo configurar AWS para confiar en el OIDC de GitHub como una identidad federada, consulte GitHub Docs - Configuring OpenID Connect in Amazon Web Services (GitHub Docs: configuración de OpenID Connect en Amazon Web Services). Para obtener información sobre las mejores prácticas para limitar el acceso a los roles asociados al IdP de IAM para GitHub, consulte Configuración de un rol para el proveedor de identidades de OIDC de GitHub en esta página.

5. Ingrese el identificador de su aplicación. La etiqueta del identificador cambia en función del proveedor que elija:

   - Si está creando un rol para Login with Amazon, ingrese el ID de la aplicación en el cuadro ID de aplicación.
   - Si está creando un rol para Facebook, ingrese el ID de la aplicación en el cuadro ID de aplicación.
   - Si está creando un rol para Google, ingrese el nombre de los destinatarios en el cuadro Audience (Público).
   - Si está creando un rol para Amazon Cognito, ingrese el ID del grupo de identidades que ha creado para sus aplicaciones de Amazon Cognito en el campo Identity Pool ID (ID de grupo de identidades).
   - Si quiere crear un rol para GitHub Actions, introduzca los siguientes detalles:
     - En Audiencia, elija `sts.amazonaws.com`.

- Para Organización de GitHub, introduzca el nombre de su organización de GitHub. El nombre de la organización de GitHub es obligatorio y debe ser alfanumérico, incluyendo guiones (-). No se pueden usar caracteres comodín (* y?) en el nombre de la organización de GitHub.
- (Opcional) Para el repositorio GitHub, introduzca el nombre del repositorio GitHub. Si no especifica un valor, se utilizará por defecto un comodín (*).
- (Opcional) Para la ramificación GitHub, introduzca el nombre de la ramificación GitHub. Si no especifica un valor, se utilizará por defecto un comodín (*).

6. (Opcional) Para Condición (opcional), elija Añadir condición para crear condiciones adicionales que deben cumplirse antes de que los usuarios de su aplicación puedan utilizar los permisos que el rol les concede. Por ejemplo, puede agregar una condición que conceda acceso a recursos de AWS únicamente a un ID de usuario IAM concreto. También puede añadir condiciones a la política de confianza después de crear el rol. Para obtener más información, consulte Modificación de una política de confianza de rol (consola).
7. Revise la información de OIDC y, a continuación, seleccione Siguiente.
8. IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente en su cuenta. Seleccione la política que desea utilizar como política de permisos o elija Create policy (Crear política) para abrir una pestaña nueva del navegador y crear una política nueva desde cero. Para obtener más información, consulte Crear políticas de IAM. Después de crear la política, cierre esa pestaña y vuelva a la pestaña original. Seleccione la casilla de verificación situada junto a las políticas de permisos que desea conceder a los usuarios de OIDC. Si lo prefiere, puede optar por no seleccionar ninguna política en ese momento y asociar las políticas al rol más adelante. De forma predeterminada, un rol no tiene permisos.
9. (Opcional) Configure un límite de permisos. Esta es una característica avanzada.

   Abra la sección Permissions boundary (Límite de permisos) y elija Use a permissions boundary to control the maximum role permissions (Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo). Seleccione la política que desea utilizar para el límite de permisos.
10. Elija Siguiente.
11. En Nombre de rol, ingrese un nombre de rol. Los nombres de rol deben ser únicos en su Cuenta de AWS. No dependen de los casos. Por ejemplo, no puede crear funciones denominadas tanto **`PRODROLE`** como **`prodrole`**. Dado que es posible que otros recursos de AWS hagan referencia al rol, no se puede editar el nombre del rol después de crearlo.
12. (Opcional) En Description (Descripción), ingrese una descripción para el nuevo rol.

13. Para editar los casos de uso y los permisos de la función, elija Edit (Editar) en las secciones Step 1: Select trusted entities (Paso 1: seleccionar entidades de confianza) o Step 2: Add permissions (Paso 2: agregar permisos).
14. (Opcional) Para agregar metadatos al rol, asocie etiquetas como pares clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
15. Revise el rol y, a continuación, seleccione Crear rol.

Configuración de un rol para el proveedor de identidades de OIDC de GitHub

Si usa GitHub como un proveedor de identidades (IdP) de OIDC, la práctica recomendada es limitar las entidades que pueden asumir el rol asociado con el IdP de IAM. Cuando incluyes una declaración de condición en la política de confianza, puedes limitar el rol a una organización, un repositorio o una rama de GitHub específica. Puede usarse la clave de condición `token.actions.githubusercontent.com:sub` con operadores de condición de cadena para limitar el acceso. Le recomendamos que limite la condición a un conjunto específico de repositorios o ramas dentro de su organización GitHub. Para obtener información acerca de cómo configurar AWS para confiar en el OIDC de GitHub como una identidad federada, consulte GitHub Docs: configuración de OpenID Connect en Amazon Web Services.

Si utiliza los entornos de GitHub en los flujos de trabajo de acción o en las políticas de OIDC, recomendamos agregar reglas de protección al entorno para una mayor seguridad. Utilice las ramas y las etiquetas de despliegue para restringir qué ramas y etiquetas se pueden implementar en el entorno. Para más información sobre la configuración de entornos con reglas de protección, consulte Ramas y etiquetas de despliegue en el artículo Uso de entornos para el despliegue de GitHub.

Cuando el IdP OIDC de GitHub es la entidad principal de confianza para su rol, IAM comprueba la condición de la política de confianza del rol para verificar que la clave de condición `token.actions.githubusercontent.com:sub` está presente y su valor no es únicamente un carácter comodín (* y?) o nulo. IAM realiza esta comprobación cuando se crea o actualiza la política de confianza. Si la clave de condición `token.actions.githubusercontent.com:sub` no está presente o el valor de la clave no cumple los criterios de valor mencionados, la solicitud fallará y devolverá un error.

> **Important**
>
> Si no limita la clave de condición `token.actions.githubusercontent.com:sub` a una organización o repositorio específicos, GitHub Actions de organizaciones o repositorios fuera

de su control pueden asumir roles asociados con el IdP de IAM de GitHub en su cuenta de AWS.

El siguiente ejemplo de política de confianza limita el acceso a la organización, el repositorio y la rama de GitHub definidos. El valor de la clave de condición `token.actions.githubusercontent.com:sub` del siguiente ejemplo es el formato de valor del asunto predeterminado documentado por GitHub.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::012345678910:oidc-provider/
token.actions.githubusercontent.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
          "token.actions.githubusercontent.com:sub":
 "repo:GitHubOrg/GitHubRepo:ref:refs/heads/GitHubBranch"
        }
      }
    }
  ]
}
```

La siguiente condición de ejemplo limita el acceso a la organización y el repositorio de GitHub definidos, pero otorga acceso a cualquier rama del repositorio.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/GitHubRepo:*"
  }
}
```

La siguiente condición de ejemplo limita el acceso a cualquier rama o repositorio de la organización de GitHub definida. Le recomendamos que limite la clave de condición `token.actions.githubusercontent.com:sub` a un valor específico que limite el acceso a GitHub Actions desde dentro de su organización de GitHub.

```
"Condition": {
  "StringEquals": {
          "token.actions.githubusercontent.com:aud": "sts.amazonaws.com"
  },
  "StringLike": {
    "token.actions.githubusercontent.com:sub": "repo:GitHubOrg/*"
  }
}
```

Para obtener más información sobre las claves de federación de OIDC disponibles para verificaciones de condición en las políticas, consulte Claves disponibles para las federaciones de identidades AWS de OIDC.

Creación de un rol para una federación SAML 2.0 (consola)

Puede utilizar la federación SAML 2.0 en lugar de crear usuarios de IAM en una Cuenta de AWS. Con un proveedor de identidad (IdP), puede administrar sus identidades de usuario fuera de AWS y conceder permisos a estas identidades de usuarios externos para que tengan acceso a los recursos de AWS de su cuenta. Para obtener más información acerca de la identidad federada y los proveedores de identidad, consulte Federación y proveedores de identidades.

> **Note**
>
> Para mejorar la resiliencia de la federación, le recomendamos que configure su IdP y su federación de AWS para que admitan varios puntos de conexión de inicio de sesión de SAML. Para obtener más información, consulte el artículo del blog sobre seguridad de AWS, How to use regional SAML endpoints for failover.

Requisitos previos para crear un rol para SAML

Para poder crear un rol de federación de SAML 2.0, antes debe completar los siguientes pasos de requisitos previos.

Preparativos para crear un rol para la federación SAML 2.0

1. Para poder crear un rol para una federación basada en SAML, debe crear un proveedor de SAML en IAM. Para obtener más información, consulte Crear un proveedor de identidades de SAML en IAM.
2. Prepare las políticas del rol que los usuarios autenticados por SAML 2.0 asumirán. Al igual que ocurre con cualquier otro rol, un rol para la federación SAML incluye dos políticas. Una es la política de confianza de rol que especifica quién puede asumir el rol. La otra es la política de permisos de IAM que especifica las acciones y los recursos de AWS a los que el usuario federado puede obtener acceso o se le deniega.

   Al crear la política de confianza para el rol, debe utilizar tres valores que garantizan que solo la aplicación pueda asumir el rol:

   - En el elemento `Action`, utilice la acción `sts:AssumeRoleWithSAML`.
   - En el elemento `Principal`, utilice la cadena `{"Federated":`*`ARNofIdentityProvider`*`}`. Sustituya *`ARNofIdentityProvider`* por el ARN del proveedor de identidad SAML que ha creado en Step 1.
   - En el elemento `Condition`, utilice una condición `StringEquals` para probar que el atributo `saml:aud` de la respuesta de SAML coincida con el punto de enlace de la federación SAML para AWS.

   La política de confianza del ejemplo siguiente está diseñada para un usuario federado de SAML:

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
         "Effect": "Allow",
         "Action": "sts:AssumeRoleWithSAML",
         "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-
   NAME"},
         "Condition": {"StringEquals": {"SAML:aud": "https://signin.aws.amazon.com/
   saml"}}
       }
     }
   ```

   Sustituya el ARN de la entidad principal por el ARN real del proveedor SAML que ha creado en IAM. Tendrá su ID de cuenta y su nombre de proveedor.

Creación de un rol para SAML

Después de completar los pasos de los requisitos previos, puede a crear el rol para la federación basada en SAML.

Para crear un rol para la federación basada en SAML

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles y, a continuación, elija Crear rol.
3. Elija el tipo de rol SAML 2.0 federation.
4. En Select a SAML provider (Seleccionar un proveedor de SAML), elija el proveedor para el rol.
5. Elija el método de nivel de acceso SAML 2.0.

   - Elija Allow programmatic access only (Permitir solo acceso mediante programación) para crear un rol que se pueda asumir mediante programación desde la API o la AWS CLI de AWS.
   - Elija Permitir el acceso AWS Management Console mediante la consola y mediante programación para crear un rol que pueda ser asumido mediante programación y desde la AWS Management Console.

   Los roles creados con ambas opciones son similares, pero el rol que puede ser asumido desde la consola incluye una política de confianza con una condición particular. Dicha condición garantiza explícitamente que el público de SAML (atributo `SAML:aud`) esté establecido en el punto de conexión de inicio de sesión de AWS para SAML (https://signin.aws.amazon.com/saml).
6. Si está creando un rol para el acceso mediante programación, elija un atributo en la lista Attribute (Atributo). A continuación, en el cuadro Value (Valor), ingrese un valor para incluirlo en el rol. Esto restringe el acceso del rol a los usuarios del proveedor de identidad cuya respuesta de autenticación SAML (aserción) incluya los atributos que especifique. Debe especificar al menos un atributo para garantiza que el rol esté limitado a un subconjunto de usuarios de su organización.

   Si está creando un rol para acceso programado y desde la consola, el atributo `SAML:aud` se añade y se establece automáticamente en la URL del punto de enlace de SAML de AWS (https://signin.aws.amazon.com/saml).

7. Para agregar más condiciones relacionadas con el atributo a la política de confianza, elija Condition (optional) (Condición [opcional]), seleccione la condición adicional y especifique un valor.

   > **Note**
   >
   > La lista incluye los atributos SAML utilizados con más frecuencia. IAM admite atributos adicionales que puede utilizar para crear condiciones. Para ver una lista de los atributos admitidos, consulte Claves disponibles para federaciones SAML. Si necesita una condición para un atributo SAML admitido que no se muestra en la lista, puede añadir manualmente dicha condición. Para ello, edite la política de confianza después de crear el rol.

8. Revise la información de confianza de SAML 2.0 y, a continuación, elija Next (Siguiente).
9. IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente en su cuenta. Seleccione la política que desea utilizar como política de permisos o elija Create policy (Crear política) para abrir una pestaña nueva del navegador y crear una política nueva desde cero. Para obtener más información, consulte Crear políticas de IAM. Después de crear la política, cierre esa pestaña y vuelva a la pestaña original. Seleccione la casilla de verificación situada junto a las políticas de permisos que desea conceder a los usuarios federados de OIDC. Si lo prefiere, puede optar por no seleccionar ninguna política en este momento y asociar las políticas al rol más adelante. De forma predeterminada, un rol no tiene permisos.
10. (Opcional) Configure un límite de permisos. Esta es una característica avanzada.

    Abra la sección Permissions boundary (Límite de permisos) y elija Use a permissions boundary to control the maximum role permissions (Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo). Seleccione la política que desea utilizar para el límite de permisos.
11. Elija Siguiente.
12. Elija Siguiente: Revisar.
13. En Nombre de rol, ingrese un nombre de rol. Los nombres de rol deben ser únicos en su Cuenta de AWS. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear funciones denominado tanto **`PRODROLE`** como **`prodrole`**. Dado que es posible que otros recursos de AWS hagan referencia al rol, no se puede editar el nombre del rol después de crearlo.
14. (Opcional) En Description (Descripción), ingrese una descripción para el nuevo rol.

15. Elija Edit (Editar) en las secciones Step 1: Select trusted entities (Paso 1: seleccionar entidades de confianza) o Step 2: Add permissions (Paso 2: agregar permisos) para editar los casos de uso y los permisos del rol.
16. De manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
17. Revise el rol y, a continuación, seleccione Crear rol.

Después de crear el rol, complete la relación de confianza de SAML configurando su software de proveedor de identidad con información sobre AWS. Esta información incluye los roles que desea que utilicen los usuarios federados. Esto se denomina configuración de la relación de confianza entre su proveedor de identidad y AWS. Para obtener más información, consulte Configuración su SAML 2.0 IdP con una relación de confianza para usuario autenticado y agregando reclamos.

Creación de un rol mediante políticas de confianza personalizadas (consola)

Puede crear una política de confianza personalizada para delegar el acceso y permitir que otros realicen acciones en su Cuenta de AWS. Para obtener más información, consulte Crear políticas de IAM.

Para obtener información sobre cómo utilizar los roles para delegar permisos, consulte Términos y conceptos de roles.

Creación de un rol de IAM mediante políticas de confianza personalizadas (consola)

Puede utilizar la AWS Management Console para crear un rol que un usuario de IAM pueda asumir. Por ejemplo, suponga que su organización tiene varias Cuentas de AWS para aislar un entorno de desarrollo de uno de producción. Para información general sobre cómo crear un rol que permita a usuarios de la cuenta de desarrollo acceder a los recursos de la cuenta de producción, consulte Situación de ejemplo en la que se usan cuentas de desarrollo y producción separadas.

Para crear un rol mediante políticas de confianza personalizadas (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Roles y, a continuación, seleccione Create role (Crear rol).
3. Elija el tipo de rol Custom trust policy (Política de confianza personalizada).

4. En la sección Custom trust policy (Política de confianza personalizada), ingrese o pegue la política de confianza personalizada para el rol. Para obtener más información, consulte Crear políticas de IAM.
5. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.
6. Seleccione la casilla situada junto a la política de confianza personalizada que ha creado.
7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio, pero no para los roles vinculados a servicios.

   Abra la sección Permissions boundary (Límite de permisos) y elija Use a permissions boundary to control the maximum role permissions (Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo). IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta. Seleccione la política que desea utilizar para el límite de permisos.
8. Elija Next (Siguiente).
9. En Role Name (Nombre del rol), el servicio define el grado de personalización del nombre del rol. Si el servicio define el nombre del rol, esta opción no es editable. En otros casos, el servicio puede definir un prefijo para el rol y permitirle escribir un sufijo opcional. Algunos servicios le permiten especificar el nombre completo de su rol.

   Si es posible, ingrese un nombre de rol o un sufijo de nombre de rol. Los nombres de rol deben ser únicos en su Cuenta de AWS. No distinguen entre mayúsculas y minúsculas. Por ejemplo, no puede crear funciones denominado tanto **`PRODROLE`** y **`prodrole`**. Dado que es posible que otros recursos de AWS hagan referencia al rol, no se puede editar el nombre del rol después de crearlo.
10. (Opcional) En Description (Descripción), ingrese una descripción para el nuevo rol.
11. Elija Edit (Editar) en las secciones Step 1: Select trusted entities (Paso 1: seleccionar entidades de confianza) o Step 2: Add permissions (Paso 2: agregar permisos) para editar la política personalizada y los permisos del rol.
12. De manera opcional, agregue metadatos al rol asociando etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
13. Revise el rol y, a continuación, seleccione Create role.

Ejemplos de políticas para delegar el acceso

En los siguientes ejemplos se muestra cómo puede permitir o conceder acceso a una Cuenta de AWS a los recursos de otra Cuenta de AWS. Para obtener información sobre cómo crear una política de IAM mediante estos documentos de políticas JSON de ejemplo, consulte the section called "Creación de políticas mediante el editor JSON".

Temas

- Uso de roles para delegar el acceso a otros recursos de la Cuenta de AWS
- Uso de una política para delegar el acceso a los servicios
- Uso de una política basada en recursos para delegar el acceso a un bucket de Amazon S3 de otra cuenta
- Uso de una política basada en recursos para delegar el acceso a una cola de Amazon SQS de otra cuenta
- No se puede delegar el acceso cuando la cuenta tiene el acceso denegado

Uso de roles para delegar el acceso a otros recursos de la Cuenta de AWS

Para ver un tutorial que muestra cómo utilizar los roles de IAM para conceder a los usuarios de una cuenta acceso a los recursos de AWS que se encuentran en otra cuenta, consulte Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.

> **Important**
>
> Puede incluir el ARN de un rol o usuario específico en el elemento `Principal` de una política de confianza de rol. Al guardar la política, AWS transforma el ARN a un ID exclusivo de entidad principal. Esto ayuda a mitigar el riesgo de que alguien aumente sus privilegios eliminando o volviendo a crear el rol o usuario. Normalmente este ID no se muestra en la consola, ya que también existe una transformación inversa al ARN cuando se muestra la política de confianza. Sin embargo, si elimina el rol o el usuario, la relación se desvincula. La política ya no se aplica, incluso si vuelva a crear el usuario o rol, ya que no coincide con el ID principal almacenado en la política de confianza. Cuando esto sucede, el ID principal se muestra en la consola, ya que AWS no puede volver a asignarlo a un ARN. El resultado es que si elimina y vuelve a crear un usuario o rol al que se hace referencia en un elemento `Principal` de la política de confianza, debe editar el rol para sustituir el ARN. Se transforma en el nuevo ID de entidad principal al guardar la política.

Uso de una política para delegar el acceso a los servicios

En el siguiente ejemplo se muestra una política que puede asociarse a un rol. La política permite que dos servicios, Amazon EMR y AWS Data Pipeline asuman el rol. Los servicios pueden realizar las tareas concedidas por la política de permisos asignada al rol (no se muestra). Para especificar varios elementos principales del servicio, no debe especificar dos elementos `Service`; solo puede tener uno. En cambio, utilice una gama de varios elementos principales del servicio como el valor de un único elemento `Service`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "elasticmapreduce.amazonaws.com",
          "datapipeline.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Uso de una política basada en recursos para delegar el acceso a un bucket de Amazon S3 de otra cuenta

En este ejemplo, la cuenta A utiliza una política basada en recursos (una política de bucket de Amazon S3) para conceder a la cuenta B acceso completo al bucket de S3 de la cuenta A. A continuación, la cuenta B crea una política de usuario de IAM para delegar el acceso del bucket de la cuenta A a uno de los usuarios de la cuenta B.

La política de bucket de S3 de la cuenta A podría ser como la siguiente política. En este ejemplo, el bucket de S3 de la cuenta A se denomina mybucket y el número de la cuenta B es 111122223333. No se especifica los usuarios individuales ni grupos de la cuenta B, solo la cuenta.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
```

```
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

De forma alternativa, la cuenta A puede utilizar las Listas de control de acceso (ACL) de Amazon S3 para conceder a la cuenta B acceso a un bucket de S3 o a un único objeto de un bucket. En tal caso, lo único que cambia es cómo la cuenta A concede acceso a la cuenta B. La cuenta B todavía utiliza una política para delegar el acceso a un grupo de IAM de la cuenta B, tal y como se describe en la próxima sección de este ejemplo. Para obtener más información sobre el control del acceso a los buckets y objetos de S3, vaya a Control de acceso en la Guía del usuario de Amazon Simple Storage Service.

El administrador de la cuenta B puede crear la siguiente muestra de política. La política permite el acceso de lectura a un grupo o usuario de la cuenta B. La política anterior concede acceso a la cuenta B. Sin embargo, los grupos y usuarios individuales de la cuenta B no pueden obtener acceso al recurso hasta que una política de grupo o usuario conceda de forma explícita los permisos al recurso. Los permisos de esta política solo pueden ser un subconjunto de los de la anterior política entre cuentas. La cuenta B no puede conceder más permisos a sus grupos y usuarios que los que la cuenta A concedió a la cuenta B en la primera política. En esta política, el elemento `Action` se define de forma explícita para permitir únicamente acciones `List` y el elemento `Resource` de esta política coincide con `Resource` para la política de bucket implementada por la cuenta A.

Para aplicar esta política, la cuenta B utiliza IAM para asociarla al usuario adecuado (o grupo) de la cuenta B.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
```

```
  }
}
```

Uso de una política basada en recursos para delegar el acceso a una cola de Amazon SQS de otra cuenta

En el siguiente ejemplo, la cuenta A tiene una cola de Amazon SQS que utiliza una política basada en recursos asociados a la cola para conceder acceso a la cola a la cuenta B. A continuación, la cuenta B utiliza una política de grupo de IAM para delegar el acceso a un grupo de la cuenta B.

En el siguiente ejemplo una política de cola concede permiso a la cuenta B para realizar las acciones `SendMessage` y `ReceiveMessage` en la cola de la cuenta A denominada queue1, pero solo entre las 12:00 h y las 15:00 h del 30 de noviembre de 2014. El número de la cuenta B es 1111-2222-3333. La cuenta A utiliza Amazon SQS para implementar esta política.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": [
      "sqs:SendMessage",
      "sqs:ReceiveMessage"
    ],
    "Resource": ["arn:aws:sqs:*:123456789012:queue1"],
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2014-11-30T12:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2014-11-30T15:00Z"}
    }
  }
}
```

La política de la cuenta B para delegar el acceso a un grupo de la cuenta B podría ser como el siguiente ejemplo. La cuenta B utiliza IAM para asociar esta política a un grupo (o usuario).

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sqs:*",
    "Resource": "arn:aws:sqs:*:123456789012:queue1"
```

```
  }
}
```

En el ejemplo anterior de la política de usuario de IAM, la cuenta B utiliza un carácter comodín para conceder acceso a su usuario a todas las acciones de Amazon SQS en la cola de la cuenta A. Sin embargo, la cuenta B puede delegar el acceso únicamente en la medida en que se haya concedido acceso a dicha cuenta. El grupo de la cuenta B que tenga la segunda política solo puede obtener acceso a la cola entre las 12:00 y las 3:00 el 30 de noviembre de 2014. El usuario solo puede realizar las acciones `SendMessage` y `ReceiveMessage`, según se define en la política de cola de Amazon SQS de la cuenta A.

No se puede delegar el acceso cuando la cuenta tiene el acceso denegado

Una Cuenta de AWS no puede delegar el acceso a los recursos de otra cuenta si la otra cuenta ha denegado de forma explícita el acceso a la cuenta principal del usuario. La denegación se propaga a los usuarios de dicha cuenta independientemente de que tengan políticas que les conceda acceso.

Por ejemplo, la cuenta A escribe una política de bucket en el bucket de S3 de la cuenta A que deniega de forma explícita el acceso de la cuenta B al bucket de la cuenta A. Pero la cuenta B escribe una política de usuario de IAM que concede a un usuario de la cuenta B acceso a un bucket de la cuenta A. La denegación explícita aplicada al bucket de S3 de la cuenta A se propaga a los usuarios de la cuenta B. Anula la política de usuario de IAM que concede acceso al usuario de la cuenta B (para obtener más información sobre se evalúan cómo los permisos, consulte Lógica de evaluación de políticas.)

La política de bucket de la cuenta A podría ser como la siguiente política. En este ejemplo, el bucket de S3 de la cuenta A se denomina mybucket y el número de la cuenta B es 1111-2222-3333. La cuenta A utiliza Amazon S3 para implementar esta política.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::mybucket/*"
  }
}
```

Esta denegación explícita anula cualquier políticas de la cuenta B que proporcione permiso para obtener acceso al bucket de S3 en la cuenta A.

Uso de roles de IAM

Para que un usuario, una aplicación o un servicio pueda utilizar un rol que usted haya creado, debe conceder permisos para cambiar al rol. Puede utilizar cualquier política asociada a grupos o usuarios para otorgar los permisos necesarios. En esta sección se describe cómo se puede conceder a los usuarios el permiso para utilizar un rol. También explica cómo el usuario puede cambiar a un rol desde el AWS Management Console, Tools for Windows PowerShell, AWS Command Line Interface (AWS CLI) y el API de `AssumeRole`.

> **Important**
>
> Cuando crea un rol mediante programación en lugar de hacerlo en la consola de IAM, tiene la opción de agregar un `Path` de 512 caracteres como máximo además del `RoleName`, que puede tener un máximo de 64 caracteres. Sin embargo, si desea utilizar un rol con la característica Cambiar rol en la consola de AWS Management Console, la combinación de `Path` y `RoleName` no puede superar los 64 caracteres.

Puede cambiar de rol desde la AWS Management Console. Puede asumir un rol llamando a una operación de la API o de la AWS CLI, o utilizando una URL personalizada. El método que utilice determinará quién puede asumir el rol y cuánto tiempo puede durar la sesión de rol. Cuando se utilizan operaciones de la API `AssumeRole*`, el rol de IAM que se asume es el recurso. El usuario o rol que llama a las operaciones de la API `AssumeRole*` es la entidad principal.

Comparación de métodos para el uso de roles

Método para asumir el papel	¿Quién puede asumir el rol?	Método para especificar la duración de las credenciales	Duración de las credencia les (mín | máx | predeterminada)
AWS Management Console	Usuario (mediante el cambio de roles)	Duración máxima de la sesión en	15 min | configuración de la duración

Método para asumir el papel	¿Quién puede asumir el rol?	Método para especificar la duración de las credenciales	Duración de las credencia les (mín | máx | predeterminada)
		la página de resumen del Rol	máxima de la sesión² | 1 h
Operación `assume-role` de la CLI u operación `AssumeRole` de la API	Usuario o rol¹	Parámetro `duration-seconds` de la CLI o parámetro `DurationSeconds` de la API	15 min | configuración de la duración máxima de la sesión² | 1 h
Operación `assume-role-with-saml` de la CLI u operación `AssumeRoleWithSAML` de la API	Cualquier usuario autenticado mediante SAML	Parámetro `duration-seconds` de la CLI o parámetro `DurationSeconds` de la API	15 min | configuración de la duración máxima de la sesión² | 1 h
Operación `assume-role-with-web-identity` de la CLI u operación `AssumeRoleWithWebIdentity` de la API	Cualquier usuario autenticado con OIDC	Parámetro `duration-seconds` de la CLI o parámetro `DurationSeconds` de la API	15 min | configuración de la duración máxima de la sesión² | 1 h

Método para asumir el papel	¿Quién puede asumir el rol?	Método para especificar la duración de las credenciales	Duración de las credencia les (mín | máx | predeterminada)
URL de la consola construida con `AssumeRole`	Usuario o rol	Parámetro `HTMLSessionDu ration` en la URL	15 min | 12 h | 1 h
URL de la consola construida con `AssumeRol eWithSAML`	Cualquier usuario autenticado mediante SAML	Parámetro `HTMLSessionDu ration` en la URL	15 min | 12 h | 1 h
URL de la consola construida con `AssumeRol eWithWebI dentity`	Cualquier usuario autenticado con OIDC	Parámetro `HTMLSessionDu ration` en la URL	15 min | 12 h | 1 h

[1] El uso de las credenciales de un rol para asumir otro rol se denomina encadenamiento de roles. Cuando se utiliza el encadenamiento de roles, las nuevas credenciales tienen una duración máxima de una hora. Cuando utiliza roles para conceder permisos a las aplicaciones que se ejecutan en instancias EC2, esas aplicaciones no están sujetas a esta limitación.

[2] Esta opción puede tener un valor comprendido entre 1 y 12 horas. Para obtener información detallada sobre modificar la configuración de la duración máxima de la sesión, consulte Modificación de un rol. Este ajuste determina la duración máxima de la sesión que se puede solicitar al obtener las credenciales del rol. Por ejemplo, cuando utilice las operaciones AssumeRole* de la API para asumir un rol, puede utilizar el parámetro `DurationSeconds` para especificar la duración de la sesión. Use este parámetro para especificar la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. A los usuarios de IAM que cambian de rol en la consola se les concede la duración máxima de la sesión, o el tiempo restante de la sesión del usuario, lo que sea menor. Supongamos que establece

una duración máxima de 5 horas en un rol. Un usuario de IAM que ha iniciado sesión en la consola durante 10 horas (fuera del máximo predeterminado de 12) cambia al rol. La duración de la sesión de rol disponible es de 2 horas. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol más adelante en esta misma página.

Notas

- La configuración de duración máxima de sesión no limita las sesiones asumidas por los servicios de AWS.
- Las credenciales del rol de IAM de Amazon EC2 no están sujetas a la duración máxima de sesión configurada en el rol.
- Para permitir que los usuarios vuelvan a asumir el rol actual dentro de una sesión de rol, especifique el ARN del rol o el ARN de la Cuenta de AWS como entidad principal en la política de confianza de rol. Los Servicios de AWS que proporcionan recursos de computación, como Amazon EC2, Amazon ECS, Amazon EKS y Lambda, brindan credenciales temporales y las actualizan automáticamente. Esto garantiza que siempre disponga de un conjunto de credenciales válido. Para estos servicios, no es necesario volver a asumir el rol actual a fin de obtener credenciales temporales. Sin embargo, si tiene la intención de aprobar etiquetas de sesión o una política de sesión, tendrá que volver a asumir el rol actual. Para obtener información sobre cómo modificar una política de confianza de roles a fin de agregar el ARN del rol o el ARN de la Cuenta de AWS para la entidad principal, consulte Modificación de una política de confianza de rol (consola).

Temas

- Revocación de las credenciales de seguridad temporales de un rol de IAM

Cómo consultar la configuración de la duración máxima de la sesión para un rol

Puede especificar la duración máxima de sesión de un rol mediante AWS Management Console o mediante el uso de la función AWS CLI o API de AWS. Cuando se utiliza una operación de la AWS CLI o de la API para asumir un rol, es posible especificar un valor para el parámetro `DurationSeconds`. Puede utilizar este parámetro para especificar la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor indicado en la opción Duración máxima de la sesión para el rol. Antes de especificar el parámetro, debe consultar este valor para su rol. Si especifica un valor para el parámetro `DurationSeconds` superior a valor máximo, la operación genera un error.

Para ver la duración máxima de la sesión de un rol (consola)

1. En el panel de navegación de la consola de IAM, elija Roles.
2. Elija el nombre del rol que desea ver.
3. Junto a Duración máxima de la sesión, vea la longitud máxima de sesión que se concede para el rol. Esta es la duración máxima de la sesión que puede especificar en su AWS CLI u operación de API.

Para ver la duración máxima de la sesión de un rol (AWS CLI)

1. Si no conoce el nombre del rol que desea asumir, ejecute el siguiente comando para enumerar los roles de su cuenta:

   - aws iam list-roles
2. Para ver la duración máxima de la sesión de rol, ejecute el siguiente comando. A continuación, consulte el parámetro de duración máxima de la sesión.

   - aws iam get-role

Para ver la duración máxima de la sesión de un rol (API de AWS)

1. Si no conoce el nombre del rol que desea asumir, llame a la siguiente operación para enumerar los roles de su cuenta:

- ListRoles

2. Para ver la duración máxima de la sesión de rol, ejecute la siguiente operación. A continuación, consulte el parámetro de duración máxima de la sesión.

   - GetRole

Conceder permisos de usuario para cambiar de rol

Cuando un administrador crea un rol para el acceso entre cuentas, establece la confianza entre la cuenta propietaria del rol, los recursos (cuenta de confianza) y la cuenta que contiene a los usuarios (cuenta de confianza). Para ello, el administrador de la cuenta de confianza especifica el número de la cuenta de confianza como `Principal` en la política de confianza del rol. Esto permite que potencialmente cualquier usuario de la cuenta de confianza asuma el rol. Para finalizar la configuración, el administrador de la cuenta de confianza debe conceder permiso a usuarios o grupos específicos de dicha cuenta para cambiar al rol.

Para conceder permiso para cambiar a un rol

1. Como administrador de la cuenta de confianza, cree una política nueva para el usuario o edite una política existente para agregar los elementos necesarios. Para obtener más información, consulte Creación o edición de la política.
2. A continuación, elija cómo desea compartir la información del rol:

   - Enlace de rol: envíe a los usuarios un enlace que los lleve a la página Switch Role (Cambiar el rol) con todos los detalles ya completados.
   - ID de cuenta o alias: proporcione a cada usuario el nombre de la función junto con el número de ID de cuenta o alias de cuenta. El usuario se dirige a la página Switch Role (Cambiar rol) y agrega los detalles manualmente.

   Para obtener más información, consulte Proporcionar información al usuario.

Tenga en cuenta que solo puede cambiar de rol cuando inicia sesión como usuario de IAM, como rol federado de SAML o como un rol federado de identidad web. No puede cambiar de rol si inicia sesión como usuario Usuario raíz de la cuenta de AWS.

Important

No puede cambiar roles en la AWS Management Console a un rol que requiera un valor ExternalId. Solo puede cambiar a dicho rol si llama a la API `AssumeRole` que admite el parámetro `ExternalId`.

Notas

- En este tema se explican las políticas para un usuario, ya que, en última instancia, concedemos permisos a un usuario para llevar a cabo una tarea. Sin embargo, no es recomendable conceder permisos a un usuario en particular. Cuando un usuario asume un rol, se le asignan los permisos asociados a ese rol.
- Al cambiar de rol en la AWS Management Console, la consola utiliza siempre sus credenciales originales para autorizar el cambio. Esto se aplica tanto si inicia sesión como usuario de IAM, como rol federado SAML o como un rol federado de identidad web. Por ejemplo, si cambia a RoleA, IAM utiliza las credenciales originales del usuario o del rol federado para determinar si está autorizado para asumir RoleA. Si trata de cambiar a RoleB mientras utiliza RoleA, las credenciales originales del usuario o del rol federado se utilizan para autorizar su intento. Las credenciales de RoleA no se utilizan para esta acción.

Temas

Creación o edición de la política

Una política que concede a un usuario permiso para asumir un rol debe incluir una instrucción con el efecto `Allow` en lo siguiente:

- La acción `sts:AssumeRole`
- El nombre de recurso de Amazon (ARN) de un rol en un elemento `Resource`

A los usuarios que obtienen la política se les permite cambiar de rol en el recurso enumerado (ya sea a través de la pertenencia a un grupo o directamente conectado).

> **Note**
>
> Si `Resource` está configurado en `*`, el usuario puede asumir cualquier rol en cualquier cuenta que tenga una relación de confianza con la cuenta del usuario. (En otras palabras, la política de confianza del rol especifica la cuenta del usuario como `Principal`). Le recomendamos que siga el principio de mínimo privilegio y especifique el ARN completo únicamente para los roles que necesite el usuario.

En el siguiente ejemplo se muestra una política que permite al usuario asumir roles solo en una cuenta. Además, la política utiliza un asterisco (*) para especificar que el usuario puede cambiar a un rol solo si el nombre del rol comienza por las letras `Test`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account-id:role/Test*"
  }
}
```

> **Note**
>
> Los permisos que el rol concede al usuario no se agregan a los permisos que se le han concedido anteriormente. Si un usuario cambia a un rol, el usuario temporalmente renuncia a sus permisos originales a cambio de los concedidos por el rol. Si el usuario deja de utilizar el rol, sus permisos originales se restablecerán automáticamente. Por ejemplo, supongamos que los permisos del usuario dejan trabajar con instancias Amazon EC2, pero la política de permisos del rol no concede esos permisos. En ese caso, al utilizar el rol, el usuario no puede trabajar con instancias Amazon EC2 en la consola. Además, las credenciales temporales obtenidas a través de `AssumeRole` no funcionarán con instancias Amazon EC2 mediante programación.

Proporcionar información al usuario

Después de crear un rol y conceder al usuario permisos para cambiar a dicho rol, debe proporcionar al usuario lo siguiente:

- El nombre del rol
- El ID o alias de la cuenta que contiene el rol

Para agilizar el acceso de sus usuarios, puede enviarles un enlace preconfigurado con el ID de la cuenta y el nombre de la función. Puede ver el enlace del rol después de completar el asistente de creación Crear rol si selecciona el banner Ver rol, o en la página Resumen del rol para cualquier rol habilitado para varias cuentas.

También puede utilizar el siguiente formato para crear manualmente el enlace. Sustituya el alias o ID de la cuenta y el nombre del rol por los dos parámetros en el siguiente ejemplo.

```
https://signin.aws.amazon.com/switchrole?
account=your_account_ID_or_alias&roleName=optional_path/role_name
```

Le recomendamos que dirija a los usuarios a Cambio a un rol (Consola) para guiarles durante el proceso. Para solucionar problemas comunes que se pueden encontrar al asumir un rol, consulte No puedo asumir un rol.

Consideraciones

- Si crea el rol mediante programación, puede crear el rol con una ruta y un nombre. Si lo hace, debe proporcionar la ruta de acceso completa y el nombre del rol a los usuarios para que puedan ingresarlos en la página Cambiar rol de la AWS Management Console. Por ejemplo: `division_abc/subdivision_efg/role_XYZ`.
- Si crea el rol mediante programación, puede agregar un `Path` de hasta 512 caracteres y un `RoleName`. El nombre del rol puede tener una longitud de hasta 64 caracteres. Sin embargo, si desea utilizar un rol con la característica Cambiar rol en AWS Management Console, la combinación de `Path` y `RoleName` no puede superar los 64 caracteres.

- Por motivos de seguridad, puede revisar los registros de AWS CloudTrail para saber quién realizó una acción en AWS. Puede utilizar la clave de condición de `sts:SourceIdentity` en la política de confianza de rol para exigir a los usuarios que especifiquen una identidad cuando asuman un rol. Por ejemplo, puede requerir que los usuarios de IAM especifiquen su propio nombre de usuario

como su identidad de origen. Esto puede ayudarle a determinar qué usuario realizó una acción específica en AWS. Para obtener más información, consulte `sts:SourceIdentity`. También puede utilizar `sts:RoleSessionName` para exigir a los usuarios que especifiquen un nombre de sesión cuando asuman un rol. Esto puede ayudarle a diferenciar entre sesiones de rol cuando un rol es utilizado por diferentes entidades.

Concesión de permisos a un usuario para transferir un rol a un servicio de AWS

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Esto permite al servicio asumir posteriormente el rol y ejecutar acciones en su nombre. Para muchos servicios, solo tiene que transferir el rol al servicio una vez durante la configuración y no cada vez que el servicio asume el rol. Por ejemplo, suponga que tiene una aplicación que se ejecuta en una instancia de Amazon EC2. Dicha aplicación requiere credenciales temporales para la autenticación, así como permisos para realizar acciones en AWS. Cuando configure la aplicación, debe transferir un rol a Amazon EC2 para que lo utilice con la instancia que proporciona dichas credenciales. Puede definir los permisos de las aplicaciones que se ejecutan en la instancia asociando una política de IAM al rol. La aplicación asume el rol cada vez que necesita realizar las acciones que este permite.

Para transferir un rol (y sus permisos) a un servicio de AWS, un usuario debe tener permisos para transferir el rol al servicio. Esto ayuda a los administradores a garantizar que solo los usuarios autorizados puedan configurar un servicio con un rol que concede permisos. Para permitir a un usuario transferir un rol a un servicio de AWS, debe conceder el permiso `PassRole` al usuario, rol o grupo de IAM del usuario.

> **Warning**
>
> - Solo puede usar el permiso `PassRole` para transferir un rol de IAM a un servicio que comparte la misma cuenta de AWS. Para transferir una función de la cuenta A a un servicio de la cuenta B, primero debe crear un rol de IAM en la cuenta B que pueda asumir la función desde la cuenta A y, a continuación, la función de la cuenta B se puede transferir al servicio. Para obtener más información, consulte Acceso a recursos entre cuentas en IAM.
> - No intente controlar quién puede pasar un rol etiquetando el rol y, a continuación, utilizando la clave de condición `ResourceTag` en una política con la acción `iam:PassRole`. Este planteamiento no da resultados fiables.

Al configurar el permiso `PassRole`, debe asegurarse de que un usuario no pase un rol en el que el rol tenga más permisos de los que usted desea que tenga el usuario. Por ejemplo, es posible que a Alice no se le permita realizar ninguna acción de Amazon S3. Si Alice pudiera transferir un rol a un servicio que permita acciones de Amazon S3, el servicio podría realizar acciones de Amazon S3 en su nombre al ejecutar el trabajo.

Si especifica un rol vinculado a un servicio, debe disponer también de permiso para transferir ese rol al servicio. Algunos servicios crean automáticamente un rol vinculado a un servicio en su cuenta al realizar una acción en dicho servicio. Por ejemplo, Amazon EC2 Auto Scaling crea el rol vinculado a un servicio `AWSServiceRoleForAutoScaling` automáticamente la primera vez que se crea un grupo de Auto Scaling. Si intenta especificar el rol vinculado a un servicio al crear un grupo de escalado automático y no posee el permiso `iam:PassRole`, recibirá un error. Si no especifica el rol de manera explícita, el permiso `iam:PassRole` no es necesario y la acción predeterminada es usar el rol `AWSServiceRoleForAutoScaling` para todas las operaciones que se realicen en ese grupo. Para saber qué servicios admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM. Para saber qué servicios crean automáticamente un rol vinculado a un servicio al realizar una acción en ellos, elija el enlace Sí y consulte la documentación relacionada con los roles vinculados a dicho servicio.

Un usuario puede pasar un ARN de rol como parámetro en cualquier operación de API que utilice el rol para asignar permisos al servicio. A continuación, el servicio comprueba si ese usuario tiene el permiso `iam:PassRole`. Para que el usuario transfiera únicamente los roles autorizados, puede filtrar el permiso `iam:PassRole` con el elemento `Resources` de la instrucción de política de IAM.

Puede utilizar el elemento `Condition` en una política JSON para probar el valor de las claves incluidas en el contexto de solicitud de todas las solicitudes de AWS. Para obtener más información acerca del uso de las claves de condición en una política, consulte Elementos de política JSON de IAM: Condition. La clave de condición `iam:PassedToService` se puede utilizar para especificar la entidad principal del servicio al que se puede pasar un rol. Para obtener más información acerca del uso de la clave de condición `iam:PassedToService` en una política, consulte iam:PassedToService.

Ejemplo 1

Supongamos que desea conceder a un usuario la capacidad de transferir cualquier conjunto de roles al servicio de Amazon EC2 cuando se lanza una instancia. Necesita tres elementos:

- Una política de permisos de IAM asociada al rol que determina lo que puede hacer dicho rol. Limite los permisos a únicamente las acciones que deba llevar a cabo el rol y únicamente los recursos

que el rol necesite para dichas acciones. Puede utilizar una política de permisos de IAM creada por el cliente o administrada por AWS.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [ "A list of the permissions the role is allowed to use" ],
        "Resource": [ "A list of the resources the role is allowed to access" ]
    }
}
```

- Una política de confianza para el rol que permita al servicio asumir dicho rol. Por ejemplo, puede adjuntar la siguiente política de confianza al rol con la acción `UpdateAssumeRolePolicy`. Esta política de confianza permite a Amazon EC2 utilizar el rol y los permisos asociados al rol.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "TrustPolicyStatementThatAllowsEC2ServiceToAssumeTheAttachedRole",
        "Effect": "Allow",
        "Principal": { "Service": "ec2.amazonaws.com" },
       "Action": "sts:AssumeRole"
    }
}
```

- Una política de permisos de IAM asociada al usuario de IAM que le permite especificar únicamente los roles aprobados. Normalmente agrega `iam:GetRole` a `iam:PassRole`, de modo que el usuario pueda obtener los detalles del rol que se transfiere. En este ejemplo, el usuario puede pasar únicamente los roles que existen en la cuenta especificada con nombres que empiezan por `EC2-roles-for-XYZ-`:

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::account-id:role/EC2-roles-for-XYZ-*"
    }]
```

```
}
```

Ahora el usuario puede iniciar una instancia de Amazon EC2 con un rol asignado. Las aplicaciones que se ejecutan en la instancia pueden acceder a credenciales temporales para el rol a través de los metadatos del perfil de instancia. Las políticas de permisos asociadas al rol determinan lo que puede hacer la instancia.

Ejemplo 2

Amazon Relational Database Service (Amazon RDS) admite una característica denominada Monitoreo mejorado. Esta característica permite a Amazon RDS monitorear una instancia de base de datos mediante un agente. También permite a Amazon RDS registrar métricas en Amazon CloudWatch Logs. Para habilitar esta característica, debe crear una función de servicio para otorgar a Amazon RDS permisos para monitorear y escribir métricas en los registros.

Para crear un rol para el monitoreo mejorado de Amazon RDS

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Roles y después Crear rol.
3. Elija el tipo de rol de Servicio de AWS y, a continuación, en Casos de uso para otros Servicios de AWS, elija el servicio RDS. Elija RDS - Supervisión mejorada y, a continuación, elija Siguiente.
4. Elija la política de permisos AmazonRDSEnhancedMonitoringRole.
5. Seleccione Siguiente.
6. En Nombre del rol, ingrese un nombre de rol que le sea útil para identificar su propósito. Los nombres de rol deben ser únicos en su Cuenta de AWS. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre del rol distingue entre mayúsculas y minúsculas. Cuando los clientes ven un nombre de rol en la consola, por ejemplo, durante el proceso de inicio de sesión, el nombre del rol no distingue entre mayúsculas y minúsculas. Dado que varias entidades pueden hacer referencia al rol, no se puede editar el nombre del rol una vez que se crea.
7. (Opcional) En Descripción, ingrese una descripción para el nuevo rol.
8. (Opcional) Adjunte etiquetas como pares de clave-valor para agregar metadatos al rol. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.

9. Revise el rol y, a continuación, seleccione Crear rol.

Automáticamente, el rol recibe una política de confianza que otorga los permisos de servicio `monitoring.rds.amazonaws.com` para asumir el rol. Después, Amazon RDS podrá realizar todas las acciones que permite la política `AmazonRDSEnhancedMonitoringRole`.

El usuario para el que desea habilitar la supervisión mejorada necesita una política que incluya una declaración que permita al usuario enumerar los roles de RDS y una declaración que le permita transferir el rol, como la siguiente. Utilice el número de cuenta y sustituya el nombre del rol por el nombre que ha facilitado en el paso 6.

```
{
  "Sid": "PolicyStatementToAllowUserToListRoles",
  "Effect": "Allow",
  "Action": ["iam:ListRoles"],
  "Resource": "*"
},
{
    "Sid": "PolicyStatementToAllowUserToPassOneSpecificRole",
    "Effect": "Allow",
    "Action": [ "iam:PassRole" ],
    "Resource": "arn:aws:iam::account-id:role/RDS-Monitoring-Role"
}
```

Puede combinar esta instrucción con instrucciones de otra política o colocarla en su propia política. En lugar de especificar que el usuario pueda transferir cualquier rol que empieza por `RDS-`, puede sustituir el nombre del rol en el ARN del recurso por un comodín, de la siguiente manera.

```
    "Resource": "arn:aws:iam::account-id:role/RDS-*"
```

Acciones **`iam:PassRole`** en registros de AWS CloudTrail

`PassRole` no es una llamada a la API. `PassRole` es un permiso, lo que significa que no se generan registros de CloudTrail para `PassRole` de IAM. Para revisar qué roles se pasan a qué Servicios de AWS en CloudTrail, debe revisar el registro de CloudTrail que creó o modificó el recurso de AWS que recibe el rol. Por ejemplo, un rol se pasa a una función de AWS Lambda cuando se crea. El registro de la acción `CreateFunction` muestra un registro del rol que se pasó a la función.

Cambio a un rol (Consola)

Un rol especifica un conjunto de permisos que puede utilizar para acceder a los recursos de AWS que necesita. En este sentido, es similar a un usuario de IAM en AWS Identity and Access Management. Al iniciar sesión como usuario, obtendrá un conjunto específico de permisos. Sin embargo, no inicia sesión en un rol propiamente, si no que al iniciar sesión puede cambiar a un rol. Esto anula temporalmente los permisos de usuario originales y, en su lugar, le otorga los permisos asignados al rol. El rol puede estar en su propia cuenta o en cualquier otra Cuenta de AWS. Para obtener más información acerca de los roles, sus ventajas y cómo crearlos, consulte Roles de IAM y Creación de roles de IAM.

> **Important**
>
> Los permisos de sus usuarios de y de cualquier rol al que cambie no se acumulan. Solo hay un conjunto de permisos activo a la vez. Cuando se cambia a un rol, se abandonan temporalmente los permisos de usuario y se trabaja con los permisos que el rol tenga asignados. Al salir del rol, los permisos de usuario se restablecen de forma automática.

Al cambiar de rol en la AWS Management Console, la consola utiliza siempre sus credenciales originales para autorizar el cambio. Esto se aplica tanto si inicia sesión como usuario de IAM, un usuario en el Centro de Identidades IAM, como rol federado SAML o como un rol federado de identidad web. Por ejemplo, si cambia a RoleA, IAM utiliza las credenciales originales del usuario o del rol federado para determinar si está autorizado para asumir RoleA. Si a continuación cambia a RoleB mientras utiliza RoleA, AWS seguirá utilizando las credenciales originales del usuario o del rol federado, no las credenciales de RoleA, para autorizar el cambio.

Cosas que debe saber acerca del cambio de roles en la consola

En esta sección se proporciona información adicional acerca del uso de la consola de IAM para cambiar a un rol.

> **Notas:**
>
> - No puede cambiar de rol si inicia sesión como usuario Usuario raíz de la cuenta de AWS. Solo puede cambiar de rol cuando inicia sesión como usuario de IAM, un usuario en el Centro de Identidades IAM, como rol federado de SAML o como un rol federado de identidad web.

- No puede cambiar roles en la AWS Management Console a un rol que requiera un valor ExternalId. Solo puede cambiar a dicho rol si llama a la API `AssumeRole` que admite el parámetro `ExternalId`.

- Si su administrador le proporciona un enlace, elíjalo y, a continuación, vaya al paso Step 5 en el siguiente procedimiento. El enlace le lleva a la página web correspondiente y rellena el ID de la cuenta (o alias) y el nombre del rol.
- Puede crear manualmente el enlace e ir directamente al paso Step 5 del procedimiento siguiente. Para crear el enlace, utilice el formato siguiente:

  `https://signin.aws.amazon.com/switchrole?account=`*`account_id_number`*`&roleName=`*`role_name`*`&displayName=`*`text_to_display`*

  Sustituya el texto siguiente:

  - *`account_id_number`* – Identificador de cuenta de 12 dígitos que le ha proporcionado el administrador. Como alternativa, el administrador puede crear un alias de cuenta de forma que la URL ya incluya el nombre de la cuenta, en lugar de un ID de cuenta. Para obtener más información, consulte Tipos de usuarios en la Guía del usuario de AWS Sign-In.
  - *`role_name`* – Nombre del rol que desea asumir. Puede obtenerlo observando el final del ARN del rol. Por ejemplo, el nombre de rol `TestRole` en el ARN de rol siguiente: `arn:aws:iam::123456789012:role/TestRole`.
  - (Opcional) *`text_to_display`* – Texto que desea que aparezca en la barra de navegación en lugar de su nombre de usuario cuando este rol esté activo.
- Puede cambiar de rol manualmente empleando la información que le proporciona el administrador mediante los siguientes procedimientos:

De forma predeterminada, cuando alterna de roles, su sesión AWS Management Console dura 1 hora. Las sesiones de usuario de IAM son de 12 horas de forma predeterminada. A los usuarios de IAM que cambian de rol en la consola se les concede la duración máxima de la sesión del rol, o el tiempo restante de la sesión del usuario, lo que sea menor. Por ejemplo, suponga que se establece una duración máxima de sesión de 10 horas para un rol. Un usuario de IAM ha iniciado sesión en la consola durante 8 horas cuando decide cambiar al rol. Quedan 4 horas en la sesión de usuario, por lo que la duración de la sesión de rol permitida es de 4 horas. En la tabla siguiente se muestra cómo determinar la duración de la sesión para un usuario de IAM al cambiar roles en la consola.

Duración de la sesión del rol de consola de los usuarios de IAM

El tiempo restante de sesión de usuario de IAM es...	La duración de la sesión de rol es…		
Duración máxima de la sesión inferior al rol	Tiempo restante en la sesión del usuario		
Duración máxima de la sesión mayor al rol	Valor de la duración máxima de la sesión		
Igual a la duración máxima de la sesión del rol	Valor de la duración máxima de la sesión (aproximado)		

> **Note**
>
> Algunas de servicio AWS pueden reiniciar automáticamente su sesión de rol cuando caduque sin que realice ninguna acción. Algunos pueden pedirle que vuelva a cargar la página del navegador para volver a autenticar su sesión.

Para solucionar problemas comunes que se pueden encontrar al asumir un rol, consulte No puedo asumir un rol.

Para cambiar de rol (consola)

1. Inicie sesión en AWS Management Console como usuario de IAM y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En la consola de IAM, seleccione su nombre de usuario en la parte superior derecha de la barra de navegación. Normalmente tiene el siguiente aspecto: ***nombre_usuario***@***número_ID_cuenta_o_alias***.
3. Elija Switch Role. Si es la primera vez que elige esta opción, aparecerá una página con más información. Después de leerla, elija Switch Role (Cambiar rol). Si borra las cookies del navegador, esta página podría aparecer de nuevo.
4. En la página Switch Role (Cambiar rol), escriba el número del ID de la cuenta o el alias de la cuenta y el nombre del rol que le proporcionó el administrador.

   > **Note**
   >
   > Si su administrador creó el rol con una ruta como, por ejemplo, `division_abc/subdivision_efg/roleToDoX`, deberá escribir dicha ruta y el nombre completos en el recuadro Role (Rol). Si escribe solamente el nombre del rol, o si la longitud combinada de `Path` y `RoleName` sobrepasa los 64 caracteres, el cambio de rol producirá un error. Se trata de un límite de las cookies del navegador que almacenan el nombre del rol. Si esto ocurre, póngase en contacto con el administrador y pídale que reduzca el tamaño de la ruta y del nombre del rol.

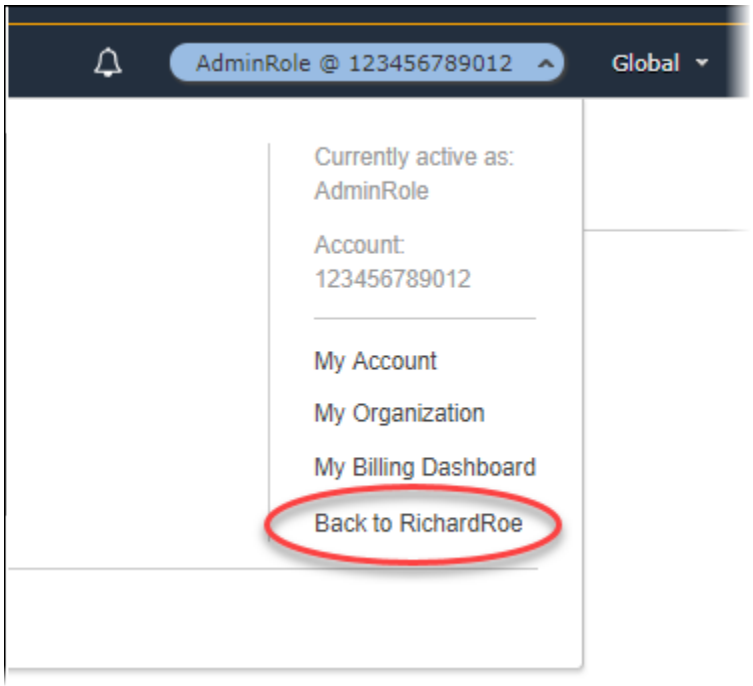
5. (Opcional) Elija un Nombre para mostrar. Escriba el texto que desea que aparezca en la barra de navegación en lugar de su nombre de usuario cuando este rol esté activo. El sistema le sugiere un nombre en función de la información de la cuenta y del rol, pero puede cambiarlo si lo desea. También puede seleccionar un color para destacar el nombre de visualización. El nombre y el color pueden ayudarle a saber cuándo está activo el rol, lo que modificará sus permisos. Por ejemplo, en el caso de un rol que le ofrece acceso al entorno de prueba, puede especificar un Nombre que mostrar de **`Test`** y seleccionar el Color verde. En el caso de un rol que le ofrece acceso al entorno de producción, puede especificar un Nombre de visualización de **`Production`** y seleccionar el Color rojo.
6. Elija Switch Role. El nombre y el color de visualización sustituyen su nombre de usuario en la barra de navegación y, a partir de ese momento, puede empezar a utilizar los permisos que le concede dicho rol.

> **Sugerencia**
>
> Los últimos roles que utilizó aparecen en el menú . La próxima vez que necesite cambiar a uno de estos roles, bastará con elegir el rol que desea. Solo tiene que escribir la información de la cuenta y del rol manualmente si dicho rol no se muestra en el menú.

Para dejar de utilizar un rol (consola)

1. En la consola de IAM, elija el Nombre de visualización del rol en la parte superior derecha de la barra de navegación. Normalmente tiene el siguiente aspecto: ***nombrerol*@*número_ID_cuenta_o_alias***.
2. Elija Back to ***nombredeusuario*** (Volver a [nombre de usuario]). El rol y sus permisos están desactivados y los permisos asociados con su usuario y grupos de IAM se restablecen de forma automática.

   Por ejemplo, supongamos que ha iniciado sesión en el número de cuenta `123456789012` con el nombre de usuario `RichardRoe`. Después de utilizar el rol `AdminRole`, desea dejar de utilizarlo y volver a los permisos originales. Para dejar de utilizar un rol, elija AdminRole @ 123456789012 y, a continuación, elija Volver a RichardRoe.



Cambio a un rol de IAM (AWS CLI)

Un rol especifica un conjunto de permisos que puede utilizar para acceder a los recursos de AWS que necesita. En este sentido, es similar a un usuario de IAM en AWS Identity and Access Management. Al iniciar sesión como usuario, obtendrá un conjunto específico de permisos. Sin embargo, no inicia sesión en una función, sino que después de iniciar sesión como usuario, puede cambiar a una función. Esto anula temporalmente los permisos de usuario originales y, en su lugar, le otorga los permisos asignados al rol. El rol puede estar en su propia cuenta o en cualquier otra Cuenta de AWS. Para obtener más información acerca de los roles, sus beneficios y cómo crearlos y configurarlos, consulte Roles de IAM y Creación de roles de IAM. Para obtener más información sobre los distintos métodos que puede utilizar para asumir un rol, consulte Uso de roles de IAM.

> **Important**
>
> Los permisos del usuario de IAM y de cualquier rol que asuma no se acumulan. Solo hay un conjunto de permisos activo a la vez. Cuando se asume un rol, se abandonan temporalmente los permisos de usuario o del rol anteriores y se trabaja con los permisos que el rol tenga asignados. Al salir del rol, los permisos de usuario se restablecen de forma automática.

Puede utilizar un rol para ejecutar un comando de la AWS CLI si ha iniciado sesión como usuario de IAM. También puede utilizar un rol para ejecutar un comando de la AWS CLI cuando haya iniciado sesión como usuario autenticado externamente (SAML o OIDC) que ya utiliza un rol. Además, puede utilizar un rol para ejecutar un comando de la AWS CLI desde una instancia de Amazon EC2 que esté asociada a un rol a través de su perfil de instancias. No puede asumir un rol si ha iniciado sesión como usuario Usuario raíz de la cuenta de AWS.

Encadenamiento de roles— También puede utilizar el encadenamiento de roles, que consiste en utilizar permisos de un rol para tener acceso a otro.

De forma predeterminada, la sesión de rol dura una hora. Cuando se asume este rol utilizando las operaciones de la CLI `assume-role*`, se puede especificar un valor para el parámetro `duration-seconds`. Este valor puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión para el rol. Si cambia de rol en la consola, la duración de la sesión se limita a un máximo de una hora. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

Si utiliza el encadenamiento de roles, la sesión tiene una duración máxima de una hora. Si utiliza a continuación el parámetro `duration-seconds` para proporcionar un valor superior a una hora, la operación generará un error.

Escenario de ejemplo: cambio a un rol de producción

Imagine que es un usuario de IAM para trabajar en el entorno de desarrollo. En esta situación, ocasionalmente necesita trabajar con el entorno de producción en la línea de comandos con la AWS CLI. Ya tiene una credencial de clave de acceso a su disposición. Este puede ser el par de claves de acceso asignado a su usuario de IAM estándar. O bien, si ha iniciado sesión como un usuario federado, puede ser el par de claves de acceso para la función que se le ha asignado inicialmente. Si sus permisos actuales le permiten asumir un rol IAM específico, puede identificar dicho rol en un "perfil" de los archivos de configuración de AWS CLI. Este comando se ejecuta con los permisos de la función de IAM especificada, no con la identidad original. Tenga en cuenta que al especificar dicho perfil en un comando de la AWS CLI, está utilizando el nuevo rol. En esta situación, no puede hacer uso de sus permisos originales en la cuenta de desarrollo al mismo tiempo. La razón es que no puede haber más de un conjunto de permisos en vigor a la vez.

> **Note**
>
> Por motivos de seguridad, los administradores pueden revisar los registros de AWS CloudTrail para saber quién realizó una acción en AWS. Es posible que el administrador requiera que especifique una identidad de origen o un nombre de la sesión de rol cuando asuma el rol. Para obtener más información, consulte `sts:SourceIdentity` y `sts:RoleSessionName`.

Para cambiar a una función de producción (AWS CLI)

1. En caso de que nunca haya utilizado AWS CLI, primero debe configurar su perfil de CLI predeterminado. Abra un símbolo del sistema y configure la instalación de AWS CLI para utilizar la clave de acceso de su usuario de IAM o de su rol federado. Para obtener más información, consulte Configuración de AWS Command Line Interface en la Guía del usuario de AWS Command Line Interface.

   Ejecute el comando aws configure de la siguiente manera:

   ```
   aws configure
   ```

Cuando se le pida, proporcione la siguiente información:

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: us-east-2
Default output format [None]: json
```

2. Cree un nuevo perfil para la función del archivo `.aws/config` en Unix o Linux, o el archivo `C:\Users\USERNAME\.aws\config` en Windows. El siguiente ejemplo crea un perfil denominado `prodaccess` que cambia a la función *`ProductionAccessRole`* en la cuenta `123456789012`. Obtendrá el ARN del rol del administrador de la cuenta que creó el rol. Si este perfil se invoca, AWS CLI utiliza las credenciales de `source_profile` para solicitar credenciales para el rol. Por esta razón, la identidad a la que se hace referencia como `source_profile` debe tener permisos de `sts:AssumeRole` para la función especificada en `role_arn`.

   ```
   [profile prodaccess]
       role_arn = arn:aws:iam::123456789012:role/ProductionAccessRole
       source_profile = default
   ```

3. Después de crear el nuevo perfil, cualquier comando de AWS CLI que especifique el parámetro `--profile prodaccess` se ejecuta bajo los permisos asociados al rol de IAM `ProductionAccessRole` en lugar de hacerlo en el usuario predeterminado.

   ```
   aws iam list-users --profile prodaccess
   ```

   Este comando funciona si los permisos asignados a `ProductionAccessRole` permiten enumerar los usuarios de la cuenta actual de AWS.

4. Para volver a los permisos concedidos por sus credenciales originales, ejecute comandos sin el parámetro `--profile`. La AWS CLI vuelve a utilizar las credenciales de su perfil predeterminado, que configuró en Step 1.

Para obtener más información, consulte Asumir un rol en la Guía del usuario de AWS Command Line Interface.

Escenario de ejemplo: Permitir que una función de perfil de instancias cambie una función en otra cuenta

Imagine que está utilizando dos Cuentas de AWS y desea permitir que una aplicación se ejecute en una instancia de Amazon EC2 para ejecutar comandos AWS CLI en ambas cuentas. Supongamos que la instancia EC2 existe en la cuenta 111111111111. Dicha instancia incluye la función de perfil de instancias `abcd` que permite que la aplicación realice tareas de solo lectura de Amazon S3 en el bucket `my-bucket-1` dentro de la misma cuenta 111111111111. Sin embargo, la aplicación también debe tener permitido asumir la función entre cuentas `efgh` para realizar tareas de la cuenta 222222222222. Para ello, la función del perfil de instancia EC2 `abcd` debe tener la siguiente política de permisos:

Política de permisos del rol de la cuenta 111111111111 ***abcd***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
```

```
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

Supongamos que la función entre cuentas `efgh` permite tareas de solo lectura de Amazon S3 en el bucket `my-bucket-2` dentro de la misma cuenta 222222222222. Para ello, la función entre cuentas `efgh` debe tener la siguiente política de permisos:

Política de permisos del rol de la cuenta 222222222222 ***efgh***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

La función `efgh` debe permitir la función de perfil de instancia `abcd` para asumirla. Para ello, la función `efgh` debe tener la siguiente política de confianza:

Política de confianza de rol ***efgh*** de la cuenta 222222222222

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

Para ejecutar los comandos de la AWS CLI en la cuenta `222222222222`, debe actualizar el archivo de configuración de la CLI. Identifique la función `efgh` como el "perfil" y la función del perfil de instancia EC2 `abcd` como la "fuente de credenciales" en el archivo de configuración de la AWS CLI. A continuación, los comandos de la CLI se ejecutan con los permisos de la función `efgh`, no la función `abcd` original.

> **Note**
>
> Por motivos de seguridad, puede utilizar AWS CloudTrail para auditar el uso de roles en la cuenta. Para diferenciar entre sesiones de rol cuando diferentes entidades principales utilizan un rol en los logs de CloudTrail, puede utilizar el nombre de sesión de rol. Si la AWS CLI asume un rol en nombre de un usuario, tal y como se describe en este tema, se crea automáticamente un nombre de sesión de rol con el formato `AWS-CLI-session-`*nnnnnnnn*. Aquí *nnnnnnnn* es un número entero que representa la hora en formato de tiempo Unix (el número de segundos desde la medianoche UTC del 1 de enero de 1970). Para obtener más información, consulte Referencia de eventos de CloudTrail en la Guía del usuario de AWS CloudTrail.

Para permitir que un perfil de instancia de EC2 cambie a una función entre cuentas (AWS CLI)

1. No tiene que configurar un perfil de CLI predeterminado. En su lugar, puede cargar las credenciales de los metadatos del perfil de instancia EC2. Cree un nuevo perfil para el rol en

el archivo `.aws/config`. En el siguiente ejemplo, se crea un perfil `instancecrossaccount` que cambia a la función *`efgh`* en la cuenta `222222222222`. Si este perfil se invoca, la AWS CLI utiliza las credenciales de los metadatos del perfil de instancia EC2 para solicitar credenciales para la función. Por esta razón, la función del perfil de instancia EC2 debe tener permisos de `sts:AssumeRole` para la función especificada en `role_arn`.

```
[profile instancecrossaccount]
role_arn = arn:aws:iam::222222222222:role/efgh
credential_source = Ec2InstanceMetadata
```

2. Después de crear el nuevo perfil, cualquier comando de AWS CLI que especifique el parámetro `--profile instancecrossaccount` se ejecuta bajo los permisos asociados a la función de `efgh` en la cuenta `222222222222`.

   ```
   aws s3 ls my-bucket-2 --profile instancecrossaccount
   ```

   Este comando funciona si los permisos asignados a la función `efgh` permiten enumerar los usuarios de la Cuenta de AWS actual.

3. Para volver a los permisos del perfil de instancia EC2 original de la cuenta `111111111111`, ejecute los comandos de la CLI sin el parámetro `--profile`.

Para obtener más información, consulte Asumir un rol en la Guía del usuario de AWS Command Line Interface.

Para cambiar a un rol de IAM (Tools for Windows PowerShell)

Un rol especifica un conjunto de permisos que puede utilizar para acceder a los recursos de AWS que necesita. En este sentido, es similar a un usuario de IAM en AWS Identity and Access Management. Al iniciar sesión como usuario, obtendrá un conjunto específico de permisos. Sin embargo, no inicia sesión en un rol propiamente, si no que al iniciar sesión puede cambiar a un rol. Esto anula temporalmente los permisos de usuario originales y, en su lugar, le otorga los permisos asignados al rol. El rol puede estar en su propia cuenta o en cualquier otra Cuenta de AWS. Para obtener más información acerca de los roles, sus beneficios y cómo crearlos y configurarlos, consulte Roles de IAM y Creación de roles de IAM.

> **Important**
>
> Los permisos de sus usuarios de IAM y de cualquier rol al que cambie no se acumulan. Solo hay un conjunto de permisos activo a la vez. Cuando se cambia a un rol, se abandonan temporalmente los permisos de usuario y se trabaja con los permisos que el rol tenga asignados. Al salir del rol, los permisos de usuario se restablecen de forma automática.

En esta sección se describe cómo cambiar de rol cuando trabaja en la línea de comando con la AWS Tools for Windows PowerShell.

Supongamos que dispone de una cuenta para trabajar en el entorno de desarrollo y que de vez en cuando tiene que trabajar en el entorno de producción en la línea de comandos con Tools for Windows PowerShell. Ya tiene una credencial de clave de acceso a su disposición. Puede ser un par de claves de acceso asignado a su usuario de IAM estándar. O bien, si ha iniciado sesión como un usuario federado, puede ser el par de claves de acceso para el rol que se le ha asignado inicialmente. Puede utilizar estas credenciales para ejecutar el cmdlet `Use-STSRole` que transfiere el ARN de un rol nuevo como parámetro. El comando devuelve credenciales de seguridad temporales para el rol solicitado. Puede utilizar estas credenciales en comandos de PowerShell posteriores con los permisos del rol para obtener acceso a los recursos de producción. Mientras utiliza el rol, no puede utilizar sus permisos de usuario de la cuenta Development ya que solo puede haber un conjunto de permisos en vigor a la vez.

> **Note**
>
> Por motivos de seguridad, los administradores pueden revisar los registros de AWS CloudTrail para saber quién realizó una acción en AWS. Es posible que el administrador requiera que especifique una identidad de origen o un nombre de la sesión de rol cuando asuma el rol. Para obtener más información, consulte `sts:SourceIdentity` y `sts:RoleSessionName`.

Tenga en cuenta que todas las claves de acceso y tokens solo son ejemplos y no se pueden utilizar tal y como se muestran. Tiene que sustituirlos por los valores adecuados de su entorno real.

Para cambiar a un rol (Tools for Windows PowerShell)

1. Abra un símbolo del sistema de PowerShell y configure el perfil predeterminado para utilizar la clave de acceso de su usuario de IAM actual o de su rol federado. Si ya ha utilizado anteriormente Tools for Windows PowerShell, probablemente esta tarea ya esté completada. Tenga en cuenta que solo puede cambiar de rol si inicia sesión como un usuario de IAM y no como el Usuario raíz de la cuenta de AWS.

   ```
   PS C:\> Set-AWSCredentials -AccessKey AKIAIOSFODNN7EXAMPLE -
   SecretKey wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY -StoreAs MyMainUserProfile
   PS C:\> Initialize-AWSDefaults -ProfileName MyMainUserProfile -Region us-east-2
   ```

   Para obtener más información, consulte la especificación de credenciales AWS en la Guía del usuario de AWS Tools for Windows PowerShell.

2. Para recuperar credenciales para el nuevo rol, ejecute el siguiente comando para cambiar al rol *RoleName* de la cuenta 123456789012. Obtendrá el ARN del rol del administrador de la cuenta que creó el rol. El comando requiere que indique también un nombre de sesión. Puede elegir cualquier texto para ello. El comando siguiente solicita las credenciales y después captura el objeto de propiedad `Credentials` del objeto de los resultados devueltos y lo almacena en la variable `$Creds`.

   ```
   PS C:\> $Creds = (Use-STSRole -RoleArn "arn:aws:iam::123456789012:role/RoleName" -
   RoleSessionName "MyRoleSessionName").Credentials
   ```

   `$Creds` es un objeto que ahora contiene los elementos `AccessKeyId`, `SecretAccessKey` y `SessionToken` que necesita en los pasos siguientes. Los siguientes comandos de ejemplo ilustran valores típicos:

   ```
   PS C:\> $Creds.AccessKeyId
   AKIAIOSFODNN7EXAMPLE

   PS C:\> $Creds.SecretAccessKey
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

   PS C:\> $Creds.SessionToken
   AQoDYXdzEGcaEXAMPLE2gsYULo
   +Im5ZEXAMPLEeYjs1M2FUIgIJx9tQqNMBEXAMPLECvSRyh0FW7jEXAMPLEW+vE/7s1HRp
   XviG7b+qYf4nD00EXAMPLEmj4wxS04L/uZEXAMPLECihzFB5lTYLto9dyBgSDyEXAMPLE9/
   g7QRUhZp4bqbEXAMPLENwGPy
   ```

```
Oj59pFA4lNKCIkVgkREXAMPLEjlzxQ7y52gekeVEXAMPLEDiB9ST3UuysgsKdEXAMPLE1TVastU1A0SKFEXAMPLEiyw
C
s8EXAMPLEpZgOs+6hz4AP4KEXAMPLERbASP+4eZScEXAMPLEsnf87eNhyDHq6ikBQ==

PS C:\> $Creds.Expiration
Thursday, June 18, 2018 2:28:31 PM
```

3. Para utilizar estas credenciales para cualquier comando posterior, inclúyalas en el parámetro `-Credential`. Por ejemplo, el comando siguiente utiliza las credenciales del rol y trabaja únicamente si se concede al rol el permiso `iam:ListRoles` y, por lo tanto, puede ejecutar el cmdlet `Get-IAMRoles`:

```
PS C:\> get-iamroles -Credential $Creds
```

4. Para volver a tener sus credenciales originales, solo tiene que dejar de utilizar el parámetro `-Credentials $Creds` y permitir que PowerShell revierta a las credenciales que están almacenadas en el perfil predeterminado.

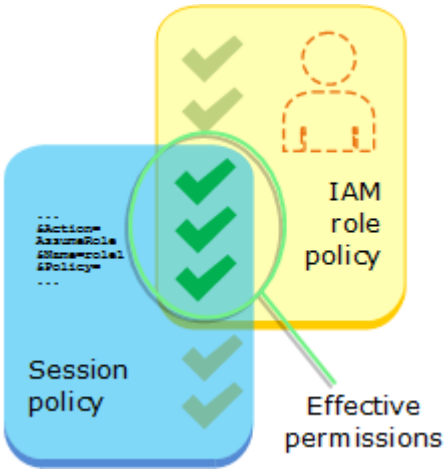
Cambio a un rol de IAM (API de AWS)

Un rol especifica un conjunto de permisos que puede utilizar para el acceso a los recursos de AWS. En este sentido, es similar a un usuario de IAM. Una entidad principal (persona o aplicación) asume un rol para recibir permisos temporales con los que realizar las tareas necesarias e interactuar con los recursos de AWS. El rol puede estar en su propia cuenta o en cualquier otra Cuenta de AWS. Para obtener más información acerca de los roles, sus beneficios y cómo crearlos y configurarlos, consulte Roles de IAM y Creación de roles de IAM. Para obtener más información sobre los distintos métodos que puede utilizar para asumir un rol, consulte Uso de roles de IAM.

> **Important**
>
> Los permisos del usuario de IAM y de cualquier rol que asuma no se acumulan. Solo hay un conjunto de permisos activo a la vez. Cuando se asume un rol, se abandonan temporalmente los permisos de usuario o del rol anteriores y se trabaja con los permisos que el rol tenga asignados. Al salir del rol, los permisos originales se restablecen de forma automática.

Para asumir un rol, una aplicación llama a la operación de la API de AWS STS `AssumeRole` y transfiere el ARN del rol que se utilizará. La operación crea una nueva sesión con credenciales temporales. Esta sesión tiene los mismos permisos que las políticas basadas en identidad aplicables al rol.

Cuando se llama a `AssumeRole`, también se puede pasar opcionalmente una política de sesión gestionada o insertada. Las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión de credenciales temporal mediante programación para un rol o un usuario federado. Puede transferir un único documento de política de sesión insertada JSON utilizando el parámetro `Policy`. Puede utilizar el parámetro `PolicyArns` para especificar hasta 10 políticas de sesión administrada. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la entidad y las políticas de la sesión. Las políticas de sesión son útiles cuando es necesario proporcionar las credenciales temporales del rol a otra persona. Pueden utilizar las credenciales temporales del rol en las llamadas posteriores a la API de AWS para tener acceso a los recursos de la cuenta propietaria del rol. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades. Para obtener más información sobre cómo determina AWS los permisos efectivos de un rol, consulte Lógica de evaluación de políticas.



Puede llamar a `AssumeRole` cuando haya iniciado sesión como usuario de IAM, o como usuario autenticado externamente (SAML o OIDC) que ya utiliza un rol. También puede utilizar el encadenamiento de roles, que consiste en utilizar un rol para asumir otro. No puede asumir un rol si ha iniciado sesión como usuario Usuario raíz de la cuenta de AWS.

De forma predeterminada, la sesión de rol dura una hora. Cuando se asume este rol utilizando las operaciones de la API de AWS STS `AssumeRole*`, se puede especificar un valor para el parámetro `DurationSeconds`. Este valor puede oscilar entre 900 segundos (15 minutos) y el valor de la

duración máxima de la sesión para el rol. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

Si utiliza el encadenamiento de roles, la sesión tiene una duración máxima de una hora. Si utiliza a continuación el parámetro `DurationSeconds` para proporcionar un valor superior a una hora, la operación generará un error.

> **Note**
>
> Por motivos de seguridad, los administradores pueden revisar los registros de AWS CloudTrail para saber quién realizó una acción en AWS. Es posible que el administrador requiera que especifique una identidad de origen o un nombre de la sesión de rol cuando asuma el rol. Para obtener más información, consulte `sts:SourceIdentity` y `sts:RoleSessionName`.

Los siguientes ejemplos de código muestran cómo crear un usuario y asumir un rol.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Crear un usuario que no tenga permisos.
- Crear un rol que conceda permiso para enumerar los buckets de Amazon S3 para la cuenta.
- Agregar una política para que el usuario asuma el rol.
- Asumir el rol y enumerar los buckets de S3 con credenciales temporales, y después limpiar los recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
```

```
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
```

```
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
```

```
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
 { UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
```

```
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

```
    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
 { UserName = userName });
        return response.User;
```

```
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
```

```
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
```

```
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
 policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
```

```
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
```

```
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
```

```
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
              $" \"AWS\": \"{userArn}\"" +
              "}," +
              "\"Action\": \"sts:AssumeRole\"" +
          "}]" +
        "}";

        // Permissions to list all buckets.
        string policyDocument = "{" +
            "\"Version\": \"2012-10-17\"," +
            " \"Statement\" : [{" +
                " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
                " \"Effect\" : \"Allow\"," +
                " \"Resource\" : \"*\"" +
            "}]" +
        "}";

        // Create an AccessKey for the user.
        uiWrapper.DisplayTitle("Create access key");
        Console.WriteLine("Now let's create an access key for the new user.");
        var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

        var accessKeyId = accessKey.AccessKeyId;
        var secretAccessKey = accessKey.SecretAccessKey;

        Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");
```

```
        Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
        var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

        // Now try listing the Amazon Simple Storage Service (Amazon S3)
        // buckets. This should fail at this point because the user doesn't
        // have permissions to perform this task.
        uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
        Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
        var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
        var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
        var buckets = await s3Wrapper.ListMyBucketsAsync();

        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);
```

```
        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();
```

```
        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
 complete.");
        Console.WriteLine("Please wait while we clean up the resources we
 created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();

        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
```

```
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
```

```
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
```

```
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
```

```
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
```

```
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
```

```
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
```

```
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
```

```
 fi

 local assume_role_policy_document="{
              \"Version\": \"2012-10-17\",
              \"Statement\": [{
                  \"Effect\": \"Allow\",
                  \"Action\": \"sts:AssumeRole\",
                  \"Resource\": \"$role_arn\"}]}"

 local assume_role_policy_name
 assume_role_policy_name=$(generate_random_name "test-assume-role-")

 # shellcheck disable=SC2181
 local assume_role_policy_arn
 assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)
```

```
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials

 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
```

```
    local bucket_count
    bucket_count=$(echo "$buckets" | wc -w | xargs)
    echo "There are $bucket_count buckets in the account. Listing buckets
 succeeded because of "
    echo "the assumed role."
  else
    errecho "Failed to list buckets. This should not happen."
    export AWS_ACCESS_KEY_ID=""
    export AWS_SECRET_ACCESS_KEY=""
    export AWS_SESSION_TOKEN=""
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
 "$policy_arn" "$assume_role_policy_arn"
    return 1
  fi

  local result=0
  export AWS_ACCESS_KEY_ID=""
  export AWS_SECRET_ACCESS_KEY=""

  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Las funciones de IAM que se usan en este escenario.

```
###############################################################################
# function iam_user_exists
#
```

```
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)

  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
```

```
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
```

```
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
# bashsupport disable=BP5008
function usage() {
  echo "function iam_create_user_access_key"
  echo "Creates an AWS Identity and Access Management (IAM) key pair."
  echo "  -u user_name   The name of the IAM user."
  echo "  [-f file_name]   Optional file name for the access key output."
  echo ""
}

# Retrieve the calling parameters.
while getopts "u:f:h" option; do
  case "${option}" in
    u) user_name="${OPTARG}" ;;
    f) file_name="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

response=$(aws iam create-access-key \
  --user-name "$user_name" \
  --output text)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-access-key operation failed.$response"
  return 1
```

```
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }
```

```
# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-role \
  --role-name "$role_name" \
  --assume-role-policy-document "$policy_document" \
  --output text \
  --query Role.Arn)

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports create-role operation failed.\n$response"
  return 1
fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
```

```
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
```

```
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
```

```
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam detach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
```

```
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
```

```
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
```

```
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$user_name" ]]; then
  errecho "ERROR: You must provide a username with the -u parameter."
  usage
  return 1
fi

if [[ -z "$access_key" ]]; then
  errecho "ERROR: You must provide an access key with the -k parameter."
  usage
  return 1
fi

iecho "Parameters:\n"
iecho "    Username:   $user_name"
iecho "    Access key:   $access_key"
iecho ""

response=$(aws iam delete-access-key \
  --user-name "$user_name" \
  --access-key-id "$access_key")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
  return 1
fi

iecho "delete-access-key response:$response"
iecho
```

```
  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Para obtener información de la API, consulte los siguientes temas en la Referencia de comandos de AWS CLI.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole

- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;
```

```
    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }
```

```
    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
```

```
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
```

```
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;

                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
```

```
                        listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
```

```
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
 role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
```

```
   bool result = true;
   if (policy.ArnHasBeenSet()) {
       // Detach the policy from the role.
       {
           Aws::IAM::Model::DetachRolePolicyRequest request;
           request.SetPolicyArn(policy.GetArn());
           request.SetRoleName(role.GetRoleName());

           Aws::IAM::Model::DetachRolePolicyOutcome outcome =
client.DetachRolePolicy(
                   request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error Detaching policy from roles. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully detached the policy with arn "
                         << policy.GetArn()
                         << " from role " << role.GetRoleName() << "." <<
std::endl;
           }
       }

       // Delete the policy.
       {
           Aws::IAM::Model::DeletePolicyRequest request;
           request.WithPolicyArn(policy.GetArn());

           Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
           if (!outcome.IsSuccess()) {
               std::cerr << "Error deleting policy. " <<
                         outcome.GetError().GetMessage() << std::endl;
               result = false;
           }
           else {
               std::cout << "Successfully deleted the policy with arn "
                         << policy.GetArn() << std::endl;
           }
       }

   }
```

```
    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for C++.
  - AttachRolePolicy
  - CreateAccessKey

- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute un escenario interactivo en un símbolo del sistema.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
```

```
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
 defer func() {
  if r := recover(); r != nil {
```

```
    log.Printf("Something went wrong with the demo.\n")
    log.Println(r)
   }
  }()

  log.Println(strings.Repeat("-", 88))
  log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
  demo.")
  log.Println(strings.Repeat("-", 88))

  user := scenario.CreateUser()
  accessKey := scenario.CreateAccessKey(user)
  role := scenario.CreateRoleAndPolicies(user)
  noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
  scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
  scenario.Cleanup(user, role)

  log.Println(strings.Repeat("-", 88))
  log.Println("Thanks for watching!")
  log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
  log.Println("Let's create an example user with no permissions.")
  userName := scenario.questioner.Ask("Enter a name for the example user:",
  demotools.NotEmpty{})
  user, err := scenario.userWrapper.GetUser(userName)
  if err != nil {
   panic(err)
  }
  if user == nil {
   user, err = scenario.userWrapper.CreateUser(userName)
   if err != nil {
    panic(err)
   }
   log.Printf("Created user %v.\n", *user.UserName)
  } else {
   log.Printf("User %v already exists.\n", *user.UserName)
  }
  log.Println(strings.Repeat("-", 88))
  return user
}
```

```
// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
```

```
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
```

```
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}

// ListBucketsWithAssumedRole performs the following actions:
//
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
```

```
  ),
 )
 if err != nil {panic(err)}

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
     *policy.PolicyName, *role.RoleName)
   }
   err = scenario.roleWrapper.DeleteRole(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)

   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
```

```
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
    if err != nil {panic(err)}
    log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
   }
   keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
    if err != nil {panic(err)}
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Defina una estructura que incluya las acciones de la cuenta.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
```

```
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Defina una estructura que incluya las acciones de la política.

```

// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
```

```
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
```

```
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
```

```
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Defina una estructura que incluya las acciones de rol.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
```

```
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
```

```
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
```

```
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
```

```
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Defina una estructura que incluya las acciones del usuario.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
```

```
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
```

```
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
```

```
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
```

```
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Go.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey

- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del usuario de IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */
```

```
public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static String userArn;

    public static void main(String[] args) throws Exception {

        final String usage = """

                Usage:
                    <username> <policyName> <roleName> <roleSessionName>
 <bucketName>\s

                Where:
                    username - The name of the IAM user to create.\s
                    policyName - The name of the policy to create.\s
                    roleName - The name of the role to create.\s
                    roleSessionName - The name of the session required for the
 assumeRole operation.\s
                    bucketName - The name of the Amazon S3 bucket from which
 objects are read.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        String policyName = args[1];
        String roleName = args[2];
```

```
        String roleSessionName = args[3];
        String bucketName = args[4];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the AWS IAM example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
```

```
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
```

```
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();
```

```
            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();
```

```
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();

            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
   public static void deleteRole(IamClient iam, String roleName, String polArn)
{

       try {
           // First the policy needs to be detached.
           DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                   .policyArn(polArn)
                   .roleName(roleName)
                   .build();

           iam.detachRolePolicy(rolePolicyRequest);

           // Delete the policy.
           DeletePolicyRequest request = DeletePolicyRequest.builder()
                   .policyArn(polArn)
                   .build();

           iam.deletePolicy(request);
           System.out.println("*** Successfully deleted " + polArn);

           // Delete the role.
           DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                   .roleName(roleName)
                   .build();

           iam.deleteRole(roleRequest);
           System.out.println("*** Successfully deleted " + roleName);

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   public static void deleteKey(IamClient iam, String username, String
accessKey) {
       try {
           DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                   .accessKeyId(accessKey)
                   .userName(username)
                   .build();

           iam.deleteAccessKey(request);
```

```
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
import {
  CreateUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
```

```
  const { User } = await iamClient.send(
    new CreateUserCommand({ UserName: userName }),
  );

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
(AWS STS).
  // It's not best practice to use access keys. For more information, see
https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
```

```
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
          AssumeRolePolicyDocument: JSON.stringify({
            Version: "2012-10-17",
            Statement: [
              {
                Effect: "Allow",
                Principal: {
                  // Allow the previously created user to assume this role.
                  AWS: User.Arn,
                },
                Action: "sts:AssumeRole",
              },
            ],
          }),
          RoleName: roleName,
        }),
      ),
  );

  if (!Role) {
    throw new Error("Role not created");
  }

  // Create a policy that allows the user to list S3 buckets.
  const { Policy: listBucketPolicy } = await iamClient.send(
    new CreatePolicyCommand({
      PolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
```

```
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
await iamClient.send(
  new AttachRolePolicyCommand({
    PolicyArn: listBucketPolicy.Arn,
    RoleName: Role.RoleName,
  }),
);

// Assume the role.
const stsClient = new STSClient({
  credentials: {
    accessKeyId: AccessKeyId,
    secretAccessKey: SecretAccessKey,
  },
});

// Retry the assume role operation until it succeeds.
const { Credentials } = await retry(
  { intervalInMs: 2000, maxRetries: 60 },
  () =>
    stsClient.send(
      new AssumeRoleCommand({
        RoleArn: Role.Arn,
        RoleSessionName: `iamBasicScenarioSession-${Math.floor(
          Math.random() * 1000000,
        )}`,
        DurationSeconds: 900,
      }),
    ),
);
```

```
  if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
    throw new Error("Credentials not created");
  }

  s3Client = new S3Client({
    credentials: {
      accessKeyId: Credentials.AccessKeyId,
      secretAccessKey: Credentials.SecretAccessKey,
      sessionToken: Credentials.SessionToken,
    },
  });

  // List the S3 buckets again.
  // Retry the list buckets operation until it succeeds. AccessDenied might
  // be thrown while the role policy is still stabilizing.
  await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
    listBuckets(s3Client),
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
```

```
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for JavaScript.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del usuario de IAM.

```
suspend fun main(args: Array<String>) {

    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
```

```
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {

    val request = CreateUserRequest {
        userName = usernameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {

    val policyDocumentValue: String = "{" +
        "  \"Version\": \"2012-10-17\"," +
        "  \"Statement\": [" +
        "    {" +
        "        \"Effect\": \"Allow\"," +
        "        \"Action\": [" +
        "            \"s3:*\"" +
```

```
        "       ]," +
        "       \"Resource\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentValue
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}

suspend fun createRole(rolenameVal: String?, fileLocation: String?): String? {

    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request = CreateRoleRequest {
        roleName = rolenameVal
        assumeRolePolicyDocument = jsonObject.toJSONString()
        description = "Created using the AWS SDK for Kotlin"
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(roleNameVal: String, policyArnVal: String) {

    val request = ListAttachedRolePoliciesRequest {
        roleName = roleNameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
```

```
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1)
                return
        }

        val policyRequest = AttachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkMyList(attachedPolicies: List<AttachedPolicy>, policyArnVal: String):
 Int {

    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(roleArnVal: String?, roleSessionNameVal: String?,
 bucketName: String) {

    val stsClient = StsClient {
        region = "us-east-1"
    }

    val roleRequest = AssumeRoleRequest {
        roleArn = roleArnVal
        roleSessionName = roleSessionNameVal
    }

    val roleResponse = stsClient.assumeRole(roleRequest)
```

```
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client {
        credentialsProvider = staticCredentials
        region = "us-east-1"
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects = ListObjectsRequest {
        bucket = bucketName
    }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(roleNameVal: String, polArn: String) {

    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest = DetachRolePolicyRequest {
        policyArn = polArn
        roleName = roleNameVal
    }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
```

```
    val request = DeletePolicyRequest {
        policyArn = polArn
    }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest = DeleteRoleRequest {
        roleName = roleNameVal
    }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request = DeleteUserRequest {
        userName = userNameVal
    }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser

- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("---------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("---------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";
```

```
$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
```

```
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for PHP.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy

- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
```

```
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 resource
                         that has permissions to create users, roles, and
 policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
```

```
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
```

```
            f"Couldn't create a policy and attach it to role {role.name}. Here's
 why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
 "
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
 "
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
```

```
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
```

```
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
```

```
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
```

```
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
 user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Principal: {'AWS': user.arn},
                   Action: "sts:AssumeRole"
                 }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
```

```
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "s3:ListAllMyBuckets",
                   Resource: "arn:aws:s3:::*"
                 }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "sts:AssumeRole",
                   Resource: role.arn
                 }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
```

```
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
  #
  # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
  def create_s3_resource(credentials)
    Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
  end

  # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
  # Because the resource uses credentials with limited access, it may not be able
to
  # list the S3 buckets.
  #
  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def list_buckets(s3_resource)
    count = 10
    s3_resource.buckets.each do |bucket|
      @logger.info "\t#{bucket.name}"
      count -= 1
      break if count.zero?
    end
  rescue Aws::Errors::ServiceError => e
    if e.code == "AccessDenied"
      puts("Attempt to list buckets with no permissions: AccessDenied.")
    else
      @logger.info("Couldn't list buckets for the account. Here's why: ")
      @logger.info("\t#{e.code}: #{e.message}")
      raise
    end
  end

  # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
```

```
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
```

```
    @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
 why:")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes a user. If the user has inline policies or access keys, they are
 deleted
  # before the user is deleted.
  #
  # @param user [Aws::IAM::User] The user to delete.
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
```

```
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Ruby.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
```

```
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;
```

```
    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
```

```
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
```

```
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
```

```
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2

Las aplicaciones que se ejecutan en una instancia de Amazon EC2 deben incluir credenciales de AWS en sus solicitudes de API de AWS. Los desarrolladores pueden almacenar las credenciales de AWS directamente en la instancia de Amazon EC2 y permitir que las aplicaciones de dicha instancia las utilicen. Sin embargo, los desarrolladores tendrían que encargarse de administrar las

credenciales, así como asegurarse de transferirlas de forma segura a cada instancia y actualizar cada instancia de Amazon EC2 cuando haya que actualizar las credenciales. Todo esto supone mucho trabajo adicional.

En su lugar, puede y debe utilizar un rol de IAM para administrar las credenciales temporales de las aplicaciones que se ejecutan en una instancia de Amazon EC2. Al utilizar un rol, no tiene que distribuir credenciales a largo plazo (como credenciales de inicio de sesión o claves de acceso) a una instancia de Amazon EC2. En vez de ello, el rol proporciona permisos temporales que las aplicaciones pueden utilizar al realizar llamadas a otros recursos de AWS. Cuando lanza una instancia de Amazon EC2, usted especifica un rol de IAM que se asocia a la instancia. Esto permite a las aplicaciones que se ejecutan en dicha instancia utilizar las credenciales temporales facilitadas por el rol para firmar las solicitudes de API.

El uso de roles para conceder permisos a aplicaciones que se ejecutan en instancias de Amazon EC2 requiere una configuración adicional. El sistema operativo virtualizado extrae de AWS una aplicación que se ejecuta en una instancia de Amazon EC2. Debido a esta separación adicional, necesita un paso más para asignar un rol de AWS y sus permisos asociados a una instancia de Amazon EC2 y para ponerlos a disposición de las aplicaciones. Este paso adicional consiste en crear un perfil de instancia asociado a la instancia. El perfil de instancia contiene el rol y puede proporcionar las credenciales temporales de este a una aplicación que se ejecute en la instancia. La aplicación, a su vez, puede utilizar estas credenciales temporales en las llamadas a la API para obtener acceso a los recursos y para restringir el acceso únicamente a aquellos recursos que el rol especifica.

> **Note**
>
> Solo se puede asignar un rol a la vez a una instancia de EC2 y todas las aplicaciones en la instancia comparten ese mismo rol y permisos. Cuando aprovecha Amazon ECS para administrar las instancias de Amazon EC2, puede asignar roles a las tareas de Amazon ECS que pueden distinguirse del rol de la instancia de Amazon EC2 en la que se ejecutan. La asignación de un rol a cada tarea se ajusta al principio de acceso con privilegio mínimo y permite un mayor control pormenorizado de acciones y recursos.
> Para obtener más información, consulte Uso de roles de IAM con tareas de Amazon ECS en la Guía de prácticas recomendadas para Amazon Elastic Container Service.

Este uso de los roles tiene varios beneficios. Dado que las credenciales del rol son temporales y se actualizan de forma automática, no es necesario administrar las credenciales ni preocuparse por

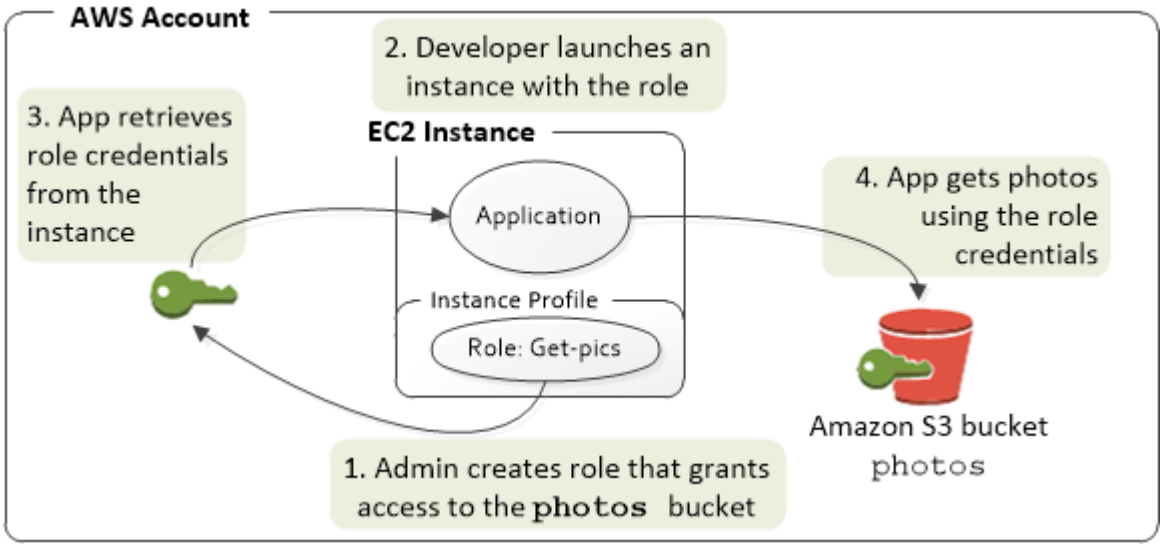
los riesgos de seguridad a largo plazo. Además, si utiliza un único rol para varias instancias, puede especificar un cambio en ese rol y el cambio se propagará automáticamente a todas las instancias.

> **Note**
>
> Aunque normalmente un rol se asigna a una instancia de Amazon EC2 cuando esta se lanza, también se puede asociar un rol a una instancia de Amazon EC2 que ya esté en ejecución. Para obtener información sobre cómo asociar una función a una instancia en ejecución, consulte Roles de IAM para Amazon EC2.

Temas

¿Cómo funcionan los roles de instancias de Amazon EC2?

En la figura siguiente, un desarrollador ejecuta una aplicación en una instancia de Amazon EC2 que necesita obtener acceso a un bucket de S3 denominado `photos`. Un administrador crea el rol de servicio `Get-pics` y lo asocia a la instancia de Amazon EC2. El rol incluye una política de permisos que otorga acceso de solo lectura al bucket de S3 especificado. También incluye una política de confianza que permite a la instancia de Amazon EC2 asumir el rol y obtener las credenciales temporales. Cuando la aplicación se ejecuta en la instancia, puede utilizar las credenciales temporales del rol para obtener acceso al bucket photos. El administrador no tiene que conceder al desarrollador permiso para acceder al bucket photos y el desarrollador no tiene que compartir ni administrar en ningún momento las credenciales.



1. El administrador usa IAM para crear el rol **`Get-pics`**. En la política de confianza del rol, el administrador especifica que solo las instancias de Amazon EC2 pueden asumir el rol. En la política de permisos del rol, el administrador especifica permisos de solo lectura para el bucket `photos`.
2. Un desarrollador lanza una instancia de Amazon EC2 y asigna el rol `Get-pics` a dicha instancia.

   > **Note**
   >
   > Si utiliza la consola de IAM, el perfil de instancias se administra de forma prácticamente transparente en su lugar. Sin embargo, si utiliza la AWS CLI o la API para crear y administrar el rol y la instancia de Amazon EC2, debe crear el perfil de instancia y asignarle el rol en pasos diferentes. A continuación, al lanzar la instancia, debe especificar el nombre del perfil de instancia en lugar del nombre del rol.

3. Cuando se ejecuta la aplicación, esta obtiene credenciales de seguridad temporales desde los metadatos de instancias de Amazon EC2, tal y como se describe en Recuperación de las credenciales de seguridad en los metadatos de la instancia. Se trata de las credenciales de seguridad temporales que representan el rol y son válidas durante un periodo de tiempo limitado.

   Con algunos AWS SDKs, el desarrollador puede utilizar un proveedor que administre las credenciales de seguridad temporales de forma transparente. (La documentación de los SDK AWS individuales describe las características compatibles con el SDK de administración de credenciales).

De forma alternativa, la aplicación puede obtener credenciales temporales directamente desde los metadatos de la instancia de Amazon EC2. Las credenciales y los valores asociados están disponibles en la categoría `iam/security-credentials/`*`role-name`* (en este caso, `iam/security-credentials/Get-pics`) de los metadatos. Si la aplicación obtiene las credenciales de los metadatos de la instancia, puede almacenar en caché las credenciales.

4. Con las credenciales temporales recuperadas, la aplicación obtiene acceso al bucket photo. Debido a la política asociada al rol **`Get-pics`**, la aplicación tiene permisos de solo lectura.

   Las credenciales de seguridad temporales disponibles en la instancia se actualizan de manera automática antes de caducar, de modo que siempre haya un conjunto válido disponible. La aplicación solo debe asegurarse de obtener un conjunto nuevo de credenciales de los metadatos de la instancia antes de que las credenciales actuales caduquen. Es posible utilizar el AWS SDK para administrar credenciales, de modo que la aplicación no necesite incluir lógica adicional para actualizar las credenciales. Por ejemplo, crear instancias de clientes con proveedores de credenciales de perfil de instancia. No obstante, si la aplicación obtiene las credenciales de seguridad temporales de los metadatos de la instancia y los almacena en caché, debería obtener un conjunto de credenciales actualizado cada hora, o al menos 15 minutos antes de que el conjunto en curso caduque. El plazo de vencimiento está indicado en la información devuelta, en la categoría `iam/security-credentials/`*`role-name`*.

Permisos necesarios para utilizar roles con Amazon EC2

Para lanzar una instancia con un rol, el desarrollador debe tener permiso para lanzar instancias de Amazon EC2 y transferir roles de IAM.

La siguiente política de ejemplo permite a los usuarios utilizar la AWS Management Console para lanzar una instancia con un rol. La política incluye comodines (*) para permitir a los usuarios transferir cualquier rol y llevar a cabo las acciones de Amazon EC2 mencionadas. La acción `ListInstanceProfiles` permite a los usuarios ver todos los roles disponibles en la Cuenta de AWS.

Example Ejemplo de política que concede permiso a un usuario para utilizar la consola de Amazon EC2 para lanzar una instancia con cualquier rol

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "IamPassRole",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        },
        {
            "Sid": "ListEc2AndListInstanceProfiles",
            "Effect": "Allow",
            "Action": [
                "iam:ListInstanceProfiles",
                "ec2:Describe*",
                "ec2:Search*",
                "ec2:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

Restricción de los roles que se pueden transferir a instancias de Amazon EC2 (usando PassRole)

Puede utilizar el permiso `PassRole` para restringir qué rol puede transferir un usuario a una instancia de Amazon EC2 cuando el usuario lanza la instancia. Esto es útil para evitar que el usuario ejecute aplicaciones que tengan más permisos de los que se le han concedido; es decir, evita que se puedan obtener privilegios superiores. Por ejemplo, supongamos que la usuaria Alice tenga permiso solo para lanzar instancias de Amazon EC2 y trabajar con buckets de Amazon S3, pero el rol que pasa a una instancia de Amazon EC2 tiene permisos para trabajar con IAM y Amazon DynamoDB. En este caso, Alice podría lanzar la instancia, iniciar sesión en ella, obtener credenciales de seguridad temporales y, a continuación, realizar acciones de IAM o DynamoDB para las que carece de autorización.

Para restringir los roles que un usuario puede transferir a una instancia de Amazon EC2, debe crear una política que permita la acción `PassRole`. A continuación, debe asociar la política al usuario (o a un grupo de IAM al que pertenezca el usuario), el cual, a su vez, lanzará las instancias de Amazon EC2. En el elemento `Resource` de la política, se genera una lista con el rol o los roles que el usuario tiene permiso para transferir a las instancias de Amazon EC2. Cuando el usuario lanza una instancia

y le asocia un rol, Amazon EC2 comprueba si el usuario tiene permiso para transmitir dicho rol. Desde luego, también debe asegurarse de que el rol que el usuario puede transferir no incluya más permisos de los que se supone que debe tener el usuario.

> **Note**
>
> `PassRole` no es una acción de API de la misma forma que `RunInstances` o `ListInstanceProfiles`. Se trata de un permiso que AWS comprueba siempre que un ARN de rol se transfiere como parámetro a una API (o la consola se encarga de ello en nombre del usuario). Sirve de ayuda para que el administrador controle qué roles pueden pasar según qué usuarios. En este caso, se asegura de que el usuario pueda asociar un rol específico a una instancia de Amazon EC2.

Example Ejemplo de política que concede permiso a un usuario para lanzar una instancia de Amazon EC2 con un rol concreto

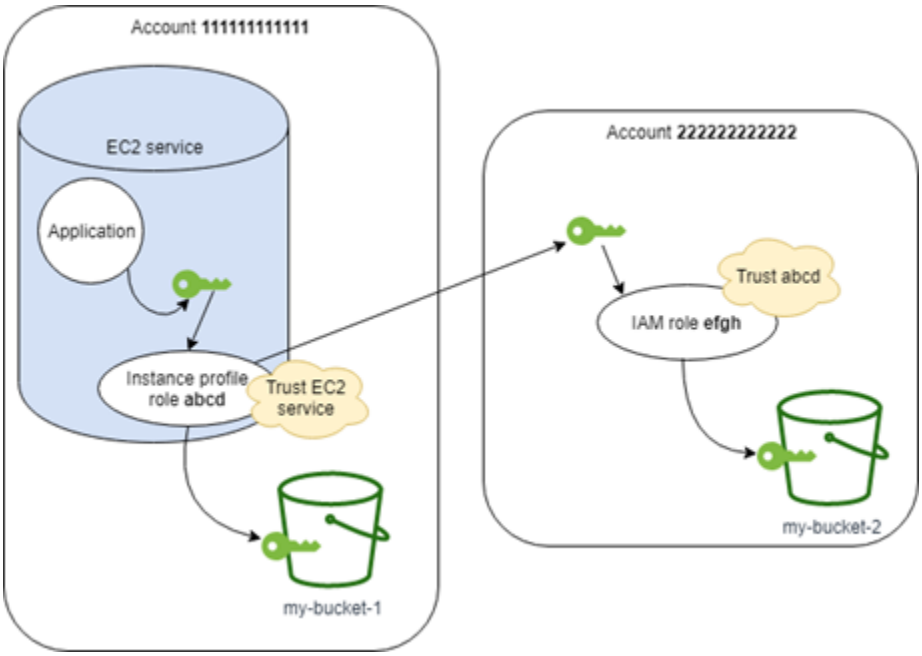
La siguiente política de ejemplo permite a los usuarios utilizar la API de Amazon EC2 para lanzar una instancia con un rol. El elemento `Resource` especifica el nombre de recurso de Amazon (ARN) de un rol. Al especificar el ARN, la política concede al usuario permiso para transferir únicamente el rol `Get-pics`. Si el usuario intenta especificar otro rol al lanzar una instancia, la acción dará un error. El usuario tiene permisos para ejecutar cualquier instancia, independientemente de si transmiten una función.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/Get-pics"
    }
  ]
}
```

Permitir que una función de perfil de instancia cambie una función en otra cuenta

Puede permitir que una aplicación que se ejecuta en una instancia de Amazon EC2 ejecute comandos en otra cuenta. Para ello, debe permitir el rol de instancia de Amazon EC2 en la primera cuenta para cambiar a una función en la segunda cuenta.

Imagine que está utilizando dos Cuentas de AWS y desea permitir que una aplicación se ejecute en una instancia de Amazon EC2 para ejecutar comandos AWS CLI en ambas cuentas. Supongamos que la instancia de Amazon EC2 existe en la cuenta `111111111111`. Dicha instancia incluye la función de perfil de instancias `abcd` que permite que la aplicación realice tareas de solo lectura de Amazon S3 en el bucket `my-bucket-1` dentro de la misma cuenta `111111111111`. Sin embargo, la aplicación también debe tener permitido asumir la función entre cuentas `efgh` para acceder al bucket `my-bucket-2` Amazon S3 de la cuenta `222222222222`.



Para ello, la función del perfil de instancia de Amazon EC2 `abcd` debe tener la siguiente política de permisos para permitir que la aplicación acceda al bucket `my-bucket-1` Amazon S3:

Política de permisos del rol de la cuenta 111111111111 ***abcd***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "Sid": "AllowIPToAssumeCrossAccountRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::222222222222:role/efgh"
        }
    ]
}
```

El rol `abcd` debe confiar en el servicio Amazon EC2 para asumir la función. Para ello, la función `abcd` debe tener la siguiente política de confianza:

Política de confianza de rol ***abcd*** de la cuenta 111111111111

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "abcdTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"Service": "ec2.amazonaws.com"}
        }
    ]
```

```
}
```

Supongamos que la función entre cuentas `efgh` permite tareas de solo lectura de Amazon S3 en el bucket `my-bucket-2` dentro de la misma cuenta `222222222222`. Para ello, la función entre cuentas `efgh` debe tener la siguiente política de permisos:

Política de permisos del rol de la cuenta 222222222222 ***efgh***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAccountLevelS3Actions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowListAndReadS3ActionOnMyBucket",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

La función `efgh` debe confiar en la función de perfil de instancia `abcd` para asumirla. Para ello, la función `efgh` debe tener la siguiente política de confianza:

Política de confianza de rol ***efgh*** de la cuenta 222222222222

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "efghTrustPolicy",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111111111111:role/abcd"}
        }
    ]
}
```

¿Cómo puedo comenzar?

Para comprender cómo funcionan los roles con instancias de Amazon EC2, debe utilizar la consola de IAM para crear un rol, lanzar una instancia de Amazon EC2 que utilice dicho rol y, a continuación, estudiar la instancia mientras se ejecuta. Puede examinar los metadatos de la instancia para ver cómo se ponen las credenciales temporales de la función a disposición de la instancia. También puede ver cómo una aplicación que ejecuta una instancia puede utilizar el rol. Utilice los siguientes recursos para obtener más información.

-
- Tutoriales de SDK. La documentación del SDK de AWS contiene explicaciones que muestran una aplicación que se ejecuta en una instancia de Amazon EC2 que utiliza credenciales temporales para los roles para leer un bucket de Amazon S3. Cada uno de los siguientes tutoriales presenta pasos similares con un lenguaje de programación diferente:
  - Configuración de roles de IAM para Amazon EC2 con el SDK para Java en la Guía para desarrolladores de AWS SDK for Java
  - Inicie una instancia de Amazon EC2 utilizando el SDK para .NET en la Guía para el desarrollador de AWS SDK for .NET
  - Creación de una instancia de Amazon EC2 con el SDK para Ruby en la Guía para el desarrollador de AWS SDK for Ruby

Información relacionada

Para obtener más información sobre cómo crear roles o roles para instancias de Amazon EC2, consulte la siguiente información:

- Para obtener más información acerca de Uso de roles de IAM con instancias Amazon EC2, diríjase a la Guía del usuario de Amazon EC2 para instancias de Linux.

- Para crear un rol, consulte Creación de roles de IAM
- Para obtener más información sobre cómo utilizar credenciales de seguridad temporales, consulte Credenciales de seguridad temporales en IAM.
- Si trabaja con la API o la CLI IAM, debe crear y administrar perfiles de instancia de IAM. Para obtener más información sobre los perfiles de instancia, consulte Uso de perfiles de instancia.
- Para obtener más información sobre las credenciales de seguridad temporales para funciones en los metadatos de la instancia, consulte Recuperación de las credenciales de seguridad en los metadatos de la instancia en la Guía del usuario de Amazon EC2 para Linux Instances.

Uso de perfiles de instancia

Utilice un perfil de instancias para pasar un rol de IAM a una instancia EC2. Para obtener más información, consulte Roles de IAM para Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux.

Administración de perfiles de instancia (consola)

Si utiliza la AWS Management Console para crear un rol para Amazon EC2, la consola crea automáticamente un perfil de instancias y le da el mismo nombre que al rol. Después, cuando utilice la consola Amazon EC2 para lanzar una instancia con un rol de IAM, puede seleccionar una función para asociarlo con la instancia. En la consola, la lista que se muestra es en realidad una lista de nombres de perfiles de instancia. La consola no crea un perfil de instancias para un rol que no está asociado a Amazon EC2.

Puede utilizar el AWS Management Console para eliminar roles de IAM y perfiles de instancia de Amazon EC2 si el rol y el perfil de instancias tienen el mismo nombre. Para obtener más información acerca de la eliminación de perfiles de instancia, consulte Eliminación de roles o perfiles de instancia.

Administración de perfiles de instancias (AWS CLI o API de AWS)

Si administra sus roles en la AWS CLI o la API de AWS, crea los roles y los perfiles de instancias como acciones independientes. Debido a que los roles y los perfiles de instancias puede tener nombres diferentes, debe saber los nombres de sus perfiles de instancia, así como los nombres de los roles que contienen. De esta forma, puede elegir el perfil de instancia correcto cuando lance una instancia EC2.

Puede asociar etiquetas a los recursos de IAM, incluidos los perfiles de instancia, a fin de identificar, organizar y controlar el acceso a ellos. Solo puede etiquetar perfiles de instancia cuando utiliza la AWS CLI o la API de AWS.

> **Note**
>
> Un perfil de instancias puede contener un único rol de IAM, aunque un rol puede incluirse en varios perfiles de instancias. Este límite de un rol por perfil de instancia no puede aumentarse. Puede eliminar el rol existente y, a continuación, agregar un rol diferente a un perfil de instancia. A continuación, debe esperar a que el cambio aparezca en todo AWS, a causa de la consistencia final. Para forzar el cambio, debe desvincular el perfil de instancia y, a continuación, asociar el perfil de instancia, o bien puede detener la instancia y después reiniciarla.

Administración de perfiles de instancias (AWS CLI)

Puede utilizar los siguientes comandos de la AWS CLI para trabajar con perfiles de instancia en una cuenta de AWS.

- Creación de un perfil de instancia: `aws iam create-instance-profile`
- Etiquetado de un perfil de instancia: `aws iam tag-instance-profile`
- Enumeración de etiquetas de un perfil de instancia: `aws iam list-instance-profile-tags`
- Desetiquetado de un perfil de instancia: `aws iam untag-instance-profile`
- Añadir un rol a un perfil de instancia: `aws iam add-role-to-instance-profile`
- Enumeración de perfiles de instancia: `aws iam list-instance-profiles`, `aws iam list-instance-profiles-for-role`
- Obtención de información sobre un perfil de instancia: `aws iam get-instance-profile`
- Eliminación de un rol de un perfil de instancia: `aws iam remove-role-from-instance-profile`
- Eliminación de un perfil de instancia: `aws iam delete-instance-profile`

También puede asociar un rol a una instancia EC2 que ya esté en ejecución ejecutando los siguientes comandos. Para obtener más información, consulte Roles de IAM para Amazon EC2.

- Asociación de un perfil de instancia con un rol a una instancia EC2 en ejecución o detenida: `aws ec2 associate-iam-instance-profile`
- Obtención de información sobre un perfil de instancia asociado a una instancia EC2: `aws ec2 describe-iam-instance-profile-associations`

- Separación de un perfil de instancia con un rol de una instancia EC2 en ejecución o detenida: `aws ec2 disassociate-iam-instance-profile`

Administración de perfiles de instancias (API de AWS)

Puede llamar a las siguientes operaciones de la API de AWS para trabajar con perfiles de instancia en una Cuenta de AWS.

- Creación de un perfil de instancia: `CreateInstanceProfile`
- Etiquetado de un perfil de instancia: `TagInstanceProfile`
- Enumeración de etiquetas de un perfil de instancia: `ListInstanceProfileTags`
- Desetiquetado de un perfil de instancia: `UntagInstanceProfile`
- Añadir un rol a un perfil de instancia: `AddRoleToInstanceProfile`
- Enumeración de perfiles de instancia: `ListInstanceProfiles`, `ListInstanceProfilesForRole`
- Obtención de información sobre un perfil de instancia: `GetInstanceProfile`
- Eliminación de un rol de un perfil de instancia: `RemoveRoleFromInstanceProfile`
- Eliminación de un perfil de instancia: `DeleteInstanceProfile`

También puede asociar un rol a una instancia EC2 que ya esté en ejecución llamando a las siguientes operaciones. Para obtener más información, consulte Roles de IAM para Amazon EC2.

- Asociación de un perfil de instancia con un rol a una instancia EC2 en ejecución o detenida: `AssociateIamInstanceProfile`
- Obtención de información sobre un perfil de instancia asociado a una instancia EC2: `DescribeIamInstanceProfileAssociations`
- Separación de un perfil de instancia con un rol de una instancia EC2 en ejecución o detenida: `DisassociateIamInstanceProfile`

Revocación de las credenciales de seguridad temporales de un rol de IAM

> **Warning**
>
> Si sigue los pasos que se indican en esta página, se denegará el acceso a todas las acciones y recursos de AWS a todos los usuarios con sesiones actuales que se hayan

creado adoptando el rol. Esto puede hacer que los usuarios pierdan el trabajo que no hayan guardado.

Cuando habilita a los usuarios para obtener acceso a la AWS Management Console en una sesión de larga duración (como, por ejemplo, 12 horas), sus credenciales temporales no caducan tan rápidamente. Si los usuarios revelan accidentalmente sus credenciales a un tercero no autorizado, ese tercero podrá obtener acceso mientras dure la sesión. Sin embargo, si es preciso, puede revocar inmediatamente todos los permisos de las credenciales del rol emitidas antes de un momento dado. Todas las credenciales temporales de dicho rol que se hayan emitido antes de ese momento dado dejarán de ser válidas. Esto obliga a todos los usuarios a volver a autenticarse y a solicitar credenciales nuevas.

Note

No se puede revocar la sesión de un rol vinculado a un servicio.

Cuando revoca los permisos de un rol ejecutando el procedimiento indicado en este tema, AWS asocia una política insertada nueva al rol que deniega todos los permisos para todas las acciones. Incluye una condición que aplica las restricciones solo si el usuario asumió el rol antes del momento en que usted revocó los permisos. Si el usuario asume el rol después de que usted revocara los permisos, la política de denegación no se aplica a ese usuario.

Para obtener más información sobre cómo denegar el acceso, consulte Deshabilitar permisos para credenciales de seguridad temporales.

Important

Esta política de denegación se aplica a todos los usuarios del rol especificado y no solo a las sesiones de la consola de más larga duración.

Permisos mínimos para revocar permisos de sesión de un rol.

Para poder revocar efectivamente los permisos de sesión de un rol, debe tener el permiso `PutRolePolicy` para el rol. Esto le permite asociar la política insertada `AWSRevokeOlderSessions` al rol.

Revocación de permisos de sesión.

Puede revocar los permisos de sesión de un rol.

Para denegar inmediatamente todos los permisos a cualquier usuario de credenciales de rol

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y, a continuación, elija el nombre (no la casilla de verificación) del rol cuyos permisos desee revocar.
3. En la página Summary (Resumen) del rol seleccionado, elija la pestaña Revoke sessions (Revocar sesiones).
4. En la pestaña Revoke sessions (Revocar sesiones), seleccione Revoke active sessions (Revocar sesiones activas).
5. AWS le solicitará que confirme la acción. Seleccione la casilla I acknowledge that I am revoking all active sessions for this role (Confirmo que voy a revocar todas las sesiones activas de este rol) y elija Revoke active sessions (Revocar sesiones activas) en el cuadro de diálogo.

   IAM asociará inmediatamente una política llamada `AWSRevokeOlderSessions` al rol. Al elegir Revocar sesiones activas, la política denegará el acceso a los usuarios que asumieron el rol en el pasado y en aproximadamente 30 segundos en el futuro. Esta elección de hora futura tiene en cuenta el retraso de propagación de la política para tratar una nueva sesión que se adquirió o renovó antes de que la política actualizada entre en vigor en una región determinada. Ningún usuario que haya asumido el rol aproximadamente 30 segundos después de que usted haya elegido Revocar sesiones activas no resultará afectado. Para saber por qué los cambios no siempre son visibles inmediatamente, consulte Los cambios que realizo no están siempre visibles inmediatamente.

> **Note**
>
> Si más tarde vuelve a elegir Revocar sesiones, la fecha y la marca temporal de la política se actualizarán y la política volverá a denegar todos los permisos a todos los usuarios que asumieron el rol antes de la nueva hora especificada.

Los usuarios válidos cuyas sesiones se revocan de esta forma deben obtener credenciales temporales para una nueva sesión para poder seguir trabajando. Las credenciales de caché de la

AWS CLI hasta que caducan. Para obligar a la CLI a eliminar y actualizar las credenciales en caché que ya no son válidas, ejecute uno de los comandos siguientes:

Linux, macOS o Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Revocación de los permisos de sesión antes de un tiempo específico

También puede revocar los permisos de sesión en el momento que elija mediante el AWS CLI o el SDK para especificar un valor para la `aws:TokenIssueTime` clave en el elemento Condición de una política.

Esta política deniega todos los permisos cuando el valor de `aws:TokenIssueTime` es anterior a la fecha y la hora especificadas. El valor de `aws:TokenIssueTime` corresponde al intervalo de tiempo exacto en el que se han creado las credenciales de seguridad temporales. El valor de `aws:TokenIssueTime` solo está presente en el contexto de las solicitudes de AWS que se firman con credenciales de seguridad temporales, de modo que la instrucción de denegación de la política no afecta a las solicitudes que se firman con las credenciales a largo plazo del usuario de IAM.

Esta política también puede asociarse a un rol. En tal caso, la política solo afecta a las credenciales de seguridad temporales que el rol ha creado antes de la fecha y la hora especificadas.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "*",
    "Resource": "*",
    "Condition": {
      "DateLessThan": {"aws:TokenIssueTime": "2014-05-07T23:47:00Z"}
    }
  }
}
```

Los usuarios válidos cuyas sesiones se revocan de esta forma deben obtener credenciales temporales para una nueva sesión para poder seguir trabajando. Las credenciales de caché de la

AWS CLI hasta que caducan. Para obligar a la CLI a eliminar y actualizar las credenciales en caché que ya no son válidas, ejecute uno de los comandos siguientes:

Linux, macOS o Unix

```
$ rm -r ~/.aws/cli/cache
```

Windows

```
C:\> del /s /q %UserProfile%\.aws\cli\cache
```

Administración de roles de IAM

A veces es necesario modificar o eliminar los roles que ha creado. Para cambiar un rol, puede hacer lo siguiente:

- Modificar las políticas asociadas al rol
- Cambiar quién puede obtener acceso al rol
- Editar los permisos que el rol concede a los usuarios
- Cambiar el valor de la duración máxima de la sesión para los roles que se asumen mediante la AWS Management Console, AWS CLI o la API

También puede eliminar los roles que ya no son necesarios. Puede administrar sus roles en la AWS Management Console, la AWS CLI y la API.

Temas

- Modificación de un rol
- Eliminación de roles o perfiles de instancia

Modificación de un rol

Puede utilizar la AWS Management Console, la AWS CLI o la API de IAM para realizar cambios en un rol.

Temas

- Ver Acceso de roles

- Generar una política basada en información de acceso
- Modificación de un rol (consola)
- Modificación de un rol (AWS CLI)
- Modificación de un rol (API de AWS)

Ver Acceso de roles

Antes de cambiar los permisos para un rol, debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Generar una política basada en información de acceso

A veces puede conceder permisos a una entidad de IAM (usuario o rol) de más allá de lo que requieren. Para ayudarle a refinar los permisos que concede, puede generar una política de IAM que esté basada en la actividad de acceso de una entidad. El analizador de acceso de IAM revisa los registros de AWS CloudTrail y genera una plantilla de política que contiene los permisos que ha utilizado la entidad en el intervalo de fechas especificado. Puede utilizar la plantilla para crear una política administrada con permisos detallados y, a continuación, adjuntarla a la entidad de IAM. De esta forma, solo concede los permisos que el usuario o rol necesita para interactuar con los recursos de AWS para su caso de uso específico. Para obtener más información, consulte Generar políticas basadas en la actividad de acceso.

Modificación de un rol (consola)

Puede utilizar la AWS Management Console para modificar un rol. Para cambiar el conjunto de etiquetas en un rol, consulte Administrar etiquetas en roles de IAM (consola).

Temas

- Modificación de una política de confianza de rol (consola)
- Modificación de una política de permisos de rol (consola)
- Modificación de una descripción de rol (consola)
- Modificación de la duración máxima de la sesión de un rol (consola)
- Modificación de un límite de permisos de rol (consola)

Modificación de una política de confianza de rol (consola)

Para cambiar quién puede asumir un rol, debe modificar la política de confianza del rol. No se puede modificar la política de confianza de un rol vinculado a un servicio.

> **Notas**
>
> - Si un usuario identificado como entidad principal de una política de confianza de un rol no puede asumir ese rol, compruebe el límite de permisos del usuario. Si hay definido un límite de permisos para el usuario, el límite deberá permitir la acción `sts:AssumeRole`.
> - Para permitir que los usuarios vuelvan a asumir el rol actual dentro de una sesión de rol, especifique el ARN del rol o el ARN de la Cuenta de AWS como entidad principal en la política de confianza de rol. Los Servicios de AWS que proporcionan recursos de computación, como Amazon EC2, Amazon ECS, Amazon EKS y Lambda, brindan credenciales temporales y las actualizan automáticamente. Esto garantiza que siempre disponga de un conjunto de credenciales válido. Para estos servicios, no es necesario volver a asumir el rol actual a fin de obtener credenciales temporales. Sin embargo, si tiene la intención de aprobar etiquetas de sesión o una política de sesión, tendrá que volver a asumir el rol actual.

Para modificar una política de confianza de rol (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles.
3. En la lista de roles de su cuenta, elija el nombre del rol que desee modificar.
4. Elija la pestaña Relaciones de confianza y, a continuación, Editar política de confianza.
5. Edite la política de confianza según sea necesario. Para añadir entidades principales adicionales que puedan asumir el rol, especifíquelas en el elemento `Principal`. Por ejemplo, el siguiente fragmento de política muestra cómo hacer referencia a dos Cuentas de AWS en el elemento `Principal`:

```
"Principal": {
  "AWS": [
```

```
    "arn:aws:iam::111122223333:root",
    "arn:aws:iam::444455556666:root"
  ]
},
```

Si especifica una entidad principal de otra cuenta, el hecho de añadir una cuenta a la política de confianza de un rol solo es una parte del establecimiento de una relación de confianza entre cuentas. De forma predeterminada, ningún usuario de las cuentas de confianza puede asumir el rol. El administrador de la cuenta en la que se acaba de establecer la relación de confianza debe conceder a los usuarios permiso para asumir el rol. Para ello, el administrador debe crear o editar una política que esté asociada al usuario para permitirle a este el acceso a la acción `sts:AssumeRole`. Para obtener más información, consulte el siguiente procedimiento o Conceder permisos de usuario para cambiar de rol.

El siguiente fragmento de política muestra cómo hacer referencia a dos servicios de AWS en el elemento `Principal`:

```
"Principal": {
  "Service": [
    "opsworks.amazonaws.com",
    "ec2.amazonaws.com"
  ]
},
```

6. Cuando haya terminado de editar la política de confianza, elija Update policy (Actualizar política) para guardar los cambios.

   Para obtener más información sobre la estructura y la sintaxis de la política, consulte las secciones Políticas y permisos en IAM y Referencia de los elementos de las políticas de JSON de IAM.

Para permitir que los usuarios de una cuenta externa de confianza utilicen el rol (consola)

Para obtener más información y detalles sobre este procedimiento, consulte Conceder permisos de usuario para cambiar de rol.

1. Inicie sesión en la Cuenta de AWS externa de confianza.
2. Decida si desea asociar los permisos a un usuario o a un grupo. En el panel de navegación de la consola de IAM, elija Users (Usuarios) o User groups (Grupos de usuarios) según corresponda.

3. Elija el nombre del usuario o el grupo al que desea conceder acceso y, a continuación, elija la pestaña Permissions (Permisos).
4. Realice una de las acciones siguientes:

   - Para editar una política administrada por el cliente, elija el nombre de la política, elija Edit policy (Editar política) y, a continuación, elija la pestaña JSON. No se puede editar una política administrada por AWS. Las políticas administradas de AWS aparecen con el icono AWS ().
     Para obtener más información acerca de la diferencia entre las políticas administradas por AWS y las políticas administradas por el cliente, consulte Políticas administradas y políticas insertadas.
   - Para editar una política insertada, seleccione la flecha que aparece junto al nombre de la política y elija Edit Policy (Editar política).
5. En el editor de políticas, añada un elemento `Statement` nuevo que especifique lo siguiente:

   ```
   {
     "Effect": "Allow",
     "Action": "sts:AssumeRole",
     "Resource": "arn:aws:iam::ACCOUNT-ID:role/ROLE-NAME"
   }
   ```

   Reemplace el ARN de la instrucción por el ARN del rol que el usuario puede asumir.
6. Siga las indicaciones que aparecen en pantalla para terminar de editar la política.

Modificación de una política de permisos de rol (consola)

Para cambiar los permisos permitidos por el rol, modifique la política (o políticas) de permisos del rol. No se puede modificar la política de permisos de un rol vinculado a un servicio en IAM. Se podría modificar la política de permisos en el servicio que depende del rol. Para comprobar si un servicio admite esta característica, consulte Servicios de AWS que funcionan con IAM y busque los servicios con Sí en la columna Roles vinculados a servicios. Elija una opción Sí con un enlace para ver la documentación acerca del rol vinculado a servicios en cuestión.

Para cambiar los permisos de un rol (consola)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles.

3. Elija el nombre del rol que desea modificar y, a continuación, la pestaña Permissions (Permisos).
4. Realice una de las acciones siguientes:

   - Para editar una política administrada por el cliente ya existente, elija el nombre de la política y seleccione Edit policy (Editar política).

     > **Note**
     >
     > No se puede editar una política administrada por AWS. La política administrada por AWS aparece con el icono de AWS ().
     >
     > Para obtener más información acerca de la diferencia entre las políticas administradas por AWS y las políticas administradas por el cliente, consulte Políticas administradas y políticas insertadas.

   - Para asociar una política administrada existente al rol, elija Add permissions (Agregar permisos) y luego Attach policies (Asociar políticas).
   - Para editar una política insertada existente, expándala y elija Edit (Editar).
   - Para integrar una nueva política insertada, elija Add permissions (Agregar permisos) y luego Create inline policy (Crear política insertada).

Modificación de una descripción de rol (consola)

Para cambiar la descripción del rol, modifique el texto de descripción.

Para cambiar la descripción del rol (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles.
3. Seleccione el nombre del rol que desea modificar.
4. En la sección Summary (Resumen), elija Edit (Editar).
5. Ingrese una descripción nueva en el cuadro y elija Save changes (Guardar cambios).

Modificación de la duración máxima de la sesión de un rol (consola)

Para especificar la duración máxima de la sesión para los roles que se asumen mediante la consola, AWS CLI o la API de AWS, modifique el valor del ajuste de duración máxima de la sesión. Esta opción puede tener un valor comprendido entre 1 y 12 horas. Si no especifica un valor, se aplicará el valor máximo predeterminado de 1 hora. Esta configuración no limita las sesiones asumidas por los servicios de AWS.

Para cambiar el valor de la duración máxima de la sesión para los roles que se asumen mediante la consola, AWS CLI o la API de AWS (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, elija Roles.
3. Seleccione el nombre del rol que desea modificar.
4. En la sección Summary (Resumen), elija Edit (Editar).
5. En Maximum session duration (Duración máxima de la sesión), elija un valor. Como alternativa, puede elegir Custom duration (Duración personalizada) e ingresar un valor (en segundos).
6. Elija Guardar cambios.

   Los cambios no entrarán en vigor hasta la próxima vez que alguien asuma este rol. Para obtener información sobre cómo revocar sesiones existentes para este rol, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

En AWS Management Console, las sesiones de usuario de IAM son de 12 horas de forma predeterminada. A los usuarios de IAM que cambian de rol en la consola se les concede la duración máxima de la sesión del rol, o el tiempo restante de la sesión del usuario, lo que sea menor.

Cualquier persona que asuma el rol desde el AWS CLI o la API de AWS puede solicitar una sesión más larga, hasta este máximo. El ajuste de la `MaxSessionDuration` determina la duración máxima de la sesión de rol que se puede solicitar.

- Para especificar una duración de sesión utilizando el AWS CLI utilice el parámetro `duration-seconds`. Para obtener más información, consulte Cambio a un rol de IAM (AWS CLI).
- Para especificar una duración de sesión mediante la API de AWS, utilice el parámetro `DurationSeconds`. Para obtener más información, consulte Cambio a un rol de IAM (API de AWS).

Modificación de un límite de permisos de rol (consola)

Para cambiar los permisos máximos permitidos para un rol, modifique el límite de permisos del rol.

Para cambiar la política que se utiliza para establecer el límite de permisos de un rol

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Roles en el panel de navegación.
3. Elija el nombre del rol con el límite de permisos que desea modificar.
4. Elija la pestaña Permisos. Si es necesario, abra la sección Permissions boundary (Límite de permisos) y, a continuación, elija Change boundary (Cambiar límite).
5. Seleccione la política que desea utilizar para el límite de permisos.
6. Elija Change boundary (Cambiar límite).

   Los cambios no entrarán en vigor hasta la próxima vez que alguien asuma este rol.

Modificación de un rol (AWS CLI)

Puede utilizar la AWS Command Line Interface para modificar un rol. Para cambiar el conjunto de etiquetas en un rol, consulte Administrar etiquetas en roles de IAM (AWS CLI o API de AWS).

Temas

- Modificación de una política de confianza de rol (AWS CLI)
- Modificación de una política de permisos de rol (AWS CLI)
- Modificación de una descripción de rol (AWS CLI)
- Modificación de la duración máxima de la sesión de un rol (AWS CLI)
- Modificación de un límite de permisos de rol (AWS CLI)

Modificación de una política de confianza de rol (AWS CLI)

Para cambiar quién puede asumir un rol, debe modificar la política de confianza del rol. No se puede modificar la política de confianza de un rol vinculado a un servicio.

Notas

- Si un usuario identificado como entidad principal de una política de confianza de un rol no puede asumir ese rol, compruebe el límite de permisos del usuario. Si hay definido un límite de permisos para el usuario, el límite deberá permitir la acción `sts:AssumeRole`.
- Para permitir que los usuarios vuelvan a asumir el rol actual dentro de una sesión de rol, especifique el ARN del rol o el ARN de la Cuenta de AWS como entidad principal en la política de confianza de rol. Los Servicios de AWS que proporcionan recursos de computación, como Amazon EC2, Amazon ECS, Amazon EKS y Lambda, brindan credenciales temporales y las actualizan automáticamente. Esto garantiza que siempre disponga de un conjunto de credenciales válido. Para estos servicios, no es necesario volver a asumir el rol actual a fin de obtener credenciales temporales. Sin embargo, si tiene la intención de aprobar etiquetas de sesión o una política de sesión, tendrá que volver a asumir el rol actual. Para obtener información sobre cómo modificar una política de confianza de roles a fin de agregar el ARN del rol o el ARN de la Cuenta de AWS para la entidad principal, consulte Modificación de una política de confianza de rol (consola).

Para modificar una política de confianza de rol (AWS CLI)

1. (Opcional) Si no conoce el nombre del rol que desea modificar, ejecute el siguiente comando para ver una lista de los roles de la cuenta:
   - aws iam list-roles
2. (Opcional) Para ver la política de confianza actual de un rol, ejecute el siguiente comando:
   - aws iam get-role
3. Para modificar las entidades principales de confianza que pueden obtener acceso al rol, cree un archivo de texto con la política de confianza actualizada. Puede utilizar cualquier editor de texto para crear la política.

   Por ejemplo, la política de confianza siguiente muestra cómo hacer referencia a dos Cuentas de AWS en el elemento `Principal`. Esto permite a los usuarios de dos Cuentas de AWS independientes asumir este rol.

   ```
   {
       "Version": "2012-10-17",
   ```

```
    "Statement": {
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::111122223333:root",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "sts:AssumeRole"
    }
}
```

Si especifica una entidad principal de otra cuenta, el hecho de añadir una cuenta a la política de confianza de un rol solo es una parte del establecimiento de una relación de confianza entre cuentas. De forma predeterminada, ningún usuario de las cuentas de confianza puede asumir el rol. El administrador de la cuenta en la que se acaba de establecer la relación de confianza debe conceder a los usuarios permiso para asumir el rol. Para ello, el administrador debe crear o editar una política que esté asociada al usuario para permitirle a este el acceso a la acción `sts:AssumeRole`. Para obtener más información, consulte el siguiente procedimiento o Conceder permisos de usuario para cambiar de rol.

4. Si desea utilizar el archivo que acaba de crear para actualizar la política de confianza, ejecute el siguiente comando:

   - aws iam update-assume-role-policy

Para permitir que los usuarios de una cuenta externa de confianza utilicen el rol (AWS CLI)

Para obtener más información y detalles sobre este procedimiento, consulte Conceder permisos de usuario para cambiar de rol.

1. Cree un archivo JSON que contenga una política de permisos que conceda permisos para asumir el rol. Por ejemplo, la política siguiente contiene los permisos mínimos necesarios:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
  }
}
```

Reemplace el ARN de la instrucción por el ARN del rol que el usuario puede asumir.

2. Ejecute el siguiente comando para cargar el archivo JSON que contiene la política de confianza en IAM:

   - aws iam create-policy

   La salida de este comando incluye el ARN de la política. Anote este ARN, ya que tendrá que utilizarlo en un paso posterior.

3. Decida a qué usuario o grupo asociará la política. Si no conoce el nombre del usuario o el grupo en cuestión, ejecute uno de los comandos siguientes para mostrar una lista de los usuarios o grupos de la cuenta:

   - aws iam list-users
   - aws iam list-groups

4. Ejecute uno de los siguientes comandos para asociar la política que ha creado en el paso anterior al usuario o al grupo:

   - aws iam attach-user-policy
   - aws iam attach-group-policy

Modificación de una política de permisos de rol (AWS CLI)

Para cambiar los permisos permitidos por el rol, modifique la política (o políticas) de permisos del rol. No se puede modificar la política de permisos de un rol vinculado a un servicio en IAM. Se podría modificar la política de permisos en el servicio que depende del rol. Para comprobar si un servicio admite esta característica, consulte Servicios de AWS que funcionan con IAM y busque los servicios con Sí en la columna Roles vinculados a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Para cambiar los permisos que permite un rol (AWS CLI)

1. (Opcional) Para ver los permisos actuales asociados a un rol, ejecute los siguientes comandos:

   1. aws iam list-role-policies para obtener una lista de políticas insertadas
   2. aws iam list-attached-role-policies para obtener una lista de políticas administradas

2. El comando para actualizar los permisos del rol varía en función de si se está actualizando una política administrada o una política insertada.

   Para actualizar una política administrada, ejecute el siguiente comando para crear una nueva versión de la política administrada:

   - aws iam create-policy-version

   Para actualizar una política insertada, ejecute el siguiente comando:

   - aws iam put-role-policy

Modificación de una descripción de rol (AWS CLI)

Para cambiar la descripción del rol, modifique el texto de descripción.

Para cambiar la descripción de un rol (AWS CLI)

1. (Opcional) Para ver la descripción actual de un rol, ejecute el siguiente comando:

   - aws iam get-role
2. Para actualizar la descripción de un rol, ejecute el siguiente comando con el parámetro de descripción:

   - aws iam update-role

Modificación de la duración máxima de la sesión de un rol (AWS CLI)

Para especificar la duración máxima de la sesión para los roles que se asumen mediante la API o la AWS CLI, modifique el valor del ajuste de duración máxima de la sesión. Esta opción puede tener un valor comprendido entre 1 y 12 horas. Si no especifica un valor, se aplicará el valor máximo predeterminado de 1 hora. Esta configuración no limita las sesiones asumidas por los servicios de AWS.

> **Note**
>
> Cualquiera que asuma el rol desde la API o la AWS CLI puede utilizar el `duration-seconds` parámetro de la CLI o el `DurationSeconds` parámetro de la API para solicitar una sesión más larga. El ajuste `MaxSessionDuration` determina la duración máxima de

la sesión de rol que se puede solicitar mediante el parámetro `DurationSeconds`. Si los usuarios no especifican un valor para el parámetro `DurationSeconds`, sus credenciales de seguridad serán válidas durante una hora.

Para cambiar el valor de la duración máxima de la sesión para los roles que se asumen mediante AWS CLI (AWS CLI)

1. (Opcional) Para ver el valor actual de la duración máxima de la sesión de un rol, ejecute el siguiente comando:

   - aws iam get-role

2. Para actualizar el valor de la duración máxima de la sesión de un rol, ejecute el siguiente comando con el parámetro `max-session-duration` de la CLI o el parámetro `MaxSessionDuration` de la API:

   - aws iam update-role

   Los cambios no entrarán en vigor hasta la próxima vez que alguien asuma este rol. Para obtener información sobre cómo revocar sesiones existentes para este rol, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

Modificación de un límite de permisos de rol (AWS CLI)

Para cambiar los permisos máximos permitidos para un rol, modifique el límite de permisos del rol.

Para cambiar la política administrada que se utiliza para establecer el límite de permisos de un rol (AWS CLI)

1. (Opcional) Para ver el límite de permisos actual de un rol, ejecute el comando siguiente:

   - aws iam get-role

2. Si desea utilizar una política administrada diferente para actualizar el límite de permisos de un rol, ejecute el comando siguiente:

   - aws iam put-role-permissions-boundary

Un rol solo puede tener una política administrada configurada como límite de permisos. Si cambia el límite de permisos, también cambiará los permisos que puede tener un rol como máximo.

Modificación de un rol (API de AWS)

Puede utilizar la API de AWS para modificar un rol. Para cambiar el conjunto de etiquetas en un rol, consulte Administrar etiquetas en roles de IAM (AWS CLI o API de AWS).

Temas

- Modificación de una política de confianza de rol (API de AWS)
- Modificación de una política de permisos de rol (API de AWS)
- Modificación de una descripción de rol (API de AWS)
- Modificación de la duración máxima de la sesión de un rol (API de AWS)
- Modificación de un límite de permisos de rol (API de AWS)

Modificación de una política de confianza de rol (API de AWS)

Para cambiar quién puede asumir un rol, debe modificar la política de confianza del rol. No se puede modificar la política de confianza de un rol vinculado a un servicio.

Notas

- Si un usuario identificado como entidad principal de una política de confianza de un rol no puede asumir ese rol, compruebe el límite de permisos del usuario. Si hay definido un límite de permisos para el usuario, el límite deberá permitir la acción `sts:AssumeRole`.
- Para permitir que los usuarios vuelvan a asumir el rol actual dentro de una sesión de rol, especifique el ARN del rol o el ARN de la Cuenta de AWS como entidad principal en la política de confianza de rol. Los Servicios de AWS que proporcionan recursos de computación, como Amazon EC2, Amazon ECS, Amazon EKS y Lambda, brindan credenciales temporales y las actualizan automáticamente. Esto garantiza que siempre disponga de un conjunto de credenciales válido. Para estos servicios, no es necesario volver a asumir el rol actual a fin de obtener credenciales temporales. Sin embargo, si tiene la intención de aprobar etiquetas de sesión o una política de sesión, tendrá que volver

a asumir el rol actual. Para obtener información sobre cómo modificar una política de confianza de roles a fin de agregar el ARN del rol o el ARN de la Cuenta de AWS para la entidad principal, consulte Modificación de una política de confianza de rol (consola).

Para modificar una política de confianza de rol (API de AWS)

1. (Opcional) Si no conoce el nombre del rol que desea modificar, llame a la siguiente operación para ver una lista de los roles de la cuenta:
   - ListRoles
2. (Opcional) Para ver la política de confianza actual de un rol, llame a la siguiente operación:
   - GetRole
3. Para modificar las entidades principales de confianza que pueden obtener acceso al rol, cree un archivo de texto con la política de confianza actualizada. Puede utilizar cualquier editor de texto para crear la política.

   Por ejemplo, la política de confianza siguiente muestra cómo hacer referencia a dos Cuentas de AWS en el elemento `Principal`. Esto permite a los usuarios de dos Cuentas de AWS independientes asumir este rol.

   ```
   {
       "Version": "2012-10-17",
       "Statement": {
           "Effect": "Allow",
           "Principal": {"AWS": [
               "arn:aws:iam::111122223333:root",
               "arn:aws:iam::444455556666:root"
           ]},
           "Action": "sts:AssumeRole"
       }
   }
   ```

   Si especifica una entidad principal de otra cuenta, el hecho de añadir una cuenta a la política de confianza de un rol solo es una parte del establecimiento de una relación de confianza entre cuentas. De forma predeterminada, ningún usuario de las cuentas de confianza puede asumir el rol. El administrador de la cuenta en la que se acaba de establecer la relación de confianza debe conceder a los usuarios permiso para asumir el rol. Para ello, el administrador

debe crear o editar una política que esté asociada al usuario para permitirle a este el acceso a la acción `sts:AssumeRole`. Para obtener más información, consulte el siguiente procedimiento o Conceder permisos de usuario para cambiar de rol.

4. Si desea utilizar el archivo que acaba de crear para actualizar la política de confianza, llame a la operación siguiente:

   - UpdateAssumeRolePolicy

Para permitir que los usuarios de una cuenta externa de confianza utilicen el rol (API de AWS)

Para obtener más información y detalles sobre este procedimiento, consulte Conceder permisos de usuario para cambiar de rol.

1. Cree un archivo JSON que contenga una política de permisos que conceda permisos para asumir el rol. Por ejemplo, la política siguiente contiene los permisos mínimos necesarios:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE:role/ROLE-NAME"
  }
}
```

   Reemplace el ARN de la instrucción por el ARN del rol que el usuario puede asumir.

2. Llame a la operación siguiente para cargar el archivo JSON que contiene la política de confianza en IAM:

   - CreatePolicy

   La salida de esta operación incluye el ARN de la política. Anote este ARN, ya que tendrá que utilizarlo en un paso posterior.

3. Decida a qué usuario o grupo asociará la política. Si no conoce el nombre del usuario o el grupo en cuestión, llame a una de las operaciones siguientes para mostrar una lista de los usuarios o grupos de la cuenta:

   - ListUsers

- ListGroups

4. Llame a una de las operaciones siguientes para asociar la política que ha creado en el paso anterior al usuario o al grupo:

   - API: AttachUserPolicy
   - AttachGroupPolicy

Modificación de una política de permisos de rol (API de AWS)

Para cambiar los permisos permitidos por el rol, modifique la política (o políticas) de permisos del rol. No se puede modificar la política de permisos de un rol vinculado a un servicio en IAM. Se podría modificar la política de permisos en el servicio que depende del rol. Para comprobar si un servicio admite esta característica, consulte Servicios de AWS que funcionan con IAM y busque los servicios con Sí en la columna Roles vinculados a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Para cambiar los permisos que permite un rol (API de AWS)

1. (Opcional) Para ver los permisos actuales asociados a un rol, llame a las siguientes operaciones:

   1. ListRolePolicies para obtener una lista de políticas insertadas
   2. ListAttachedRolePolicies para obtener una lista de políticas administradas

2. La operación para actualizar los permisos del rol varía en función de si se está actualizando una política administrada o una política insertada.

   Para actualizar una política administrada, llame a la siguiente operación para crear una nueva versión de la política administrada:

   - CreatePolicyVersion

   Para actualizar una política insertada, llame a la siguiente operación:

   - PutRolePolicy

Modificación de una descripción de rol (API de AWS)

Para cambiar la descripción del rol, modifique el texto de descripción.

Para cambiar la descripción de un rol (API de AWS)

1. (Opcional) Para ver la descripción actual de un rol, llame a la siguiente operación:
   - GetRole
2. Para actualizar la descripción de un rol, llame a la siguiente operación con el parámetro de descripción:
   - UpdateRole

Modificación de la duración máxima de la sesión de un rol (API de AWS)

Para especificar la duración máxima de la sesión para los roles que se asumen mediante la API o la AWS CLI, modifique el valor del ajuste de duración máxima de la sesión. Esta opción puede tener un valor comprendido entre 1 y 12 horas. Si no especifica un valor, se aplicará el valor máximo predeterminado de 1 hora. Esta configuración no limita las sesiones asumidas por los servicios de AWS.

> **Note**
>
> Cualquiera que asuma el rol desde la API o la AWS CLI puede utilizar el `duration-seconds` parámetro de la CLI o el `DurationSeconds` parámetro de la API para solicitar una sesión más larga. El ajuste `MaxSessionDuration` determina la duración máxima de la sesión de rol que se puede solicitar mediante el parámetro `DurationSeconds`. Si los usuarios no especifican un valor para el parámetro `DurationSeconds`, sus credenciales de seguridad serán válidas durante una hora.

Para cambiar el valor de la duración máxima de la sesión para los roles que se asumen mediante la API (API de AWS)

1. (Opcional) Para ver el valor actual de la duración máxima de la sesión de un rol, llame a la siguiente operación:
   - GetRole
2. Para actualizar el valor de la duración máxima de la sesión de un rol, llame a la siguiente operación con el parámetro `max-sessionduration` de la CLI o el parámetro `MaxSessionDuration` de la API:

- UpdateRole

Los cambios no entrarán en vigor hasta la próxima vez que alguien asuma este rol. Para obtener información sobre cómo revocar sesiones existentes para este rol, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

Modificación de un límite de permisos de rol (API de AWS)

Para cambiar los permisos máximos permitidos para un rol, modifique el límite de permisos del rol.

Para cambiar la política administrada que se utiliza para establecer el límite de permisos de un rol (API de AWS)

1. (Opcional) Para ver el límite de permisos actual de un rol, llame a la operación siguiente:
   - GetRole
2. Si desea utilizar una política administrada diferente para actualizar el límite de permisos de un rol, llame a la operación siguiente:
   - PutRolePermissionsBoundary

   Un rol solo puede tener una política administrada configurada como límite de permisos. Si cambia el límite de permisos, también cambiará los permisos que puede tener un rol como máximo.

Eliminación de roles o perfiles de instancia

Si ya no necesita un rol, le recomendamos que elimine el rol y sus permisos asociados. De esta forma no tiene una entidad no utilizada que no se monitoree ni mantenga de forma activa.

Si el rol se asoció a una instancia EC2, puede también eliminarlo del perfil de instancia y, a continuación, eliminar dicho perfil.

> **Warning**
>
> Asegúrese de que no tiene ninguna instancia de Amazon EC2 ejecutándose con el rol o el perfil de instancias que va a eliminar. Al eliminar un perfil de instancias o rol asociado a

una instancia en ejecución se romperá cualquier aplicación que se esté ejecutando en la instancia.

Si prefiere no eliminar permanentemente un rol, puede deshabilitarlo. Para ello, cambie las políticas del rol y, a continuación, revoque todas las sesiones actuales. Por ejemplo, podría añadir una política al rol que denegó el acceso a todos de AWS. También puede editar la política de confianza para denegar el acceso a cualquier persona que intente asumir el rol. Para obtener más información acerca de cómo revocar sesiones, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

Temas

Ver Acceso de roles

Antes de eliminar un rol, le recomendamos que revise la fecha en que se usó el rol por última vez. Para ello, utilice la AWS Management Console, la AWS CLI o la API de AWS. Debe ver esta información porque no debería eliminar el acceso de alguien que utilice el rol.

Es posible que la fecha de la última actividad del rol no coincida con la última fecha notificada en la ficha Asesor de acceso. La ficha Asesor de acceso informa de la actividad solo para los servicios permitidos por las políticas de permisos del rol. La fecha de la última actividad del rol incluye el último intento de acceder a cualquier servicio de AWS.

Note

El periodo de seguimiento de la última actividad de un rol y los datos del Asesor de acceso es para los últimos 400 días. Este período puede ser más corto si su Región comenzó a admitir estas características en el último año. El rol podría haberse utilizado hace más de 400

días. Para obtener más información sobre el período de seguimiento, consulte Dónde AWS se hace un seguimiento de la información de acceso reciente.

Para ver cuándo se utilizó un rol por última vez (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Roles (Roles) en el panel de navegación.
3. Busque la fila del rol con la actividad que desee ver. Puede utilizar el campo de búsqueda para filtrar los resultados. Vea la columna Última actividad para ver el número de días transcurridos desde la última vez que se utilizó el rol. Si el rol no se ha utilizado dentro del período de seguimiento, la tabla muestra Ninguno.
4. Elija el nombre del rol para ver más información. La página Resumen del rol también incluye Última actividad, que muestra la fecha en que se utilizó el rol por última vez. Si el rol no se ha utilizado en los últimos 400 días, Última actividad muestra No se ha accedido en el período de seguimiento.

Para ver cuándo se utilizó un rol por última vez (AWS CLI)

`aws iam get-role` - Ejecute este comando para devolver información sobre un rol, incluido el objeto `RoleLastUsed`. Este objeto contiene el `LastUsedDate` y el `Region` en el que se utilizó el rol por última vez. Si `RoleLastUsed` está presente pero no contiene un valor, el rol no se ha utilizado dentro del período de seguimiento.

Para ver cuándo se usó un rol por última vez (AWS API)

`GetRole` - Llame a esta operación para devolver información sobre un rol, incluido el objeto `RoleLastUsed`. Este objeto contiene el `LastUsedDate` y el `Region` en el que se utilizó el rol por última vez. Si `RoleLastUsed` está presente pero no contiene un valor, el rol no se ha utilizado dentro del período de seguimiento.

Eliminar un rol vinculado a servicios

Si el rol es un rol vinculado a un servicio, consulte la documentación del servicio vinculado para obtener información acerca de cómo eliminar el rol. Puede consultar los roles vinculados a servicios en su cuenta en cualquier momento a través de la página de IAM Roles de la consola. Los roles vinculados con servicios aparecen con el texto (Service-linked role (Función vinculada al servicio)) en

la columna Trusted entities (Entidades de confianza) de la tabla. Un banner en la página Resumen del rol también indica que es un tipo de rol vinculado a un servicio.

Si el servicio no incluye documentación acerca de cómo eliminar el rol vinculado al servicio, puede utilizar la consola de IAM, la AWS CLI o la API para eliminarlo. Para obtener más información, consulte Eliminar un rol vinculado a servicios.

Eliminar un rol de IAM (consola)

Cuando se utiliza la AWS Management Console para eliminar un rol, IAM elimina automáticamente las políticas administradas asociadas al rol. También elimina automáticamente cualquier política en línea asociada con el rol y cualquier perfil de instancia de Amazon EC2 que contenga el rol.

> **Important**
>
> En algunos casos, un rol podría estar asociado a un perfil de instancias de Amazon EC2 y tanto el rolo como el perfil de instancias podrían tener el mismo nombre. En ese caso, puede utilizar la AWS Management Console para eliminar el rol y el perfil de instancia. Este vínculo se produce de forma automática para roles y perfiles de instancia que crea en la consola. Si creó el rol desde la API de AWS CLI, Tools for Windows PowerShell o API de AWS, el rol y el perfil de instancias podrían tener distintos nombres. En ese caso no puede utilizar la consola para eliminarlos. En cambio, debe utilizar la API de AWS CLI, Tools for Windows PowerShell o API de AWS para primero eliminar el rol del perfil de instancias. A continuación, debe realizar un paso independiente para eliminar el rol.

Para eliminar un rol (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Roles y, a continuación, seleccione la casilla de verificación junto al nombre del rol que desee eliminar.
3. En la parte superior de la página, elija Delete (Eliminar).
4. En el cuadro de diálogo de confirmación, revise la información de acceso reciente, donde se indica cuándo accedió cada uno de los roles seleccionados a un servicio de AWS por última vez. Esto le ayuda a confirmar si el rol está actualmente activo. Si desea continuar, escriba el nombre del rol en el campo de entrada de texto y elija Eliminar. Si está seguro, puede continuar con la eliminación, aunque la información de acceso reciente siga cargándose.

> **Note**
>
> No puede utilizar la consola para eliminar un perfil de instancia, a no ser que tenga el mismo nombre que el rol. El perfil de instancias se elimina como parte del proceso de eliminación de un rol, tal y como se describe en el procedimiento anterior. Para eliminar un perfil de instancia sin eliminar también el rol, debe utilizar la AWS CLI o la API de AWS. Para obtener más información, consulte las siguientes secciones.

Eliminar un rol de IAM (AWS CLI)

Cuando utiliza la AWS CLI para eliminar un rol, antes debe eliminar las políticas insertadas asociadas al rol. También debe desvincular las políticas administradas asociadas al rol. Si desea eliminar el perfil de instancia asociado que incluye el rol, debe eliminarlo por separado.

Para eliminar un rol (AWS CLI)

1. Si no conoce el nombre del rol que desea eliminar, escriba el siguiente comando para enumerar los roles de su cuenta:

   ```
   aws iam list-roles
   ```

   La lista incluye el Nombre de recurso de Amazon (ARN) de cada rol. Utilice el nombre del rol, no el ARN, para hacer referencia a los roles con los comandos de CLI. Por ejemplo, si un rol tiene el ARN `arn:aws:iam::123456789012:role/myrole`, debe referirse a él como **`myrole`**.
2. Elimine el rol de todos los perfiles de instancia en los que está asociado.

   a. Para enumerar todos los perfiles de instancia con los que se asocia el rol, escriba el siguiente comando:

      ```
      aws iam list-instance-profiles-for-role --role-name role-name
      ```

   b. Para eliminar el rol de un perfil de instancia, escriba el siguiente comando para cada perfil de instancia:

      ```
      aws iam remove-role-from-instance-profile --instance-profile-name instance-profile-name --role-name role-name
      ```
3. Elimine todas las políticas asociadas al rol.

a. Para enumerar todas las políticas insertadas que están en el rol, ingrese el siguiente comando:

```
aws iam list-role-policies --role-name role-name
```

b. Para eliminar cada política insertada del rol, ingrese el siguiente comando para cada política:

```
aws iam delete-role-policy --role-name role-name --policy-name policy-name
```

c. Para enumerar todas las políticas administradas que están asociadas al rol, ingrese el siguiente comando:

```
aws iam list-attached-role-policies --role-name role-name
```

d. Para desvincular cada política administrada del rol, ingrese el siguiente comando para cada política:

```
aws iam detach-role-policy --role-name role-name --policy-arn policy-arn
```

4. Escriba el comando siguiente para eliminar el rol:

```
aws iam delete-role --role-name role-name
```

5. Si no tiene pensado reutilizar los perfiles de instancia que se asociaron al rol, puede escribir el siguiente comando para eliminarlos:

```
aws iam delete-instance-profile --instance-profile-name instance-profile-name
```

Eliminar un rol de IAM (API de AWS)

Si utiliza la API de IAM para eliminar un rol, antes debe eliminar las políticas insertadas asociadas al rol. También debe desvincular las políticas administradas asociadas al rol. Si desea eliminar el perfil de instancia asociado que incluye el rol, debe eliminarlo por separado.

Para eliminar un rol (API de AWS)

1. Para enumerar todos los perfiles de instancia asociados a un rol, llame a ListInstanceProfilesForRole.

Para eliminar el rol de un perfil de instancia, llame a RemoveRoleFromInstanceProfile. Debe transmitir el nombre del rol y el nombre del perfil de instancia.

Si no va a volver a utilizar un perfil de instancia que estaba asociado al rol, llame a DeleteInstanceProfile para eliminarlo.

2. Para enumerar todas las políticas insertadas de un rol, llame a ListRolePolicies.

   Para eliminar todas las políticas insertadas asociadas al rol, llame a DeleteRolePolicy. Debe pasar el nombre del rol y el nombre de la política insertada.

3. Para enumerar todas las políticas administradas que se encuentran asociadas a un rol, llame a ListAttachedRolePolicies.

   Para desvincular políticas administradas que se encuentran asociadas al rol, llame a DetachRolePolicy. Debe pasar el nombre del rol y el ARN de la política administrada.

4. Llame a DeleteRole para eliminar el rol.

Información relacionada

Para obtener información general sobre los perfiles de instancia, consulte Uso de perfiles de instancia.

Para obtener información general acerca de los roles vinculados a servicios, consulte Uso de roles vinculados a servicios.

Federación y proveedores de identidades

Si ya administra identidades de usuarios fuera de AWS, puede utilizar los proveedores de identidades en lugar de crear usuarios de IAM en su Cuenta de AWS. Con un proveedor de identidades (IdP), puede administrar sus identidades de usuario fuera de AWS y conceder permisos a estas identidades de usuarios externos para utilizar los recursos de AWS en su cuenta. Esto resulta útil si su organización ya tiene su propio sistema de identidad, como por ejemplo, un directorio de usuario corporativo. También resulta útil si crea una aplicación móvil o web que necesita acceso a los recursos de AWS.

Un IdP externo proporciona información de identidad de AWS mediante OpenID Connect (OIDC) o SAML 2.0 (lenguaje de marcado de aserciones de seguridad). El OIDC conecta aplicaciones, como

GitHub Actions, que no se ejecutan en AWS los AWS recursos. Algunos ejemplos de proveedores de identidad SAML conocidos son: Shibboleth y Active Directory Federation Services.

> **Note**
>
> Como práctica recomendada de seguridad, le recomendamos que administre usuarios humanos en IAM Identity Center con un proveedor de identidad SAML externo en lugar de utilizar la federación SAML de IAM. Para obtener más información acerca de situaciones específicas en las que se requiere un usuario de IAM, consulte Cuándo crear un usuario de IAM (en lugar de un rol).

Cuando se utiliza un proveedor de identidad, no es necesario crear un código de inicio de sesión personalizado ni administrar sus propias identidades de usuario. El proveedor de identidad lo hace automáticamente. Sus usuarios externos inician sesión a través de un IdP, y usted puede conceder permisos a las identidades externas para utilizar los recursos de AWS en su cuenta. Los proveedores de identidades le ayudan a proteger su Cuenta de AWS, ya que no tiene que distribuir ni integrar credenciales de seguridad a largo plazo, como por ejemplo, claves de acceso, en su aplicación.

Esta guía aborda la federación de IAM. Su caso de uso podría ser más adecuado para IAM Identity Center o Amazon Cognito. Los siguientes resúmenes y tabla proporcionan una visión general de los métodos que sus usuarios pueden emplear para obtener acceso federado a los recursos de AWS.

	Tipo de cuenta	Administración de acceso a	Fuente de identidad compatible
Federación con IAM Identity Center	Cuentas múltiples administradas por AWS Organizations	Usuarios humanos de su plantilla	• SAML 2.0 • Active Directory administrado • Directorio de Identity Center
Federación con IAM	Cuenta única e independiente	• Usuarios humanos en implement aciones a corto plazo y a pequeña escala	• SAML 2.0 • OIDC

	Tipo de cuenta	Administración de acceso a	Fuente de identidad compatible
		• Usuarios no humanos	
Federación con grupos de identidades de Amazon Cognito	Cualquiera	Los usuarios de aplicaciones que requieren autorización de IAM para acceder a los recursos	• SAML 2.0 • OIDC • Seleccionar proveedores de identidad social OAuth 2.0

Federación con IAM Identity Center

Si desea administrar el acceso de usuarios humanos de manera centralizada, le recomendamos utilizar IAM Identity Center para administrar el acceso a las cuentas y los permisos dentro de esas cuentas. A los usuarios de IAM Identity Center se les conceden credenciales a corto plazo para sus recursos de AWS. Puede utilizar Active Directory, un proveedor de identidades (IdP) externo o un directorio del Centro de identidades de IAM como fuente de identidad para los usuarios y grupos con el fin de asignar el acceso a sus recursos de AWS.

IAM Identity Center admite la federación de identidades con SAML (Security Assertion Markup Language) 2.0, que proporciona acceso federado de inicio de sesión único para los usuarios autorizados a utilizar aplicaciones dentro del portal de acceso de AWS. A continuación, los usuarios pueden realizar un inicio de sesión único que admita SAML, incluida la AWS Management Console y aplicaciones de terceros, como Microsoft 365, SAP Concur y Salesforce.

Federación con IAM

Aunque recomendamos la administración de usuarios humanos en IAM Identity Center, puede habilitar el acceso de usuario federado con IAM para usuarios humanos en implementaciones a corto plazo y a pequeña escala. IAM le permite utilizar IdPs SAML 2.0 y Open ID Connect (OIDC) por separado y utilizar atributos de usuario federado para el control de acceso. Con IAM, puede proporcionar atributos de usuario, como el centro de costes, el cargo o la ubicación, desde sus IdP a AWS, e implementar permisos de acceso detallados basados en estos atributos.

Una carga de trabajo es un conjunto de recursos y código que ofrece valor comercial, como una aplicación o un proceso de backend. Su carga de trabajo puede requerir una identidad IAM para hacer peticiones a servicios de AWS, aplicaciones, herramientas operativas y componentes. Estas identidades incluyen máquinas que se ejecutan en los entornos de AWS, como instancias de Amazon EC2 o funciones de AWS Lambda.

También se pueden administrar identidades de máquina para las partes externas que necesiten acceso. Para dar acceso a las identidades de máquina, puede utilizar roles de IAM. Los roles de IAM tienen permisos específicos y ofrecen una forma de acceder a AWS empleando credenciales de seguridad temporales con una sesión de rol. Además, es posible que tenga máquinas fuera de AWS que necesiten acceso a los entornos de AWS. Para máquinas que se ejecuten fuera de AWS, puede utilizar Funciones de IAM en cualquier lugar. Para obtener más información acerca de los roles de , consulte Roles de IAM. Para obtener detalles sobre cómo utilizar roles para delegar el acceso en Cuentas de AWS, consulte Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.

Para vincular un IdP directamente a IAM, debe crear una entidad de proveedor de identidad para establecer una relación de confianza entre su Cuenta de AWS y el IdP. IAM admite proveedores de identidades (IdP) que son compatibles con OpenID Connect (OIDC) o SAML 2.0 (Security Assertion Markup Language 2.0). Para obtener más información sobre el uso de uno de estos proveedores de identidad (IdP) con AWS, consulte las siguientes secciones:

- Federación OIDC
- Federación SAML 2.0

Federación con grupos de identidades de Amazon Cognito

Amazon Cognito está diseñado para desarrolladores que desean autenticar y autorizar usuarios en sus aplicaciones móviles y web. Los grupos de usuarios de Amazon Cognito añaden características de inicio de sesión y registro a su aplicación, y los grupos de identidades proporcionan credenciales de IAM que conceden a sus usuarios acceso a recursos protegidos que usted administra en AWS. Los grupos de identidades adquieren credenciales para sesiones temporales a través de la operación `AssumeRoleWithWebIdentity` de la API.

Amazon Cognito funciona con proveedores de identidad externos compatibles con SAML y OpenID Connect, y con proveedores de identidad social como Facebook, Google y Amazon. Su aplicación puede iniciar sesión en un usuario con un grupo de usuarios o un IdP externo y, a continuación, recuperar recursos en su nombre con sesiones temporales personalizadas en un rol de IAM.

Escenarios habituales

> **Note**
>
> Se recomienda exigir a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. ¿Ha considerado la posibilidad de usar AWS IAM Identity Center? Puede usar IAM Identity Center para administrar de forma centralizada el acceso a múltiples Cuentas de AWS y proporcionar a los usuarios un acceso protegido por MFA y de inicio de sesión único a todas sus cuentas asignadas desde un solo lugar. Con IAM Identity Center, puede crear y administrar identidades de usuario en IAM Identity Center o conectarse fácilmente a su proveedor de identidades existente compatible con SAML 2.0. Para obtener más información, consulte ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center.

Puede utilizar un proveedor de identidades (IdP) externo para gestionar las identidades de los usuarios externos a AWS y el IdP externo. Un IdP externo puede proporcionar información de identidad de AWS mediante OpenID Connect (OIDC) o el lenguaje de marcado de aserciones de seguridad (SAML). El OIDC se suele utilizar cuando una aplicación que no se ejecuta en AWS necesita acceder a los recursos AWS.

Si desea configurar la federación con un IdP externo, debe crear un proveedor de identidades de IAM para informar a AWS sobre el IdP externo y su configuración. Esto establece una relación de confianza entre su Cuenta de AWS y el IdP externo. En los siguientes temas, se proporcionan escenarios comunes para usar proveedores de identidades de IAM.

Temas

- Uso de Amazon Cognito para aplicaciones móviles
- Uso de las operaciones de API de federación de OIDC para aplicaciones móviles

Uso de Amazon Cognito para aplicaciones móviles

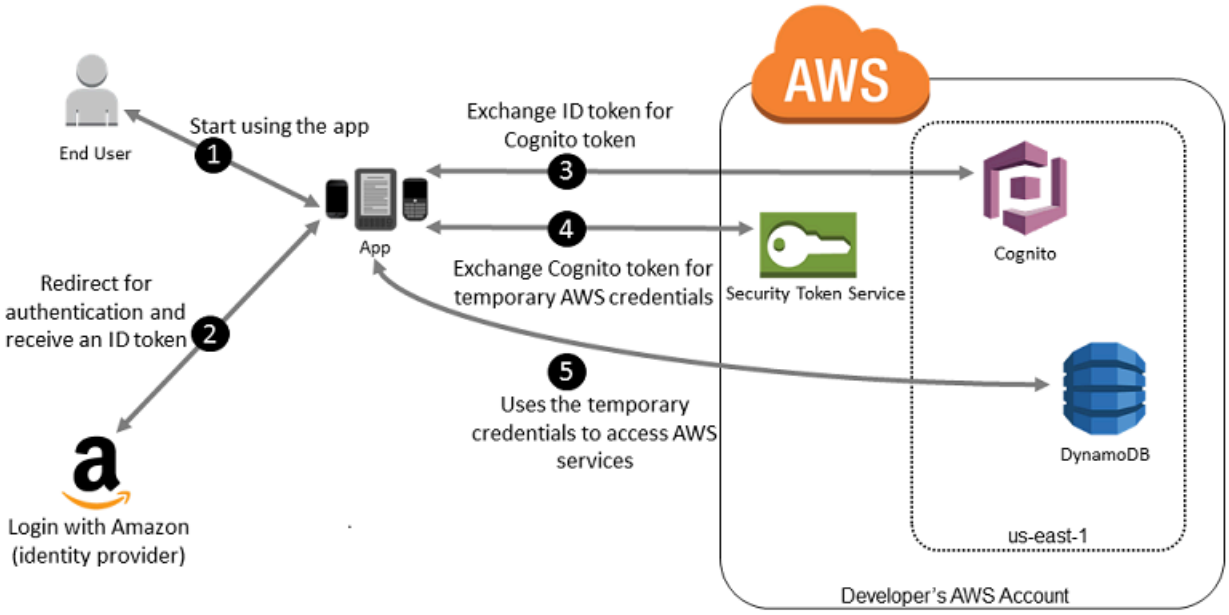
La mejor forma de utilizar las federaciones de OIDC es emplear Amazon Cognito. Por ejemplo, Adele, la desarrolladora, está creando un juego para un dispositivo móvil donde los datos de usuario, como perfiles y puntuaciones, se almacenan en Amazon S3 y Amazon DynamoDB. Adele también podría almacenar estos datos localmente en el dispositivo y utilizar Amazon Cognito para mantenerlos sincronizados en todos los dispositivos. Ella sabe que por razones de seguridad y

mantenimiento, las credenciales de seguridad de AWS a largo plazo no deben distribuirse con el juego. También sabe que el juego podría tener un gran número de usuarios. Por todo ello, no quiere crear nuevas identidades de usuario en IAM para cada jugador. En cambio, diseña el juego de manera que los usuarios puedan iniciar sesión con una identidad que ya han establecido con un proveedor de identidad (IdP) externo bien conocido, como Login with Amazon, Facebook, Google o cualquier IdP compatible con OpenID Connect (OIDC). Su juego puede aprovechar el mecanismo de autenticación de uno de estos proveedores para validar la identidad del usuario.

Para permitir que la aplicación móvil pueda acceder a sus recursos de AWS, Adele registra primero un ID de desarrollador con el proveedor de identidad elegido. También configura la aplicación con cada uno de estos proveedores. En su Cuenta de AWS que contiene el bucket de Amazon S3 y la tabla de DynamoDB para el juego, Adele utiliza Amazon Cognito para crear roles de IAM que definen de forma precisa los permisos que necesita el juego. Si está utilizando un IdP de OIDC, también crea una entidad de proveedor de identidades de OIDC de IAM para establecer una relación de confianza entre un grupo de identidades de Amazon Cognito de la Cuenta de AWS y el IdP.

En el código de aplicación, Adele llama a la interfaz de inicio de sesión para el proveedor de identidad que ha configurado anteriormente. El proveedor de identidad gestiona todos los detalles que permiten iniciar sesión al usuario y la aplicación obtiene un token de acceso OAuth o un token de ID de OIDC del proveedor. La aplicación de Adele puede intercambiar esta información de autenticación por un conjunto de credenciales de seguridad temporales que constan de un ID de clave de acceso de AWS, una clave de acceso secreta y un token de sesión. A continuación, la aplicación puede utilizar estas credenciales para obtener acceso a los servicios web que ofrece AWS. La aplicación está limitada por los permisos que están definidos en el rol que asume.

La imagen siguiente muestra un flujo simplificado de su funcionamiento, utilizando Login with Amazon como proveedor de identidad. En el paso 2, la aplicación también puede utilizar Facebook, Google o cualquier IdP compatible con OIDC, pero no se muestra aquí.



1. Un cliente inicia la aplicación en un dispositivo móvil. La aplicación solicita al usuario que inicie la sesión.
2. La aplicación utiliza los recursos de Login with Amazon para aceptar las credenciales del usuario.
3. La aplicación utiliza operaciones `GetId` y `GetCredentialsForIdentity` de la API de Amazon Cognito para intercambiar el token de ID de Login with Amazon por un token de Amazon Cognito. Amazon Cognito, que se ha configurado para confiar en su proyecto de Login with Amazon, genera un token que intercambia por credenciales de sesión temporales con AWS STS.
4. La aplicación recibe credenciales de seguridad temporales de Amazon Cognito. La aplicación también puede usar el flujo de trabajo básico (clásico) de Amazon Cognito para recuperar tokens de AWS STS con `AssumeRoleWithWebIdentity`. Para obtener más información acerca del modo en que ayuda a la autenticación de usuarios, consulte Identity pools (federated identities) authentication flow (Flujo de autenticación de grupo de identidades [federadas]) en la Amazon Cognito Developer Guide (Guía para desarrolladores de Amazon Cognito).
5. La aplicación puede utilizar las credenciales de seguridad temporales para acceder a los recursos de AWS que necesita la aplicación para funcionar. El rol asociado con las credenciales de seguridad temporales y sus políticas asignadas determina a qué se puede acceder.

Utilice el siguiente proceso para configurar la aplicación para utilizar Amazon Cognito para autenticar los usuarios y otorgar a la aplicación acceso a los recursos de AWS. Para conocer los pasos específicos en este escenario, consulte la documentación de Amazon Cognito.

1. (Opcional) Regístrese como desarrollador en Login with Amazon, Facebook, Google o cualquier otro IdP compatible con OpenID Connect (OIDC) y configure una o varias aplicaciones con el proveedor. Este paso es opcional, ya que Amazon Cognito también admite acceso sin autenticar (como invitado) para los usuarios.
2. Diríjase a Amazon Cognito de AWS Management Console. Utilice el asistente de Amazon Cognito para crear un grupo de identidades, que es un contenedor que Amazon Cognito utiliza para mantener las identidades de los usuarios finales organizados para las aplicaciones. Puede compartir grupos de identidades entre aplicaciones. Al configurar un grupo de identidades, Amazon Cognito crea uno o dos roles de IAM (uno para las identidades autenticadas y otro para las identidades "invitadas" sin autenticar) que definen los permisos para los usuarios de Amazon Cognito.
3. Integre AWS Amplify con su aplicación e importe los archivos necesarios para usar Amazon Cognito.
4. Cree una instancia del proveedor de credenciales de Amazon Cognito, transmitiendo el ID de grupo de identidades, el número de Cuenta de AWS y el nombre de recurso de Amazon (ARN) de los roles que asocia con el grupo de identidades. El asistente de Amazon Cognito en la AWS Management Console proporciona código de muestra para ayudarle a comenzar.
5. Cuando la aplicación accede a un recurso de AWS, transfiere la instancia de credenciales de proveedor al objeto de cliente, que a su vez transmite las credenciales de seguridad temporales al cliente. Los permisos de las credenciales se basan en el rol o roles que ha definido previamente.

Para más información, consulte los siguientes temas:

- Sign in (Android) (Iniciar sesión [Android]) en la documentación de AWS Amplify Framework.
- Sign in (iOS) (Iniciar sesión [iOS]) en la documentación de AWS Amplify Framework.

Uso de las operaciones de API de federación de OIDC para aplicaciones móviles

Para obtener los mejores resultados, utilice Amazon Cognito como su agente de identidades para casi todos los casos de federación de OIDC. Amazon Cognito es fácil de utilizar y ofrece capacidades adicionales, tales como acceso anónimo (no autenticado) y sincronización de datos del usuario entre varios dispositivos y proveedores. Sin embargo, si ya ha creado una aplicación que utiliza federación de OIDC llamando manualmente a la API `AssumeRoleWithWebIdentity`, puede seguir utilizándola y sus aplicaciones seguirán funcionando correctamente.

El esquema general del proceso de uso de una federación de OIDC sin Amazon Cognito es el siguiente:

1. Inscríbase como desarrollador con el proveedor de identidad (IdP) externo y configure su aplicación con él, que le proporcionará un ID único para su aplicación. (Cada IdP utiliza una terminología diferente para este proceso. En este esquema se utiliza el término configurar para el proceso de identificación de su aplicación con el IdP). Cada proveedor de identidad (IdP) le ofrece una ID de aplicación exclusivo para dicho IdP, de modo que si configura la misma aplicación con varios proveedores de identidad (IdP), la aplicación tendrá varios ID de aplicación. Puede configurar varias aplicaciones en un mismo proveedor.

   Los siguientes enlaces externos proporcionan información sobre el uso de algunos de los proveedores de identidad (IdP) habituales:

   - Login with Amazon Developer Center
   - Añadir inicio de sesión con Facebook a su aplicación o sitio web en el sitio de desarrolladores de Facebook.
   - Uso de OAuth 2.0 para iniciar sesión (OpenID Connect) en el sitio de desarrolladores de Google.

   > **Important**
   >
   > Si utiliza un proveedor de identidades OIDC de Google, Facebook o Amazon Cognito, no cree un proveedor de identidad de IAM independiente en el AWS Management Console. AWS tiene estos proveedores de identidad OIDC incorporados y disponibles para su uso. Omita el siguiente paso y vaya directamente a crear nuevos roles con su proveedor de identidades.

2. Si utiliza un proveedor de identidad (IdP) distinto de Google, Facebook o Amazon Cognito compatible con OIDC, cree una entidad de proveedor de identidades de IAM para dicho proveedor.
3. En IAM, cree uno o varios roles. Para cada rol, defina quién puede asumir el rol (la política de confianza) y los permisos que los usuarios de la aplicación tienen (la política de permisos). Por lo general, debe crear un rol para cada proveedor de identidad (IdP) que una aplicación admite. Por ejemplo, puede crear un rol que asume una aplicación si el usuario inicia sesión mediante Login with Amazon, un segundo rol para la misma aplicación si el usuario inicia sesión en Facebook y un tercer rol para la aplicación si el usuario inicia sesión a través de Google. En el caso de la relación de confianza, especifique el proveedor de identidad (IdP) (como Amazon.com) como

`Principal` (la entidad de confianza) e incluya un elemento `Condition` que coincida con el ID de la aplicación asignado al proveedor de identidad (IdP). En Creación de un rol para un proveedor de identidad de terceros (federación) podrá encontrar ejemplos de roles para distintos proveedores.

4. En su aplicación, autentique los usuarios con el proveedor de identidad (IdP). Los detalles de cómo hacerlo varían en función del proveedor de identidad que utilice (Login with Amazon, Facebook o Google) y de la plataforma de la aplicación. Por ejemplo, el método de autenticación de una aplicación Android puede diferir del método de una aplicación iOS o una aplicación web basada en JavaScript.

   Por lo general, si el usuario aún no ha iniciado sesión, el proveedor de identidad (IdP) se encarga de mostrar una página de inicio de sesión. Una vez que el proveedor de identidad (IdP) autentica al usuario, el IdP devuelve un token de autenticación con información sobre el usuario a su aplicación. La información que se incluye depende de lo que el proveedor de identidad (IdP) expone y de la información que el usuario está dispuesto a compartir. Puede utilizar esta información en su aplicación.

5. En su aplicación, realice una llamada sin firma a la acción `AssumeRoleWithWebIdentity` para solicitar credenciales de seguridad temporales. En la solicitud, transmita el token de autenticación del proveedor de identidades (IdP) y especifique el Nombre de recurso de Amazon (ARN) para el rol de IAM que ha creado para dicho proveedor. AWS verifica que el token es de confianza y válido y, en tal caso, devuelve las credenciales de seguridad temporales a su aplicación que tienen los permisos para el rol que indica en la solicitud. La respuesta también incluye los metadatos sobre el usuario del proveedor de identidad (IdP), tales como el ID de usuario único que el proveedor de identidad (IdP) asocia al usuario.

6. Al utilizar las credenciales de seguridad temporales de la respuesta de `AssumeRoleWithWebIdentity`, la aplicación envía solicitudes firmadas a las operaciones de la API de AWS. La información del ID de usuario del IdP puede distinguir entre los usuarios de su aplicación, por ejemplo, puede colocar objetos en las carpetas de Amazon S3 que incluyan el ID de usuario como prefijos o sufijos. Esto le permite crear políticas de control de acceso que bloqueen la carpeta para que solo el usuario con dicho ID pueda tener acceso a ella. Para obtener más información, consulte Identifique a los usuarios con la federación OIDC más adelante en este tema.

7. Su aplicación debe almacenar en caché las credenciales de seguridad temporales para que no tenga que obtener unas nuevas cada vez que la aplicación tenga que realizar una solicitud a AWS. De forma predeterminada, las credenciales son válidas durante una hora. Cuando las credenciales caduquen (o antes), realice otra llamada a `AssumeRoleWithWebIdentity`

para obtener un nuevo conjunto de credenciales de seguridad temporales. En función del proveedor de identidad (IdP) y el modo en que administra sus tokens, es posible que tenga que actualizar el token del proveedor de identidad (IdP) antes de realizar una nueva llamada a `AssumeRoleWithWebIdentity`, ya que los tokens del proveedor de identidad (IdP) también suelen caducar después de un tiempo establecido. Si utiliza el SDK for iOS de AWS o el SDK for Android de AWS, puede utilizar la acción AmazonSTSCredentialsProvider, que administra las credenciales temporales de IAM, incluida su actualización según sea necesario.

Federación OIDC

Imagine que está creando una aplicación que accede a los recursos de AWS, como, por ejemplo, GitHib Actions, que utiliza flujos de trabajo para acceder a Amazon S3 y DynamoDB.

Cuando utiliza estos flujos de trabajo, realizará solicitudes a los servicios de AWS que deben estar firmadas con una clave de acceso de AWS. Sin embargo, te recomendamos encarecidamente que no guardes las AWS credenciales a largo plazo en aplicaciones externas AWS. En cambio, cree su aplicación de forma que solicite credenciales de seguridad de AWS cuando las necesite utilizando la federación de OIDC. Las credenciales temporales suministradas se asignan a un rol de AWS que solo tiene los permisos necesarios para realizar las tareas requeridas por la aplicación.

Con la federación de OIDC no necesita crear código de inicio de sesión personalizado ni administrar sus propias identidades de usuario. En su lugar, puedes usar el OIDC en aplicaciones, como GitHub Actions o cualquier otro IdP compatible con OpenID Connect (OIDC), para autenticarte con AWS. Pueden recibir un token de autenticación, conocido como JSON Web Token (JWT) y luego intercambiarlo por credenciales de seguridad temporales en AWS que tienen asignado un rol de IAM con permisos para utilizar los recursos de la Cuenta de AWS. El uso de un IdP le permite tener una Cuenta de AWS más segura, ya que no tiene que integrar ni distribuir credenciales de seguridad a largo plazo con su aplicación.

Para la mayoría de las situaciones, le recomendamos que utilice Amazon Cognito ya que actúa como agente de identidades y realiza gran parte de la federación por usted. Para obtener más detalles, consulte la siguiente sección Uso de Amazon Cognito para aplicaciones móviles.

> **Note**
>
> Los JSON Web Tokens (JWT) emitidos por los proveedores de identidad de OpenID Connect (OIDC) tienen un tiempo de vencimiento especificado en el campo `exp` que indica cuándo expira el token. IAM proporciona un período de cinco minutos más allá del

tiempo de caducidad especificado en el JWT para tener en cuenta la desviación del reloj, según lo permitido por el estándar OpenID Connect (OIDC) Core 1.0. Esto significa que los documentos JWT del OIDC recibidos por IAM después de la fecha de caducidad, pero dentro de este plazo de cinco minutos, se aceptan para su posterior evaluación y procesamiento.

Temas

- Crear un proveedor de identidad de IAM OpenID Connect (OIDC)
- Obtención de la huella digital de un proveedor de identidades OpenID Connect
- Identifique a los usuarios con la federación OIDC
- Recursos adicionales para federaciones de OIDC

Crear un proveedor de identidad de IAM OpenID Connect (OIDC)

Los proveedores de identidad de IAM OIDC son entidades de que describen un servicio de proveedor de identidad (IdP) externo compatible con el estándar OpenID Connect (OIDC), como Google o Salesforce. Puede utilizar un proveedor de identidad OIDC de IAM cuando desee establecer una relación de confianza entre un IdP compatible con OIDC y su Cuenta de AWS. Esto resulta útil al crear una aplicación móvil o aplicación web que requiere acceso a los recursos de AWS, pero no desea crear un código de inicio de sesión personalizado ni administrar sus propias identidades de usuario. Para obtener más información acerca de esta situación, consulte the section called “Federación OIDC”.

Puede crear y administrar un proveedor de identidad OIDC de IAM con AWS Management Console, el AWS Command Line Interface, Tools for Windows PowerShell o la API de IAM.

Después de crear un proveedor de identidades OIDC de IAM, debe crear uno más roles de IAM. Un rol es una identidad en AWS que no tiene sus propias credenciales (como las tiene un usuario). Sin embargo, en este contexto, un rol se asigna dinámicamente a un usuario federado que autentica el IdP de la organización. El rol permite al proveedor de identidad (IdP) de la organización solicitar credenciales de seguridad temporales para obtener acceso a AWS. Las políticas asignadas al rol determinan lo que los usuarios federados pueden realizar en AWS. Para crear un rol para un proveedor de identidades de terceros, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

> **Important**
>
> Al configurar políticas basadas en la identidad para acciones que admiten recursos `oidc-provider`, IAM evalúa la URL completa del proveedor de identidades del OIDC, incluidas las rutas especificadas. Si la URL de su proveedor de identidad OIDC tiene una ruta, debe incluirla en el ARN `oidc-provider` como un valor del elemento `Resource`. También tiene la opción de añadir una barra inclinada y un comodín (`/*`) al dominio URL o de utilizar caracteres comodín (`*` y `?`) en cualquier punto de la ruta URL. Si la URL del proveedor de identidad del OIDC de la solicitud no coincide con el valor establecido en el elemento `Resource` de la política, la solicitud falla.

Temas

- Creación y administración de un proveedor OIDC (consola)
- Creación y administración de un proveedor de identidad de OIDC de IAM (AWS CLI)
- Creación y administración de un proveedor de identidad de OIDC (API de AWS)

Creación y administración de un proveedor OIDC (consola)

Siga estas instrucciones para crear y administrar un proveedor de identidad de IAM OIDC en la AWS Management Console.

> **Important**
>
> Si utiliza un proveedor de identidades OIDC de Google, Facebook o Amazon Cognito, no cree un proveedor de identidades IAM independiente mediante este procedimiento. Estos proveedores de identidad de OIDC ya están integrados a AWS y están disponibles para su uso. En su lugar, siga los pasos para crear nuevos roles para su proveedor de identidades, consulte Creación de un rol para una federación de OpenID Connect (consola).

Para crear un proveedor de identidad de IAM OIDC (consola)

1. Antes de crear un proveedor de identidad de IAM OIDC, debe registrar la aplicación en el IdP para recibir un ID de cliente. El ID de cliente (también conocido como público) es un identificador único para la aplicación que se emite al registrar la aplicación con el proveedor de identidades.

Para obtener más información sobre cómo obtener un ID de cliente, consulte la documentación de su proveedor de identidad.

> **Note**
>
> AWS garantiza la comunicación con algunos proveedores de identidad (IdP) de OIDC a través de nuestra biblioteca de entidades de certificación raíz de confianza (CA) en lugar de utilizar una huella digital de certificado para verificar el certificado del servidor IdP. En estos casos, la huella digital heredada permanece en su configuración, pero ya no se utiliza para la validación. Estos proveedores de identidades de OIDC incluyen a Auth0, GitHub, GitLab, Google y aquellos que utilizan un bucket de Amazon S3 para alojar un punto de conexión JSON Web Key Set (JWKS).

2. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
3. En el panel de navegación, elija Proveedores de identidades y, a continuación, Agregar proveedor.
4. En Configurar proveedor, elija OpenID Connect.
5. En URL del proveedor, escriba la dirección URL del proveedor de identidades. La dirección URL debe cumplir las siguientes restricciones:

   - La dirección URL distingue entre mayúsculas y minúsculas.
   - La dirección URL debe comenzar por **`https://`**.
   - La dirección URL no debe contener un número de puerto.
   - En la Cuenta de AWS, cada proveedor de identidad OIDC en IAM debe utilizar una dirección URL única.

6. Elija Obtener huella digital para verificar el certificado de servidor de su proveedor de identidades. Para saber cómo hacerlo, consulte Obtención de la huella digital de un proveedor de identidades OpenID Connect.

> **Note**
>
> La cadena de certificados del proveedor de identidades OIDC debe empezar con el dominio o la URL del emisor, luego con el certificado intermedio y terminar con el certificado raíz. Si el orden de la cadena de certificados es diferente o incluye certificados duplicados o adicionales, se produce un error de discordancia de firmas y STS no podrá validar el token web JSON (JWT). Corrija el orden de los certificados de

la cadena devuelta por el servidor para resolver el error. Para obtener más información sobre los estándares de la cadena de certificados, consulte el documento certificate_list en el RFC 5246 en el sitio web de la serie RFC.

7. En Público, escriba el ID del cliente de la aplicación que ha registrado en el proveedor de identidades y recibido en Step 1. De este modo se realizarán solicitudes a AWS. Si tiene más ID de cliente (también conocido como público) para este proveedor de identidades, puede añadirlos más adelante en la página de detalles de proveedor.
8. (Opcional) En Agregar etiquetas, puede agregar pares clave-valor para ayudarlo a identificar y organizar los proveedores de identidad. También puede utilizar etiquetas para controlar el acceso a los recursos de AWS. Para obtener más información sobre cómo etiquetar proveedores de identidad OIDC IAM, consulte Etiquetado de proveedores de identidad OpenID Connect (OIDC). Seleccione Agregar etiqueta. Introduzca valores para cada par clave-valor de etiqueta.
9. Compruebe la información que ha proporcionado. Cuando haya terminado, elija Agregar proveedor.
10. Asigne un rol de IAM a su proveedor de identidades para otorgar a las identidades de usuarios externas administradas por su proveedor de identidades permisos para acceder a los recursos de AWS de su cuenta. Para obtener más información sobre cómo crear roles para la federación de identidades, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

> **Note**
>
> Los proveedores de identidad de OIDC que se utilizan en una política de confianza de roles deben estar en la misma cuenta en la que se encuentra el rol.

Para agregar o eliminar una huella digital de un proveedor de identidades OIDC de IAM (consola)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Proveedores de identidades. A continuación, elija el nombre del proveedor de identidades de IAM que desea actualizar.
3. En la sección Huellas digitales, elija Administrar. Para introducir un nuevo valor de huella digital, elija Agregar huella digital. Para eliminar una huella digital, elija Eliminar que está ubicado junto a la huella digital que desea eliminar.

> **Note**
>
> Un proveedor de identidad de IAM OIDC debe tener un mínimo de una y un máximo de cinco huellas digitales.

Cuando haya terminado, elija Guardar cambios.

Para agregar un público a un proveedor de identidades OIDC de IAM (consola)

1. En el panel de navegación, elija Proveedores de identidades y, a continuación, elija el nombre del proveedor de identidades de IAM que desea actualizar.
2. En la sección Públicos, elija Acciones y seleccione Agregar público.
3. Escriba el ID del cliente de la aplicación que registró en el proveedor de identidades y recibió en Step 1. De este modo, se realizarán solicitudes a AWS. A continuación, seleccione Agregar públicos.

> **Note**
>
> Un proveedor de identidades OIDC de IAM debe tener un mínimo de 1 y un máximo de 100 públicos.

Para eliminar un público de un proveedor de identidades OIDC de IAM (consola)

1. En el panel de navegación, elija Proveedores de identidades y, a continuación, elija el nombre del proveedor de identidades de IAM que desea actualizar.
2. En la sección Públicos, seleccione el botón de opción situado junto al público que desea eliminar y, a continuación, seleccione Acciones.
3. Elija Eliminar público. Se abrirá una nueva ventana.
4. Si elimina un público, las identidades federadas con el público no pueden asumir los roles asociados con aquel. En la ventana, lea la advertencia y confirme que desea eliminar el público escribiendo la palabra `remove` en el campo.
5. Elija Eliminar para eliminar el público.

Para eliminar un proveedor de identidad de OIDC de IAM (consola)

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Proveedores de identidades.
3. Seleccione la casilla de verificación situada junto al proveedor de identidad de IAM que desea eliminar. Se abrirá una nueva ventana.
4. Confirme que desea eliminar el proveedor escribiendo la palabra `delete` en el campo. A continuación, elija Eliminar.

Creación y administración de un proveedor de identidad de OIDC de IAM (AWS CLI)

Puede utilizar los siguientes comandos de la AWS CLI para crear y administrar proveedores de identidad de OIDC de IAM.

Para crear un proveedor de identidad OIDC de IAM (AWS CLI)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, ejecute el siguiente comando:
   - `aws iam list-open-id-connect-providers`
2. Para crear un nuevo proveedor de identidad de IAM OIDC, ejecute este comando:
   - `aws iam create-open-id-connect-provider`

Para actualizar la lista de huellas digitales de certificado de servidor para un proveedor de identidad de IAM OIDC existente (AWS CLI)

- Para actualizar la lista de huellas digitales de certificado de servidor de un proveedor de identidad de IAM OIDC, ejecute el siguiente comando:
  - `aws iam update-open-id-connect-provider-thumbprint`

Para etiquetar un proveedor de identidad OIDC de IAM existente (AWS CLI)

- Para etiquetar un proveedor de identidad de OIDC de IAM existente, ejecute este comando:
  - `aws iam tag-open-id-connect-provider`

Para enumerar etiquetas para un proveedor de identidad OIDC de IAM existente (AWS CLI)

- Para enumerar etiquetas de un proveedor de identidad OIDC de IAM existente, ejecute el siguiente comando:
  - `aws iam list-open-id-connect-provider-tags`

Para quitar etiquetas de un proveedor de identidad de OIDC de IAM (AWS CLI)

- Para quitar etiquetas de un proveedor de identidad de OIDC de IAM existente, ejecute el siguiente comando:
  - `aws iam untag-open-id-connect-provider`

Para agregar o eliminar un ID de cliente de un proveedor de identidad de IAM OIDC existente (AWS CLI)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, ejecute el siguiente comando:
   - `aws iam list-open-id-connect-providers`
2. (Opcional) Para obtener información detallada sobre un proveedor de identidad de IAM OIDC, ejecute el siguiente comando:
   - `aws iam get-open-id-connect-provider`
3. Para agregar un ID de cliente nuevo a un proveedor de identidad de IAM OIDC existente, ejecute el siguiente comando:
   - `aws iam add-client-id-to-open-id-connect-provider`
4. Para eliminar un cliente de un proveedor de identidad de IAM OIDC existente, ejecute el siguiente comando:
   - `aws iam remove-client-id-from-open-id-connect-provider`

Para eliminar un proveedor de identidad IAM OIDC (AWS CLI)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, ejecute el siguiente comando:

- `aws iam list-open-id-connect-providers`

2. (Opcional) Para obtener información detallada sobre un proveedor de identidad de IAM OIDC, ejecute el siguiente comando:

   - `aws iam get-open-id-connect-provider`

3. Para eliminar un proveedor de identidad de IAM OIDC, ejecute este comando:

   - `aws iam delete-open-id-connect-provider`

Creación y administración de un proveedor de identidad de OIDC (API de AWS)

Puedes utilizar los siguientes comandos de la API de IAM para crear y administrar proveedores OIDC.

Para crear un proveedor de identidad OIDC de IAM (API de AWS)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, llame a la operación siguiente:

   - `ListOpenIDConnectProviders`

2. Para crear un nuevo proveedor de identidad de IAM OIDC, llame a la siguiente operación:

   - `CreateOpenIDConnectProvider`

Para actualizar la lista de huellas digitales de certificado de servidor para un proveedor de identidad de IAM OIDC existente (API de AWS)

- Para actualizar la lista de huellas digitales del certificado de servidor de un proveedor de identidad de IAM OIDC, llame a la siguiente operación:

  - `UpdateOpenIDConnectProviderThumbprint`

Para etiquetar un proveedor de identidad OIDC de IAM existente (API de AWS)

- Para etiquetar un proveedor de identidad OIDC de IAM existente, llame a la siguiente operación:

  - `TagOpenIDConnectProvider`

Para enumerar etiquetas de un proveedor de identidad OIDC de IAM existente (API de AWS)

- Para enumerar etiquetas de un proveedor de identidad OIDC de IAM existente, llame a la siguiente operación:
  - `ListOpenIDConnectProviderTags`

Para quitar etiquetas de un proveedor de identidad OIDC de IAM existente (API de AWS)

- Para eliminar etiquetas de un proveedor de identidad OIDC de IAM existente, llame a la siguiente operación:
  - `UntagOpenIDConnectProvider`

Para agregar OIDC eliminar un ID de cliente de un proveedor de identidad de IAM OIDC existente (API de AWS)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, llame a la operación siguiente:
   - `ListOpenIDConnectProviders`
2. (Opcional) Para obtener información detallada sobre un proveedor de identidad de IAM OIDC, llame a la siguiente operación:
   - `GetOpenIDConnectProvider`
3. Para agregar un ID de cliente nuevo a un proveedor de identidad de IAM OIDC existente, llame a la siguiente operación:
   - `AddClientIDToOpenIDConnectProvider`
4. Para eliminar un ID de cliente de un proveedor de identidad de IAM OIDC existente, llame a la siguiente operación:
   - `RemoveClientIDFromOpenIDConnectProvider`

Para eliminar un proveedor de identidad IAM OIDC (API de AWS)

1. (Opcional) Para obtener una lista de todos los proveedores de identidad de IAM OIDC de su cuenta de AWS, llame a la operación siguiente:

- `ListOpenIDConnectProviders`

2. (Opcional) Para obtener información detallada sobre un proveedor de identidad de IAM OIDC, llame a la siguiente operación:

   - `GetOpenIDConnectProvider`

3. Para eliminar un proveedor de identidad de IAM OIDC, llame a la siguiente operación:

   - `DeleteOpenIDConnectProvider`

Obtención de la huella digital de un proveedor de identidades OpenID Connect

Cuando crea un proveedor de identidades OpenID Connect (OIDC) en la consola de IAM, debe proporcionar una huella digital. IAM requiere la huella digital de la entidad de certificación (CA) superior intermedia que firmó el certificado utilizado por el proveedor de identidades (IdP) externo. La huella digital es una firma para el certificado de CA que se utilizó para emitir el certificado para el proveedor de identidad compatible con OIDC. Cuando crea un proveedor de identidad OIDC en IAM, está dando acceso a su Cuenta de AWS a las identidades autenticadas por ese IdP. Al proporcionar la huella digital del certificado de CA, confía en cualquier certificado emitido por esa CA con el mismo nombre DNS que el registrado. Esto elimina la necesidad de actualizar las relaciones de confianza de cada cuenta al renovar el certificado de firma del proveedor de identidad.

> **Important**
>
> En la mayoría de los casos, el servidor de federación utiliza dos certificados diferentes:
>
> - El primero establece una conexión HTTPS entre AWS y el proveedor de identidades. Este lo debe emitir una CA raíz pública conocida, como AWS Certificate Manager. Esto permite al cliente comprobar la fiabilidad y el estado del certificado.
> - El segundo se emplea para cifrar tokens y debe estar firmado por una CA raíz privada o pública.

Puede crear y administrar un proveedor de identidad OIDC de IAM con AWS Command Line Interface, Tools for Windows PowerShell o la API de IAM. Cuando utilice estos métodos, la huella digital será opcional. Si decide no incluir una huella digital, IAM recuperará la huella digital de la CA intermedia superior del certificado del servidor IdP del OIDC. Si opta por incluir una huella digital, deberá obtenerla manualmente y proporcionarla a AWS.

Cuando crea un proveedor de identidades OIDC con la consola de IAM, esta intenta recuperar la huella digital en su lugar. Le recomendamos que obtenga también manualmente la huella digital del IdP OIDC y que verifique que la consola obtuvo la huella digital correcta. Para obtener más información sobre cómo obtener huellas digitales de certificados, consulte las siguientes secciones.

Obtener huellas digitales de certificados

Utilice un navegador Web y la herramienta de línea de comandos OpenSSL para obtener la huella digital de un proveedor OIDC. Sin embargo, no es necesario que obtenga manualmente la huella digital del certificado para crear un proveedor de identidades OIDC de IAM. Puede usar el siguiente procedimiento para obtener la huella digital del certificado de su proveedor OIDC.

> **Note**
>
> AWS garantiza la comunicación con algunos proveedores de identidad (IdP) de OIDC a través de nuestra biblioteca de entidades de certificación raíz de confianza (CA) en lugar de utilizar una huella digital de certificado para verificar el certificado del servidor IdP. En estos casos, la huella digital heredada permanece en su configuración, pero ya no se utiliza para la validación. Estos proveedores de identidades de OIDC incluyen a Auth0, GitHub, GitLab, Google y aquellos que utilizan un bucket de Amazon S3 para alojar un punto de conexión JSON Web Key Set (JWKS).

Para obtener la huella digital de un proveedor de identidades OIDC

1. Para poder obtener la huella digital de un proveedor de identidad OIDC, debe obtener primero la herramienta de línea de comandos OpenSSL. Esta herramienta le permite descargar la cadena de certificados del proveedor de identidades OIDC IdP y generar una huella digital del certificado final en la cadena de certificados. Si necesita instalar y configurar OpenSSL, siga las instrucciones que figuran en Instalación de OpenSSL y Configuración de OpenSSL.
2. Comience con la URL del IdP OIDC (por ejemplo, `https://server.example.com`) y, a continuación, agregue `/.well-known/openid-configuration` para formar la dirección URL del documento de configuración del IdP, de este modo:

   `https://`*`server.example.com`*`/.well-known/openid-configuration`

   Abra esta URL en un navegador web y sustituya *`servidor.ejemplo.com`* por el nombre del servidor del proveedor de identidades.

3. En el documento que se muestra, utilice la característica de su navegador web Buscar para localizar el texto de `"jwks_uri"`. Justo después del texto `"jwks_uri"`, verá dos puntos (:) seguidos de una URL. Copie el nombre completo del dominio de la URL. No incluya `https://` ni ninguna ruta que vaya después del dominio de nivel superior.

```
{
 "issuer": "https://accounts.example.com",
 "authorization_endpoint": "https://accounts.example.com/o/oauth2/v2/auth",
 "device_authorization_endpoint": "https://oauth2.exampleapis.com/device/code",
 "token_endpoint": "https://oauth2.exampleapis.com/token",
 "userinfo_endpoint": "https://openidconnect.exampleapis.com/v1/userinfo",
 "revocation_endpoint": "https://oauth2.exampleapis.com/revoke",
 "jwks_uri": "https://www.exampleapis.com/oauth2/v3/certs",
...
```

4. Utilice la herramienta de línea de comandos OpenSSL para ejecutar el siguiente comando. Sustituya *claves.ejemplo.com* por el nombre de dominio que ha obtenido en Step 3.

```
openssl s_client -servername keys.example.com -showcerts -
connect keys.example.com:443
```

5. En la ventana de comandos, desplácese hacia arriba hasta que vea un certificado similar al del ejemplo siguiente. Si ve más de un certificado, busque el último certificado que se muestre (en la parte inferior de la salida del comando). Este contendrá el certificado de la CA superior intermedia en la cadena de autoridad de certificación.

```
-----BEGIN CERTIFICATE-----
 MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
 VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
 b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
 BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
 MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
 VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
 b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
 YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
 21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
 rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
 Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
 nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
 FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
 NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE=
```

```
-----END CERTIFICATE-----
```

Copie todo el certificado (incluidas las líneas `-----BEGIN CERTIFICATE-----` y `-----END CERTIFICATE-----`) y péguelo en un archivo de texto. A continuación, guarde el archivo con el nombre de archivo **`certificate.crt`**.

> **Note**
>
> La cadena de certificados del proveedor de identidades OIDC debe empezar con el dominio o la URL del emisor, luego con el certificado intermedio y terminar con el certificado raíz. Si el orden de la cadena de certificados es diferente o incluye certificados duplicados o adicionales, se produce un error de discordancia de firmas y STS no podrá validar el token web JSON (JWT). Corrija el orden de los certificados de la cadena devuelta por el servidor para resolver el error. Para obtener más información sobre los estándares de la cadena de certificados, consulte el documento certificate_list en el RFC 5246 en el sitio web de la serie RFC.

6. Utilice la herramienta de línea de comandos OpenSSL para ejecutar el siguiente comando.

```
openssl x509 -in certificate.crt -fingerprint -sha1 -noout
```

La ventana de comandos muestra la huella digital del certificado, que tiene un aspecto similar al del siguiente ejemplo:

```
SHA1 Fingerprint=99:0F:41:93:97:2F:2B:EC:F1:2D:DE:DA:52:37:F9:C9:52:F2:0D:9E
```

Elimine todos los dos puntos (:) de esta cadena para generar la huella digital final, tal y como se muestra a continuación:

```
990F4193972F2BECF12DDEDA5237F9C952F20D9E
```

7. Si desea crear el proveedor de identidad de IAM OIDC con la AWS CLI, las Tools for Windows PowerShell o la API de IAM, proporcione esta huella digital al crear el proveedor.

   Si desea crear el proveedor de identidad IAM OIDC en la consola de IAM, compare esta huella digital con la que se muestra en la página Verificar información de proveedor de la consola al crear un proveedor OIDC.

> **Important**
>
> Si la huella digital que ha obtenido no coincide con la que se muestra en la consola, no debe crear el proveedor OIDC en la consola. Espere un rato, vuelva a intentar crear el proveedor OIDC y asegúrese de que las huellas digitales coincidan antes de crear el proveedor. Si siguen sin coincidir después de intentarlo por segunda vez, utilice el foro de IAM para ponerse en contacto con AWS.

Instalación de OpenSSL

Si aún no dispone de OpenSSL instalado, siga las instrucciones indicadas en esta sección.

Para instalar OpenSSL en Linux o Unix

1. Vaya a OpenSSL: Fuente, Tarballs (https://openssl.org/source/).
2. Descargue la fuente más reciente y cree el paquete.

Para instalar OpenSSL en Windows

1. Vaya a OpenSSL: distribuciones binarias (https://wiki.openssl.org/index.php/Binaries) para obtener una lista de sitios desde los que puede instalar la versión de Windows.
2. Siga las instrucciones del sitio seleccionado para iniciar la instalación.
3. Si se le solicita que instale Microsoft Visual C++ 2008 Redistributables y aún no está instalado en su sistema, elija el enlace de descarga adecuado para su entorno. Siga las instrucciones proporcionadas por el Asistente para la instalación de Microsoft Visual C++ 2008 Redistributable.

> **Note**
>
> Si no está seguro de si Microsoft Visual C++ 2008 Redistributables ya está instalado en el sistema, puede intentar instalar OpenSSL primero. El instalador de OpenSSL muestra una alerta si Microsoft Visual C++ 2008 Redistributables aún no está instalado. Asegúrese de instalar la arquitectura (32 bits o 64 bits) que coincide con la versión de OpenSSL que instale.

4. Después de instalar Microsoft Visual C++ 2008 Redistributables, seleccione la versión adecuada de los archivos binarios OpenSSL para su entorno y guarde el archivo localmente. Inicie el Asistente de instalación de OpenSSL.
5. Siga las instrucciones descritas en el Asistente de instalación de OpenSSL.

Configuración de OpenSSL

Antes de utilizar los comandos de OpenSSL, debe configurar el sistema operativo para que tenga información sobre la ubicación donde está instalado OpenSSL.

Para configurar OpenSSL en Linux o Unix

1. En la línea de comandos, establezca la variable `OpenSSL_HOME` a la ubicación de la instalación de OpenSSL:

   ```
   $ export OpenSSL_HOME=path_to_your_OpenSSL_installation
   ```

2. Establezca la ruta para incluir la instalación de OpenSSL:

   ```
   $ export PATH=$PATH:$OpenSSL_HOME/bin
   ```

   > **Note**
   >
   > Los cambios que haga en las variables de entorno mediante el comando `export` solo son válidas para la sesión actual. Puede realizar cambios persistentes en las variables de entorno configurándolas en el archivo de configuración de su shell. Para obtener más información, consulte la documentación del sistema operativo.

Para configurar OpenSSL en Windows

1. Abra una ventana del símbolo del sistema.
2. Establezca la variable `OpenSSL_HOME` a la ubicación de la instalación de OpenSSL:

   ```
   C:\> set OpenSSL_HOME=path_to_your_OpenSSL_installation
   ```

3. Establezca la variable `OpenSSL_CONF` a la ubicación del archivo de configuración en su instalación OpenSSL:

```
C:\> set OpenSSL_CONF=path_to_your_OpenSSL_installation\bin\openssl.cfg
```

4. Establezca la ruta para incluir la instalación de OpenSSL:

```
C:\> set Path=%Path%;%OpenSSL_HOME%\bin
```

> **Note**
>
> Cualquier cambio que realice en las variables de entorno de Windows en una ventana del Símbolo del sistema es válido solo para la sesión actual de la línea de comandos. Puede realizar cambios persistentes en las variables de entorno configurándolas como propiedades del sistema. Los procedimientos exactos dependen de la versión de Windows que esté utilizando. (Por ejemplo, en Windows 7, abra Panel de control, Seguridad y sistema, Sistema. A continuación, elija Configuración avanzada del sistema, pestaña Avanzado, Variables de entorno.) Para obtener más información, consulte la documentación de Windows.

Identifique a los usuarios con la federación OIDC

Al crear políticas de acceso en IAM, suele ser útil poder especificar los permisos en función de las aplicaciones configuradas y el ID de los usuarios que se han autenticado con un proveedor de identidad (IdP) externo. Por ejemplo, la aplicación móvil que utiliza la federación OIDC podría mantener información en Amazon S3 con una estructura de este tipo:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
...
myBucket/app2/user1
myBucket/app2/user2
myBucket/app2/user3
...
```

Es posible que también quiera distinguir estas rutas de acceso según el proveedor. En tal caso, la estructura podría ser la siguiente (solo se muestran dos proveedores para ahorrar espacio):

```
myBucket/Amazon/app1/user1
```

```
myBucket/Amazon/app1/user2
myBucket/Amazon/app1/user3
...
myBucket/Amazon/app2/user1
myBucket/Amazon/app2/user2
myBucket/Amazon/app2/user3

myBucket/Facebook/app1/user1
myBucket/Facebook/app1/user2
myBucket/Facebook/app1/user3
...
myBucket/Facebook/app2/user1
myBucket/Facebook/app2/user2
myBucket/Facebook/app2/user3
...
```

En el caso de estas estructuras, `app1` y `app2` representan distintas aplicaciones, tales como juegos diferentes, y cada usuario de la aplicación tiene una carpeta distinta. Los valores de `app1` y `app2` pueden ser nombres fáciles de recordar que usted asigna (por ejemplo, `mynumbersgame`) o pueden ser los ID de las aplicaciones que los proveedores asignan al configurar su aplicación. Si decide incluir nombres de proveedor en la ruta de acceso, estos también pueden ser nombres fáciles de recordar, tales como `Cognito`, `Amazon`, `Facebook` y `Google`.

Por lo general, puede crear las carpetas de `app1` y `app2` a través de la AWS Management Console, ya que los nombres de las aplicaciones son valores estáticos. Eso es así también si incluye el nombre del proveedor en la ruta de acceso, ya que el nombre del proveedor es también un valor estático. En cambio, las carpetas específicas de usuarios (*`user1`*, *`user2`*, *`user3`*, etc.) deben crearse en el tiempo de ejecución desde la aplicación con el ID de usuario que está disponible en el valor `SubjectFromWebIdentityToken` devuelto por la solicitud realizada a `AssumeRoleWithWebIdentity`.

Para escribir políticas que permitan el acceso exclusivo a los recursos por parte de usuarios individuales, puede hacer coincidir el nombre de la carpeta completa, incluido el nombre del proveedor y el nombre de la aplicación, si está utilizando esta opción. A continuación, puede incluir las siguientes claves de contexto específicas del proveedor que hagan referencia al ID de usuario que el proveedor devuelve:

- `cognito-identity.amazonaws.com:sub`
- `www.amazon.com:user_id`
- `graph.facebook.com:id`

- `accounts.google.com:sub`

En el caso de los proveedores de OIDC, utilice la dirección URL completa del proveedor de OIDC con la clave de subcontexto, tal y como se indica en el siguiente ejemplo:

- *`server.example.com`*`:sub`

En el siguiente ejemplo se muestra una política de permisos que concede acceso a un bucket de Amazon S3 solo si el prefijo del bucket coincide con la cadena:

`myBucket/Amazon/mynumbersgame/`*`user1`*

En este ejemplo se presupone que el usuario ha iniciado sesión con Login with Amazon y que el usuario utiliza una aplicación denominada `mynumbersgame`. El ID único del usuario se presenta como un atributo denominado `user_id`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::myBucket"],
      "Condition": {"StringLike": {"s3:prefix": ["Amazon/mynumbersgame/
${www.amazon.com:user_id}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}",
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}/*"
      ]
    }
  ]
}
```

Podría crear políticas similares para los usuarios que inician sesión con Amazon Cognito, Facebook, Google o cualquier otro proveedor de identidades (IdP) compatible con OpenID Connect. Estas políticas utilizarían un nombre de proveedor diferente como parte de la ruta de acceso, así como diferentes ID de aplicaciones.

Para obtener más información sobre las claves de federación de OIDC disponibles para verificaciones de condición en las políticas, consulte Claves disponibles para las federaciones de identidades AWS de OIDC.

Recursos adicionales para federaciones de OIDC

Los siguientes recursos pueden ayudarle a obtener más información sobre la federaciones de OIDC:

- Usa OpenID Connect en tus flujos de trabajo de GitHub mediante la configuración de OpenID Connect en Amazon Web Services
- Amazon Cognito Identity en la Guía de bibliotecas de Amplify para Android y Amazon Cognito Identity en la Guía de bibliotecas de Amplify para Swift.
- Automatizar las funciones de identidad Web de IAM AWS basadas en OpenID Connect con Microsoft Entra ID en el blog AWS Partner Network (APN) explica cómo autenticar procesos automatizados en segundo plano o aplicaciones que se ejecutan fuera del uso de la autorización OIDC de máquina a máquina AWS.
- En el artículo Web Identity Federation with Mobile Applications se explican las federaciones de OIDC y se muestra un ejemplo de cómo utilizarlas para obtener acceso al contenido de Amazon S3.

Federación SAML 2.0

AWS admite la federación de identidades con SAML 2.0 (Lenguaje de marcado para confirmaciones de seguridad 2.0), un estándar abierto que utilizan muchos proveedores de identidad (IdP). Esta característica permite el inicio de sesión único (SSO) federado para que los usuarios puedan iniciar sesión en la AWS Management Console o invocar las operaciones de la API de AWS sin necesidad de crear un usuario de IAM para cada persona de la organización. Al utilizar SAML, puede simplificar el proceso de configuración de la federación con AWS, ya que puede utilizar el servicio del proveedor de identidades en lugar de escribir el código proxy de identidad personalizado.

La federación de IAM admite estos casos de uso:

- Acceso federado para permitir que un usuario o aplicación de su organización llame a las operaciones de API de AWS. Utiliza una afirmación SAML (como parte de la respuesta de autenticación) que se genera en la organización para obtener credenciales de seguridad temporales. Esta situación es similar a otras situaciones de federación que admite IAM, como las descritas en Solicitud de credenciales de seguridad temporales y Federación OIDC. Sin embargo, los proveedores de identidad SAML 2.0 de su organización gestionan gran parte de los detalles en el tiempo de ejecución para realizar la comprobación de la autenticación y la autorización. Esta es la situación que se ha tratado en este tema.
- Inicio de sesión único (SSO) basado en web en la AWS Management Console desde su organización. Los usuarios pueden iniciar sesión en un portal de la organización alojado por un proveedor de identidades compatible con SAML 2.0, seleccionar una opción para ir a AWS y ser redireccionados hacia la consola sin tener que proporcionar información de inicio de sesión adicional. Puede utilizar un IdP SAML de terceros para establecer el acceso SSO a la consola, o también puede crear un IdP personalizado para conceder acceso a la consola a los usuarios externos. Para obtener más información acerca de la creación de un proveedor de identidad personalizado, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

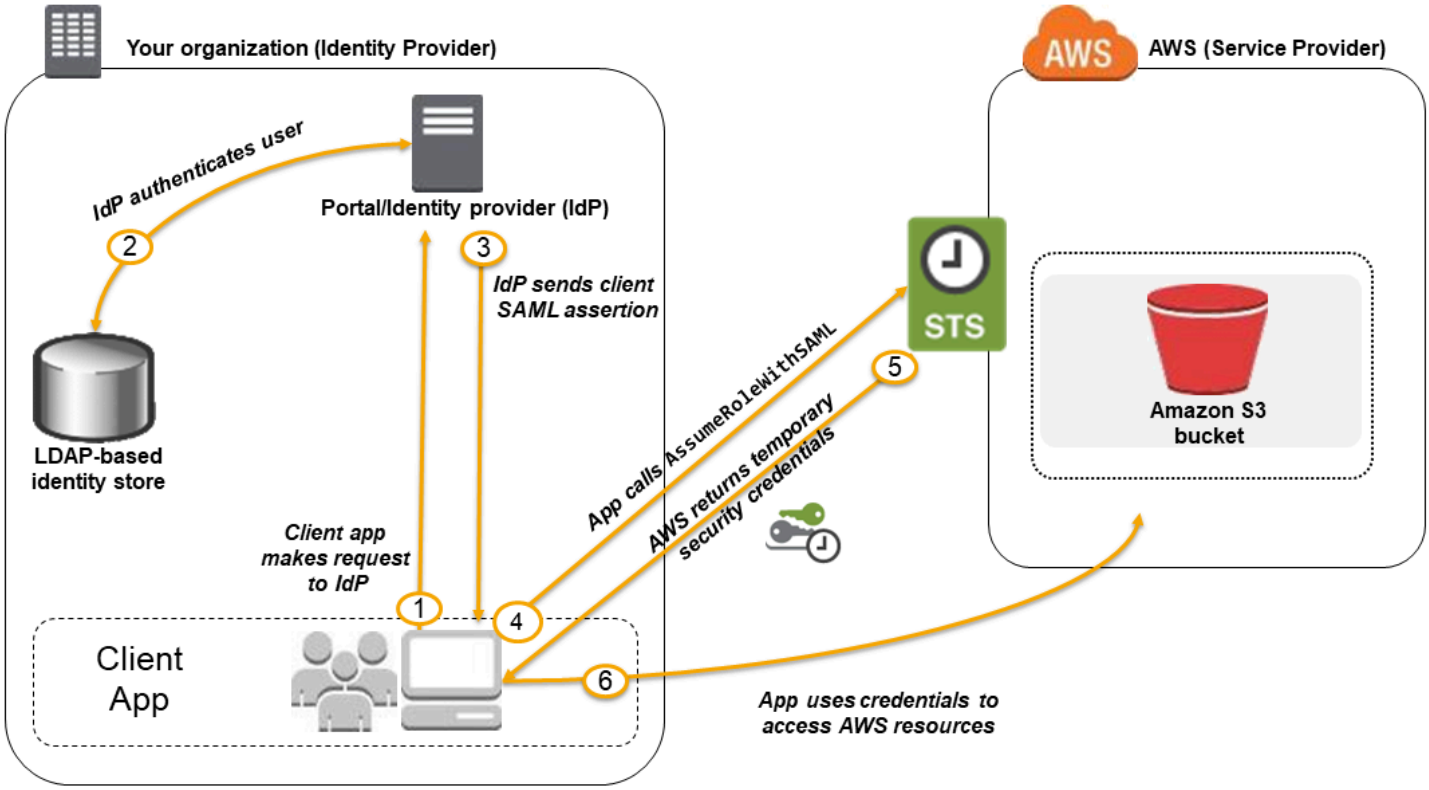
Temas

- Uso de la federación basada en SAML para el acceso a la API de AWS
- Información general sobre la configuración de la federación basada en SAML 2.0
- Información general acerca del rol que permite el acceso federado SAML a los recursos de AWS
- Identificación única de los usuarios en la federación basada en SAML
- Crear un proveedor de identidades de SAML en IAM
- Configuración su SAML 2.0 IdP con una relación de confianza para usuario autenticado y agregando reclamos
- Integración de proveedores de soluciones SAML externos con AWS
- Configure aserciones SAML para la respuesta de autenticación
- Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0

Uso de la federación basada en SAML para el acceso a la API de AWS

Imagine que desea proporcionar un método para que los empleados copien datos de sus equipos a una carpeta de copia de seguridad. Puede crear una aplicación que los usuarios pueden ejecutar en

sus equipos. En la etapa final, la aplicación lee y escribe objetos en un bucket de S3. Los usuarios no tienen acceso directo a AWS. En su lugar, se utiliza el siguiente proceso:



1. Un usuario de su organización utiliza una aplicación cliente para solicitar autenticación del proveedor de identidad de su organización.
2. El proveedor de identidad autentica al usuario en función del almacén de identidades de la organización.
3. El proveedor de identidad construye una aserción de SAML con información sobre el usuario y envía dicha aserción a la aplicación cliente.
4. La aplicación cliente llama a la API AWS STS`AssumeRoleWithSAML de` , transfiriendo el ARN del proveedor SAML, el ARN del rol a asumir y la aserción de SAML del proveedor de identidad.
5. La respuesta de la API a la aplicación cliente incluye credenciales de seguridad temporales.
6. La aplicación cliente usa las credenciales de seguridad temporales para llamar a las operaciones de API de Amazon S3.

Información general sobre la configuración de la federación basada en SAML 2.0

Antes de poder utilizar la federación basada en SAML 2.0 tal y como se describe en la situación anterior y en el diagrama, debe configurar el proveedor de identidad de su organización y su Cuenta

de AWS para que confíen el uno en el otro. El proceso general para configurar esta confianza se describe en los pasos siguientes. Dentro de su organización, debe contar con un proveedor de identidades compatible con SAML 2.0, como Microsoft Active Directory Federation Service (AD FS, parte de Windows Server), Shibboleth u otro proveedor SAML 2.0 compatible.

> **Note**
>
> Para mejorar la resiliencia de la federación, le recomendamos que configure su IdP y su federación de AWS para que admitan varios puntos de conexión de inicio de sesión de SAML. Para obtener más información, consulte el artículo del blog sobre seguridad de AWS, How to use regional SAML endpoints for failover.

Para configurar que el proveedor de identidad de la organización y la cuenta de AWS confíen el uno en el otro

1. Registre AWS como proveedor de servicios (SP) con el IdP de su organización. Utilice el documento de metadatos SAML de `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`

   Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS.

   Opcionalmente, puede usar el documento de metadatos SAML de `https://signin.aws.amazon.com/static/saml-metadata.xml` .
2. Con el IdP de la organización puede generar un archivo XML de metadatos equivalente que describe su IdP como proveedor de identidad de IAM en AWS. Debe incluir el nombre de emisor, una fecha de creación, una fecha de vencimiento y claves que AWS puede utilizar para validar las respuestas de autenticación (aserciones) de la organización.
3. En la consola de IAM, debe crear una entidad de proveedor de identidad SAML. Como parte de este proceso, puede cargar el documento de metadatos SAML producido por el proveedor de identidad en la organización en Step 2. Para obtener más información, consulte Crear un proveedor de identidades de SAML en IAM.
4. En IAM, debe crear uno o varios roles de IAM. En la política de confianza del rol, se establece el proveedor SAML como principal, que establece una relación de confianza entre la organización y AWS. La política de permisos del rol establece qué usuarios de la organización pueden

realizar qué cosas en AWS. Para obtener más información, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

> **Note**
>
> Los proveedores de identidad de SAML que se utilizan en una política de confianza de roles deben estar en la misma cuenta en la que se encuentra el rol.

5. En el proveedor de identidad de la organización, defina aserciones que asignen usuarios o grupos de la organización a los roles de IAM. Tenga en cuenta que los distintos usuarios y grupos de la organización pueden asignarse a diferentes roles de IAM. Los pasos exactos para llevar a cabo la asignación dependerán del proveedor de identidad que esté utilizando. En la situación anterior de una carpeta de Amazon S3 para los usuarios, es posible que todos los usuarios estén asignados al mismo rol que proporciona los permisos de Amazon S3. Para obtener más información, consulte Configure aserciones SAML para la respuesta de autenticación.

   Si el proveedor de identidades activa el inicio de sesión único en la consola de AWS, puede configurar la duración máxima de las sesiones de la consola. Para obtener más información, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 .

   > **Note**
   >
   > La implementación de AWS de la federación SAML 2.0 no es compatible con aserciones SAML cifradas entre el proveedor de identidades IAM y AWS. Sin embargo, el tráfico entre los sistemas del cliente y AWS se transmite a través de un canal (TLS) cifrado.

6. En la aplicación que esté creando, llame a la API AWS Security Token Service `AssumeRoleWithSAML`, que transfiere el ARN del proveedor SAML que creó en Step 3, el ARN del rol a asumir que creó en Step 4 y la aserción de SAML sobre el usuario actual que obtiene del proveedor de identidades. AWS se asegurará de que la solicitud para asumir el rol procede del proveedor de identidades al que se hace referencia en el proveedor SAML.

   Para obtener más información, consulte AssumeRoleWithSAML en la Referencia de la API de AWS Security Token Service.

7. Si la solicitud se realiza correctamente, la API devuelve un conjunto de credenciales de seguridad temporales, que su aplicación puede utilizar para realizar solicitudes firmadas a AWS.

Su aplicación tiene información sobre el usuario actual y puede acceder a carpetas específicas del usuario en Amazon S3, tal y como se describe en la situación anterior.

Información general acerca del rol que permite el acceso federado SAML a los recursos de AWS

El rol o roles que crea en IAM definen lo que los usuarios federados de la organización pueden realizar en AWS. Al crear la política de confianza para el rol, debe especificar el proveedor SAML que creó anteriormente como `Principal`. Además puede ampliar la política de confianza con una `Condition` para permitir únicamente a los usuarios que cumplen determinados atributos SAML acceder al rol. Por ejemplo, puede especificar que solo los usuarios cuya afiliación SAML es `staff` (tal como afirma https://openidp.feide.no) estén autorizados para acceder al rol, tal y como se muestra en la siguiente política de muestra:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/saml",
        "saml:iss": "https://openidp.feide.no"
      },
      "ForAllValues:StringLike": {"saml:edupersonaffiliation": ["staff"]}
    }
  }]
}
```

> **Note**
>
> Los proveedores de identidad de SAML que se utilizan en una política de confianza de roles deben estar en la misma cuenta en la que se encuentra el rol.

Para obtener más información sobre las claves de SAML que puede revisar en una política, consulte Claves disponibles para la federación AWS STS basada en SAML.

Puede incluir puntos de conexión regionales para el atributo `saml:aud` en `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS.

Para la política de permisos del rol, debe especificar los permisos de la misma forma que haría para cualquier rol. Por ejemplo, si los usuarios de la organización pueden administrar las instancias Amazon Elastic Compute Cloud (EC2), debe permitir de forma explícita las acciones de Amazon EC2 en la política de permisos, como los de la política administrada AmazonEC2FullAccess.

Identificación única de los usuarios en la federación basada en SAML

Al crear políticas de acceso en IAM, a menudo es útil poder especificar los permisos basados en la identidad de los usuarios. Por ejemplo, en el caso de los usuarios que se han federado mediante SAML, es posible que una aplicación quiera conservar información en Amazon S3 utilizando una estructura de este tipo:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

Puede crear el bucket (`myBucket`) y la carpeta (`app1`) a través de la consola de Amazon S3 o la AWS CLI, pues son valores estáticos. Sin embargo, las carpetas específicas de usuarios (*user1*, *user2*, *user3*, etc.) deben crearse en el tiempo de ejecución utilizando código, ya que el valor que identifica al usuario es desconocido hasta la primera vez que el usuario inicia sesión a través del proceso de federación.

Para escribir políticas que hacen referencia a detalles específicos del usuario como parte de un nombre de recurso, la identidad del usuario tiene que estar disponible en claves SAML que puedan utilizarse en las condiciones de políticas. Las siguientes claves están disponibles para la federación basada en SAML 2.0 para utilizarlas en las políticas de IAM. Puede utilizar los valores devueltos por las siguientes claves para crear identificadores de usuario únicos para recursos como carpetas de Amazon S3.

- `saml:namequalifier`. Un valor hash basado en la concatenación del `Issuer` valor de respuesta (`saml:iss`) y una cadena con el ID de cuenta `AWS` y el nombre fácil de recordar (la última parte del ARN) del proveedor SAML en IAM. La concatenación del ID de cuenta y del nombre fácil de recordar del proveedor SAML está disponible para las políticas de IAM como clave `saml:doc`. El ID de cuenta y el nombre de proveedor deben separarse con una "/" como en

"123456789012/provider_name". Para obtener más información, consulte la clave `saml:doc` en Claves disponibles para la federación AWS STS basada en SAML.

La combinación de `NameQualifier` y `Subject` se puede utilizar para identificar un usuario federado de forma unívoca. El pseudocódigo siguiente muestra cómo se calcula este valor. En este pseudocódigo, + indica una concatenación, `SHA1` representa una función que genera un resumen del mensaje utilizando SHA-1 y `Base64` representa una función que genera una versión con codificación Base64 de la salida del hash.

```
Base64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
MySAMLIdP" ) )
```

Para obtener más información acerca de las claves de la política que están disponibles para la federación basada en SAML, consulte Claves disponibles para la federación AWS STS basada en SAML.

- `saml:sub` (cadena). Se trata del asunto de la demanda, que incluye un valor que identifica de forma unívoca a un usuario individual dentro de una organización (por ejemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).
- `saml:sub_type` (cadena). Esta clave puede ser `persistent`, `transient` o la URI completa `Format` de los elementos `Subject` y `NameID` utilizados en la aserción SAML. Un valor de `persistent` indica que el valor de `saml:sub` es el mismo para un usuario en todas las sesiones. Si el valor es `transient`, el usuario tendrá un valor `saml:sub` diferente para cada sesión. Para obtener información sobre el atributo `NameID` del elemento `Format`, consulte Configure aserciones SAML para la respuesta de autenticación.

El siguiente ejemplo muestra una política de permisos que utiliza las claves anteriores para conceder permisos a una carpeta específica de usuario en Amazon S3. La política presupone que los objetos de Amazon S3 se identifican utilizando un prefijo que incluye tanto `saml:namequalifier` y `saml:sub`. Observe que el elemento `Condition` incluye una prueba para asegurarse de que `saml:sub_type` está fijado en `persistent`. Si se establece como `transient`, el valor `saml:sub` para el usuario puede ser diferente en cada sesión, por lo que la combinación de los valores no debe utilizarse para identificar las carpetas específicas del usuario.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
```

```
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}/*"
    ],
    "Condition": {"StringEquals": {"saml:sub_type": "persistent"}}
  }
}
```

Para obtener más información sobre la asignación de aserciones del proveedor de identidad a claves de políticas, consulte Configure aserciones SAML para la respuesta de autenticación.

Crear un proveedor de identidades de SAML en IAM

Un proveedor de identidad de IAM SAML 2.0 es una entidad de IAM que describe un servicio de proveedor de identidad (IdP) externo compatible con el estándar SAML 2.0 (Lenguaje de marcado para confirmaciones de seguridad 2.0). Un proveedor de identidad de IAM SAML se utiliza para establecer una relación de confianza entre un IdP compatible con SAML, como Shibboleth o Active Directory Federation Services, y AWS, de modo que los usuarios de su organización tengan acceso a los recursos de AWS. Los proveedores de identidad de IAM SAML se utilizan como entidades principales en una política de confianza de IAM.

Para obtener más información acerca de esta situación, consulte Federación SAML 2.0.

Puede crear y administrar un proveedor de identidad de IAM en el AWS Management Console o con AWS CLI, Tools for Windows PowerShell o la API de IAM AWS.

Después de crear un proveedor SAML, debe crear uno varios roles de IAM. Un rol es una identidad en AWS que no tiene sus propias credenciales (como las tiene un usuario). Sin embargo, en este contexto, un rol se asigna dinámicamente a un usuario federado que autentica el IdP de la organización. El rol permite al proveedor de identidad (IdP) de la organización solicitar credenciales de seguridad temporales para obtener acceso a AWS. Las políticas asignadas al rol determinan lo que los usuarios federados pueden realizar en AWS. Para crear un rol para la federación SAML, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

Por último, después de crear el rol, complete la relación de confianza SAML configurando su proveedor de identidad (IdP) con información sobre AWS y los roles que desea que los usuarios federados utilicen. Esto se denomina configuración de la relación de confianza para usuario

autenticado entre su proveedor de identidades y AWS. Para configurar una relación de confianza, consulte Configuración su SAML 2.0 IdP con una relación de confianza para usuario autenticado y agregando reclamos.

Temas

- Creación y administración de un proveedor de identidad de IAM SAML (consola)
- Creación y administración de un proveedor de identidad SAML de IAM (AWS CLI)
- Creación y administración de un proveedor de identidad SAML de IAM (API de AWS)

Creación y administración de un proveedor de identidad de IAM SAML (consola)

Puede utilizar la AWS Management Console para crear y eliminar proveedores de identidad IAM SAML.

Para crear un proveedor de identidades SAML de IAM (consola)

1. Antes de poder crear un proveedor de identidades SAML de IAM, necesita el documento de metadatos de SAML que se obtiene del proveedor de identidades. Este documento incluye el nombre del emisor, la información del vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) que se reciben del proveedor de identidades. Para generar el documento de metadatos, utilice el software de administración de identidades que su organización utiliza como proveedor de identidad (IdP). Para obtener instrucciones sobre cómo configurar muchos de los proveedores de identidades (IdP) que funcionan con AWS y sobre cómo generar el documento de metadatos de SAML, consulte Integración de proveedores de soluciones SAML externos con AWS.

   > **Important**
   >
   > Este archivo de metadatos incluye el nombre del emisor, información de vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) recibida desde el IdP. El archivo de metadatos debe estar codificado en formato UTF-8 sin una marca de orden de bytes (BOM). Para eliminar la BOM, puede codificar el archivo en UTF-8 con una herramienta de edición de texto, como Notepad++. El certificado x.509 incluido como parte del documento de metadatos de SAML debe utilizar un tamaño de clave de al menos 1024 bits. Además, el certificado x.509 no debe contener extensiones repetidas. Puede utilizar extensiones, pero estas solo pueden aparecer una vez en el certificado. Si el certificado x.509 no cumple ninguna

condición, se produce un error en la creación de proveedor de identidad (IdP) y devuelve el mensaje “Unable to parse metadata” (No se pueden analizar los metadatos). Según se define en la versión 1.0 del perfil de interoperabilidad de metadatos SAML V2.0, IAM no evalúa ni toma medidas en relación con la caducidad del certificado X.509 del documento de metadatos.

2. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
3. En el panel de navegación, elija Identity providers (Proveedores de identidades) y, a continuación, Add provider (Agregar proveedor).
4. En Configure provider (Configurar proveedor), elija SAML.
5. Escriba un nombre para el proveedor de identidad.
6. En Metadata document (Documento de metadatos), haga clic en Choose File (Elegir archivo) y especifique el documento de metadatos de SAML que descargó en Step 1.
7. (Opcional) En Add tags (Agregar etiquetas), puede agregar pares clave-valor para ayudarlo a identificar y organizar los proveedores de identidad. También puede utilizar etiquetas para controlar el acceso a los recursos de AWS. Para obtener más información sobre cómo etiquetar proveedores de identidad SAML, consulte Etiquetado de proveedores de identidad SAML de IAM.

   Seleccione Agregar etiqueta. Introduzca valores para cada par clave-valor de etiqueta.
8. Compruebe la información que ha proporcionado. Cuando haya terminado, elija Add provider (Agregar proveedor).
9. Asigne un rol de IAM a su proveedor de identidades para otorgar a las identidades de usuarios externas administradas por su proveedor de identidades permisos para acceder a los recursos de AWS de su cuenta. Para obtener más información sobre cómo crear roles para la federación de identidades, consulte Creación de un rol para un proveedor de identidad de terceros (federación).

> **Note**
>
> Los proveedores de identidad de SAML que se utilizan en una política de confianza de roles deben estar en la misma cuenta en la que se encuentra el rol.

Para eliminar un proveedor SAML (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Proveedores de identidades.
3. Seleccione el botón de opción situado junto al proveedor de identidades que desea eliminar.
4. Elija Eliminar. Se abrirá una nueva ventana.
5. Confirme que desea eliminar el proveedor escribiendo la palabra `delete` en el campo. A continuación, elija Eliminar.

Creación y administración de un proveedor de identidad SAML de IAM (AWS CLI)

Puedes utilizar la AWS CLI para crear y administrar proveedores SAML.

Antes de poder crear un proveedor de identidades de IAM, necesita el documento de metadatos de SAML que se obtiene del proveedor de identidades. Este documento incluye el nombre del emisor, la información del vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) que se reciben del proveedor de identidades. Para generar el documento de metadatos, utilice el software de administración de identidades que su organización utiliza como proveedor de identidad (IdP). Para obtener instrucciones sobre cómo configurar muchos de los proveedores de identidades (IdP) que funcionan con AWS y sobre cómo generar el documento de metadatos de SAML, consulte Integración de proveedores de soluciones SAML externos con AWS.

> **Important**
>
> Este archivo de metadatos incluye el nombre del emisor, información de vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) recibida desde el IdP. El archivo de metadatos debe estar codificado en formato UTF-8 sin una marca de orden de bytes (BOM). Para eliminar la BOM, puede codificar el archivo en UTF-8 con una herramienta de edición de texto, como Notepad++.
> El certificado x.509 incluido como parte del documento de metadatos de SAML debe utilizar un tamaño de clave de al menos 1024 bits. Además, el certificado x.509 no debe contener extensiones repetidas. Puede utilizar extensiones, pero estas solo pueden aparecer una vez en el certificado. Si el certificado x.509 no cumple ninguna condición, se produce un error en la creación de proveedor de identidad (IdP) y devuelve el mensaje “Unable to parse metadata” (No se pueden analizar los metadatos).

Según se define en la versión 1.0 del perfil de interoperabilidad de metadatos SAML V2.0, IAM no evalúa ni toma medidas en relación con la caducidad del certificado X.509 del documento de metadatos.

Para crear un proveedor de identidad de IAM y cargar un documento de metadatos (AWS CLI)

- Ejecute este comando: `aws iam create-saml-provider`

Para cargar un nuevo documento de metadatos para un proveedor de identidad de IAM (AWS CLI)

- Ejecute este comando: `aws iam update-saml-provider`

Para etiquetar un proveedor de identidad existente de IAM (AWS CLI)

- Ejecute este comando: `aws iam tag-saml-provider`

Para enumerar etiquetas del proveedor de identidad de IAM existente (AWS CLI)

- Ejecute este comando: `aws iam list-saml-provider-tags`

Para quitar etiquetas de un proveedor de identidad de IAM existente (AWS CLI)

- Ejecute este comando: `aws iam untag-saml-provider`

Para eliminar un proveedor de identidad de IAM SAML (AWS CLI)

1. (Opcional) Para mostrar información de todos los proveedores. como el ARN o la fecha de creación y vencimiento, ejecute el siguiente comando:
   - `aws iam list-saml-providers`
2. (Opcional) Para obtener información sobre un determinado proveedor, como ARN, fecha de creación y vencimiento, ejecute el siguiente comando:
   - `aws iam get-saml-provider`
3. Para eliminar un proveedor de identidad de IAM, ejecute este comando:

- `aws iam delete-saml-provider`

Creación y administración de un proveedor de identidad SAML de IAM (API de AWS)

Puedes utilizar la API de AWS para crear y administrar proveedores SAML.

Antes de poder crear un proveedor de identidades de IAM, necesita el documento de metadatos de SAML que se obtiene del proveedor de identidades. Este documento incluye el nombre del emisor, la información del vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) que se reciben del proveedor de identidades. Para generar el documento de metadatos, utilice el software de administración de identidades que su organización utiliza como proveedor de identidad (IdP). Para obtener instrucciones sobre cómo configurar muchos de los proveedores de identidades (IdP) que funcionan con AWS y sobre cómo generar el documento de metadatos de SAML, consulte Integración de proveedores de soluciones SAML externos con AWS.

> **Important**
>
> El archivo de metadatos debe estar codificado en formato UTF-8 sin una marca de orden de bytes (BOM). Además, el certificado X.509, que se incluye como parte del documento de metadatos de SAML debe utilizar un tamaño de clave de al menos 1024 bits. Si el tamaño de clave es menor, se produce un error en la creación de proveedor de identidad (IdP) y aparece el mensaje "Unable to parse metadata". Para eliminar la BOM, puede codificar el archivo en UTF-8 con una herramienta de edición de texto, como Notepad++.

Para crear un proveedor de identidad de IAM y cargar un documento de metadatos (API de AWS)

- Llame a esta operación: `CreateSAMLProvider`

Para cargar un nuevo documento de metadatos para un proveedor de identidad de IAM (API de AWS)

- Llame a esta operación: `UpdateSAMLProvider`

Para etiquetar un proveedor de identidad de IAM existente (API de AWS)

- Llame a esta operación: `TagSAMLProvider`

Para enumerar etiquetas de un proveedor de identidad de IAM existente (API de AWS)

- Llame a esta operación: `ListSAMLProviderTags`

Para quitar etiquetas de un proveedor de identidad de IAM existente (API de AWS)

- Llame a esta operación: `UntagSAMLProvider`

Para eliminar un proveedor de identidad de IAM (API de AWS)

1. (Opcional) Para mostrar información de todos los proveedores de identidad (IdP), como ARN, fecha de creación y vencimiento, llame a la siguiente operación:
   - `ListSAMLProviders`
2. (Opcional) Para obtener información sobre un determinado proveedor, como ARN, fecha de creación y vencimiento, llame a la siguiente operación:
   - `GetSAMLProvider`
3. Para eliminar un IdP, llame a la siguiente operación:
   - `DeleteSAMLProvider`

Configuración su SAML 2.0 IdP con una relación de confianza para usuario autenticado y agregando reclamos

Cuando crea un proveedor de identidad de IAM SAML y un rol para el acceso con SAML, está informando a AWS sobre el proveedor de identidad (IdP) externo y lo que los usuarios pueden hacer. Su siguiente paso consistirá en informar al proveedor de identidades sobre AWS como proveedor de servicios. Esto se denomina añadir una relación de confianza para usuario autenticado entre su proveedor de identidades y AWS. El proceso exacto para añadir una relación de confianza entre partes depende del IdP que se use. Para obtener más información, consulte la documentación del software de administración de identidades.

Muchos proveedores de identidades permiten especificar una URL en la que el proveedor de identidades pueda leer un documento XML que contiene información sobre la parte que confía y certificados. Para AWS, use `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` o `https://signin.aws.amazon.com/static/saml-metadata.xml`. Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS.

Si no puede especificar directamente una URL, descargue el documento XML de la URL anterior e impórtelo a su software de proveedor de identidades.

También necesita crear reglas de notificación adecuadas en su proveedor de identidades que indiquen que AWS es la parte que confía. Cuando el proveedor de identidades envía una respuesta SAML al punto de enlace de AWS, que contiene una aserción SAML con una o varias notificaciones. Una notificación es información sobre el usuario y sus grupos. Una regla de notificación asigna dicha información a atributos SAML. Esto le permite asegurarse de que las respuestas de autenticación SAML de su proveedor de identidades contengan los atributos necesarios que AWS utiliza en las políticas de IAM para comprobar los permisos de los usuarios federados. Para obtener más información, consulte los temas siguientes:

- Información general acerca del rol que permite el acceso federado SAML a los recursos de AWS. En este tema se trata el uso de claves específicas de SAML en políticas de IAM y cómo utilizarlas para restringir los permisos para usuarios federados de SAML.
- Configure aserciones SAML para la respuesta de autenticación. En este tema se explica cómo configurar las notificaciones SAML que contienen información sobre el usuario. Las notificaciones están empaquetadas en una aserción SAML y se incluyen en la respuesta de SAML que se envía a AWS. Debe asegurarse de que la información que las políticas de AWS necesitan esté incluida en la aserción SAML en un formato que AWS pueda reconocer y utilizar.
- Integración de proveedores de soluciones SAML externos con AWS. En este tema se proporcionan enlaces a documentación provista por organizaciones externas sobre cómo integrar las soluciones de identidad con AWS.

> **Note**
>
> Para mejorar la resiliencia de la federación, le recomendamos que configure su IdP y su federación de AWS para que admitan varios puntos de conexión de inicio de sesión

de SAML. Para obtener más información, consulte el artículo del blog sobre seguridad de AWS, How to use regional SAML endpoints for failover.

Integración de proveedores de soluciones SAML externos con AWS

> **Note**
>
> Se recomienda exigir a los usuarios humanos que utilicen credenciales temporales cuando accedan a AWS. ¿Ha considerado la posibilidad de usar AWS IAM Identity Center? Puede usar IAM Identity Center para administrar de forma centralizada el acceso a múltiples Cuentas de AWS y proporcionar a los usuarios un acceso protegido por MFA y de inicio de sesión único a todas sus cuentas asignadas desde un solo lugar. Con IAM Identity Center, puede crear y administrar identidades de usuario en IAM Identity Center o conectarse fácilmente a su proveedor de identidades existente compatible con SAML 2.0. Para obtener más información, consulte ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center.

Los siguientes enlaces le ayudarán a configurar las soluciones de proveedores de identidad (IdP) SAML 2.0 externos para que funcionen con la federación de AWS.

> **Tip**
>
> Los ingenieros de soporte de AWS pueden ayudar a los clientes que tienen Planes de soporte Business y Enterprise con algunas tareas de integración que implican software de terceros. Para ver una lista actual de las plataformas y aplicaciones compatibles, consulte ¿Qué software de terceros se admite? en la página Preguntas frecuentes de AWS Support.

Solución	Más información
Auth0	Integrate with Amazon Web Services (Integración con Amazon Web Services): esta página del sitio web de documentación de Auth0 contiene enlaces a los recursos en los que se describe cómo configurar el inicio de sesión único (SSO) con la AWS Management Console e incluye

Solución	Más información
	un ejemplo de JavaScript. Puede configurar Auth0 para que pase las etiquetas de sesión. Para obtener más información, consulte Auth0 Announces Partnership con AWS para etiquetas de sesión de IAM.
Microsoft Entra	Tutorial: Integración del inicio de sesión único (SSO) de Microsoft Entra con AWS Single-Account Access: este tutorial del sitio web de Microsoft describe cómo configura r Microsoft Entra (conocido anteriormente como Azure AD) como un proveedor de identidades (IdP) mediante la federación SAML.
Centrify	Configure Centrify and Use SAML for SSO to AWS AWS En esta página del sitio web de Centrify se explica cómo configurar Centrify para utilizar el inicio de sesión único (SSO) basado en SAML en .
CyberArk	Configure CyberArk para proporcionar acceso a Amazon Web Services (AWS) a los usuarios que inician sesión mediante el inicio de sesión único (SSO) de SAML desde el portal de usuarios de CyberArk.
ForgeRock	ForgeRock Identity Platform se integra con AWS. Puede configurar ForgeRock para que pase las etiquetas de sesión. Para obtener más información, consulte el tema sobre control de acceso basado en atributos para Amazon Web Services.
Workspace de Google	Amazon Web Services cloud application - En este artículo del sitio Google Workspace Admin Help se describe cómo configurar Google Workspace como IdP SAML 2.0 con AWS como el proveedor de servicio.

Solución	Más información
IBM	Puede configurar IBM para que pase las etiquetas de sesión. Para obtener más información, consulte el tema IBM Cloud Identity IDaaS como uno de los primeros en admitir etiquetas de sesión de AWS.
JumpCloud	Concesión de acceso mediante roles de IAM para inicio de sesión único (SSO) con Amazon AWS: en este artículo del sitio web de JumpCloud se describe cómo configurar y habilitar el inicio de sesión único (SSO) en función de los roles de IAM para AWS.
Matrix42	Guía de introducción a MyWorkspace: en esta guía se describe cómo integrar los servicios de identidad de AWS con Matrix42 MyWorkspace.
Microsoft Active Directory Federation Services (AD FS)	Field Notes: Integrating Active Directory Federation Service with AWS IAM Identity Center (Notas de campo: integración de Active Directory Federation Service con): esta publicación del blog de arquitectura de AWS explica el flujo de autenticación entre AD FS y AWS IAM Identity Center (IAM Identity Center). IAM Identity Center admite la federación de identidades con SAML 2.0, lo que permite la integración con las soluciones de AD FS. Los usuarios pueden iniciar sesión en el portal de IAM Identity Center con sus credenciales corporativas, lo que reduce la sobrecarga administrativa de mantener credenciales separadas en IAM Identity Center. Puede configurar AD FS para que pase las etiquetas de sesión. Para obtener más información, consulte el tema sobre uso del control de acceso basado en atributos con AD FS para simplificar la administración de permisos de IAM.

Solución	Más información
miniOrange	SSO para AWS - En esta página del sitio web de miniOrange se describe cómo establecer un acceso seguro a AWS para empresas y tener control completo sobre el acceso de las aplicaciones de AWS.
Okta	Integrar la interface de línea de comandos de Amazon Web Services utilizando Okta - En esta página del sitio de soporte de Okta puede consultar cómo configurar Okta para utilizarlo con AWS. Puede configurar Okta para que pase las etiquetas de sesión. Para obtener más informaci ón, consulte Okta y AWS se unen para simplificar el acceso mediante etiquetas de sesión.
Okta	Federación de cuentas de AWS: esta sección del sitio web de Okta describe cómo configurar y habilitar Centro de identidades de IAM para AWS.
OneLogin	En OneLogin Knowledgebase, busque **SAML AWS** para obtener una lista de artículos que expliquen cómo configurar la funcionalidad de IAM Identity Center entre OneLogin y AWS en situaciones de una sola función y de varias funciones. Puede configurar OneLogin para que pase las etiquetas de sesión. Para obtener más informaci ón, consulte OneLogin y etiquetas de sesión: control de acceso basado en atributos para recursos de AWS.

Solución	Más información
Ping Identity	PingFederate AWS Connector: consulte sobre PingFederate AWS Connector, una plantilla de conexión rápida para configurar fácilmente un inicio de sesión único (SSO) y una conexión de aprovisionamiento. Lea la documentación y descargue la versión de PingFederate AWS Connector más reciente para las integraciones con AWS. Puede configurar Ping Identity para que pase las etiquetas de sesión. Para obtener más información, consulte Announcing Ping Identity Support for Attribute-Based Access Control in AWS (Anuncio de compatibilidad de Ping Identity para el control de acceso basado en atributos en AWS).
RadiantLogic	Radiant Logic Technology Partners - RadiantOne Federated Identity Service de Radiant Logic se integra con AWS para proporcionar un centro de identidades para el inicio de sesión único (SSO) basado en SAML.
RSA	La Guía de implementación de Amazon Web Services y RSA Ready proporciona orientación para la integración de AWS y RSA. Para obtener más información sobre la configuración de SAML, consulte la Guía de implementación de Amazon Web Services - Configuración de SSO de My Page de SAML - RSA Ready.
Salesforce.com	How to configure SSO from Salesforce to AWS - En este artículo sobre procedimientos del sitio para desarrolladores de Salesforce.com se describe cómo configurar un proveedor de identidades (IdP) en Salesforce y configurar AWS como proveedor de servicio.
SecureAuth	AWS - SecureAuth SAML SSO - En este artículo del sitio web de SecureAuth se describe cómo configurar la integración SAML con AWS para un dispositivo SecureAuth.

Solución	Más información
Shibboleth	How to Use Shibboleth for SSO to the AWS Management Console - En esta entrada del blog de seguridad de AWS se ofrece un tutorial que explica paso a paso cómo configurar Shibboleth como proveedor de identidades para AWS. Puede configurar Shibboleth para que pase las etiquetas de sesión.

Para obtener más información, consulte la página Socios de IAM del sitio web de AWS.

Configure aserciones SAML para la respuesta de autenticación

En su organización, después de verificar la identidad de un usuario, el proveedor de identidades (IdP) externo envía una respuesta de autenticación al punto de conexión SAML de AWS en `https://region-code.signin.aws.amazon.com/saml`. Para obtener una lista de los posibles reemplazos de *`region-code`*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS. Esta respuesta es una solicitud POST que contiene un token de SAML que cumple el estándar HTTP POST Binding for SAML 2.0 y que incluye los siguientes elementos o notificaciones. Estas notificaciones se configuran en su proveedor de identidades compatible con SAML. Consulte la documentación de su proveedor de identidad para obtener instrucciones sobre cómo ingresar esos reclamos.

Cuando el proveedor de identidades envía la respuesta que contiene las notificaciones a AWS, muchas de las notificaciones entrantes se mapean a claves de contexto de AWS. Estas claves de contexto pueden comprobarse en las políticas de IAM utilizando el elemento `Condition`. Encontrará una lista de los mapeos disponibles en la sección Mapeo de atributos SAML con claves de contexto de una política de confianza de AWS.

`Subject` y **`NameID`**

En el fragmento siguiente se muestra un ejemplo. Solo tiene que cambiar sus propios valores por los valores marcados. Debe haber exactamente un elemento `SubjectConfirmation` con un elemento `SubjectConfirmationData` que contenga tanto el atributo `NotOnOrAfter` como un atributo `Recipient`. Estos atributos incluyen un valor que debe coincidir con el punto de conexión de AWS `https://region-code.signin.aws.amazon.com/saml`. Para obtener una lista de los posibles valores de *`region-code`*, consulte la columna Region (Región) en Puntos de conexión de inicio de

sesión de AWS. Para el valor AWS, también puede utilizar `https://signin.aws.amazon.com/static/saml`, como se muestra en el ejemplo siguiente.

El valor de los elementos `NameID` puede ser “persistent” o “transient”, o bien el URI de formato completo proporcionado por la solución del proveedor de identidades. El valor “persistent” indica que el valor de `NameID` es el mismo para un usuario entre sesiones. Si el valor es “transient”, el usuario tendrá un valor de `NameID` diferente para cada sesión. Las interacciones mediante inicio de sesión único permiten los siguientes tipos de identificadores:

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:entity`

```
<Subject>
  <NameID Format="urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent">_cbb88bf52c2510eabe00c1642d4643f41430fe25e3</NameID>
  <SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
    <SubjectConfirmationData NotOnOrAfter="2013-11-05T02:06:42.876Z"
 Recipient="https://signin.aws.amazon.com/saml"/>
  </SubjectConfirmation>
</Subject>
```

Important

La clave de contexto `saml:aud` proviene del atributo recipient (destinatario) de SAML, ya que es el equivalente de SAML del campo de público de OIDC; por ejemplo, `accounts.google.com:aud`.

Atributo **`PrincipalTag`** de SAML

(Opcional) Puede utilizar un elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/PrincipalTag:{TagKey}`. Este elemento le permite pasar atributos como etiquetas de sesión en la aserción SAML. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

Para pasar atributos como etiquetas de sesión, incluya el elemento `AttributeValue` que especifica el valor de la etiqueta. Por ejemplo, para pasar los pares clave-valor de etiquetas `Project` = `Marketing` y `CostCenter` = `12345`, utilice el siguiente atributo. Incluya un elemento `Attribute` separado para cada etiqueta.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Marketing</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
```

Para establecer las etiquetas anteriores como transitivas, incluya otro elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Este es un atributo opcional multivalor que establece las etiquetas de sesión como transitivas. Las etiquetas transitivas persisten cuando se utiliza la sesión de SAML para asumir otro rol en AWS. Esto se conoce como encadenamiento de roles. Por ejemplo, para establecer tanto las etiquetas `CostCenter` como las `Principal` como transitivas, utilice el siguiente atributo para especificar las claves.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>CostCenter</AttributeValue>
</Attribute>
```

Atributo **`Role`** de SAML

Puede utilizar un elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/Role`. Este elemento contiene uno o más elementos `AttributeValue` que indican el proveedor de identidad de IAM y el rol que el IdP asigna al usuario. El rol de IAM y el proveedor de identidades de IAM se especifican como un par de ARN delimitados con comas en el mismo formato que los parámetros `RoleArn` y `PrincipalArn` que se transfieren a AssumeRoleWithSAML. Este elemento debe contener al menos un par de proveedores de roles

(elemento `AttributeValue`) y puede contener varios pares. Si el elemento contiene varios pares, se le pide al usuario que seleccione qué rol quiere asumir cuando utilice WebSSO para iniciar sesión en la AWS Management Console.

> **Important**
>
> El valor del atributo `Name` de la etiqueta `Attribute` distingue entre mayúsculas y minúsculas. Debe establecerse en `https://aws.amazon.com/SAML/Attributes/Role` con precisión.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/Role">
  <AttributeValue>arn:aws:iam::account-number:role/role-name1,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name2,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
  <AttributeValue>arn:aws:iam::account-number:role/role-name3,arn:aws:iam::account-
number:saml-provider/provider-name</AttributeValue>
</Attribute>
```

Atributo **`RoleSessionName`** de SAML

Puede utilizar un elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. Este elemento contiene un elemento `AttributeValue` que proporciona un identificador para las credenciales temporales que se generan cuando se asume el rol. Puede utilizar esto para asociar las credenciales temporales con el usuario que está utilizando la aplicación. Este elemento se utiliza para mostrar información del usuario en la AWS Management Console. El valor del elemento `AttributeValue` debe tener entre 2 y 64 caracteres, solo puede contener caracteres alfanuméricos, guiones bajos y los siguientes caracteres: . , + = @ - (guion). No puede contener espacios. El valor suele ser un ID de usuario (`johndoe`) o una dirección de correo electrónico (`johndoe@example.com`). No debe ser un valor que contenga un espacio, como el nombre de visualización de un usuario (`John Doe`).

> **Important**
>
> El valor del atributo `Name` de la etiqueta `Attribute` distingue entre mayúsculas y minúsculas. Debe establecerse en `https://aws.amazon.com/SAML/Attributes/RoleSessionName` con precisión.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/RoleSessionName">
  <AttributeValue>user-id-name</AttributeValue>
</Attribute>
```

Atributo **`SessionDuration`** de SAML

(Opcional) Puede utilizar un elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/SessionDuration"`. Este elemento contiene un elemento `AttributeValue` que especifica cuánto tiempo puede obtener acceso el usuario a la AWS Management Console antes de tener que solicitar credenciales temporales nuevas. El valor es un número entero que representa el número de segundos para la sesión. Este valor puede oscilar entre 900 segundos (15 minutos) y 43 200 segundos (12 horas). Si este atributo no está presente, las credenciales serán válidas durante una hora (el valor predeterminado del parámetro `DurationSeconds` de la API `AssumeRoleWithSAML`).

Para utilizar este atributo, debe configurar el proveedor SAML para que proporcione un acceso de inicio de sesión único a la AWS Management Console a través del punto de enlace web de inicio de sesión de la consola en `https://region-code.signin.aws.amazon.com/saml`. Para obtener una lista de los posibles valores de *`region-code`*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS. Opcionalmente, puede utilizar la siguiente URL: `https://signin.aws.amazon.com/static/saml`. Tenga en cuenta que este atributo amplía las sesiones únicamente en la AWS Management Console. No puede ampliar la duración de otras credenciales. Sin embargo, si está presente en una llamada a la API `AssumeRoleWithSAML`, se puede utilizar para acortar la duración de la sesión. La duración predeterminada de las credenciales devueltas por la llamada es de 60 minutos.

Además, tenga en cuenta que si se ha definido también un atributo `SessionNotOnOrAfter`, el valor inferior de los dos atributos, `SessionDuration` o `SessionNotOnOrAfter`, establecerá la duración máxima de la sesión de la consola.

Si habilita sesiones de consola con una duración ampliada, aumenta el riesgo de que las credenciales se filtren. Para mitigar este riesgo, puede desactivar inmediatamente las sesiones de consola activas de cualquier rol si elige Revoke Sessions en la página Role Summary de la consola de IAM. Para obtener más información, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

> **Important**
>
> El valor del atributo `Name` de la etiqueta `Attribute` distingue entre mayúsculas y minúsculas. Debe establecerse en `https://aws.amazon.com/SAML/Attributes/SessionDuration` con precisión.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SessionDuration">
  <AttributeValue>1800</AttributeValue>
</Attribute>
```

Atributo **`SourceIdentity`** de SAML

(Opcional) Puede utilizar un elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Este elemento contiene un `AttributeValue` elemento que proporciona un identificador para la persona o aplicación que utiliza un rol de IAM. El valor de la identidad de origen persiste cuando se utiliza la sesión de SAML para asumir otro rol en AWS conocido como Encadenamiento de roles. El valor de la identidad de origen está presente en la solicitud para cada acción realizada durante la sesión de rol. El valor que se establece no se puede cambiar durante la sesión de rol. A continuación, los administradores pueden utilizar registros de AWS CloudTrail para monitorear y auditar la información de identidad de origen para determinar quién realizó acciones con roles compartidos.

El valor del elemento `AttributeValue` debe tener entre 2 y 64 caracteres, solo puede contener caracteres alfanuméricos, guiones bajos y los siguientes caracteres: . , + = @ - (guion). No puede contener espacios. El valor suele ser un atributo asociado con el usuario, como un ID de usuario (`johndoe`) o una dirección de correo electrónico (`johndoe@example.com`). No debe ser un valor que contenga un espacio, como el nombre de visualización de un usuario (`John Doe`). Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

> **Important**
>
> Si la aserción SAML está configurada para utilizar el atributo `SourceIdentity`, la política de confianza también debe incluir la acción `sts:SetSourceIdentity`, de otro modo, la operación de rol asumido fallará. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

Para pasar un atributo de identidad de origen, incluya el elemento `AttributeValue` que especifica el valor de la identidad de origen. Por ejemplo, para pasar la identidad de origen `DiegoRamirez` utilice el atributo siguiente.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</AttributeValue>
```

Mapeo de atributos SAML con claves de contexto de una política de confianza de AWS

En las tablas de esta sección se enumeran los atributos SAML utilizados con más frecuencia y se muestra su correspondencia con las claves de contexto de condición de una política de confianza de AWS. Puede utilizar estas claves para controlar el acceso a un rol. Para ello, compare las claves con los valores que se incluyen en las aserciones que acompañan a una solicitud de acceso SAML.

> **Important**
>
> Estas claves solo están disponibles en las políticas de confianza de IAM (políticas que determinan quién puede asumir un rol) y no se pueden aplicar a políticas de permisos.

En la tabla de atributos eduPerson y eduOrg, los valores se indican como cadenas o como listas de cadenas. En el caso de los valores de cadenas, puede probar estos valores en las políticas de confianza de IAM utilizando las condiciones `StringEquals` o `StringLike`. En cuanto a los valores que contienen una lista de cadenas, puede utilizar los `ForAnyValue`operadores de definición de políticas`ForAllValues` y para probar los valores de las políticas de confianza.

> **Note**
>
> Debe incluir únicamente una notificación por clave de contexto de AWS. Si incluye más de una, solo se asignará una notificación.

Atributos eduPerson y eduOrg

Atributo eduPerson o eduOrg (clave **`Name`**)	Mapas a esta AWS clave de contexto AWS (clave **`FriendlyName`**)	Tipo
`urn:oid:1.3.6.1.4.1.5923.1.1.1.1`	`eduPerson Affiliation`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.1.1.2`	`eduPersonNickname`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.1.1.3`	`eduPersonOrgDN`	Cadena
`urn:oid:1.3.6.1.4.1.5923.1.1.1.4`	`eduPerson OrgUnitDN`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.1.1.5`	`eduPerson PrimaryAf filiation`	Cadena
`urn:oid:1.3.6.1.4.1.5923.1.1.1.6`	`eduPerson PrincipalName`	Cadena
`urn:oid:1.3.6.1.4.1.5923.1.1.1.7`	`eduPerson Entitlement`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.1.1.8`	`eduPerson PrimaryOrgUnitDN`	Cadena
`urn:oid:1.3.6.1.4.1.5923.1.1.1.9`	`eduPerson ScopedAff iliation`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.10`	`eduPerson TargetedID`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1. 1.1.11`	`eduPerson Assurance`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.2.1.2`	`eduOrgHomePageURI`	Lista de cadenas

Atributo eduPerson o eduOrg (clave **`Name`**)	Mapas a esta AWS clave de contexto AWS (clave **`FriendlyName`**)	Tipo
`urn:oid:1.3.6.1.4.1.5923.1.2.1.3`	`eduOrgIdentityAuthNPolicyURI`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.2.1.4`	`eduOrgLegalName`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.2.1.5`	`eduOrgSuperiorURI`	Lista de cadenas
`urn:oid:1.3.6.1.4.1.5923.1.2.1.6`	`eduOrgWhitePagesURI`	Lista de cadenas
`urn:oid:2.5.4.3`	`cn`	Lista de cadenas

Atributos de Active Directory

Atributo de AD	Se asigna con esta clave de contexto AWS	Tipo
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name`	`name`	Cadena
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	Cadena
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname`	`givenName`	Cadena
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname`	`surname`	Cadena
`http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`	`mail`	Cadena

Atributo de AD	Se asigna con esta clave de contexto AWS	Tipo
`http://schemas.microsoft.com/ws/2008/06/identity/claims/primarygroupsid`	`uid`	Cadena

Atributos X.500

Atributo X.500	Se asigna con esta clave de contexto AWS	Tipo
`2.5.4.3`	`commonName`	Cadena
`2.5.4.4`	`surname`	Cadena
`2.4.5.42`	`givenName`	Cadena
`2.5.4.45`	`x500UniqueIdentifier`	Cadena
`0.9.2342.19200300100.1.1`	`uid`	Cadena
`0.9.2342.19200300100.1.3`	`mail`	Cadena
`0.9.2342.19200300.100.1.45`	`organizationStatus`	Cadena

Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0

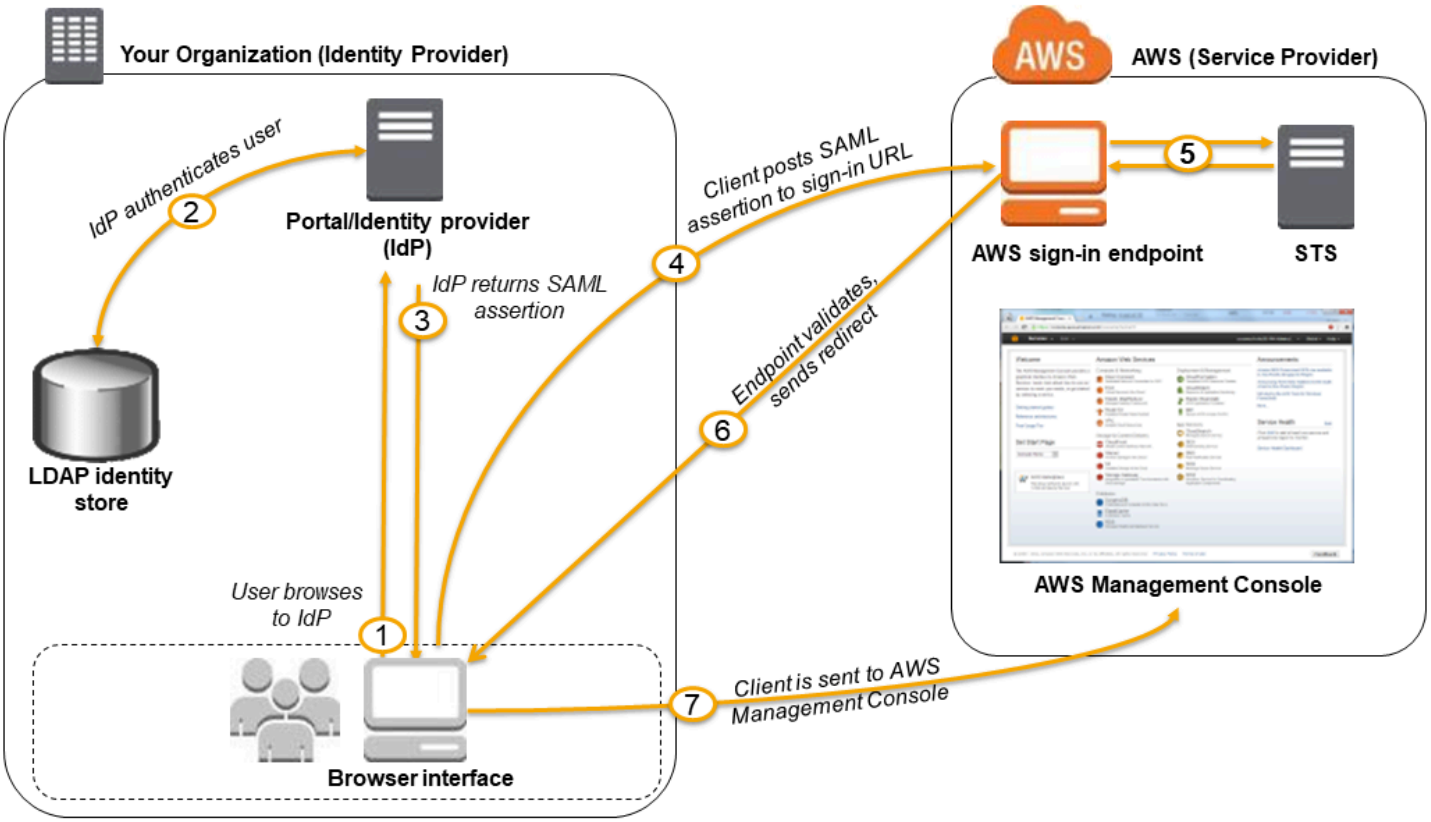
Puede utilizar un rol para configurar un proveedor de identidad compatible con SAML 2.0 y AWS de modo que se permita a los usuarios federados el acceso a la AWS Management Console. El rol otorga los permisos de usuario para realizar tareas en la consola. Si desea ofrecer a los usuarios federados de SAML otras formas para acceder a AWS, consulte uno de estos temas:

- AWS CLI: Cambio a un rol de IAM (AWS CLI)
- Tools for Windows PowerShell: Para cambiar a un rol de IAM (Tools for Windows PowerShell)
- API de AWS: Cambio a un rol de IAM (API de AWS)

Información general

El siguiente diagrama ilustra el flujo para el inicio de sesión único habilitado para SAML.

> **Note**
>
> Este uso específico de SAML difiere del más general que se muestra en Federación SAML 2.0 porque este flujo de trabajo abre la AWS Management Console en nombre del usuario. Esto requiere el uso del punto de conexión de inicio de sesión de AWS en lugar de llamar directamente a la API `AssumeRoleWithSAML`. El punto de enlace llama a la API para el usuario y devuelve una URL que redirige automáticamente al navegador del usuario a la AWS Management Console.



El siguiente diagrama muestra los siguientes pasos:

1. El usuario navega al portal de su organización y selecciona la opción para ir a la AWS Management Console. En su organización, el portal suele ser una función del IdP que gestiona el intercambio de confianza entre su organización y AWS. Por ejemplo, en Active Directory

Federation Services, la dirección URL del portal es: `https://ADFSServiceName/adfs/ls/IdpInitiatedSignOn.aspx`

2. El portal verifica la identidad del usuario de la organización.
3. El portal genera una respuesta de autenticación SAML que incluye aserciones que identifican al usuario e incluyen atributos sobre el usuario. También puede configurar su proveedor de identidad para incluir un atributo de la aserción de SAML llamado `SessionDuration` que especifica durante cuánto tiempo es válida la sesión de la consola. También puede configurar el IdP para que pase atributos como las etiquetas de sesión. El portal envía esta respuesta al navegador del cliente.
4. Se redirige al navegador del cliente al punto de enlace de inicio de sesión único de AWS y publica esta aserción de SAML.
5. El punto de enlace solicita credenciales de seguridad temporales en nombre del usuario y crea una dirección URL de inicio de sesión de la consola que utiliza dichas credenciales.
6. AWS envía la dirección URL de inicio de sesión al cliente en forma de redireccionamiento.
7. El navegador del cliente se redirige hacia la AWS Management Console. Si la respuesta de autenticación de SAML incluye atributos asociados a varios roles de IAM, el usuario primero deberá seleccionar el rol para acceder a la consola.

Desde el punto de vista del usuario, el proceso se realiza de forma transparente: el usuario comienza en el portal interno de la organización y acaba en la AWS Management Console, sin haber tenido que escribir las credenciales de AWS.

Consulte las secciones siguientes para obtener información general acerca de cómo configurar este comportamiento junto con enlaces a pasos detallados.

Configurar la red como proveedor SAML para AWS

En la red de su organización, configure el almacén de identidades (como Windows Active Directory) de modo que funcione con un IdP basado en SAML, como Windows Active Directory Federation Services, Shibboleth, etc. Utilice el IdP para generar un documento de metadatos que describa la organización como IdP e incluya claves de autenticación. También puede configurar el portal de su organización para dirigir las solicitudes de los usuarios para la AWS Management Console al punto de enlace SAML de AWS para la autenticación mediante aserciones SAML. Cómo configurar el proveedor de identidad para producir el archivo metadata.xml depende del proveedor de identidad. Consulte la documentación del proveedor de identidad para obtener instrucciones o consulte

Integración de proveedores de soluciones SAML externos con AWS para encontrar enlaces a la documentación web de muchos de los proveedores SAML compatibles.

Crear un proveedor SAML en IAM

A continuación, inicie sesión en la AWS Management Console y vaya la consola de IAM. Cree un nuevo proveedor SAML, que es una entidad de IAM que contiene la información sobre el IdP de la organización. Durante este proceso, cargue el documento de metadatos generado por el software del proveedor de identidad de la organización de la sección anterior. Para obtener más información, consulte Crear un proveedor de identidades de SAML en IAM.

Configurar permisos en AWS para los usuarios federados

El siguiente paso consiste en crear un rol de IAM que establezca una relación de confianza entre IAM y el IdP de su organización. Este rol debe identificar su IdP como una entidad principal (entidad de confianza) a efectos de la federación. El rol también define qué pueden hacer los usuarios autenticados mediante el proveedor de identidad de la organización en AWS. Puede utilizar la consola de IAM para crear este rol. Al crear la política de confianza que indica quién puede asumir el rol, especifique el proveedor SAML que creó anteriormente en IAM. Especifique también uno o más atributos de SAML que debe tener un usuario para poder asumir el rol. Por ejemplo, puede especificar que solo los usuarios cuyo valor SAML `eduPersonOrgDN` es `ExampleOrg` puedan iniciar sesión. El asistente de rol añade automáticamente una condición para probar el atributo `saml:aud` para asegurarse de que el rol se asume solo para iniciar sesión en la AWS Management Console. La política de confianza del rol podría tener el siguiente aspecto:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  }]
}
```

> **Note**
>
> Los proveedores de identidad de SAML que se utilizan en una política de confianza de roles deben estar en la misma cuenta en la que se encuentra el rol.

Puede incluir puntos de conexión regionales para el atributo `saml:aud` en `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml`. Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS.

Para la política de permisos del rol, debe especificar los permisos de la misma forma que haría para un rol, usuario o grupo. Por ejemplo, si los usuarios de su organización pueden administrar instancias Amazon EC2, usted permite de forma explícita las acciones de Amazon EC2 en la política de permisos. Puede hacer esto asignando una política administrada, como la política administrada Amazon EC2 Full Access.

Para obtener más información acerca de cómo crear un rol para un proveedor de identidad SAML, consulte Creación de un rol para una federación SAML 2.0 (consola).

Finalizar la configuración y crear aserciones de SAML

Notifique a su IdP de SAML de que AWS es su proveedor de servicios mediante la instalación del archivo `saml-metadata.xml` que se encuentra en `https://region-code.signin.aws.amazon.com/static/saml-metadata.xml` o `https://signin.aws.amazon.com/static/saml-metadata.xml`. Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS.

La instalación de dicho archivo depende del proveedor de identidad. Algunos proveedores ofrecen la opción de escribir la URL, en cuyo caso el IdP obtiene e instala el archivo automáticamente. Otros requieren que descargue el archivo de la URL y, a continuación, lo proporcione como archivo local. Consulte la documentación del proveedor de identidad para obtener información detallada o consulte Integración de proveedores de soluciones SAML externos con AWS para encontrar enlaces a la documentación web de cualquiera de los proveedores SAML compatibles.

También puede configurar la información que quiere que el proveedor de identidad traspase como atributos SAML a AWS dentro de la respuesta de autenticación. La mayor parte de esta información

aparece en AWS como claves de contexto de condición que puede evaluar en sus políticas. Estas claves de condición garantizan que solo los usuarios autorizados en los contextos adecuados tengan permisos para acceder a los recursos de AWS. Puede especificar períodos de tiempo que restrinjan cuándo se puede utilizar la consola. También puede especificar el tiempo máximo (hasta 12 horas) durante el cuál los usuarios pueden acceder a la consola antes de tener que renovar sus credenciales. Para obtener información, consulte Configure aserciones SAML para la respuesta de autenticación.

Credenciales de seguridad temporales en IAM

Puede utilizar AWS Security Token Service (AWS STS) para crear credenciales de seguridad temporales que pueden controlar el acceso a sus recursos de AWS y proporcionárselas a usuarios de confianza. Las credenciales de seguridad temporales funcionan prácticamente igual que las credenciales de clave de acceso a largo plazo, con las siguientes diferencias:

- Las credenciales de seguridad temporales son a corto plazo, tal como su nombre indica. Se pueden configurar para durar entre unos cuantos minutos y varias horas. Cuando las credenciales caduquen, AWS dejará de reconocerlas o permitirá todo tipo de acceso a las solicitudes realizadas desde API que las utilicen.
- Las credenciales de seguridad temporales no se guardan con el usuario, sino que se generan de forma dinámica y se proporcionan al usuario cuando se solicitan. Cuando las credenciales de seguridad temporales caducan (o incluso antes), el usuario puede solicitar nuevas credenciales, siempre y cuando el usuario que las solicite tenga permiso para hacerlo.

Como resultado, las credenciales temporales tienen las siguientes ventajas con respecto a las credenciales a largo plazo:

- No tiene que distribuir ni incrustar credenciales de seguridad de AWS a largo plazo con una aplicación.
- Puede proporcionar acceso a sus recursos de AWS a usuarios, sin necesidad de definir una identidad de AWS para ellos. Las credenciales temporales son la base de los roles y la federación de identidades.
- Las credenciales de seguridad temporales tienen un ciclo de vida limitado, por lo que no tiene que actualizarlas ni revocarlas de forma explícita cuando ya no las necesite. Cuando las credenciales de seguridad temporales caducan, ya no se pueden volver a utilizar. Puede especificar el tiempo de validez de las credenciales, hasta un límite máximo.

Regiones de AWS STS y AWS

AWS STS genera las credenciales de seguridad temporales;. De forma predeterminada, AWS STS es un servicio global con un único punto de enlace en `https://sts.amazonaws.com`. Sin embargo, también puede optar por realizar llamadas de API de AWS STS a puntos de enlace de cualquier otra región compatible. Esto puede reducir la latencia de servidor (retraso del servidor) enviando las solicitudes a servidores de una región que está más cerca de usted. No importa de qué región vienen sus credenciales, funcionan en todo el mundo. Para obtener más información, consulte Administrar AWS STS en una Región de AWS.

Escenarios habituales en las credenciales temporales

Las credenciales temporales son útiles en escenarios en los que entren en juego las identidades federadas, la delegación, el acceso entre cuentas y los roles de IAM.

Identidad federada

Puede administrar sus identidades de usuario en un sistema externo situado fuera de AWS y conceder a los usuarios que inician sesión desde dichos sistemas acceso para realizar tareas de AWS y obtener acceso a sus recursos de AWS. IAM admite dos tipos de identidades federadas. En ambos casos, las identidades se almacenan fuera de AWS. La diferencia radica dónde reside el sistema externo, en su centro de datos o un tercero externo en la web. Para obtener más información acerca de proveedores de identidades externos, consulte Federación y proveedores de identidades.

- Federación de SAML: Puede autenticar usuarios de la red de su organización y, a continuación, dar a dichos usuarios acceso a AWS sin tener que crearles nuevas identidades de AWS ni exigirles que inicien sesión con credenciales diferentes. Este procedimiento se denomina inicio de sesión único para el acceso temporal. AWS STS es compatible con estándares abiertos como Security Assertion Markup Language (SAML) 2.0, con el que puede utilizar Microsoft AD FS para aprovechar su Microsoft Active Directory. También puede utilizar SAML 2.0 para administrar su propia solución de identidades federadas de usuarios. Para obtener más información, consulte Federación SAML 2.0.
  - Agente de federación personalizado AWS Puede utilizar el sistema de autenticación de su organización para conceder acceso a los recursos de . Si desea ver un escenario de ejemplo, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.
  - Federación con SAML 2.0 Puede utilizar el sistema de autenticación de su organización y SAML para conceder acceso a los recursos de AWS. Para obtener más información y ver un escenario de ejemplo, consulte Federación SAML 2.0.

- Federación de OpenID Connect (OIDC): puede dejar que los usuarios inicien sesión con un proveedor de identidades de terceros conocido, como Inicio de sesión con Amazon, Facebook, Google o cualquier proveedor compatible con OIDC 2.0 para su aplicación móvil o Web, no necesita crear un código de inicio de sesión personalizado ni administrar sus propias identidades de usuario. La federación de OIDC le permite tener una Cuenta de AWS más segura, ya que no tiene que distribuir credenciales de seguridad a largo plazo, como claves de acceso de usuario de IAM, con su aplicación. Para obtener más información, consulte Federación OIDC.

  La federación de OIDC AWS STS admite Inicio de sesión con Amazon, Facebook, Google o cualquier proveedor de identidad compatible con OpenID Connect (OIDC).

  > **Note**
  >
  > Para aplicaciones móviles, le recomendamos que utilice Amazon Cognito. Puede utilizar el servicio con los AWS SDK para desarrollo móvil para crear identidades únicas para usuarios y autenticarlos para proteger el acceso a los recursos de AWS. Amazon Cognito es compatible con los mismos proveedores de identidades que AWS STS y con el acceso sin autenticar (invitado), y le permite migrar datos de usuario cuando un usuario inicia sesión. Amazon Cognito también ofrece operaciones de API para sincronizar los datos del usuario de modo que se preserven cuando cambia de un dispositivo a otro. Para obtener más información, consulte Autenticación con Amplify en la documentación de Amplify.

Roles para el acceso entre cuentas

Muchas organizaciones mantienen más de una Cuenta de AWS. Con los roles y el acceso entre cuentas, puede definir identidades de usuarios en una cuenta y utilizar esas mismas identidades para obtener acceso a recursos de AWS de otras cuentas que pertenezcan a su organización. Este procedimiento se denomina delegación del acceso temporal. Para obtener más información sobre la creación de roles entre cuentas, consulte Creación de un rol para delegar permisos a un usuario de IAM. Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Roles para Amazon EC2

Si ejecuta aplicaciones en instancias de Amazon EC2 y dichas aplicaciones necesitan obtener acceso a recursos de AWS, puede proporcionar credenciales de seguridad temporales a las

instancias cuando las lanza. Dichas credenciales de seguridad temporales están disponibles para todas las aplicaciones que se ejecutan en la instancia, por lo que no es necesario almacenar credenciales a largo plazo en ella. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2.

Otros servicios de AWS

Puede utilizar credenciales de seguridad temporales para obtener acceso a la mayoría de los servicios de AWS. Para obtener una lista de los servicios que aceptan credenciales de seguridad temporales, consulte Servicios de AWS que funcionan con IAM.

Solicitud de credenciales de seguridad temporales

Para solicitar credenciales de seguridad temporales, puede utilizar las operaciones AWS Security Token Service (AWS STS) en la API de AWS. Esto incluye operaciones para crear credenciales de seguridad temporales que pueden controlar el acceso a sus recursos de AWS y proporcionárselas a usuarios de confianza. Para obtener más información acerca de AWS STS, consulte Credenciales de seguridad temporales en IAM. Para obtener más información sobre los distintos métodos que puede utilizar para solicitar credenciales de seguridad temporales asumiendo un rol, consulte Uso de roles de IAM.

Para llamar a las operaciones de la API, puede utilizar uno de los SDK de AWS. Los SDK están disponibles para una gran variedad de entornos y lenguajes de programación, tales como Java, .NET, Python, Ruby, Android e iOS. Los SDK se encargan de tareas como firmar solicitudes criptográficamente, reintentar solicitudes si fuera necesario y administrar respuestas a errores. También puede utilizar la API de consulta de AWS STS, que se describe en Referencia de API de AWS Security Token Service. Por último, dos herramientas de línea de comandos admiten los comandos de AWS STS: AWS Command Line Interface y AWS Tools for Windows PowerShell.

Las operaciones de API de AWS STS crean una nueva sesión con credenciales de seguridad temporales formadas por un par de claves de acceso y un token de sesión. El par de claves de acceso consta de un ID de clave de acceso y una clave secreta. Los usuarios (o una aplicación que el usuario ejecute) pueden utilizar estas credenciales para obtener acceso a los recursos. Puede crear una sesión de rol y pasar políticas de sesión y etiquetas de sesión con programación mediante las operaciones de la API de AWS STS. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Para obtener más información acerca de las políticas de sesión, consulte Políticas de sesión. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

> **Note**
>
> El tamaño del token de seguridad que devuelven las operaciones de la API de AWS STS no es fijo. Recomendamos encarecidamente que no realice suposiciones sobre el tamaño máximo. El tamaño típico del token es inferior a 4096 bytes, pero puede variar.

Uso de AWS STS con regiones de AWS

Puede enviar llamadas a la API de AWS STS tanto a un punto de enlace global como a uno de los puntos de enlace regionales. Si elige el punto de enlace más cercano, puede reducir la latencia y mejorar el rendimiento de las llamadas a la API. También puede elegir enviar sus llamadas a un punto de enlace regional alternativo si ya no puede comunicarse con el punto de enlace original. Si utiliza uno de los distintos SDK de AWS, utilice el método del SDK para especificar una región antes de realizar la llamada a la API. Si va realiza manualmente solicitudes de API HTTP, debe enviar usted mismo la solicitud al punto de conexión correcto. Para obtener más información, consulte la AWS STS sección de Regiones y puntos de enlace y Administrar AWS STS en una Región de AWS.

Las siguientes son operaciones de API que puede utilizar para obtener credenciales temporales para su uso en aplicaciones y entornos de AWS.

AssumeRole: delegación y federación entre cuentas a través de un agente de identidades personalizado

La operación de API `AssumeRole` es útil para permitir a los usuarios existentes de IAM el acceso a recursos de AWS a los que todavía no tienen acceso. Por ejemplo, el usuario puede necesitar acceso a los recursos de otra Cuenta de AWS. También es útil para obtener temporalmente acceso privilegiado por ejemplo, para proporcionar autenticación multifactor (MFA). Debe llamar a esta API con las credenciales de usuario activas. Para saber quién puede llamar a esta operación, consulte Comparación de las operaciones de la API de AWS STS. Para obtener más información, consulte Creación de un rol para delegar permisos a un usuario de IAM y Configuración del acceso a una API protegido por MFA.

Esta llamada debe realizarse con credenciales de seguridad de AWS válidas. Al realizar esta llamada, transfiere la siguiente información:

- El Nombre de recurso de Amazon (ARN) del rol que la aplicación debe asumir.
- (Opcional) La duración, que especifica cuánto tiempo son válidas las credenciales de seguridad temporales. Use el parámetro `DurationSeconds` para especificar la duración de la sesión de

rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol. Si no pasa este parámetro, las credenciales temporales caducan en una hora. El parámetro `DurationSeconds` de esta API es distinto del parámetro HTTP `SessionDuration` que se utiliza para especificar la duración de una sesión de consola. Utilice el parámetro HTTP `SessionDuration` en la solicitud del punto de conexión de federación para un token de inicio de sesión en la consola. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

- Nombre de sesión de rol. Utilice este valor de cadena para identificar la sesión cuando un rol es utilizado por diferentes entidades. Por motivos de seguridad, los administradores pueden ver este campo en registros de AWS CloudTrail para ayudar a identificar quién realizó una acción en AWS. Es posible que el administrador requiera que especifique su nombre de usuario de IAM como nombre de sesión cuando asuma el rol. Para obtener más información, consulte `sts:RoleSessionName`.
- (Opcional) Identidad de origen. Puede exigir a los usuarios que especifiquen una identidad de origen cuando asuman un rol. Una vez establecida la identidad de origen, el valor no se puede cambiar. Está presente en la solicitud de todas las acciones que se realizan durante la sesión de rol. El valor de identidad de origen persiste en sesiones de rol encadenado. Puede utilizar la información de identidad de origen en registros de AWS CloudTrail para determinar quién realizó acciones con un rol. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.
- (Opcional) Políticas de sesión administradas o insertadas. Estas políticas limitan los permisos de la política basada en identidades del rol que están asignados a la sesión de rol. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades del rol que se asume. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.
- (Opcional) Etiquetas de sesión. Puede asumir un rol y, a continuación, utilizar las credenciales temporales para realizar una solicitud. Cuando lo haga, las etiquetas principales de la sesión incluyen las etiquetas del rol y las etiquetas de sesión pasadas. Si realiza esta llamada con credenciales temporales, la nueva sesión también hereda las etiquetas de sesión transitivas de la sesión de llamada. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

- (Opcional) Información sobre MFA. Si se configura para utilizar Multi-Factor Authentication (MFA), incluya el identificador de un dispositivo MFA y el código de un solo uso proporcionado por dicho dispositivo.
- (Opcional) Un valor `ExternalId` que se puede utilizar al delegar el acceso a su cuenta a un tercero. Este valor permite garantizar que solo el tercero especificado pueda obtener acceso al rol. Para obtener más información, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.

En el siguiente ejemplo se muestra una solicitud y respuesta de muestra que utiliza `AssumeRole`. Esta solicitud de ejemplo asume el rol `demo` durante la duración especificada con la política de sesión incluida, las etiquetas de sesión, el ID externo y la identidad de origen. La sesión resultante se denomina `John-session`.

Example Ejemplo de solicitud

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=John-session
&RoleArn=arn:aws::iam::123456789012:role/demo
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%20%22Stmt1%22%2C%22Effect%22%3A%20%22Allow%22%2C%22Action%22%3A%20%22s3%3A*%22%2C
%22Resource%22%3A%20%22*%22%7D%5D%7D
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&ExternalId=123ABC
&SourceIdentity=DevUser123
&AUTHPARAMS
```

El valor de política que se muestra en el ejemplo anterior es la versión codificada como URL de la siguiente política:

```
{"Version":"2012-10-17","Statement":
[{"Sid":"Stmt1","Effect":"Allow","Action":"s3:*","Resource":"*"}]}
```

El parámetro `AUTHPARAMS` en el ejemplo es un marcador de posición para su firma. Una firma es la información de autenticación que debe incluir en las solicitudes de la API HTTP de AWS.

Recomendamos utilizar los SDK de AWS para crear solicitudes de API; un beneficio de esto es que los SDK gestionan la firma de solicitudes en su nombre. Si debe crear y firmar manualmente solicitudes de API, diríjase a Firma de solicitudes de AWS con Signature Version 4 en la Referencia general de Amazon Web Services para averiguar cómo firmar una solicitud.

Además de las credenciales de seguridad temporales, la respuesta incluye el Nombre de recurso de Amazon (ARN) para el usuario federado y el plazo de vencimiento de las credenciales.

Example Ejemplo de respuesta

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<AssumeRoleResult>
<SourceIdentity>DevUser123</SourceIdentity>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
  </SessionToken>
  <SecretAccessKey>
   wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-07-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
<AssumedRoleUser>
  <Arn>arn:aws:sts::123456789012:assumed-role/demo/John</Arn>
  <AssumedRoleId>ARO123EXAMPLE123:John</AssumedRoleId>
</AssumedRoleUser>
<PackedPolicySize>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Una conversión de AWS comprime las políticas de sesión pasadas y las etiquetas de sesión en un formato binario empaquetado que tiene un límite separado. Su solicitud puede fallar

para este límite incluso si el texto sin formato cumple con los demás requisitos. El elemento de respuesta `PackedPolicySize` indica por porcentaje lo cerca que están las políticas y etiquetas de su solicitud al límite de tamaño superior.

AssumeRoleWithWebIdentity: federación a través de un proveedor de identidades basado en la web

La operación de API `AssumeRoleWithWebIdentity` devuelve un conjunto de credenciales de seguridad temporales para los usuarios federados autenticados a través de un proveedor de identidad público. Entre los ejemplos de proveedores de identidad públicos se incluyen Login with Amazon, Facebook, Google o cualquier proveedor de identidad compatible con OpenID Connect (OIDC). Esta operación es útil para crear aplicaciones móviles o aplicaciones web basadas en el cliente que requieren acceso a AWS. El uso de esta operación significa que los usuarios no tienen sus propias identidades de AWS o IAM. Para obtener más información, consulte Federación OIDC.

En lugar de llamar directamente a `AssumeRoleWithWebIdentity`, le recomendamos que utilice Amazon Cognito y las credenciales del proveedor de Amazon Cognito con los SDK de AWS para el desarrollo de aplicaciones móviles. Para obtener más información, consulte Autenticación con Amplify en la documentación de Amplify.

Si no usa Amazon Cognito, llame a la acción `AssumeRoleWithWebIdentity` de AWS STS. Se trata de una llamada sin firma, lo que significa que la aplicación no necesita tener acceso a las credenciales de seguridad de AWS para realizar la llamada. Al realizar esta llamada, transfiere la siguiente información:

- El Nombre de recurso de Amazon (ARN) del rol que la aplicación debe asumir. Si su aplicación admite varias formas de inicio de sesión para los usuarios, debe definir varios roles, uno por cada proveedor de identidad. La llamada a `AssumeRoleWithWebIdentity` debe incluir el ARN del rol que es específico para el proveedor a través del que el usuario ha iniciado sesión.
- El token que la aplicación obtiene del proveedor de identidad (IdP) después de que la aplicación autentique al usuario.
- Puede configurar su IdP para que pase atributos a su token como etiquetas de sesión.
- (Opcional) La duración, que especifica cuánto tiempo son válidas las credenciales de seguridad temporales. Use el parámetro `DurationSeconds` para especificar la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. Para obtener información sobre cómo ver el valor máximo para

el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol. Si no pasa este parámetro, las credenciales temporales caducan en una hora. El parámetro `DurationSeconds` de esta API es distinto del parámetro HTTP `SessionDuration` que se utiliza para especificar la duración de una sesión de consola. Utilice el parámetro HTTP `SessionDuration` en la solicitud del punto de conexión de federación para un token de inicio de sesión en la consola. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

- Nombre de sesión de rol. Utilice este valor de cadena para identificar la sesión cuando un rol es utilizado por diferentes entidades. Por motivos de seguridad, los administradores pueden ver este campo en registros de AWS CloudTrail para saber quién realizó una acción en AWS. Es posible que el administrador requiera que proporcione un valor específico para el nombre de la sesión cuando asuma el rol. Para obtener más información, consulte `sts:RoleSessionName`.
- (Opcional) Identidad de origen. Puede requerir que los usuarios federados especifiquen una identidad de origen cuando asuman un rol. Una vez establecida la identidad de origen, el valor no se puede cambiar. Está presente en la solicitud de todas las acciones que se realizan durante la sesión de rol. El valor de identidad de origen persiste en sesiones de rol encadenado. Puede utilizar la información de identidad de origen en registros de AWS CloudTrail para determinar quién realizó acciones con un rol. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.
- (Opcional) Políticas de sesión administradas o insertadas. Estas políticas limitan los permisos de la política basada en identidades del rol que están asignados a la sesión de rol. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades del rol que se asume. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.

> **Note**
>
> Una llamada a `AssumeRoleWithWebIdentity` no se firma (cifrada). Por lo tanto, solo debe incluir estas políticas de sesión opcionales si la solicitud se transmite a través de un intermediario de confianza. En este caso, alguien podría modificar la política para eliminar las restricciones.

Si llama a `AssumeRoleWithWebIdentity`, AWS verifica la autenticidad del token. Por ejemplo, en función del proveedor de AWS, se podría realizar una llamada al proveedor e incluir el token

que la aplicación ha transmitido. Suponiendo que el proveedor de identidad valida el token, AWS le devuelve la siguiente información:

- Un conjunto de credenciales de seguridad temporales. Estas incluyen un ID de clave de acceso, una clave de acceso secreta y un token de sesión.
- El ID de rol y el ARN del rol asumido.
- Un valor `SubjectFromWebIdentityToken` que incluye el ID de usuario único.

Cuando tiene las credenciales de seguridad temporales, puede utilizarlas para realizar llamadas a la API de AWS. Se trata del mismo proceso que para hacer una llamada a la API de AWS con credenciales de seguridad a largo plazo. La diferencia es que debe incluir el token de sesión, que permite a AWS verificar que las credenciales de seguridad temporales son válidas.

La aplicación debe almacenar en caché las credenciales. Tal y como se ha mencionado, las credenciales caducan después de una hora de forma predeterminada. Si no utiliza la operación AmazonSTSCredentialsProvider en el SDK de AWS, depende de usted y de su aplicación volver a llamar a `AssumeRoleWithWebIdentity`. Llame a esta operación para obtener un nuevo conjunto de credenciales de seguridad temporales antes de que caduquen las antiguas.

AssumeRoleWithSAML: federación a través de un proveedor de identidades empresarial compatible con SAML 2.0

La operación de API `AssumeRoleWithSAML` devuelve un conjunto de credenciales de seguridad temporales para los usuarios federados que se autentican a través del sistema de identidad existente de su organización. Los usuarios también deben utilizar SAML 2.0 (lenguaje de marcado para confirmaciones de seguridad) para transmitir la información de autenticación y autorización a AWS. Esta operación de la API es útil en organizaciones que han integrado sus sistemas de identidad (como Windows Active Directory u OpenLDAP) con software que puede producir aserciones SAML. Esta integración proporciona información sobre los permisos e identidad del usuario (como Active Directory Federation Services o Shibboleth). Para obtener más información, consulte Federación SAML 2.0.

> **Note**
>
> Una llamada a `AssumeRoleWithSAML` no se firma (cifrada). Por lo tanto, solo debe incluir estas políticas de sesión opcionales si la solicitud se transmite a través de un intermediario de confianza. En este caso, alguien podría modificar la política para eliminar las restricciones.

Se trata de una llamada sin firma, lo que significa que la aplicación no necesita tener acceso a las credenciales de seguridad de AWS para realizar la llamada. Al realizar esta llamada, transfiere la siguiente información:

- El Nombre de recurso de Amazon (ARN) del rol que la aplicación debe asumir.
- El ARN del proveedor SAML creado en IAM que describe el proveedor de identidad.
- La aserción SAML, codificada en base 64, que el proveedor de identidad SAML ha proporcionado en su respuesta de autenticación a la solicitud de inicio de sesión de la aplicación.
- Puede configurar su IdP para que pase atributos a su aserción SAML como etiquetas de sesión.
- (Opcional) La duración, que especifica cuánto tiempo son válidas las credenciales de seguridad temporales. Use el parámetro `DurationSeconds` para especificar la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol. Si no pasa este parámetro, las credenciales temporales caducan en una hora. El parámetro `DurationSeconds` de esta API es distinto del parámetro HTTP `SessionDuration` que se utiliza para especificar la duración de una sesión de consola. Utilice el parámetro HTTP `SessionDuration` en la solicitud del punto de conexión de federación para un token de inicio de sesión en la consola. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.
- (Opcional) Políticas de sesión administradas o insertadas. Estas políticas limitan los permisos de la política basada en identidades del rol que están asignados a la sesión de rol. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades del rol que se asume. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.
- Nombre de sesión de rol. Utilice este valor de cadena para identificar la sesión cuando un rol es utilizado por diferentes entidades. Por motivos de seguridad, los administradores pueden ver este campo en registros de AWS CloudTrail para saber quién realizó una acción en AWS. Es posible que el administrador requiera que proporcione un valor específico para el nombre de la sesión cuando asuma el rol. Para obtener más información, consulte `sts:RoleSessionName`.
- (Opcional) Identidad de origen. Puede requerir que los usuarios federados especifiquen una identidad de origen cuando asuman un rol. Una vez establecida la identidad de origen, el valor no se puede cambiar. Está presente en la solicitud de todas las acciones que se realizan durante la sesión de rol. El valor de identidad de origen persiste en sesiones de rol encadenado. Puede

utilizar la información de identidad de origen en registros de AWS CloudTrail para determinar quién realizó acciones con un rol. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

Si llama a `AssumeRoleWithSAML`, AWS verifica la autenticidad de la aserción SAML. Suponiendo que el proveedor de identidad valida la aserción, AWS le devuelve la siguiente información:

- Un conjunto de credenciales de seguridad temporales. Estas incluyen un ID de clave de acceso, una clave de acceso secreta y un token de sesión.
- El ID de rol y el ARN del rol asumido.
- Un valor `Audience` que incluye el valor del atributo `Recipient` del elemento `SubjectConfirmationData` de la aserción SAML.
- Un valor `Issuer` que incluye el valor del elemento `Issuer` de la aserción SAML.
- Un elemento `NameQualifier` que incluye un valor hash creado a partir del valor `Issuer`, el ID de la Cuenta de AWS y el nombre fácil de recordar del proveedor SAML. Cuando se combina con el elemento `Subject`, pueden identificar exclusivamente al usuario federado.
- Un elemento `Subject` que incluye el valor del elemento `NameID` en el elemento `Subject` de la aserción SAML.
- Un elemento `SubjectType` que indica el formato del elemento `Subject`. El valor puede ser `persistent`, `transient` o la URI completa `Format` de los elementos `Subject` y `NameID` utilizados en su aserción SAML. Para obtener información sobre el atributo `NameID` del elemento `Format`, consulte Configure aserciones SAML para la respuesta de autenticación.

Cuando tiene las credenciales de seguridad temporales, puede utilizarlas para realizar llamadas a la API de AWS. Se trata del mismo proceso que para hacer una llamada a la API de AWS con credenciales de seguridad a largo plazo. La diferencia es que debe incluir el token de sesión, que permite a AWS verificar que las credenciales de seguridad temporales son válidas.

La aplicación debe almacenar en caché las credenciales. Las credenciales caducan después de una hora de forma predeterminada. Si no utiliza la acción AmazonSTSCredentialsProvider en el SDK de AWS, depende de usted y de su aplicación volver a llamar a `AssumeRoleWithSAML`. Llame a esta operación para obtener un nuevo conjunto de credenciales de seguridad temporales antes de que caduquen las antiguas.

GetFederationToken: federación a través de un agente de identidades personalizadas

La operación de API `GetFederationToken` devuelve un conjunto de credenciales de seguridad temporales para los usuarios federados. Esta API difiere de `AssumeRole` en que el periodo de vencimiento predeterminado es bastante mayor (12 horas en lugar de una hora). Además, puede utilizar el parámetro `DurationSeconds` para especificar una duración de validez para las credenciales de seguridad temporales. Las credenciales resultantes son válidas durante el tiempo especificado, entre 900 segundos (15 minutos) y 129 600 segundos (36 horas). Un periodo de vencimiento mayor puede ayudar a reducir el número de llamadas a AWS, ya que no es necesario obtener credenciales nuevas con tanta frecuencia.

Al realizar esta solicitud, se utilizan las credenciales de un usuario específico de IAM. Los permisos para las credenciales de seguridad temporales los determinan las políticas de sesión que se transfieren cuando se llama a `GetFederationToken`. Los permisos de la sesión resultantes son la intersección de las políticas de usuario de IAM y las políticas de sesión que transfiere. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades del usuario de IAM que solicita la federación. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.

Cuando utiliza las credenciales temporales devueltas por la operación `GetFederationToken`, las etiquetas principales de la sesión incluyen las etiquetas del usuario y las etiquetas de sesión pasadas. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

La llamada `GetFederationToken` devuelve credenciales de seguridad temporales que incluyen el token de seguridad, la clave de acceso, la clave secreta y el vencimiento. Puede utilizar `GetFederationToken` si desea administrar los permisos de su organización (por ejemplo, la utilización de la aplicación de proxy para asignar permisos).

El siguiente ejemplo muestra una solicitud y respuesta de muestra que utiliza `GetFederationToken`. En este ejemplo de solicitud se federa al usuario que llama durante la duración especificada con el ARN de la póliza de sesión y las etiquetas de sesión. La sesión resultante se denomina `Jane-session`.

Example Ejemplo de solicitud

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Jane-session
```

```
&PolicyArns.member.1.arn==arn%3Aaws%3Aiam%3A%3A123456789012%3Apolicy%2FRole1policy
&DurationSeconds=1800
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&AUTHPARAMS
```

El ARN de la política que se muestra en el ejemplo anterior incluye el siguiente ARN codificado en URL:

```
arn:aws:iam::123456789012:policy/Role1policy
```

Además, tenga en cuenta que el parámetro &AUTHPARAMS del ejemplo se entiende como marcador de posición para la información de autenticación. Esta es la firma, que debe incluir con las solicitudes API de HTTP de AWS. Recomendamos utilizar los SDK de AWS para crear solicitudes de API; un beneficio de esto es que los SDK gestionan la firma de solicitudes en su nombre. Si debe crear y firmar manualmente solicitudes de API, diríjase a Firma de solicitudes de AWS con Signature Version 4 en la Referencia general de Amazon Web Services para averiguar cómo firmar una solicitud.

Además de las credenciales de seguridad temporales, la respuesta incluye el Nombre de recurso de Amazon (ARN) para el usuario federado y el plazo de vencimiento de las credenciales.

Example Ejemplo de respuesta

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetFederationTokenResult>
<Credentials>
  <SessionToken>
   AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
   LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
   QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
   9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
   +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCEXAMPLE==
  </SessionToken>
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2019-04-15T23:28:33.359Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE;</AccessKeyId>
</Credentials>
<FederatedUser>
```

```
  <Arn>arn:aws:sts::123456789012:federated-user/Jean</Arn>
  <FederatedUserId>123456789012:Jean</FederatedUserId>
</FederatedUser>
<PackedPolicySize>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Una conversión de AWS comprime las políticas de sesión pasadas y las etiquetas de sesión en un formato binario empaquetado que tiene un límite separado. Su solicitud puede fallar para este límite incluso si el texto sin formato cumple con los demás requisitos. El elemento de respuesta `PackedPolicySize` indica por porcentaje lo cerca que están las políticas y etiquetas de su solicitud al límite de tamaño superior.

AWS recomienda conceder permisos en el nivel de recursos (por ejemplo, adjuntar una política basada en recursos a un bucket de Amazon S3), puede omitir el parámetro `Policy`. No obstante, si no incluye una política para el usuario federado, las credenciales de seguridad temporales no concederán los permisos. En este caso, debe utilizar las políticas de recursos para conceder acceso a sus recursos de AWS al usuario federado.

Por ejemplo, supongamos que el número de Cuenta de AWS es 111122223333 y que dispone de un bucket de Amazon S3 al que desea que Susan obtenga acceso. Las credenciales de seguridad temporales de Susan no incluyen una política para el bucket. En ese caso, tendría que asegurarse de que el bucket tiene una política con un ARN que coincida con el ARN de Susan, como `arn:aws:sts::111122223333:federated-user/Susan`.

GetSessionToken: credenciales temporales para usuarios de entornos que no son de confianza

La operación de API `GetSessionToken` devuelve un conjunto de credenciales de seguridad temporales para un usuario de IAM existente. Es útil para proporcionar mayor seguridad, por ejemplo, para permitir las solicitudes de AWS únicamente cuando la función MFA está habilitada para el usuario de IAM. Dado que las credenciales son temporales, proporcionan mayor seguridad cuando se dispone de un usuario de IAM que tiene acceso a los recursos a través de un entorno

menos seguro. Algunos ejemplos de entornos menos seguros son un dispositivo móvil o un navegador web. Para obtener más información, consulte Solicitud de credenciales de seguridad temporales o GetBucketEncryption en la Referencia de la API de AWS Security Token Service.

De forma predeterminada, las credenciales de seguridad temporales de un usuario de IAM son válidas durante un máximo de 12 horas. Sin embargo, puede solicitar una duración mínima de 15 minutos o una duración máxima de 36 horas mediante el parámetro `DurationSeconds`. Por motivos de seguridad, un token para un usuario Usuario raíz de la cuenta de AWS está limitado a una hora.

`GetSessionToken` devuelve credenciales de seguridad temporales que incluyen un token de sesión, un ID de clave de acceso y una clave de acceso secreta. En el siguiente ejemplo se muestra una solicitud y respuesta de muestra que utiliza `GetSessionToken`. La respuesta también incluye el plazo de vencimiento de las credenciales de seguridad temporales.

Example Ejemplo de solicitud

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=1800
&AUTHPARAMS
```

El parámetro `AUTHPARAMS` en el ejemplo es un marcador de posición para su firma. Una firma es la información de autenticación que debe incluir en las solicitudes de la API HTTP de AWS. Recomendamos utilizar los SDK de AWS para crear solicitudes de API; un beneficio de esto es que los SDK gestionan la firma de solicitudes en su nombre. Si debe crear y firmar manualmente solicitudes de API, diríjase a Firma de solicitudes de AWS con Signature Version 4 en la Referencia general de Amazon Web Services para averiguar cómo firmar una solicitud.

Example Ejemplo de respuesta

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
<GetSessionTokenResult>
<Credentials>
  <SessionToken>
   AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
   To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
   rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
   Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
  </SessionToken>
```

```
  <SecretAccessKey>
  wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
  </SecretAccessKey>
  <Expiration>2011-07-11T19:55:29.611Z</Expiration>
  <AccessKeyId>AKIAIOSFODNN7EXAMPLE</AccessKeyId>
</Credentials>
</GetSessionTokenResult>
<ResponseMetadata>
<RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
</ResponseMetadata>
</GetSessionTokenResponse>
```

De forma opcional, la solicitud `GetSessionToken` puede incluir los valores `SerialNumber` y `TokenCode` para la verificación con autenticación multifactor (MFA) de AWS. Si los valores facilitados son válidos, AWS STS proporciona credenciales de seguridad temporales que incluyen el estado de la autenticación MFA. A continuación, las credenciales de seguridad temporales pueden utilizarse para obtener acceso a las acciones de la API protegidas por MFA o a los sitios web de AWS durante el periodo en que la autenticación MFA sea válida.

El siguiente ejemplo muestra una solicitud `GetSessionToken` que incluye un código de verificación de MFA y un número de serie del dispositivo.

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=7200
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
&AUTHPARAMS
```

> **Note**
>
> La llamada a AWS STS se puede realizar al punto de conexión global o a cualquiera de los puntos de conexión regionales que active en su Cuenta de AWS. Para obtener más información, consulte la sección de AWS STS de Regiones y puntos de enlace.
> El parámetro `AUTHPARAMS` en el ejemplo es un marcador de posición para su firma. Una firma es la información de autenticación que debe incluir en las solicitudes de la API HTTP de AWS. Recomendamos utilizar los SDK de AWS para crear solicitudes de API; un beneficio de esto es que los SDK gestionan la firma de solicitudes en su nombre. Si debe crear y firmar manualmente solicitudes de API, diríjase a Firma de solicitudes de AWS con Signature

Version 4 en la Referencia general de Amazon Web Services para averiguar cómo firmar una solicitud.

Comparación de las operaciones de la API de AWS STS

La siguiente tabla compara las características de las operaciones de la API de AWS STS que devuelven credenciales de seguridad temporales. Para obtener más información sobre los distintos métodos que puede utilizar para solicitar credenciales de seguridad temporales asumiendo un rol, consulte Uso de roles de IAM. Para obtener información sobre las diferentes operaciones de API de AWS STS que le permiten pasar etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

Comparación de opciones de API

API de AWS STS	Quién puede llamar	Duración de las credenciales (mín | máx | predeterminada)	Soporte de MFA[1]	Compatibilidad con políticas de sesión[2]	Restricciones aplicables a las credenciales temporales obtenidas
AssumeRole	Usuario de IAM o rol de IAM con credenciales de seguridad temporales existentes	15 min | configuración de la duración máxima de la sesión[3] | 1 h	Sí	Sí	No se puede llamar a `GetFederationToken` ni a `GetSessionToken`.
AssumeRoleWithSAML	Cualquier usuario: el intermediario debe transmitir una respuesta de autenticación de	15 min | configuración de la duración	No	Sí	No se puede llamar a `GetFederationToken` ni a `GetSessionToken`.

API de AWS STS	Quién puede llamar	Duración de las credenciales (mín | máx | predeterminada)	Soporte de MFA[1]	Compatibilidad con políticas de sesión[2]	Restricciones aplicables a las credenciales temporales obtenidas
	SAML que indique la autenticación de un proveedor de identidad conocido	máxima de la sesión[3] | 1 h			
AssumeRoleWithWebIdentity	Cualquier usuario; la persona que llama debe pasar un token JWT compatible con OIDC que indique la autenticación de un proveedor de identidad conocido	15 min | configuración de la duración máxima de la sesión[3] | 1 h	No	Sí	No se puede llamar a `GetFederationToken` ni a `GetSessionToken` .
GetFederationToken	Usuario de IAM o Usuario raíz de la cuenta de AWS	Usuario de IAM: 15 m | 36 h | 12 h Usuario raíz: 15 m | 1 h | 1 h	No	Sí	No se puede llamar a operaciones de IAM mediante AWS CLI o la API de AWS. Esta limitación no se aplica a las sesiones de consola. No se puede llamar a operaciones de AWS STS excepto `GetCallerIdentity` .[4] Se permite el inicio de sesión único (SSO) en la consola.[5]

API de AWS STS	Quién puede llamar	Duración de las credenciales (mín | máx | predeterminada)	Soporte de MFA[1]	Compatibilidad con políticas de sesión[2]	Restricciones aplicables a las credenciales temporales obtenidas
GetSessionToken	Usuario de IAM o Usuario raíz de la cuenta de AWS	Usuario de IAM: 15 m | 36 h | 12 h Usuario raíz: 15 m | 1 h | 1 h	Sí	No	No se puede llamar a las operaciones de API de IAM, a no ser que se incluya en la solicitud la información de MFA. No se puede llamar a las operaciones de API de AWS STS excepto `AssumeRole` o `GetCallerIdentity` . No se permite el inicio de sesión único (SSO) en la consola.[6]

[1] Compatibilidad con MFA. Puede incluir información acerca del dispositivo de autenticación multifactor (MFA) cuando llama a las operaciones de API AssumeRole y GetSessionToken. De este modo, se garantiza que las credenciales de seguridad temporales que se derivan de la llamada a la API solo las puedan utilizar los usuarios que se autentiquen con un dispositivo MFA. Para obtener más información, consulte Configuración del acceso a una API protegido por MFA.

[2] Soporte con políticas de sesión. Las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Esta política limita los permisos de la política basada en identidad del rol o usuario que se asignan a la sesión. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la entidad y las políticas de la sesión. Las políticas de sesión no se pueden utilizar para conceder más permisos que los permitidos por la política basada en identidades del rol que se asume. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.

[3] Configuración de la duración máxima de la sesión. Use el parámetro `DurationSeconds` para especificar la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

[4] GetCallerIdentity. No se requieren permisos para realizar esta operación. Si un administrador añade una política a su usuario o rol de IAM que deniega explícitamente el acceso a la acción `sts:GetCallerIdentity`, puede realizar esta operación. Los permisos no son necesarios porque se devuelve la misma información cuando se deniega el acceso a un usuario o rol de IAM. Para ver un ejemplo de respuesta, consulte No tengo autorización para realizar la operación iam:DeleteVirtualMFADevice.

[5] .[6]. Para facilitar el inicio de sesión único (SSO), AWS le permite llamar a un punto de enlace de federación (`https://signin.aws.amazon.com/federation`) y transmitir las credenciales de seguridad temporales. El punto de enlace devuelve un token que puede utilizar para crear una dirección URL con la que un usuario inicia sesión directamente en la consola sin necesidad de una contraseña. Para obtener más información, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 y How to Enable Cross-Account Access to the AWS Management Console en el blog de seguridad de AWS.

[6] Tras recuperar las credenciales temporales, no puede acceder a la AWS Management Console transmitiendo las credenciales al punto de enlace de inicio de sesión único de la federación. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

Uso de credenciales temporales con recursos de AWS

Puede utilizar credenciales de seguridad temporales para realizar solicitudes programáticas de recursos de AWS mediante AWS CLI o la API de AWS o (mediante losSDK de AWS). Las credenciales temporales proporcionan los mismos permisos que las credenciales de seguridad a largo plazo, como las credenciales de usuario de IAM. Sin embargo, hay algunas diferencias:

- Cuando realice una llamada utilizando las credenciales de seguridad temporales, la llamada debe incluir un token de sesión, que se devuelve junto con las credenciales temporales en cuestión. AWS utiliza el token de sesión para validar las credenciales de seguridad temporales.
- Las credenciales temporales vencen después de un intervalo especificado. Después de que las credenciales temporales venzan, cualquier llamada que hagas con esas credenciales fallará, por lo

que deberás generar un nuevo conjunto de credenciales temporales. Las credenciales temporales no pueden extenderse o actualizarse más allá del intervalo original especificado.

- Cuando utiliza credenciales temporales para realizar una solicitud, la entidad principal puede incluir un conjunto de etiquetas. Estas etiquetas provienen de etiquetas de sesión y etiquetas asociadas al rol que asume. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

Si está utilizando los SDK de AWS, AWS Command Line Interface (AWS CLI) o Tools for Windows PowerShell, la forma de obtener y usar credenciales de seguridad temporales difiere según el contexto. Si ejecuta código o comandos de la AWS CLI o las Tools for Windows PowerShell en una instancia EC2, puede sacar partido de los roles de Amazon EC2. De lo contrario, puede llamar a una API de AWS STS para obtener las credenciales temporales y, a continuación, utilizarlas de forma explícita para realizar llamadas a los servicios de AWS.

> **Note**
>
> Puede utilizar AWS Security Token Service (AWS STS) para crear credenciales de seguridad temporales que pueden controlar el acceso a sus recursos de AWS y proporcionárselas a usuarios de confianza. Para obtener más información sobre AWS STS, consulte Credenciales de seguridad temporales en IAM. AWS STS es un servicio global que tiene un punto de enlace predeterminado en `https://sts.amazonaws.com`. Este punto de conexión se encuentra en la región Este de EE. UU. (Norte de Virginia), aunque las credenciales que obtiene de este y otros puntos de conexión son válidas a nivel global. Estas credenciales funcionan con servicios y recursos de cualquier región. También puede optar por realizar llamadas a API de AWS STS a los puntos de enlace de cualquier región compatible. Esto puede reducir la latencia realizando las solicitudes desde servidores de una región que está más cerca de usted. No importa de qué región vienen sus credenciales, funcionan en todo el mundo. Para obtener más información, consulte Administrar AWS STS en una Región de AWS.

Contenido

- Más información

Uso de credenciales temporales en instancias Amazon EC2

Si desea ejecutar código o comandos de AWS CLI en una instancia EC2, la forma recomendada de obtener credenciales es utilizar roles para Amazon EC2. Cree un rol de IAM que especifique los permisos que desea conceder a las aplicaciones que se ejecutan en las instancias EC2. Al lanzar la instancia, asocie el rol con la instancia.

Las aplicaciones y los comandos de la AWS CLI y las Tools for Windows PowerShell que se ejecutan en la instancia pueden obtener credenciales de seguridad temporales automáticas desde los metadatos de la instancia. No es necesario obtener explícitamente las credenciales de seguridad temporales. Los SDK de AWS, AWS CLI y Tools for Windows PowerShell obtienen automáticamente las credenciales del servicio de metadatos de instancia de EC2 y las utilizan. Las credenciales temporales tienen los permisos que usted defina para el rol asociado a la instancia.

Para obtener más información y ejemplos, consulte lo siguiente:

- Uso de roles de IAM para conceder acceso a recursos de AWS en Amazon Elastic Compute Cloud (EC2) — AWS SDK for Java
- Granting Access Using an IAM Role — AWS SDK for .NET
- Crear un rol: AWS SDK for Ruby

Uso de credenciales de seguridad temporales con los SDK de AWS

Para utilizar credenciales de seguridad temporales en el código, se llama mediante programación a una API AWS STS similar a `AssumeRole` y se extraen las credenciales y el token de sesión resultantes. A continuación, utilice esos valores como credenciales para las llamadas posteriores a AWS. En el siguiente ejemplo se muestra un pseudocódigo en el que se indica cómo utilizar credenciales de seguridad temporales si utiliza un SDK de AWS:

```
assumeRoleResult = AssumeRole(role-arn);
tempCredentials = new SessionAWSCredentials(
   assumeRoleResult.AccessKeyId,
   assumeRoleResult.SecretAccessKey,
   assumeRoleResult.SessionToken);
s3Request = CreateAmazonS3Client(tempCredentials);
```

Para ver un ejemplo escrito en Python (usando AWS SDK for Python (Boto)), consulte Cambio a un rol de IAM (API de AWS). En este ejemplo se muestra cómo llamar a `AssumeRole` para obtener credenciales de seguridad temporales y, a continuación, utilizar esas credenciales para realizar una llamada a Amazon S3.

Para obtener más información sobre cómo llamar a `AssumeRole`, `GetFederationToken` y otras operaciones de API, consulte la Referencia de la API de AWS Security Token Service. Para obtener información sobre cómo obtener las credenciales de seguridad temporales y el token de sesión a partir del resultado, consulte la documentación para el SDK con el que está trabajando. Encontrará la documentación para todos los SDK de AWS en la página de documentación de AWS principal, en la sección SDK y conjuntos de herramientas.

Debe asegurarse de que obtiene un nuevo conjunto de credenciales antes de que caduquen las antiguas. En algunos SDK, puede utilizar un proveedor que administre en su nombre el proceso de actualización de credenciales; compruebe la documentación del SDK que esté utilizando.

Uso de credenciales de seguridad temporales con AWS CLI

Puede utilizar las de credenciales de seguridad temporales con AWS CLI. Esto puede resultar útil para probar políticas.

Mediante la AWS CLI, puede llamar a una API de AWS STS como `AssumeRole` o `GetFederationToken` y, a continuación, capturar la salida resultante. En el siguiente ejemplo se muestra una llamada a `AssumeRole` que envía la salida a un archivo. En el ejemplo, se supone que el parámetro `profile` es un perfil en el archivo de configuración de AWS CLI. También se supone que hace referencia a las credenciales de un usuario de IAM que tiene permisos para asumir el rol.

```
aws sts assume-role --role-arn arn:aws:iam::123456789012:role/role-name --role-session-
name "RoleSession1" --profile IAM-user-name > assume-role-output.txt
```

Cuando el comando está terminado, puede extraer el ID de clave de acceso, clave de acceso secreta y token de sesión del lugar en el que lo haya enrutado. Puede hacerlo manualmente o mediante un script. A continuación, puede asignar estos valores a las variables de entorno.

Cuando ejecuta comandos AWS CLI, AWS CLI buscará credenciales en un orden determinado, primero en las variables de entorno y, a continuación, en el archivo de configuración. Por lo tanto, cuando haya colocado las credenciales temporales en las variables de entorno, AWS CLI utilizará dicha credenciales de forma predeterminada (Si especifica un parámetro `profile` en el comando, la AWS CLI omite las variables de entorno. En lugar de eso, el AWS CLI busca en el archivo de

configuración, que le permite anular las credenciales en las variables de entorno, en el caso de ser necesario).

En el siguiente ejemplo se muestra cómo puede establecer las variables de entorno para las credenciales de seguridad temporales y, a continuación, llamar a un comando de AWS CLI. Dado que no se incluye el parámetro `profile` en el comando de AWS CLI, AWS CLI busca primero credenciales en las variables de entorno y, de este modo, utiliza las credenciales temporales.

Linux

```
$ export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
$ export AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of session token>
$ aws ec2 describe-instances --region us-west-1
```

Windows

```
C:\> SET AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
C:\> SET AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
C:\> SET AWS_SESSION_TOKEN=AQoDYXdzEJr...<remainder of token>
C:\> aws ec2 describe-instances --region us-west-1
```

Uso de credenciales de seguridad temporales con operaciones de API

Si está realizando solicitudes directas de la API HTTPS a AWS, puede firmar dichas solicitudes con las credenciales de seguridad temporales que obtiene de AWS Security Token Service (AWS STS). Para ello, utilice el ID de clave de acceso y la clave de acceso secreta que recibe de AWS STS. Utilice el ID de clave de acceso y la clave de acceso secreta de la misma forma que utilizaría las credenciales a largo plazo para firmar una solicitud. También puede añadir a su solicitud de la API el token de sesión que reciba de AWS STS. Puede añadir el token de sesión a un encabezado HTTP o a un parámetro de cadena de consulta denominado `X-Amz-Security-Token`. Añada el token de sesión al encabezado HTTP o al parámetro de cadena de consulta, pero no a ambos. Para obtener más información sobre la firma de las solicitudes de la API HTTPS, consulte Firma de solicitudes de la API de AWS en la Referencia general de AWS.

Más información

Para obtener más información acerca de AWS STS con otros servicios de AWS, consulte los siguientes vínculos:

- Amazon S3. Consulte Realización de solicitudes con las credenciales temporales de usuario de IAM o Realización de solicitudes con credenciales temporales de usuario federado en la Guía del usuario de Amazon Simple Storage Service.
- Amazon SNS. Consulte Uso de políticas basadas en identidades con Amazon SNS en la Guía para desarrolladores de Amazon Simple Queue Service.
- Amazon SQS. Consulte Administración de identidades y acceso en Amazon SQS en la Guía para desarrolladores de Amazon Simple Queue Service.
- Amazon SimpleDB. Consulte Utilizar credenciales de seguridad temporales en la Guía para desarrolladores de Amazon SimpleDB.

Control de los permisos para credenciales de seguridad temporales

Puede utilizar AWS Security Token Service (AWS STS) para crear credenciales de seguridad temporales que pueden controlar el acceso a sus recursos de AWS y proporcionárselas a usuarios de confianza. Para obtener más información acerca de AWS STS, consulte Credenciales de seguridad temporales en IAM. Las credenciales de seguridad temporales que emite AWS STS son válidas durante el periodo de vencimiento y no se pueden revocar. Sin embargo, los permisos asignados a credenciales de seguridad temporales se evalúan cada vez que una solicitud utiliza las credenciales, por lo que puede conseguir el efecto de revocar las credenciales cambiando sus permisos de acceso incluso después de que se hayan emitido.

En los siguientes temas se presupone que tiene experiencia trabajando con permisos y políticas de AWS. Para obtener más información sobre estos temas, consulte Recursos de AWS para administración de acceso.

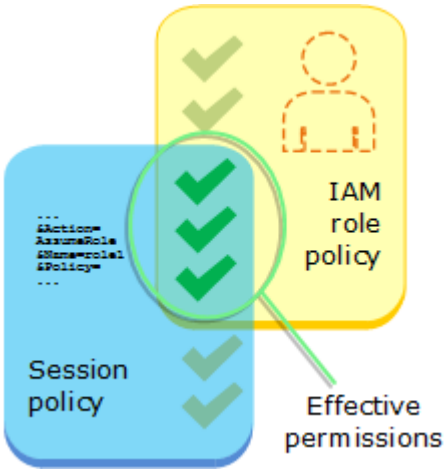
Temas

- Permisos de AssumeRole, AssumeRoleWithSAML y AssumeRoleWithWebIdentity
- Monitorear y controlar las acciones realizadas con roles asumidos
- Permisos para GetFederationToken
- Permisos para GetSessionToken
- Deshabilitar permisos para credenciales de seguridad temporales
- Concesión de permisos para crear credenciales de seguridad temporales

Permisos de AssumeRole, AssumeRoleWithSAML y AssumeRoleWithWebIdentity

La política de permisos del rol que se asume determina los permisos de las credenciales de seguridad temporales devueltas por `AssumeRole`, `AssumeRoleWithSAML` y `AssumeRoleWithWebIdentity`. Defina estos permisos al crear o actualizar el rol.

De forma opcional, puede pasar políticas de sesión administradas o insertadas como parámetros de las operaciones de API `AssumeRole`, `AssumeRoleWithSAML` o `AssumeRoleWithWebIdentity`. Las políticas de sesión limitan los permisos para la sesión de credenciales temporales de la función. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la función y las políticas de la sesión. Puede utilizar las credenciales temporales del rol en llamadas posteriores a la API de AWS para tener acceso a los recursos de la cuenta propietaria del rol. No puede utilizar las políticas de sesión para conceder más permisos que los permitidos por la política basada en identidades de la función que se asume. Para obtener más información sobre cómo determina AWS los permisos efectivos de un rol, consulte Lógica de evaluación de políticas.



AWS no evalúa las políticas asociadas a las credenciales con las que se hizo la llamada original a `AssumeRole` para tomar la decisión de autorización "permitir" o "denegar". El usuario abandona temporalmente sus permisos originales en favor de los permisos asignados al rol asumido. En el caso de las operaciones de API `AssumeRoleWithSAML` y `AssumeRoleWithWebIdentity`, no existen políticas que evaluar, porque el intermediario de la API no es una identidad de AWS.

Ejemplo: asignación de permisos utilizando AssumeRole

Puede utilizar la operación de API `AssumeRole` con diferentes tipos de políticas. A continuación se ofrecen algunos ejemplos.

Política de permisos del rol

En este ejemplo, se llama a la operación de API `AssumeRole` sin especificar la política de sesión en el parámetro opcional `Policy`. Los permisos asignados a las credenciales temporales vienen determinados por la política de permisos del rol que se asume. En el ejemplo siguiente, la política de permisos concede el permiso de rol para enumerar todos los objetos contenidos en un bucket de S3 denominado `productionapp`. También permite al rol obtener, agregar y eliminar objetos en ese bucket.

Example Ejemplo de política de permisos del rol

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Política de sesión pasada como parámetro

Imagine que desea permitir a un usuario asumir el mismo rol que en el ejemplo anterior. Sin embargo, en este caso desea que la sesión del rol solo tenga permiso para obtener y colocar objetos en el bucket de S3 `productionapp`. No desea permitirle eliminar objetos. Una forma de conseguirlo es crear un nuevo rol y especificar los permisos deseados en su política de permisos. Otra forma de conseguirlo consiste en llamar a la API `AssumeRole` e incluir políticas de sesión en el parámetro opcional `Policy` como parte de la llamada a la operación de API. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la función y las políticas de la sesión. Las políticas de sesión no se pueden utilizar para conceder más permisos que

los permitidos por la política basada en identidades de la función que se asume. Para obtener más información sobre los permisos de sesión de un rol, consulte Políticas de sesión.

Después de recuperar las credenciales temporales de la nueva sesión, puede pasarlas al usuario que desea que tenga esos permisos.

Por ejemplo, imagine que las siguientes políticas se transfieren como parámetro de la llamada a la API. La persona que ha iniciado la sesión solo tiene permisos para ejecutar las acciones siguientes:

- Realizar una lista de todos los objetos del bucket `productionapp`.
- Obtener y colocar objetos en el bucket `productionapp`.

Con la política de sesión siguiente, el permiso `s3:DeleteObject` se descarta y en la sesión no se concede el permiso `s3:DeleteObject`. La política establece los permisos máximos para la sesión del rol, anulando todas las políticas de permisos que tuviera el rol.

Example Ejemplo de política de sesión pasada con la llamada a la API **`AssumeRole`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
    }
  ]
}
```

Política basada en recursos

Algunos recursos de AWS admiten políticas basadas en recursos y dichas políticas proporcionan otro mecanismo para definir los permisos que afectan directamente a las credenciales de seguridad

temporales. Solo unos pocos recursos, como buckets de Amazon S3, temas de Amazon SNS y colas de Amazon SQS admiten políticas basadas en recursos. El siguiente ejemplo amplía los ejemplos anteriores y utiliza un bucket de S3 denominado `productionapp`. La política siguiente se asocia al bucket.

Al adjuntar la siguiente política basada en recursos al bucket `productionapp`, se deniega a todos los usuarios el permiso para eliminar objetos del bucket. (Consulte el elemento `Principal` en la política). Esto incluye todos los usuarios con el rol asumido, aunque la política de permisos de rol conceda el permiso `DeleteObject`. Una instrucción `Deny` explícita siempre prevalece sobre una instrucción `Allow`.

Example Ejemplo de política de bucket

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "*"},
    "Effect": "Deny",
    "Action": "s3:DeleteObject",
    "Resource": "arn:aws:s3:::productionapp/*"
  }
}
```

Para obtener más información sobre el modo en que AWS combina y evalúa varios tipos de políticas, consulte Lógica de evaluación de políticas.

Monitorear y controlar las acciones realizadas con roles asumidos

Un rol de IAM es un objeto en IAM que está asignado a permisos. Cuando usted asume ese rol utilizando una identidad de IAM o una identidad de fuera de AWS, recibirá una sesión con los permisos que se asignan al rol.

Cuando lleva a cabo acciones en AWS, la información sobre su sesión se puede registrar en AWS CloudTrail para que el administrador de la cuenta la supervise. Los administradores pueden configurar roles para requerir que las identidades pasen una cadena personalizada que identifique a la persona o aplicación que está realizando acciones en AWS. Esta información de identidad se almacena como Identidad de origen en AWS CloudTrail. Cuando el administrador revisa la actividad en CloudTrail, puede ver la información de la identidad de origen para determinar quién o qué realizó acciones con las sesiones de rol asumidas.

Después de establecer una identidad de origen, está presente en las solicitudes de cualquier acción de AWS realizada durante la sesión de rol. El valor que se establece persiste cuando se utiliza un rol para asumir otro rol a través de AWS CLI o API de AWS, conocida como encadenamiento de roles. El valor que se establece no se puede cambiar durante la sesión de rol. Los administradores pueden configurar permisos pormenorizados basados en la presencia o el valor de la identidad de origen para controlar aún más las acciones de AWS que se realizan con roles compartidos. Puede decidir si se puede utilizar el atributo de identidad de origen, si es necesario y qué valor se puede utilizar.

La forma en que utiliza la identidad de origen difiere del nombre de sesión de rol y las etiquetas de sesión de forma importante. El valor de identidad de origen no se puede cambiar una vez establecido y persiste para cualquier acción adicional que se realice con la sesión de rol. A continuación se muestra cómo puede utilizar las etiquetas de sesión y el nombre de la sesión de rol:

- Etiquetas de sesión - Puede pasar etiquetas de sesión al asumir un rol o federar un usuario. Las etiquetas de sesión están presentes cuando se asume un rol. Puede definir políticas que utilicen claves de condición de etiqueta para conceder permisos a sus entidades principales en función de sus etiquetas. Luego puede utilizar CloudTrail para ver las solicitudes realizadas para asumir roles o federar usuarios. Para obtener más información sobre las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.
- Nombre de la sesión de rol – Puede utilizar la clave de condición `sts:RoleSessionName` en una política de confianza de rol para exigir que los usuarios proporcionen un nombre de sesión específico cuando asuman un rol. El nombre de sesión de rol se puede utilizar para diferenciar sesiones de rol cuando un rol es utilizado por diferentes entidades principales. Para obtener más información sobre el nombre de la sesión de rol, consulte sts:RoleSessionName.

Le recomendamos que utilice la identidad de origen cuando desee controlar la identidad que asume un rol. La identidad de origen también es útil para extraer registros de CloudTrail para determinar quién utilizó el rol para realizar acciones.

Temas

- Configuración para utilizar la identidad de origen
- Cosas que debe saber sobre la identidad de origen
- Permisos necesarios para establecer la identidad de origen
- Especificar una identidad de origen al asumir un rol
- Utilizar la identidad de origen con AssumeRole

- Utilizar la identidad de origen con AssumeRoleWithSAML
- Utilizar la identidad de origen con AssumeRoleWithWebIdentity
- Controlar el acceso mediante información de identidad de origen
- Visualización de identidad de origen en CloudTrail

Configuración para utilizar la identidad de origen

La forma en que se configura para utilizar la identidad de origen depende del método utilizado cuando se asumen los roles. Por ejemplo, los usuarios de IAM pueden asumir roles directamente mediante la operación `AssumeRole`. Si tiene identidades empresariales, también conocidas como identidades de personal, es posible que accedan a sus recursos de AWS utilizando `AssumeRoleWithSAML`. Si los usuarios finales acceden a sus aplicaciones móviles o web, es posible que lo hagan mediante`AssumeRoleWithWebIdentity`. A continuación se ofrece información general del flujo de trabajo de alto nivel para ayudarle a comprender cómo puede configurar la utilización de la información de identidad de origen en su entorno existente.

1. Configurar usuarios y roles de prueba - Mediante un entorno de preproducción, configure los usuarios y roles de prueba y configure sus políticas para permitir establecer una identidad de origen.

   Si utiliza un proveedor de identidades (IdP) para sus identidades federadas, configure su IdP para que pase un atributo de usuario de su elección para la identidad de origen en la aserción o el token.
2. Asuma el rol — Pruebe asumir roles y pasar una identidad de origen con los usuarios y roles que configuró para la prueba.
3. Revisión de CloudTrail – Revise la información de identidad de origen para sus roles de prueba en los registros de CloudTrail.
4. Forme a sus usuarios – Después de haber probado en su entorno de preproducción, asegúrese de que sus usuarios sepan cómo transmitir la información de identidad de origen, si es necesario. Establezca una fecha límite para cuándo requerirá a los usuarios que proporcionen una identidad de origen en su entorno de producción.
5. Configuración de políticas de producción – Configure las políticas para su entorno de producción y, a continuación, agréguelas a los usuarios y roles de producción.
6. Monitoreo de actividad – Monitoree la actividad de su rol de producción mediante los registros de CloudTrail.

Cosas que debe saber sobre la identidad de origen

Tenga en cuenta lo siguiente cuando trabaje con identidad de origen.

- Las políticas de confianza de roles para todos los roles conectados a un proveedor de identidades (IdP) deben tener el permiso `sts:SetSourceIdentity`. En el caso de los roles que no tienen este permiso en la política de confianza de rol, la operación `AssumeRole*` producirá un error. Si no desea actualizar la política de confianza de rol para cada rol, puede utilizar una instancia de proveedor de identidades independiente para pasar la identidad de origen. A continuación, agregue el permiso `sts:SetSourceIdentity` solo a los roles que están conectados al proveedor de identidades independiente.
- Cuando una identidad establece una identidad de origen, la clave `sts:SourceIdentity` está presente en la solicitud. Para las acciones posteriores realizadas durante la sesión de rol, la clave `aws:SourceIdentity` está presente en la solicitud. AWS no controla el valor de la identidad de origen en claves `sts:SourceIdentity` o `aws:SourceIdentity`. Si decide requerir una identidad de origen, debe elegir un atributo que desea que proporcionen sus usuarios o IdP. Por motivos de seguridad, debe asegurarse de que puede controlar cómo se proporcionan esos valores.
- El valor de la identidad de origen debe tener entre 2 y 64 caracteres, solo puede contener caracteres alfanuméricos, guiones bajos y los siguientes caracteres: . , + = @ - (guion). No puede utilizar un valor que comience con el texto **`aws:`**. Este prefijo se reserva para uso interno de AWS.
- La información de la identidad de origen no es capturada por CloudTrail cuando un servicio o rol vinculado a un servicio de AWS realiza una acción en nombre de una identidad federada o del personal.

> **Important**
>
> No puede cambiar a un rol en AWS Management Console que requiera que se establezca una identidad de origen cuando se asume el rol. Para asumir tal rol, puede utilizar el AWS CLI o API de AWS para llamar a la operación `AssumeRole` y especifique el parámetro de identidad de origen.

Permisos necesarios para establecer la identidad de origen

Además de la acción que coincide con la operación de la API, debe tener la siguiente acción de solo permisos en la política:

```
sts:SetSourceIdentity
```

- Para especificar una identidad de origen, las entidades principales (usuarios y roles de IAM) deben tener permisos para `sts:SetSourceIdentity`. Como administrador, puede configurarlo en la política de confianza de rol y en la política de permisos de seguridad de la entidad principal.
- Cuando asume un rol con otro rol, llamado encadenamiento de roles, se requieren permisos para `sts:SetSourceIdentity` tanto en la política de permisos de la entidad principal que está asumiendo el rol como en la política de confianza de rol del rol de destino. De lo contrario, la operación de rol asumido no se llevará a cabo correctamente.
- Cuando se utiliza la identidad de origen, las políticas de confianza de roles para todos los roles conectados a un proveedor de identidades (IdP) deben tener el permiso `sts:SetSourceIdentity`. La operación `AssumeRole*` producirá un error para cualquier rol conectado a un IdP sin este permiso. Si no desea actualizar la política de confianza de rol para cada rol, puede utilizar una instancia de proveedor de identidades independiente para pasar la identidad de origen y agregar el permiso de `sts:SetSourceIdentity` solamente a los roles que están conectados al IdP separado.
- Para establecer una identidad de origen a través de los límites de la cuenta, debe incluir el permiso de `sts:SetSourceIdentity` en dos lugares. Debe estar en la política de permisos de la entidad principal en la cuenta de origen y en la política de confianza de rol del rol en la cuenta de destino. Es posible que tenga que hacer esto, por ejemplo, cuando se usa un rol para asumir un rol en otra cuenta con encadenamiento de roles.

Como administrador de la cuenta, imagine que desea permitir que el usuario de IAM `DevUser` en su cuenta para que asuma el `Developer_Role` en la misma cuenta. Pero desea permitir esta acción solo si el usuario ha establecido la identidad de origen en su nombre de usuario de IAM. Puede asignar la siguiente política al usuario IAM.

Example Ejemplos de política basada en identidades adjunta a DevUser

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRole",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role"
```

```
    },
    {
      "Sid": "SetAwsUserNameAsSourceIdentity",
      "Effect": "Allow",
      "Action": "sts:SetSourceIdentity",
      "Resource": "arn:aws:iam::123456789012:role/Developer_Role",
      "Condition": {
        "StringLike": {
          "sts:SourceIdentity": "${aws:username}"
        }
      }
    }
  ]
}
```

Para aplicar los valores de identidad de origen aceptables, puede configurar la siguiente política de confianza de rol. La política proporciona al usuario de IAM permisos de `DevUser` para asumir el rol y establecer una identidad de origen. La clave de condición `sts:SourceIdentity` define el valor de identidad de origen aceptable.

Example Ejemplo de política de confianza de rol para identidad de origen

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDevUserAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/DevUser"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "sts:SourceIdentity": "DevUser"
        }
      }
    }
  ]
}
```

Al utilizar las credenciales para el usuario de IAM de `DevUser`, el usuario intenta asumir el `DeveloperRole` utilizando la siguiente solicitud AWS CLI.

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer_Role \
--role-session-name Dev-project \
--source-identity DevUser \
```

Cuando AWS evalúa la solicitud, el contexto de la solicitud contiene el `sts:SourceIdentity` de `DevUser`.

Especificar una identidad de origen al asumir un rol

Puede especificar una identidad de origen cuando utilice uno de las operaciones AWS STS API de `AssumeRole*` para obtener credenciales de seguridad temporales de un rol. La operación API que utilice difiere en función de su caso de uso. Por ejemplo, si utiliza roles de IAM para dar a los usuarios de IAM acceso a los recursos AWS que normalmente no tienen acceso, puede utilizar la operación de `AssumeRole`. Si utiliza la identidad federada de empresa para administrar los usuarios del personal, puede utilizar la operación de `AssumeRoleWithSAML`. Si utiliza la federación de OIDC para permitir que los usuarios finales accedan a sus aplicaciones móviles o Web, puede utilizar la operación de `AssumeRoleWithWebIdentity`. En las secciones siguientes se explica cómo utilizar la identidad de origen con cada operación. Para obtener más información sobre los escenarios comunes de las credenciales temporales, consulte Escenarios habituales en las credenciales temporales.

Utilizar la identidad de origen con AssumeRole

La operación `AssumeRole` devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Puede utilizar credenciales de usuario de IAM o credenciales de rol para llamar a `AssumeRole`. Para pasar identidad de origen mientras asume un rol, utilice la opción `--source-identity` de AWS CLI o el parámetro `SourceIdentity` de la API de AWS. En el siguiente ejemplo, se muestra cómo especificar la identidad de origen con la herramienta de AWS CLI.

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/developer \
```

```
--role-session-name Audit \
--source-identity Admin \
```

Utilizar la identidad de origen con AssumeRoleWithSAML

La entidad principal de llamada a la operación `AssumeRoleWithSAML` se autentica mediante la federación basada en SAML. Esta operación devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Para obtener más información acerca del uso de la federación basada en SAML para el acceso a la AWS Management Console, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 . Para obtener información detallada sobre el acceso a la API de AWS CLI o AWS, consulte Federación SAML 2.0. Para obtener un tutorial sobre cómo configurar la federación de SAML para los usuarios de Active Directory, consulte autenticación federada Active Directory Federation Services (ADFS) de AWS en el blog de seguridad de AWS.

Como administrador, puede permitir que los miembros del directorio de su empresa se federen en AWS mediante la operación AWS STS `AssumeRoleWithSAML`. Para ello, debe completar las siguientes tareas:

1. Configurar un proveedor SAML en su organización.
2. Crear un proveedor SAML en IAM.
3. Configure un rol y sus permisos en AWS para los usuarios federados.
4. Finalice la configuración del proveedor de identidad SAML y cree aserciones para la respuesta de autenticación SAML.

Para establecer un atributo SAML para la identidad de origen, incluya el elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/SourceIdentity`. Utilice el elemento `AttributeValue` para especificar el valor de la identidad de origen. Por ejemplo, suponga que desea pasar los siguientes atributos de identidad como la identidad de origen.

`SourceIdentity:DiegoRamirez`

Para pasar este atributo, incluya el siguiente elemento en su aserción de SAML.

Example Ejemplo de fragmento de una aserción de SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
```

```
<AttributeValue>DiegoRamirez</AttributeValue>
</Attribute>
```

Utilizar la identidad de origen con AssumeRoleWithWebIdentity

La entidad principal que llama a la operación `AssumeRoleWithWebIdentity` se autentica mediante la federación compatible con OpenID Connect (OIDC). Esta operación devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Para obtener más información acerca del uso de la federación de OIDC para el acceso a la AWS Management Console, consulte Federación OIDC.

Para pasar la identidad de origen desde OpenID Connect (OIDC), debe incluir la identidad de origen en el Token Web JSON (JWT). Incluya identidad de origen en el espacio de nombres `https://aws.amazon.com/ source_identity` en el token cuando envíe la solicitud `AssumeRoleWithWebIdentity`. Para obtener más información sobre los tokens y las notificaciones de OIDC, consulte Uso de tokens con grupos de usuarios en la Guía para desarrolladores de Amazon Cognito.

Por ejemplo, el siguiente JWT descodificado es un token que se utiliza para llamar a `AssumeRoleWithWebIdentity` con la identidad de origen de `Admin`.

Example Ejemplo de token web JSON decodificado

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/source_identity":"Admin"
}
```

Controlar el acceso mediante información de identidad de origen

Cuando se establece inicialmente una identidad de origen, la clave sts:SourceIdentity está presente en la solicitud. Después de establecer una identidad de origen, la clave AWS:SourceIdentity está presente en todas las solicitudes posteriores realizadas durante la sesión de rol. Como

administrador, puede escribir políticas que otorguen autorización condicional para realizar acciones de AWS basadas en la existencia o el valor del atributo de identidad de origen.

Imagine que desea exigirle a sus desarrolladores que establezcan una identidad de origen para que asuman un rol crítico que tenga permiso para escribir en un recurso de producción crítico de AWS. Imagínese también que concede acceso de AWS a las identidades de su personal mediante `AssumeRoleWithSAML`. Solo desea que los desarrolladores sénior Saanvi y Diego tengan acceso al rol, de modo que cree la siguiente política de confianza para el rol.

Example Ejemplo de política de confianza de rol para identidad de origen (SAML)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SAMLProviderAssumeRoleWithSAML",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:AssumeRoleWithSAML"
      ],
      "Condition": {
        "StringEquals": {
          "SAML:aud": "https://signin.aws.amazon.com/saml"
        }
      }
    },
    {
      "Sid": "SetSourceIdentitySrEngs",
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:saml-provider/name-of-identity-
provider"
      },
      "Action": [
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
```

```
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
  ]
}
```

La política de confianza contiene una condición para `sts:SourceIdentity` que requiere una identidad de origen de Saanvi o Diego a fin de asumir el rol crítico.

Alternativamente, si utiliza un proveedor OIDC para la federación y los usuarios se autentican con `AssumeRoleWithWebIdentity`, su política de confianza de rol podría tener el siguiente aspecto.

Example Ejemplo de política de confianza de rol para identidad de origen (proveedor OIDC)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/server.example.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "server.example.com:aud": "oidc-audience-id"
        },
        "StringLike": {
          "sts:SourceIdentity": [
            "Saanvi",
            "Diego"
          ]
        }
      }
    }
```

```
  ]
}
```

Encadenamiento de funciones y requisitos de cuentas cruzadas

Imagine que desea permitir a los usuarios que han asumido `CriticalRole` que asuman una `CriticalRole_2` en otra cuenta. Las credenciales de sesión de rol que se obtuvieron para asumir `CriticalRole` se utilizan para encadenamiento de roles a un segundo rol, `CriticalRole_2`, en una cuenta de diferente. El rol se asume a través de un límite de cuenta. Por lo tanto, el permio `sts:SetSourceIdentity` debe concederse tanto en la política de permisos en `CriticalRole` como en la política de confianza de rol en `CriticalRole_2`.

Example Ejemplo de política de permisos en CriticalRole

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AssumeRoleAndSetSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Resource": "arn:aws:iam::222222222222:role/CriticalRole_2"
    }
  ]
}
```

Para proteger la identidad de origen de configuración a través del límite de la cuenta, la siguiente política de confianza de rol confía solo en la entidad principal de rol de `CriticalRole` para establecer la identidad de origen.

Example Ejemplo de política de confianza de rol en CriticalRole_2

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "AWS": "arn:aws:iam::111111111111:role/CriticalRole"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringLike": {
          "aws:SourceIdentity": ["Saanvi","Diego"]
        }
      }
    }
  ]
}
```

El usuario realiza la siguiente llamada utilizando las credenciales de sesión de rol obtenidas al asumir CriticalRole. La identidad de origen se estableció durante el supuesto de CriticalRole, por lo que no necesita configurarse explícitamente de nuevo. Si el usuario intenta establecer una identidad de origen diferente del valor establecido cuando se asumió `CriticalRole`, se denegará la solicitud de rol de asunción.

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::222222222222:role/CriticalRole_2 \
--role-session-name Audit \
```

Cuando la entidad principal de llamada asume el rol, la identidad de origen en la solicitud persiste desde la primera sesión de rol asumida. En consecuencia, tanto `aws:SourceIdentity` como `sts:SourceIdentity` están presentes en el contexto de la solicitud.

Visualización de identidad de origen en CloudTrail

Usted puede utilizar CloudTrail para ver las solicitudes realizadas para asumir roles o federar usuarios. También puede ver las solicitudes de rol o usuario para realizar acciones en AWS. El archivo de registro de CloudTrail incluye información sobre la configuración de identidad de origen para la sesión de usuario federado o de rol asumido. Para obtener más información, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail

Por ejemplo, suponga que un usuario hace una solicitud AWS STS `AssumeRole`, y configura una identidad de origen. Puede encontrar la información `sourceIdentity` en la clave `requestParameters` de su sesión de CloudTrail.

Example Ejemplo de sección requestParameters en una sesión AWS CloudTrail

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "111122223333"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/DevRole",
        "roleSessionName": "Dev1",
        "sourceIdentity": "source-identity-value-set"
    }
```

Si el usuario utiliza la sesión de rol asumida para realizar una acción, la información de identidad de origen está presente en la clave `userIdentity` en la sesión de CloudTrail.

Example Ejemplo de clave userIdentity en una sesión AWS CloudTrail

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE:Dev1",
    "arn": "arn:aws:sts::123456789012:assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/DevRole",
```

```
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23:46:28Z"
      },
      "sourceIdentity": "source-identity-value-present"
    }
  }
}
```

Para ver el ejemplo de eventos de API AWS STS en las sesiones de CloudTrail, consulte Ejemplo de eventos API de IAM en el registro de CloudTrail. Para obtener más información sobre la información incluida en los archivos de sesión de CloudTrail, consulte Referencia de eventos de CloudTrail en la Guía del usuario de AWS CloudTrail.

Permisos para GetFederationToken

Un usuario de IAM llama a la operación `GetFederationToken` y devuelve credenciales temporales para ese usuario. Esta operación federa al usuario. Los permisos asignados a un usuario federado están definidos en dos lugares:

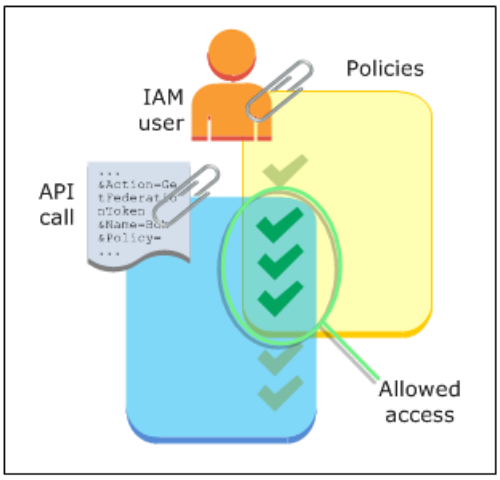
- Las políticas de sesión que se pasan como parámetro en la llamada a la API `GetFederationToken`. (Este es el caso más frecuente).
- Una política basada en recursos que nombra explícitamente al usuario federado en el elemento `Principal` de la política. (Este es el caso menos frecuente).

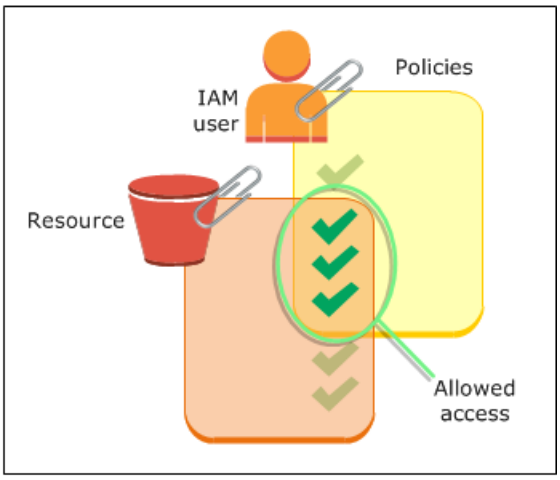
Las políticas de sesión son políticas avanzadas que se pasan como parámetros cuando se crea una sesión temporal mediante programación. Al crear una sesión de usuario federado y pasar las políticas de sesión, los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del usuario de y las políticas de la sesión. No puede utilizar la política de sesión para conceder más permisos que los permitidos por la política basada en identidades del usuario que se federa.

En la mayoría de los casos, si no transfiere una política con la llamada a la API `GetFederationToken`, las credenciales de seguridad temporales obtenidas no tendrán permisos. Sin embargo, una política basada en recursos puede proporcionar permisos adicionales para la

sesión. Puede acceder a un recurso con una política basada en recursos que especifica la sesión como el principal permitido.

Las figuras siguientes muestran una representación visual de cómo las políticas interactúan para determinar los permisos de las credenciales de seguridad temporales devueltos por una llamada a `GetFederationToken`.





Ejemplo: asignación de permisos con GetFederationToken

Puede utilizar la acción de la API `GetFederationToken` con diferentes tipos de políticas. A continuación se ofrecen algunos ejemplos.

Política asociada al usuario de IAM

En este ejemplo, tiene una aplicación cliente basada en navegador que se utiliza dos servicios web de backend. Un servicio de backend es su propio servidor de autenticación, que utiliza su propio sistema de identidad para autenticar la aplicación de cliente. El otro servicio de backend es un servicio de AWS que proporciona algunas de las funcionalidades de la aplicación cliente. Su servidor autentica la aplicación cliente y crea o recupera la política de permisos correspondiente. A continuación, llama a la API `GetFederationToken` para obtener credenciales de seguridad temporales y devuelve dichas credenciales a la aplicación cliente. Esta puede realizar solicitudes directamente al servicio de AWS con las credenciales de seguridad temporales. Esta arquitectura permite a la aplicación cliente realizar solicitudes de AWS sin integrar credenciales de AWS a largo plazo.

El servidor de autenticación llama a la API de `GetFederationToken` con las credenciales de seguridad a largo plazo de un usuario IAM llamado `token-app`. Sin embargo, las credenciales de usuario de IAM a largo plazo permanecen en el servidor y nunca se distribuyen al cliente. La política del ejemplo siguiente está asociada al usuario de IAM `token-app` y define el conjunto más

amplio de permisos que sus usuarios federados (clientes) necesitarán. Tenga en cuenta que el permiso `sts:GetFederationToken` es obligatorio para que su servicio de autenticación obtenga credenciales de seguridad temporales para los usuarios federados.

> **Note**
>
> AWS proporciona una aplicación Java de muestra para este fin; la puede descargar aquí: Token Vending Machine for Identity Registration - Sample Java Web Application.

Example Ejemplo de política asociada al usuario de IAM **`token-app`** que llama a **`GetFederationToken`**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:GetFederationToken",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "dynamodb:ListTables",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sqs:ReceiveMessage",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:ListSubscriptions",
      "Resource": "*"
    }
  ]
```

```
}
```

La política anterior concede varios permisos al usuario de IAM. Sin embargo, esta política por sí sola no concede ningún permiso al usuario federado. Si este usuario de IAM llama a `GetFederationToken` y no transfiere una política como parámetro de la llamada a la API, el usuario federado obtenido no tendrá permisos efectivos.

Política de sesión pasada como parámetro

La forma más frecuente de asegurarse de que se le asigne al usuario federado el permiso adecuado consiste en pasar políticas de sesión en la llamada a la API `GetFederationToken`. Profundizando en el ejemplo anterior, supongamos que `GetFederationToken` se llama con las credenciales del usuario de IAM `token-app`. Entonces, imagine que las siguientes políticas de sesión se transfieren como parámetro de la llamada a la API. El usuario federado resultante tiene permiso para crear una lista del contenido del bucket de Amazon S3 denominado `productionapp`. El usuario no puede realizar las acciones de Amazon S3 `GetObject`, `PutObject`, y `DeleteObject` en elementos del bucket `productionapp`.

Estos permisos se asignan al usuario federado porque son la intersección de las políticas del usuario de IAM y las políticas de la sesión que transfiere.

El usuario federado no podrá realizar acciones en Amazon SNS, Amazon SQS, Amazon DynamoDB, ni en ningún bucket de S3, excepto `productionapp`. Estas acciones se deniegan a pesar de que el usuario de IAM asociado a la llamada a `GetFederationToken` cuenta con los permisos correspondientes.

Example Ejemplo de política de sesión transferida como parámetro de la llamada a la API **`GetFederationToken`**.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::productionapp"]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
```

```
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": ["arn:aws:s3:::productionapp/*"]
    }
  ]
}
```

Políticas basadas en recursos

Algunos recursos de AWS admiten políticas basadas en recursos, y dichas políticas proporcionan otro mecanismo para conceder permisos directamente a un usuario federado. Solo algunos servicios de AWS admiten políticas basadas en recursos. Por ejemplo, Amazon S3 tiene buckets, Amazon SNS tiene temas y Amazon SQS tiene colas a las que puede asociar políticas. Para obtener una lista de todos los servicios que admiten políticas basadas en recursos, consulte Servicios de AWS que funcionan con IAM y observe la columna de políticas basadas en recursos de las tablas. Puede utilizar políticas basadas en recursos para asignar permisos directamente a un usuario federado. Para ello, debe especificar el Nombre de recurso de Amazon (ARN) del usuario federado en el elemento `Principal` de la política basada en recursos El siguiente ejemplo ilustra esto y amplía los ejemplos anteriores, utilizando un bucket de S3 denominado `productionapp`.

La política basada en recursos siguiente se asocia al bucket. Esta política del bucket permite a una usuaria federada llamada Carol obtener acceso al bucket. Cuando la política de ejemplo descrita anteriormente se asocia al usuario de IAM `token-app`, la usuaria federada Carol tiene permiso para realizar las acciones`s3:GetObject`, `s3:PutObject`, y `s3:DeleteObject` en el bucket denominado `productionapp`. Esto es válido incluso cuando no se pasa ninguna política de sesión como parámetro en la llamada a la API `GetFederationToken`. En este caso, la explicación radica en que la siguiente política basada en recursos ha concedido permisos explícitos a la usuaria federada denominada Carol.

Recuerde, solo se conceden permisos a un usuario federado cuando estos se conceden de forma explícita al usuario de IAM y al usuario federado. También se pueden conceder (dentro de la cuenta) mediante una política basada en recursos que nombre explícitamente al usuario federado en el elemento `Principal` de la política, como en el siguiente ejemplo.

Example Ejemplo de política de bucket que permite el acceso a un usuario federado

```
{
  "Version": "2012-10-17",
  "Statement": {
```

```
    "Principal": {"AWS": "arn:aws:sts::account-id:federated-user/Carol"},
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": ["arn:aws:s3:::productionapp/*"]
  }
}
```

Para obtener más información sobre cómo se evalúan las políticas, consulte Policy evaluation logic (Lógica de evaluación de las políticas).

Permisos para GetSessionToken

La ocasión principal para llamar a la operación de API `GetSessionToken` o al comando `get-session-token` de la CLI es cuando un usuario debe autenticarse con la autenticación multifactor (MFA). Se puede escribir una política que permite determinadas acciones solo cuando dichas acciones las solicita un usuario autenticado con MFA. Para transferir correctamente la comprobación de autorización MFA, el usuario debe llamar primero a `GetSessionToken` e incluir los parámetros adicionales `SerialNumber` y `TokenCode`. Si el usuario se ha autenticado correctamente con un dispositivo MFA, las credenciales devueltas por la operación de API `GetSessionToken` incluyen el contexto de MFA. Este contexto indica que el usuario se ha autenticado con MFA y está autorizado a realizar operaciones de API que requieren la autenticación MFA.

Permisos requeridos para GetSessionToken

No se requiere ningún permiso para que un usuario obtenga un token de sesión. La finalidad de la operación `GetSessionToken` es autenticar al usuario mediante MFA. No se pueden utilizar políticas para controlar las operaciones de autenticación.

Para conceder permisos para realizar la mayoría de las operaciones de AWS, es preciso agregar la acción del mismo nombre a una política. Por ejemplo, para crear un usuario, debe utilizar la operación API `CreateUser`, el comando `create-user` de la CLI o la AWS Management Console. Para realizar estas operaciones, debe disponer de una política que le permita obtener acceso a la acción `CreateUser`.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": "iam:CreateUser",
            "Resource": "*"
        }
    ]
}
```

Puede incluir la acción `GetSessionToken` en sus políticas, pero no tiene efecto en la capacidad de un usuario para realizar la operación `GetSessionToken`.

Permisos concedidos por GetSessionToken

Si se llama a `GetSessionToken` con las credenciales de un usuario de IAM, las credenciales de seguridad temporales tienen los mismos permisos que el usuario de IAM. Del mismo modo, si se llama a `GetSessionToken` con credenciales de Usuario raíz de la cuenta de AWS, las credenciales de seguridad temporales tienen permisos de usuario raíz.

> **Note**
>
> Se recomienda evitar el uso de las credenciales de usuario raíz para llamar a `GetSessionToken`. En lugar de esto, siga nuestras prácticas recomendadas y cree uno o más usuarios de IAM con los permisos que necesitan. A continuación, utilice estos usuarios de IAM para la interacción diaria con AWS.

Las credenciales temporales que obtiene si llama a `GetSessionToken` tienen las siguientes funciones y limitaciones:

- Puede utilizar las credenciales para acceder a la AWS Management Console transmitiendo las credenciales al punto de enlace de inicio de sesión único de la federación en `https://signin.aws.amazon.com/federation`. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.
- Las credenciales no se pueden utilizar para llamar a las operaciones de API de AWS STS o IAM. Puede utilizarlas para llamar a las operaciones de API de otros servicios de AWS.

Compare esta operación de API y sus limitaciones y posibilidades con las demás operaciones de API que crean credenciales de seguridad temporales en Comparación de las operaciones de la API de AWS STS.

Para obtener más información sobre el acceso mediante API protegidas por MFA utilizando `GetSessionToken`, consulte Configuración del acceso a una API protegido por MFA.

Deshabilitar permisos para credenciales de seguridad temporales

Las credenciales de seguridad temporales son válidas hasta que caducan. Estas credenciales son válidas durante el tiempo especificado, desde 900 segundos (15 minutos) hasta un máximo de 129 600 segundos (36 horas). La duración predeterminada de una sesión es de 43 200 segundos (12 horas). Puede revocar estas credenciales, pero también debe cambiar los permisos del rol a fin de detener el uso de credenciales comprometidas para actividades malintencionadas en la cuenta. Los permisos asignados a las credenciales de seguridad temporales se evalúan cada vez que se utilizan para realizar una solicitud de AWS. Una vez que haya eliminado todos los permisos de las credenciales, las solicitudes de AWS que las utilizan fallarán.

Es posible que las actualizaciones de la política tarden unos minutos en hacerse efectivas. Revoque las credenciales de seguridad temporales del rol para obligar a todos los usuarios que asuman el rol a volver a autenticarse y solicitar credenciales nuevas.

No puede cambiar los permisos para un usuario Usuario raíz de la cuenta de AWS. Del mismo modo, no puede cambiar los permisos de las credenciales de seguridad temporales que se han creado llamando a `GetFederationToken` o `GetSessionToken` al iniciar sesión como usuario raíz. Por este motivo, le recomendamos que no llame a `GetFederationToken` ni a `GetSessionToken` como usuario raíz.

> **Important**
>
> En el caso de los usuarios de IAM Identity Center, consulte Deshabilitar el acceso de los usuarios en la Guía del usuario de AWS IAM Identity Center. También puede Eliminar el acceso de los usuarios a aplicaciones en la nube o SAML 2.0 personalizadas en la consola de IAM Identity Center.

Temas

- Denegar el acceso a todas las sesiones asociadas a un rol
- Denegar el acceso a una sesión específica
- Denegar una sesión de usuario con claves de contexto de condición
- Denegar a un usuario de sesión con políticas basadas en recursos

Denegar el acceso a todas las sesiones asociadas a un rol

Utilice este enfoque cuando le preocupe el acceso sospechoso por parte de:

- Entidades principales de otra cuenta que utilizan el acceso entre cuentas
- Identidades de usuarios externos con permisos para acceder a recursos de AWS en su cuenta
- Usuarios que se han autenticado en una aplicación Web o móvil con un proveedor de OIDC

Este procedimiento deniega los permisos a todos los usuarios que cuentan con permisos para asumir un rol.

A fin de cambiar o eliminar los permisos asignados a las credenciales de seguridad temporales que se obtienen al llamar a `AssumeRole`, `AssumeRoleWithSAML`, `AssumeRoleWithWebIdentity`, `GetFederationToken` o `GetSessionToken`, puede editar o eliminar la política de permisos que define los permisos para el rol.

> **Important**
>
> Si hay una política basada en recursos que permite el acceso de una entidad principal, también debe agregar una denegación explícita para ese recurso. Para obtener más información, consulte Denegar a un usuario de sesión con políticas basadas en recursos.

1. Inicie sesión en la AWS Management Console y abra la consola de IAM.
2. En el panel de navegación, elija el nombre del rol que desea editar. Puede utilizar el cuadro de búsqueda para filtrar la lista.
3. Seleccione la política correspondiente.
4. Elija la pestaña Permisos.
5. Elija la pestaña JSON y actualice la política para denegar todos los recursos y acciones.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*"
    }
```

```
  ]
}
```

6. En la página Review (Revisar), revise el Summary (Resumen) de la política y seleccione Save changes (Guardar cambios) para guardar su trabajo.

Al editar la política, los cambios afectan a los permisos de todas las credenciales de seguridad temporales asociadas al rol, incluidas las credenciales que se han emitido antes de cambiar la política de permisos del rol. Después de actualizar la política, puede revocar las credenciales de seguridad temporales del rol para revocar de inmediato todos los permisos de las credenciales que ha emitido el rol.

Denegar el acceso a una sesión específica

Cuando actualiza los roles que puede asumir un IdP con una política de denegación total o elimina el rol por completo, todos los usuarios que tienen acceso al rol se ven afectados. Puede denegar el acceso en función del elemento `Principal` sin afectar a los permisos de todas las demás sesiones asociadas al rol.

Se pueden denegar permisos a `Principal` mediante claves de contexto de condición o políticas basadas en recursos.

> **Tip**
>
> Puede encontrar los ARN de los usuarios federados mediante los registros de AWS CloudTrail. Para obtener más información, consulte Cómo identificar con facilidad a sus usuarios federados mediante AWS CloudTrail.

Denegar una sesión de usuario con claves de contexto de condición

Puede utilizar claves de contexto de condición en situaciones en las que desea denegar el acceso a sesiones de credenciales de seguridad temporales específicas sin afectar a los permisos del rol o usuario de IAM que creó las credenciales.

Para obtener más información sobre las claves de contexto de condición, consulte Claves de contexto de condición globales de AWS.

> **Note**
>
> Si hay una política basada en recursos que permite el acceso de una entidad principal, también debe agregar una instrucción de denegación explícita en la política basada en recursos después de completar estos pasos.

Después de actualizar la política, puede revocar las credenciales de seguridad temporales del rol para revocar de inmediato todas las credenciales emitidas.

aws:PrincipalArn

Puede utilizar la clave de contexto de condición aws:PrincipalArn para denegar el acceso a un ARN de entidad principal específico. Para ello, especifique el identificador único (ID) del usuario federado, rol o usuario de IAM al que se encuentran asociadas las credenciales de seguridad temporales en el elemento Condition de una política.

1. En el panel de navegación de la consola de IAM, elija el nombre del rol que desea editar. Puede utilizar el cuadro de búsqueda para filtrar la lista.
2. Seleccione la política correspondiente.
3. Elija la pestaña Permisos.
4. Elija la pestaña JSON y agregue una instrucción de denegación para el ARN de la entidad principal, como se muestra en el siguiente ejemplo.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "ArnEquals": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
          ]
        }
      }
```

```
    }
  ]
}
```

5. En la página Review (Revisar), revise el Summary (Resumen) de la política y seleccione Save changes (Guardar cambios) para guardar su trabajo.

aws:userid

Puede utilizar la clave de contexto de condición aws:userid para denegar el acceso a todas las sesiones de credenciales de seguridad temporales, o a específicas, asociadas al rol o usuario de IAM. Para ello, especifique el identificador único (ID) del usuario federado, rol o usuario de IAM al que se encuentran asociadas las credenciales de seguridad temporales en el elemento `Condition` de una política.

En la siguiente política se muestra un ejemplo de cómo puede denegar el acceso a las sesiones de credenciales de seguridad temporales mediante la clave de contexto de condición `aws:userid`.

- `AIDAXUSER1` representa el identificador único para un usuario de IAM. Especificar el identificador único de un usuario de IAM como valor para la clave de contexto `aws:userid` denegará todas las sesiones que se encuentran asociadas a dicho usuario.
- `AR0AXR0LE1` representa el identificador único para un rol de IAM. Especificar el identificador único de un rol de IAM como valor para la clave de contexto `aws:userid` denegará todas las sesiones que se encuentran asociadas al rol.
- `AR0AXR0LE2` representa el identificador único para una sesión de rol asumido. En la parte del nombre de sesión del rol especificado por el autor de la llamada del identificador único del rol asumido, puede especificar un nombre de sesión de rol o un carácter comodín si se utiliza el operador de condición StringLike. Si especifica el nombre de la sesión del rol, denegará la sesión del rol nombrada sin afectar a los permisos del rol que creó las credenciales. Si especifica un comodín para el nombre de la sesión del rol, denegará todas las sesiones asociadas al rol.
- `account-id:<federated-user-caller-specified-name>` representa el identificador único para una sesión de usuario federado. Un usuario de IAM crea un usuario federado que llama a la API GetFederationToken. Si especifica el identificador único para un usuario federado, denegará la sesión del usuario federado nombrado sin afectar los permisos del rol que creó las credenciales.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "account-id:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Para ver ejemplos específicos de valores de clave de una entidad principal, consulte Valores clave principales. Para obtener información sobre los identificadores únicos de IAM, consulte Identificadores únicos.

Denegar a un usuario de sesión con políticas basadas en recursos

Si el ARN de la entidad principal también se encuentra incluido en alguna política basada en recursos, también debe revocar el acceso basado en los valores `principalId` o `sourceIdentity` del usuario específico en el elemento `Principal` de una política basada en recursos. Si solo actualiza la política de permisos del rol, el usuario podrá seguir realizando las acciones que se permiten en la política basada en recursos.

1. Consulte Servicios de AWS que funcionan con IAM para comprobar si el servicio admite políticas basadas en recursos.
2. Inicie sesión en la AWS Management Console y abra la consola del servicio. Cada servicio tiene una ubicación diferente en la consola para adjuntar políticas.
3. Edite la instrucción de la política para especificar la información de identificación de la credencial:
   a. En `Principal`, ingrese el ARN de la credencial que desee denegar.
   b. En `Effect`, escriba “Denegar”.

c. En `Action`, ingrese el espacio de nombres del servicio y el nombre de la acción que se denegará. Para denegar todas las acciones, utilice el carácter comodín (*). Por ejemplo: “s3:*”.

d. En `Resource`, ingrese el ARN del recurso de destino. Por ejemplo: “arn:aws:s3:::EXAMPLE-BUCKET”.

```
{
"Version": "2012-10-17",
  "Statement": {
    "Principal": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::EXAMPLE-BUCKET"
  }
}
```

4. Guarde su trabajo.

Concesión de permisos para crear credenciales de seguridad temporales

De forma predeterminada, los usuarios de IAM no tienen permiso para crear credenciales de seguridad temporales para usuarios federados y roles. Debe utilizar una política para proporcionar estos permisos a los usuarios. Aunque pueda conceder permisos directamente a un usuario, le recomendamos encarecidamente que conceda permisos a los grupos. Esto facilita en gran medida la administración de los permisos. Cuando alguien ya no tenga que realizar las tareas asociadas a los permisos, solo tiene que retirarlo del grupo. Si otra persona necesita realizar dicha tarea, añádala al grupo para concederle los permisos.

Para conceder permiso a un grupo de IAM para crear credenciales de seguridad temporales para usuarios federados o roles, asocie una política que conceda a ambos los siguientes privilegios:

- Para que los usuarios federados obtengan acceso a un rol de IAM, conceda acceso a AWS STS de `AssumeRole`.

- Para los usuarios federados que no necesiten un rol, conceda acceso a `GetFederationToken` de AWS STS.

Para obtener información sobre las diferencias entre las operaciones de API `AssumeRole` y `GetFederationToken`, consulte Solicitud de credenciales de seguridad temporales.

Los usuarios de IAM también pueden llamar a `GetSessionToken` para crear credenciales de seguridad temporales. No se requieren permisos para que un usuario llame a `GetSessionToken`. La finalidad de esta operación es autenticar al usuario mediante MFA. No se pueden utilizar políticas para controlar la autenticación. Esto significa que no se puede impedir que los usuarios de IAM llamen a `GetSessionToken` para crear credenciales temporales.

Example Ejemplo de política que concede permiso para asumir un rol

La siguiente política de ejemplo concede permiso para llamar a `AssumeRole` para el rol `UpdateApp` en la Cuenta de AWS `123123123123`. Cuando se usa `AssumeRole`, el usuario (o la aplicación) que crea las credenciales de seguridad en nombre de un usuario federado no puede delegar ningún permiso que no se haya especificado ya en la política de permisos de la función.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::123123123123:role/UpdateAPP"
  }]
}
```

Example Ejemplo de política que concede permiso para crear credenciales de seguridad temporales para un usuario federado

La siguiente política de ejemplo concede permiso para obtener acceso a `GetFederationToken`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
```

```
}
```

> **Important**
>
> Cuando dé a los usuarios de IAM permiso para crear credenciales de seguridad temporales para usuarios federados con `GetFederationToken`, debe ser consciente de que esto permite a esos usuarios delegar sus propios permisos. Para obtener más información sobre cómo delegar permisos entre usuarios de IAM y Cuentas de AWS, consulte Ejemplos de políticas para delegar el acceso. Para obtener más información sobre cómo controlar los permisos en las credenciales de seguridad temporales, consulte Control de los permisos para credenciales de seguridad temporales.

Example Ejemplo de política que concede a un usuario un permiso limitado para crear credenciales de seguridad temporales para usuarios federados

Cuando se permite que un usuario de IAM llame a `GetFederationToken`, la práctica recomendada es restringir los permisos que el usuario de IAM puede delegar. Por ejemplo, la política siguiente muestra cómo dejar que un usuario de IAM cree credenciales de seguridad temporales únicamente para usuarios federados cuyos nombres comiencen con Manager.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Administrar AWS STS en una Región de AWS

De forma predeterminada, AWS Security Token Service (AWS STS) está disponible como servicio global y todas las solicitudes de AWS STS se dirigen a un único punto de enlace en `https://sts.amazonaws.com`. AWS recomienda utilizar puntos de enlace regionales de AWS STS para reducir la latencia, compilar la redundancia y aumentar la validez del token de sesión.

- Reducir la latencia - Al realizar las llamadas a AWS STS a un punto de enlace que esté geográficamente más cerca de sus servicios y aplicaciones, puede obtener acceso a los servicios de AWS STS con una latencia menor y tiempos de respuesta mejores.
- Redundancia incorporada: puede limitar los efectos de una falla dentro de una carga de trabajo a una cantidad limitada de componentes con un alcance predecible de contención de impacto. El uso de puntos de conexión de AWS STS regionales le permite alinear el alcance de sus componentes con el alcance de sus identificadores de sesión. Para obtener más información sobre este pilar de fiabilidad, consulte Usar el aislamiento de errores para proteger su carga de trabajo en AWS Well-Architected Framework.
- Aumentar la validez del token de sesión: los tokens de sesión de puntos de enlace de AWS STS regionales son válidos en todas las Regiones de AWS. Los tokens de sesión del punto de conexión de STS global son válidos únicamente en las Regiones de AWS que están habilitadas de forma predeterminada. Si va a habilitar una nueva región en su cuenta, puede utilizar tokens de sesión de los puntos enlace de AWS STS regionales. Si decide utilizar el punto de conexión global, debe cambiar la compatibilidad de la región de tokens de sesión de AWS STS para el punto de enlace global. De esta forma, se garantiza que los tokens sean válidos en todas las Regiones de AWS.

Administración de tokens de sesión de punto de enlace global

La mayoría de las Regiones de AWS están habilitadas para llevar a cabo operaciones en todos los Servicios de AWS de forma predeterminada. Estas regiones se activan automáticamente para su uso con AWS STS. Algunas regiones, como, por ejemplo, Asia Pacífico (Hong Kong), se deben habilitar manualmente. Para obtener más información sobre cómo habilitar y deshabilitar Regiones de AWS, consulte Especificar qué Regiones de AWS puede usar su cuenta en la Guía de referencia de AWS Account Management. Cuando habilita estos regiones de AWS, se activan automáticamente para su uso con AWS STS. No puede activar el punto de conexión de AWS STS para una región que está deshabilitada. Los tokens que son válidos en todas las Regiones de AWS incluyen más caracteres que los tokens que son válidos en regiones que están habilitadas de forma predeterminada. Cambiar esta configuración podría afectar a los sistemas existentes en los que almacena tokens temporalmente.

Puede cambiar esta configuración mediante laAWS Management Console, la AWS CLI o la API de AWS.

Para cambiar la compatibilidad de la región de los tokens de sesión para el punto de enlace global (consola)

1. Inicie sesión como usuario raíz o usuario con permisos para realizar tareas de administración de IAM. Para cambiar la compatibilidad de los tokens de sesión, debe tener una política que permita la acción `iam:SetSecurityTokenServicePreferences`.
2. Abra la consola de IAM. En el panel de navegación, elija Configuración de cuenta.
3. En la sección Tokens de sesión de los puntos de conexión de STS de Security Token Service (STS). El punto de conexión global indica `Valid only in Regiones de AWS enabled by default`. Elija Change.
4. En el cuadro de diálogo Cambiar compatibilidad de región, seleccione Todas las Regiones de AWS. A continuación, elija Guardar cambios.

> **Note**
>
> Los tokens que son válidos en todas las Región de AWS incluyen más caracteres que los tokens que son válidos en regiones que están habilitadas de forma predeterminada. Cambiar esta configuración podría afectar a los sistemas existentes en los que almacena tokens temporalmente.

Para cambiar la compatibilidad de la región de los tokens de sesión para el punto de enlace global (AWS CLI)

Establezca la versión del token de sesión. Los tokens de la versión 1 son válidos únicamente en las Regiones de AWS que están disponibles de forma predeterminada. Estos tokens no funcionan en regiones habilitadas manualmente, como, por ejemplo, Asia Pacífico (Hong Kong). Los tokens de la versión 2 son válidos en todas las regiones. Sin embargo, los tokens de versión 2 incluyen más caracteres y podrían afectar a los sistemas en los que almacena tokens temporalmente.

- `aws iam set-security-token-service-preferences`

Para cambiar la compatibilidad de la región de los tokens de sesión para el punto de enlace global (API de AWS)

Establezca la versión del token de sesión. Los tokens de la versión 1 son válidos únicamente en las Regiones de AWS que están disponibles de forma predeterminada. Estos tokens no funcionan en

regiones habilitadas manualmente, como, por ejemplo, Asia Pacífico (Hong Kong). Los tokens de la versión 2 son válidos en todas las regiones. Sin embargo, los tokens de versión 2 incluyen más caracteres y podrían afectar a los sistemas en los que almacena tokens temporalmente.

- `SetSecurityTokenServicePreferences`

Activación y desactivación de AWS STS en una Región de AWS

Al activar puntos de enlace de STS para una región, AWS STS puede emitir credenciales temporales para los usuarios y roles de dicha cuenta que realizan una solicitud de AWS STS. Esas credenciales se pueden utilizar en cualquier región que esté habilitada de forma predeterminada o habilitada manualmente. En el caso de las regiones habilitadas de forma predeterminada, debe activar el punto de conexión regional de STS en la cuenta en la que se generan las credenciales temporales. No importa si el usuario ha iniciado sesión en la misma cuenta o en una cuenta diferente al realizar la solicitud. En el caso de las regiones habilitadas de forma manual, debe activar la región en la cuenta que realiza la solicitud y en la cuenta en la que se generan las credenciales temporales.

Por ejemplo, imagínese que un usuario de la cuenta A quiere enviar una solicitud de API `sts:AssumeRole` al punto de conexión regional de AWS STS `https://sts.us-east-2.amazonaws.com`. La solicitud se realiza para las credenciales temporales del rol denominado `Developer` de la cuenta B. Dado que la solicitud se realiza para crear credenciales para una entidad de la cuenta B, la cuenta B debe activar la región `us-east-2`. Los usuarios de la cuenta A (o cualquier otra cuenta) pueden llamar al punto de enlace `us-east-2` para solicitar credenciales para la cuenta B, independientemente de si la región está activada en sus cuentas.

> **Note**
>
> Las regiones activas están a disposición de todos los usuarios que utilizan las credenciales temporales de esa cuenta. Para controlar qué usuarios o roles de IAM pueden acceder a la región, utilice la clave de condición `aws:RequestedRegion` en sus políticas de permisos.

Para activar o desactivar AWS STS en una región que está activada de forma predeterminada (consola)

1. Inicie sesión como usuario raíz o usuario con permisos para realizar tareas de administración de IAM.
2. Abra la consola de IAM y en el panel de navegación elija Configuración de cuenta.

3. En la sección Puntos de conexión de Security Token Service (STS), busque la región que quiere configurar y, a continuación, elija Activa o Inactiva en la columna Estado de STS.
4. En el cuadro de diálogo que se abre, elija Activar o Desactivar.

En las regiones que deben estar activadas, activamos AWS STS automáticamente cuando se activa la región. Después de activar una región, AWS STS siempre estará activa en la región y no se podrá desactivar. Para obtener información sobre cómo habilitar regiones que están deshabilitadas de forma predeterminada, consulte Especificar qué Regiones de AWS puede usar su cuenta en la Guía de referencia de AWS Account Management.

Código de escritura para utilizar en regiones de AWS STS

Después de activar una región, podrá dirigir las llamadas a la API de AWS STS a esa región. El siguiente fragmento de código Java demuestra cómo configurar un objeto `AWSSecurityTokenService` para realizar solicitudes desde la región de Europa (Irlanda) (eu-west-1).

```
EndpointConfiguration regionEndpointConfig = new EndpointConfiguration("https://sts.eu-
west-1.amazonaws.com", "eu-west-1");
AWSSecurityTokenService stsRegionalClient =
 AWSSecurityTokenServiceClientBuilder.standard()
.withCredentials(credentials)
.withEndpointConfiguration(regionEndpointConfig)
.build();
```

AWS STS recomienda realizar llamadas a un punto de enlace regional. Para aprender cómo habilitar una región de forma manual, consulte Especificar qué Regiones de AWS puede utilizar su cuenta en la Guía de referencia de AWS Account Management.

En el ejemplo, la primera línea crea una instancia de un objeto `EndpointConfiguration` llamado `regionEndpointConfig`, que incluye la URL del punto de conexión y la Región de AWS como parámetros.

Para obtener información acerca de cómo configurar puntos de conexión regionales de AWS STS con una variable de entorno para AWS SDK, consulte Puntos de conexión regionalizados de AWS STS en la Guía de referencia de herramientas y AWS SDK.

Si desea obtener información sobre el resto de combinaciones de entornos de programación y lenguajes, consulte la documentación del correspondiente SDK.

Regiones y puntos de conexión

En la siguiente tabla se muestran las regiones y sus puntos de enlace. Indica cuáles están activadas de forma predeterminada y cuáles puede activar o desactivar.

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
--Global--	sts.amazonaws.com	 Sí	 No
Este de EE. UU. (Ohio)	sts.us-east-2.amazonaws.com	 Sí	 Sí
Este de EE. UU. (Norte de Virginia)	sts.us-east-1.amazonaws.com	 Sí	 No
Oeste de EE. UU. (Norte de California)	sts.us-west-1.amazonaws.com	 Sí	 Sí
Oeste de EE. UU. (Oregón)	sts.us-west-2.amazonaws.com	 Sí	 Sí

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
África (Ciudad del Cabo)	sts.af-south-1.amazonaws.com	 No[1]	 No
Asia-Pacífico (Hong Kong)	sts.ap-east-1.amazonaws.com	 No[1]	 No
Asia-Pacífico (Hyderabad)	sts.ap-south-2.amazonaws.com	 No[1]	 No
Asia-Pacífico (Yakarta)	sts.ap-southeast-3.amazonaws.com	 No[1]	 No
Asia-Pacífico (Melbourne)	sts.ap-southeast-4.amazonaws.com	 No[1]	 No
Asia-Pacífico (Bombay)	sts.ap-south-1.amazonaws.com	 Sí	 Sí

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
Asia-Pacífico (Osaka)	sts.ap-northeast-3.amazonaws.com	 Sí	 Sí
Asia-Pacífico (Seúl)	sts.ap-northeast-2.amazonaws.com	 Sí	 Sí
Asia-Pacífico (Singapur)	sts.ap-southeast-1.amazonaws.com	 Sí	 Sí
Asia-Pacífico (Sídney)	sts.ap-southeast-2.amazonaws.com	 Sí	 Sí
Asia-Pacífico (Tokio)	sts.ap-northeast-1.amazonaws.com	 Sí	 Sí
Canadá (centro)	sts.ca-central-1.amazonaws.com	 Sí	 Sí

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
Oeste de Canadá (Calgary)	sts.ca-west-1.amazonaws.com	 Sí	 Sí
China (Pekín)	sts.cn-north-1.amazonaws.com.cn	 Sí[2]	 No
China (Ningxia)	sts.cn-northwest-1.amazonaws.com.cn	 Sí[2]	 Sí
Europa (Fráncfort)	sts.eu-central-1.amazonaws.com	 Sí	 Sí
Europa (Irlanda)	sts.eu-west-1.amazonaws.com	 Sí	 Sí
Europa (Londres)	sts.eu-west-2.amazonaws.com	 Sí	 Sí

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
Europa (Milán)	sts.eu-south-1.amazonaws.com	 No[1]	 No
Europa (París)	sts.eu-west-3.amazonaws.com	 Sí	 Sí
Europa (España)	sts.eu-south-2.amazonaws.com	 No[1]	 No
Europa (Estocolmo)	sts.eu-north-1.amazonaws.com	 Sí	 Sí
Europa (Zúrich)	sts.eu-central-2.amazonaws.com	 No[1]	 No
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	 No[1]	 No

Nombre de la región de	Punto de conexión	Activo de forma predeterminada	Activación/desactivación manual
Medio Oriente (Baréin)	sts.me-south-1.amazonaws.com	 No[1]	 No
Medio Oriente (EAU)	sts.me-central-1.amazonaws.com	 No[1]	 No
América del Sur (São Paulo)	sts.sa-east-1.amazonaws.com	 Sí	 Sí

[1]Debe habilitar la región para utilizarla. Esto activa automáticamente AWS STS. No puede activar o desactivar manualmente AWS STS en estas regiones.

[2]Para usar AWS en China, necesita una cuenta y unas credenciales específicas para AWS en China.

AWS CloudTrail y puntos de enlace regionales

Las llamadas a puntos de conexión regionales y globales se registran en el campo `tlsDetails` en AWS CloudTrail. Las llamadas a puntos de conexión regionales, como por ejemplo `us-east-2.amazonaws.com`, se registran en CloudTrail en la región correspondiente. Las llamadas al punto de enlace global, `sts.amazonaws.com`, se registran como llamadas a un servicio global. Los eventos de los puntos de conexión globales AWS STS se registran en us-east-1.

> **Note**
>
> `tlsDetails` solo se puede ver para los servicios que admiten este campo. Consulte Servicios que admiten detalles de TLS en CloudTrail en la Guía del usuario de AWS CloudTrail

Para obtener más información, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail.

Uso de tokens al portador

Algunos servicios de AWS requieren que tenga permiso para obtener un token al portador del servicio AWS STS para poder acceder a sus recursos mediante programación. Estos servicios admiten un protocolo que requiere que utilice un token al portador en lugar de una solicitud Signature Version 4 firmada tradicional. Cuando realiza operaciones de la API de AWS CLI o AWS que requieren tokens al portador, el servicio de AWS solicita un token al portador en su nombre. El servicio le proporciona el token, que puede utilizar más adelante para realizar futuras operaciones en ese servicio.

Los tokens al portador del servicio AWS STS incluyen información original de su entidad principal que podría afectar a sus permisos. Esta información puede incluir etiquetas de entidad principal, etiquetas de sesión y políticas de sesión. El ID de clave de acceso del token comienza con el prefijo `ABIA`. Esto le ayuda a identificar las operaciones que se realizaron mediante tokens al portador del servicio en sus registros de CloudTrail.

Important

El token al portador solo se puede utilizar para hacer llamadas al servicio que lo genera y en la región en que se generó. No puede utilizar el token al portador para realizar operaciones en otros servicios o regiones.

Un ejemplo de un servicio que admite tokens al portador es AWS CodeArtifact. Para poder interactuar con AWS CodeArtifact a través de un administrador de paquetes como NPM, Maven o PIP, primero debe llamar a la operación `aws codeartifact get-authorization-token`. Esta operación devuelve un token al portador que puede utilizar para realizar operaciones de AWS CodeArtifact. De forma alternativa, puede utilizar el comando `aws codeartifact login`, que completa la misma operación y luego configura su cliente automáticamente.

Si realiza una acción en un servicio de AWS que genera un token al portador para usted, debe tener los siguientes permisos en su política de IAM:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServiceBearerToken",
            "Effect": "Allow",
            "Action": "sts:GetServiceBearerToken",
            "Resource": "*"
        }
    ]
}
```

Para ver un ejemplo de token de portador de servicio, consulte Uso de políticas basadas en identidades para AWS CodeArtifact en la Guía del usuario de AWS CodeArtifact .

Aplicaciones de ejemplo que utilizan credenciales temporales

Puede utilizar AWS Security Token Service (AWS STS) para crear credenciales de seguridad temporales que pueden controlar el acceso a sus recursos de AWS y proporcionárselas a usuarios de confianza. Para obtener más información acerca de AWS STS, consulte Credenciales de seguridad temporales en IAM. Para ver cómo puede utilizar AWS STS para administrar credenciales de seguridad temporales, puede descargar las siguientes aplicaciones de ejemplo que implementan escenarios de ejemplo completos:

- Habilitar la federación a AWS para utilizar Windows Active Directory, ADFS y SAML 2.0. Demuestra cómo delegar el acceso mediante la federación empresarial a AWS mediante Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0 y SAML (Security Assertion Markup Language) 2.0.
- Permitir el acceso del agente de identidades personalizadas a la consola de AWS. Muestra cómo crear un proxy de federación personalizada que permite el inicio de sesión único (SSO), de modo que los usuarios de Active Directory puedan iniciar sesión en AWS Management Console.
- Cómo utilizar Shibboleth para el inicio de sesión único en AWS Management Console.. En esta página se muestra cómo utilizar Shibboleth y SAML para proporcionar a los usuarios un acceso de inicio de sesión único (SSO) a la AWS Management Console.

Muestras para la federación OIDC

Las siguientes aplicaciones de muestra ilustran cómo utilizar la federación OIDC con proveedores como Inicio de sesión con Amazon, Amazon Cognito, Facebook, o Google. Puede intercambiar la

autenticación de estos proveedores por credenciales de seguridad de AWS temporales para obtener acceso a los servicios de AWS.

- Tutoriales de Amazon Cognito: le recomendamos que utilice Amazon Cognito con las SDK de AWS para el desarrollo móvil. Amazon Cognito es la forma más sencilla de administrar la identidad en aplicaciones móviles y ofrece características adicionales, como la sincronización y la identidad en todos los dispositivos. Para obtener más información acerca de Amazon Cognito, consulte Autenticación con Amplify en la documentación de Amplify.

Permitir el acceso del agente de identidades personalizadas a la consola de AWS

Puede escribir y ejecutar código para crear una dirección URL que permita a los usuarios que inicien sesión en la red de su organización obtener acceso de forma segura a la AWS Management Console. La dirección URL incluye un token de inicio de sesión que obtiene de AWS y que autentica al usuario en AWS. La sesión de la consola resultante puede incluir una función `AccessKeyId` distinta debido a la federación. Para rastrear el uso de la clave de acceso para el inicio de sesión de la federación a través de eventos de CloudTrail relacionados, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail y los eventos de inicio de sesión de AWS Management Console.

> **Note**
>
> Si su organización usa un proveedor de identidad (IdP) que es compatible con SAML, puede configurar el acceso a la consola sin necesidad de escribir código. Funciona con proveedores como Active Directory Federation Services de Microsoft o Shibboleth de código abierto. Para obtener más información, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 .

Para permitir que los usuarios de su organización tengan acceso a la AWS Management Console, puede crear un agente de identidades que realice los pasos siguientes:

1. Comprobar que el sistema de identidad local autentique al usuario.
2. Llame a las operaciones de la API AssumeRole (recomendado) o GetFederationToken de AWS Security Token Service (AWS STS) para obtener credenciales de seguridad temporales para el usuario. Para obtener más información sobre los distintos métodos que puede utilizar para asumir

un rol, consulte Uso de roles de IAM. Para obtener información sobre cómo pasar etiquetas de sesión opcionales al obtener las credenciales de seguridad, consulte Transferencia de etiquetas de sesión en AWS STS.

- Si utiliza una de las operaciones `AssumeRole*` de la API para obtener las credenciales de seguridad temporales de un rol, puede incluir el parámetro `DurationSeconds` en la llamada. Este parámetro especifica la duración de la sesión de rol, que puede oscilar entre 900 segundos (15 minutos) y el valor de la duración máxima de la sesión especificado para el rol. Cuando se utiliza `DurationSeconds` en una operación `AssumeRole*`, debe llamarla como un usuario de IAM con credenciales a largo plazo. De lo contrario, la llamada al punto de enlace de federación en el paso 3 produce un error. Para obtener información sobre cómo ver o cambiar el valor máximo para un rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.
- Si utiliza la operación `GetFederationToken` de la API para obtener las credenciales, puede incluir el parámetro `DurationSeconds` en la llamada. Este parámetro especifica la duración de la sesión de rol. Este valor puede oscilar entre 900 segundos (15 minutos) y 129 600 segundos (36 horas). Solo podrá realizar esta llamada a la API utilizando las credenciales de seguridad de AWS a largo plazo de un usuario de IAM. También puede realizar estas llamadas con las credenciales de Usuario raíz de la cuenta de AWS, pero no lo recomendamos. Si realiza esta llamada como usuario raíz, la duración predeterminada de la sesión es de una hora. También puede especificar una sesión que dure entre 900 segundos (15 minutos) y 3 600 segundos (una hora).

3. Llame al punto de enlace de federación de AWS y proporcione las credenciales de seguridad temporales para solicitar un token de inicio de sesión.
4. Representar una URL de la consola que incluya el token:
   - Si utiliza una de las operaciones `AssumeRole*` de la API en la URL, puede incluir el parámetro HTTP `SessionDuration`. Este parámetro especifica la duración de la sesión de consola, que oscila entre 900 segundos (15 minutos) y 43 200 segundos (12 horas).
   - Si utiliza la operación `GetFederationToken` de la API en la URL, puede incluir el parámetro `DurationSeconds`. Este parámetro especifica la duración de la sesión de consola federada. Este valor puede oscilar entre 900 segundos (15 minutos) y 129 600 segundos (36 horas).

> **Note**
>
> - No utilice el parámetro HTTP `SessionDuration` si obtuvo las credenciales temporales con `GetFederationToken`. Si lo hace, la operación producirá un error.

- Utilizar las credenciales de un rol para asumir otro rol se denomina encadenamiento de roles. Cuando se utiliza el encadenamiento de roles, las nuevas credenciales tienen una duración máxima de una hora. Cuando utiliza roles para conceder permisos a las aplicaciones que se ejecutan en instancias EC2, esas aplicaciones no están sujetas a esta limitación.

5. Proporcione la dirección URL al usuario o invóquela en nombre del usuario.

La dirección URL que el punto de enlace de federación proporciona es válida durante 15 minutos después de su creación. Esto difiere de la duración (en segundos) de la sesión de credenciales de seguridad temporales que está asociada a la URL. Esas credenciales son válidas durante el periodo que especificó al crearlas, a partir del momento en que se crearon.

> **Important**
>
> La dirección URL concede el acceso a sus recursos de AWS a través de la AWS Management Console si ha habilitado los permisos en las credenciales de seguridad temporales asociadas. Por este motivo, debe tratar a la dirección URL como un secreto. Recomendamos devolver la dirección URL a través de un redireccionamiento seguro, por ejemplo, mediante la utilización de un código de estado de respuesta HTTP 302 a través de una conexión SSL. Para obtener más información sobre el código de estado de respuesta HTTP 302, diríjase a RFC 2616, sección 10.3.3.

Para finalizar estas tareas, puede utilizar la API de consulta HTTPS para AWS Identity and Access Management (IAM) y AWS Security Token Service (AWS STS). O bien, puede utilizar lenguajes de programación, tales como Java, Ruby o C#, junto con el SDK de AWS apropiado. En las siguientes secciones, se describe cada uno de estos métodos.

Temas

- Código de ejemplo con operaciones de API de consulta de IAM
- Ejemplo de código que utiliza Python
- Ejemplo de código que utiliza Java
- Ejemplo que muestra cómo crear la dirección URL (Ruby)

Código de ejemplo con operaciones de API de consulta de IAM

Puede crear una dirección URL que ofrezca a los usuarios federados acceso directo a la AWS Management Console. Esta tarea usa IAM y la API de consulta HTTPS de AWS STS. Para obtener más información sobre cómo realizar solicitudes de consulta, consulte Making Query Requests.

> **Note**
>
> El siguiente procedimiento incluye ejemplos de cadenas de texto. Para mejorar la legibilidad, se han agregado saltos de línea a algunos de los ejemplos más largos. Al crear estas cadenas para su propio uso, debe omitir cualquier salto de línea.

Para proporcionar a un usuario federado acceso a los recursos de la AWS Management Console

1. Autentique al usuario en su sistema de autorización e identidad.
2. Obtenga credenciales de seguridad temporales para el usuario. Las credenciales temporales incluyen un ID de clave de acceso, una clave de acceso secreta y un token de seguridad. Para obtener más información sobre la creación de instancias de contenedor, consulte Credenciales de seguridad temporales en IAM.

   Para obtener credenciales temporales, puede llamar a la API AssumeRole de AWS STS (recomendado) o a la API GetFederationToken. Para obtener más información sobre las diferencias entre estas operaciones API, consulte Descripción de las opciones de la API para delegar de forma segura el acceso a su cuenta deAWS en el blog de seguridad de AWS.

   > **Important**
   >
   > Si utiliza la API GetFederationToken para crear credenciales de seguridad temporales, debe especificar los permisos que dichas credenciales conceden al usuario que asume el rol. Para cualquiera de las operaciones API que empiezan por `AssumeRole*`, utilice un rol de IAM para asignar permisos. Para el resto de las operaciones API, el mecanismo varía según la API. Para obtener más información, consulte Control de los permisos para credenciales de seguridad temporales. Además, si utiliza las operaciones `AssumeRole*` de la API, debe llamarlas como usuario de IAM con credenciales a largo plazo. De lo contrario, la llamada al punto de enlace de federación en el paso 3 produce un error.

3. Después de obtener las credenciales de seguridad temporales, intégrelas en una cadena de sesión JSON para intercambiarlas por un token de inicio de sesión. El siguiente ejemplo muestra cómo codificar las credenciales. Sustituya el texto del marcador de posición con los valores adecuados de las credenciales que reciba en el paso anterior.

```
{"sessionId":"*** temporary access key ID ***",
"sessionKey":"*** temporary secret access key ***",
"sessionToken":"*** session token ***"}
```

4. Aplique el código URL a la cadena de sesión del paso anterior. Dado que la información que está codificando es confidencial, recomendamos que evite utilizar un servicio web para esta codificación. En su lugar, utilice una función o característica instalada localmente en su conjunto de herramientas de desarrollo para codificar esta información de forma segura. Puede utilizar la función `urllib.quote_plus` en Python, la función `URLEncoder.encode` en Java o la función `CGI.escape` en Ruby. Consulte los ejemplos más adelante en este tema.

5.
> **Note**
>
> AWS admite solicitudes POST aquí.

Envíe su solicitud al punto de conexión de federación de AWS:

`https://region-code.signin.aws.amazon.com/federation`

Para obtener una lista de los posibles valores de *region-code*, consulte la columna Region (Región) en Puntos de conexión de inicio de sesión de AWS. Opcionalmente, puede utilizar un punto de conexión de federación de inicio de sesión de AWS predeterminado:

`https://signin.aws.amazon.com/federation`

La solicitud debe incluir los parámetros `Action` y `Session` y (opcionalmente) si ha utilizado una operación `AssumeRole*` de la API, un parámetro HTTP `SessionDuration`, tal como se muestra en el siguiente ejemplo.

```
Action = getSigninToken
SessionDuration = time in seconds
Session = *** the URL encoded JSON string created in steps 3 & 4 ***
```

> **Note**
>
> Las siguientes instrucciones de este paso solo funcionan con solicitudes GET.

El parámetro HTTP `SessionDuration` especifica la duración de la sesión de consola. Esta es distinta de la duración de las credenciales temporales especificada mediante el parámetro `DurationSeconds`. Puede especificar un valor máximo de `SessionDuration` de 43 200 (12 horas). Si falta el parámetro `SessionDuration`, el valor predeterminado de la sesión será el correspondiente a la duración de las credenciales que ha recuperado de AWS STS en el paso 2 (que tienen el valor predeterminado de una hora). Consulte la documentación de la API `AssumeRole` para obtener más información sobre cómo especificar la duración mediante el parámetro `DurationSeconds`. La posibilidad de crear una sesión de consola con una duración superior a una hora es intrínseca a la operación `getSigninToken` del punto de enlace de federación.

> **Note**
>
> - No utilice el parámetro HTTP `SessionDuration` si obtuvo las credenciales temporales con `GetFederationToken`. Si lo hace, la operación producirá un error.
> - Utilizar las credenciales de un rol para asumir otro rol se denomina encadenamiento de roles. Cuando se utiliza el encadenamiento de roles, las nuevas credenciales tienen una duración máxima de una hora. Cuando utiliza roles para conceder permisos a las aplicaciones que se ejecutan en instancias EC2, esas aplicaciones no están sujetas a esta limitación.

Cuando habilita sesiones de consola con una duración prolongada, aumenta el riesgo de exposición de credenciales. Para ayudarle a mitigar este riesgo, puede deshabilitar inmediatamente las sesiones de consola activas para cualquier rol eligiendo Revoke Sessions (Revocar sesiones) en la página de la consola de IAM Role Summary (Resumen de roles). Para obtener más información, consulte Revocación de las credenciales de seguridad temporales de un rol de IAM.

El siguiente es un ejemplo de lo que su solicitud podría parecer. Las líneas se ajustan aquí para facilitar su legibilidad, pero debe enviarla como una cadena de una sola línea.

```
https://signin.aws.amazon.com/federation
?Action=getSigninToken
&SessionDuration=1800
&Session=%7B%22sessionId%22%3A+%22ASIAJUMHIZPTOKTBMK5A%22%2C+%22sessionKey%22
%3A+%22LSD7LWI%2FL%2FN%2BgYpan5QFz0XUpc8s7HYjRsgcsrsm%22%2C+%22sessionToken%2
2%3A+%22FQoDYXdzEBQaDLbj3VWv2u50NN%2F3yyLSASwYtWhPnGPMNmzZFfZsL0Qd3vtYHw5A5dW
AjOsrkdPkghomIe3mJip5%2F0djDBbo7SmO%2FENDEiCdpsQKodTpleKA8xQq0CwFg6a69xdEBQT8
FipATnLbKoyS4b%2FebhnsTUjZZQWp0wXXqFF7gSm%2FMe2tXe0jzsdP0O12obez9lijPSdF1k2b5
PfGhiuyAR9aD5%2BubM0pY86fKex1qsytjvyTbZ9nXe6DvxVDcnCOhOGETJ7XFkSFdH0v%2FYR25C
UAhJ3nXIkIbG7Ucv9cOEpCf%2Fg23ijRgILIBQ%3D%3D%22%7D
```

La respuesta del punto de enlace de federación es un documento JSON con un valor de `SigninToken`. Tendrá un aspecto similar al siguiente ejemplo.

```
{"SigninToken":"*** the SigninToken string ***"}
```

6.

> **Note**
>
> AWS admite solicitudes POST aquí.

Por último, cree la dirección URL que los usuarios federados pueden utilizar para obtener acceso a la AWS Management Console. La dirección URL es el mismo punto de enlace URL de federación que usó en Step 5, además de los siguientes parámetros:

```
?Action = login
&Issuer = *** the form-urlencoded URL for your internal sign-in page ***
&Destination = *** the form-urlencoded URL to the desired AWS console page ***
&SigninToken = *** the value of SigninToken received in the previous step ***
```

> **Note**
>
> Las siguientes instrucciones de este paso solo funcionan con la API de GET.

El siguiente ejemplo muestra lo que la dirección URL final podría parecer. La dirección URL es válida durante 15 minutos desde el momento en que se crea. Las credenciales de seguridad

temporales y la sesión de consola integradas dentro de la dirección URL son válidas durante el periodo especificado en el parámetro HTTP `SessionDuration` al solicitarlas inicialmente.

```
https://signin.aws.amazon.com/federation
?Action=login
&Issuer=https%3A%2F%2Fexample.com
&Destination=https%3A%2F%2Fconsole.aws.amazon.com%2F
&SigninToken=VCQgs5qZZt3Q6fn8Tr5EXAMPLEmLnwB7JjUc-SHwnUUWabcRdnWsi4DBn-dvC
CZ85wrD0nmldUcZEXAMPLE-vXYH4Q__mleuF_W2BE5HYexbe9y4Of-kje53SsjNNecATfjIzpW1
WibbnH6YcYRiBoffZBGExbEXAMPLE5aiKX4THWjQKC6gg6alHu6JFrnOJoK3dtP6I9a6hi6yPgm
iOkPZMmNGmhsvVxetKzr8mx3pxhHbMEXAMPLETv1pij0rok3IyCR2YVcIjqwfWv32HU2Xlj471u
3fU6uOfUComeKiqTGX974xzJOZbdmX_t_lLrhEXAMPLEDDIisSnyHGw2xaZZqudm4mo2uTDk9Pv
9l5K0ZCqIgEXAMPLEcA6tgLPykEWGUyH6BdSC6166n4M4JkXIQgac7_7821YqixsNxZ6rsrpzwf
nQoS14O7R0eJCCJ684EXAMPLEZRdBNnuLbUYpz2Iw3vIN0tQgOujwnwydPscM9F7foaEK3jwMkg
Apeb1-6L_OB12MZhuFxx55555EXAMPLEhyETEd4ZulKPdXHkgl6T9ZkIlHz2Uy1RUTUhhUxNtSQ
nWc5xkbBoEcXqpoSIeK7yhje9Vzhd61AEXAMPLElbWeouACEMG6-Vd3dAgFYd6i5FYoyFrZLWvm
0LSG7RyYKeYN5VIzUk3YWQpyjP0RiT5KUrsUi-NEXAMPLExMOMdoODBEgKQsk-iu2ozh6r8bxwC
RNhujg
```

Ejemplo de código que utiliza Python

En los siguientes ejemplos se muestra cómo utilizar Python para crear mediante programación una dirección URL que ofrezca a los usuarios federados acceso directo a la AWS Management Console. Se incluyen dos ejemplos:

- Federación mediante solicitudes GET para AWS
- Federación mediante solicitudes POST para AWS

Ambos ejemplos usan la API AWS SDK for Python (Boto3) y AssumeRole para obtener credenciales de seguridad temporales.

Uso de solicitudes GET

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your Cuenta de AWS,
```

```
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,
# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = "?Action=getSigninToken"
request_parameters += "&SessionDuration=43200"
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
        return urllib.parse.quote_plus(s)
request_parameters += "&Session=" +
 quote_plus_function(json_string_with_temp_credentials)
request_url = "https://signin.aws.amazon.com/federation" + request_parameters
r = requests.get(request_url)
```

```
# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create URL where users can use the sign-in token to sign in to
# the console. This URL must be used within 15 minutes after the
# sign-in token was issued.
request_parameters = "?Action=login"
request_parameters += "&Issuer=Example.org"
request_parameters += "&Destination=" + quote_plus_function("https://
console.aws.amazon.com/")
request_parameters += "&SigninToken=" + signin_token["SigninToken"]
request_url = "https://signin.aws.amazon.com/federation" + request_parameters

# Send final URL to stdout
print (request_url)
```

Uso de solicitudes POST

```
import urllib, json, sys
import requests # 'pip install requests'
import boto3 # AWS SDK for Python (Boto3) 'pip install boto3'
import os
from selenium import webdriver # 'pip install selenium', 'brew install chromedriver'

# Step 1: Authenticate user in your own identity system.

# Step 2: Using the access keys for an IAM user in your ACuenta de AWS,
# call "AssumeRole" to get temporary access keys for the federated user

# Note: Calls to AWS STS AssumeRole must be signed using the access key ID
# and secret access key of an IAM user or using existing temporary credentials.
# The credentials can be in Amazon EC2 instance metadata, in environment variables,

# or in a configuration file, and will be discovered automatically by the
# client('sts') function. For more information, see the Python SDK docs:
# http://boto3.readthedocs.io/en/latest/reference/services/sts.html
# http://boto3.readthedocs.io/en/latest/reference/services/
sts.html#STS.Client.assume_role
if sys.version_info[0] < 3:
    def quote_plus_function(s):
        return urllib.quote_plus(s)
else:
    def quote_plus_function(s):
```

```
        return urllib.parse.quote_plus(s)

sts_connection = boto3.client('sts')

assumed_role_object = sts_connection.assume_role(
    RoleArn="arn:aws:iam::account-id:role/ROLE-NAME",
    RoleSessionName="AssumeRoleDemoSession",
)

# Step 3: Format resulting temporary credentials into JSON
url_credentials = {}
url_credentials['sessionId'] =
 assumed_role_object.get('Credentials').get('AccessKeyId')
url_credentials['sessionKey'] =
 assumed_role_object.get('Credentials').get('SecretAccessKey')
url_credentials['sessionToken'] =
 assumed_role_object.get('Credentials').get('SessionToken')
json_string_with_temp_credentials = json.dumps(url_credentials)

# Step 4. Make request to AWS federation endpoint to get sign-in token. Construct the
 parameter string with
# the sign-in action request, a 12-hour session duration, and the JSON document with
 temporary credentials
# as parameters.
request_parameters = {}
request_parameters['Action'] = 'getSigninToken'
request_parameters['SessionDuration'] = '43200'
request_parameters['Session'] = json_string_with_temp_credentials

request_url = "https://signin.aws.amazon.com/federation"
r = requests.post( request_url, data=request_parameters)

# Returns a JSON document with a single element named SigninToken.
signin_token = json.loads(r.text)

# Step 5: Create a POST request where users can use the sign-in token to sign in to
# the console. The POST request must be made within 15 minutes after the
# sign-in token was issued.
request_parameters = {}
request_parameters['Action'] = 'login'
request_parameters['Issuer']='Example.org'
request_parameters['Destination'] = 'https://console.aws.amazon.com/'
request_parameters['SigninToken'] =signin_token['SigninToken']
```

```
jsrequest = '''
var form = document.createElement('form');
form.method = 'POST';
form.action = '{request_url}';
request_parameters = {request_parameters}
for (var param in request_parameters) {{
    if (request_parameters.hasOwnProperty(param)) {{
        const hiddenField = document.createElement('input');
        hiddenField.type = 'hidden';
        hiddenField.name = param;
        hiddenField.value = request_parameters[param];
        form.appendChild(hiddenField);
    }}
}}
document.body.appendChild(form);
form.submit();
'''.format(request_url=request_url, request_parameters=request_parameters)

driver = webdriver.Chrome()
driver.execute_script(jsrequest);
```

Ejemplo de código que utiliza Java

El siguiente ejemplo muestra cómo utilizar Java para crear de forma programada una dirección URL que ofrezca a los usuarios federados acceso directo a la AWS Management Console. El siguiente fragmento de código usa AWS SDK para Java.

```
import java.net.URLEncoder;
import java.net.URL;
import java.net.URLConnection;
import java.io.BufferedReader;
import java.io.InputStreamReader;
// Available at http://www.json.org/java/index.html
import org.json.JSONObject;
import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClient;
import com.amazonaws.services.securitytoken.model.Credentials;
import com.amazonaws.services.securitytoken.model.GetFederationTokenRequest;
import com.amazonaws.services.securitytoken.model.GetFederationTokenResult;


/* Calls to AWS STS API operations must be signed using the access key ID
```

```
   and secret access key of an IAM user or using existing temporary
   credentials. The credentials should not be embedded in code. For
   this example, the code looks for the credentials in a
   standard configuration file.
*/
AWSCredentials credentials =
  new PropertiesCredentials(
         AwsConsoleApp.class.getResourceAsStream("AwsCredentials.properties"));

AWSSecurityTokenServiceClient stsClient =
  new AWSSecurityTokenServiceClient(credentials);

GetFederationTokenRequest getFederationTokenRequest =
  new GetFederationTokenRequest();
getFederationTokenRequest.setDurationSeconds(1800);
getFederationTokenRequest.setName("UserName");

// A sample policy for accessing Amazon Simple Notification Service (Amazon SNS) in the
 console.

String policy = "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Action\":\"sns:*\"," +
  "\"Effect\":\"Allow\",\"Resource\":\"*\"}]}";

getFederationTokenRequest.setPolicy(policy);

GetFederationTokenResult federationTokenResult =
  stsClient.getFederationToken(getFederationTokenRequest);

Credentials federatedCredentials = federationTokenResult.getCredentials();

// The issuer parameter specifies your internal sign-in
// page, for example https://mysignin.internal.mycompany.com/.
// The console parameter specifies the URL to the destination console of the
// AWS Management Console. This example goes to Amazon SNS.
// The signin parameter is the URL to send the request to.

String issuerURL = "https://mysignin.internal.mycompany.com/";
String consoleURL = "https://console.aws.amazon.com/sns";
String signInURL = "https://signin.aws.amazon.com/federation";

// Create the sign-in token using temporary credentials,
// including the access key ID,  secret access key, and session token.
String sessionJson = String.format(
  "{\"%1$s\":\"%2$s\",\"%3$s\":\"%4$s\",\"%5$s\":\"%6$s\"}",
```

```
  "sessionId", federatedCredentials.getAccessKeyId(),
  "sessionKey", federatedCredentials.getSecretAccessKey(),
  "sessionToken", federatedCredentials.getSessionToken());

// Construct the sign-in request with the request sign-in token action, a
// 12-hour console session duration, and the JSON document with temporary
// credentials as parameters.

String getSigninTokenURL = signInURL +
                           "?Action=getSigninToken" +
                           "&DurationSeconds=43200" +
                           "&SessionType=json&Session=" +
                           URLEncoder.encode(sessionJson,"UTF-8");

URL url = new URL(getSigninTokenURL);

// Send the request to the AWS federation endpoint to get the sign-in token
URLConnection conn = url.openConnection ();

BufferedReader bufferReader = new BufferedReader(new
  InputStreamReader(conn.getInputStream()));
String returnContent = bufferReader.readLine();

String signinToken = new JSONObject(returnContent).getString("SigninToken");

String signinTokenParameter = "&SigninToken=" + URLEncoder.encode(signinToken,"UTF-8");

// The issuer parameter is optional, but recommended. Use it to direct users
// to your sign-in page when their session expires.

String issuerParameter = "&Issuer=" + URLEncoder.encode(issuerURL, "UTF-8");

// Finally, present the completed URL for the AWS console session to the user

String destinationParameter = "&Destination=" + URLEncoder.encode(consoleURL,"UTF-8");
String loginURL = signInURL + "?Action=login" +
                     signinTokenParameter + issuerParameter + destinationParameter;
```

Ejemplo que muestra cómo crear la dirección URL (Ruby)

El siguiente ejemplo muestra cómo utilizar Ruby para crear de forma programada una dirección URL que ofrezca a los usuarios federados acceso directo a la AWS Management Console. Este fragmento de código usa AWS SDK para Ruby.

```
require 'rubygems'
require 'json'
require 'open-uri'
require 'cgi'
require 'aws-sdk'

# Create a new STS instance
#
# Note: Calls to AWS STS API operations must be signed using an access key ID
# and secret access key. The credentials can be in EC2 instance metadata
# or in environment variables and will be automatically discovered by
# the default credentials provider in the AWS Ruby SDK.
sts = Aws::STS::Client.new()

# The following call creates a temporary session that returns
# temporary security credentials and a session token.
# The policy grants permissions to work
# in the AWS SNS console.

session = sts.get_federation_token({
  duration_seconds: 1800,
  name: "UserName",
  policy: "{\"Version\":\"2012-10-17\",\"Statement\":{\"Effect\":\"Allow\",\"Action\":
\"sns:*\",\"Resource\":\"*\"}}",
})

# The issuer value is the URL where users are directed (such as
# to your internal sign-in page) when their session expires.
#
# The console value specifies the URL to the destination console.
# This example goes to the Amazon SNS console.
#
# The sign-in value is the URL of the AWS STS federation endpoint.
issuer_url = "https://mysignin.internal.mycompany.com/"
console_url = "https://console.aws.amazon.com/sns"
signin_url = "https://signin.aws.amazon.com/federation"

# Create a block of JSON that contains the temporary credentials
# (including the access key ID, secret access key, and session token).
session_json = {
  :sessionId => session.credentials[:access_key_id],
  :sessionKey => session.credentials[:secret_access_key],
  :sessionToken => session.credentials[:session_token]
```

```
}.to_json

# Call the federation endpoint, passing the parameters
# created earlier and the session information as a JSON block.
# The request returns a sign-in token that's valid for 15 minutes.
# Signing in to the console with the token creates a session
# that is valid for 12 hours.
get_signin_token_url = signin_url +
                       "?Action=getSigninToken" +
                       "&SessionType=json&Session=" +
                       CGI.escape(session_json)

returned_content = URI.parse(get_signin_token_url).read

# Extract the sign-in token from the information returned
# by the federation endpoint.
signin_token = JSON.parse(returned_content)['SigninToken']
signin_token_param = "&SigninToken=" + CGI.escape(signin_token)

# Create the URL to give to the user, which includes the
# sign-in token and the URL of the console to open.
# The "issuer" parameter is optional but recommended.
issuer_param = "&Issuer=" + CGI.escape(issuer_url)
destination_param = "&Destination=" + CGI.escape(console_url)
login_url = signin_url + "?Action=login" + signin_token_param +
  issuer_param + destination_param
```

Recursos adicionales para las credenciales de seguridad temporales

Los escenarios y aplicaciones siguientes pueden orientarlo sobre el uso de credenciales de seguridad temporales:

- Cómo integrar AWS STS SourceIdentity con su proveedor de identidades. Esta publicación muestra cómo configurar el atributo AWS STS `SourceIdentity` al usar Okta, Ping o OneLogin como su IdP.
- Federación OIDC. En esta sección se explica cómo configurar roles de IAM cuando utiliza las federaciones OIDC y la API `AssumeRoleWithWebIdentity`.
- Configuración del acceso a una API protegido por MFA. En este tema se explica cómo utilizar roles para exigir una autenticación multifactor (MFA) para proteger acciones de API confidenciales en su cuenta.

Para obtener más información sobre las políticas y permisos de AWS, consulte los temas siguientes:

- Recursos de AWS para administración de acceso
- Lógica de evaluación de políticas.
- Administración de los permisos de acceso a los recursos de Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.
- Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Etiquetado de recursos de IAM

Una etiqueta es un atributo personalizado que puede asignar a un recurso de AWS. Cada etiqueta tiene dos partes:

- Una clave de etiqueta (por ejemplo, `CostCenter`, `Environment`, `Project` o `Purpose`).
- Un campo opcional que se denomina valor de etiqueta (por ejemplo, `111122223333` o `Production` o el nombre de un equipo). Omitir el valor de etiqueta es lo mismo que utilizar una cadena vacía.

En conjunto, se conocen como pares clave-valor. Para ver los límites de la cantidad de etiquetas que puede tener en los recursos de IAM, consulte IAM y cuotas de AWS STS.

> **Note**
>
> Para obtener más información sobre la distinción entre mayúsculas y minúsculas en las claves de etiqueta y los valores de las claves de etiqueta, consulte Case sensitivity.

Las etiquetas le ayudan a identificar y organizar los recursos de AWS. Muchos servicios de AWS admiten el etiquetado, por lo que puede asignar la misma etiqueta a los recursos de diferentes servicios para indicar que los recursos están relacionados. Por ejemplo, puede asignar la misma etiqueta a un rol de IAM que se asigna a un bucket de Amazon S3. Para obtener más información sobre estrategias de etiquetado, consulte la Guía del usuario del etiquetado de recursos de AWS.

Además de utilizar etiquetas para identificar, organizar y realizar el seguimiento de sus recursos de IAM, puede utilizarlas en las políticas de IAM para ayudar a controlar quién puede ver e interactuar

con el recurso. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Elija una convención de nomenclatura de etiquetas de AWS

Cuando comience a asociar etiquetas a sus recursos de IAM, elija la convención de nomenclatura de etiquetas con cuidado. Aplique la misma convención para todas las etiquetas de AWS. Esto es especialmente importante si utiliza etiquetas en las políticas para controlar el acceso a recursos de AWS. Si ya utiliza etiquetas en AWS, revise la convención de nomenclatura y ajústela según corresponda.

> **Note**
>
> Si su cuenta es miembro de AWS Organizations, consulte Políticas de etiquetas en la guía del usuario de Organizaciones para obtener más información sobre el uso de etiquetas en Organizaciones.

Prácticas recomendadas para la denominación de etiquetas

Estas son algunas prácticas recomendadas y convenciones de nomenclatura para las etiquetas.

Asegúrese de que los nombres de las etiquetas se utilicen de forma coherente. Por ejemplo, las etiquetas `CostCenter` y `costcenter` son diferentes, por lo que una podría configurarse como etiqueta de asignación de costos para análisis e informes financieros y la otra podría no serlo. Del mismo modo, la etiqueta `Name` aparece en la consola de AWS para muchos recursos, pero no así la etiqueta `name`. Para obtener más información sobre la distinción entre mayúsculas y minúsculas en las claves de etiqueta y los valores de las claves de etiqueta, consulte Case sensitivity.

Varias etiquetas están predefinidas por AWS o son creadas automáticamente por varios servicios de AWS. Muchos de los nombres de las etiquetas definidas por AWS utilizan todas minúsculas, con guiones que separan las palabras del nombre y prefijos para identificar el servicio de origen de la etiqueta. Por ejemplo:

- `aws:ec2spot:fleet-request-id` identifica la solicitud de instancia puntual de Amazon EC2 que lanzó la instancia.

- `aws:cloudformation:stack-name` identifica la pila de AWS CloudFormation que creó el recurso.
- `elasticbeanstalk:environment-name` identifica la aplicación que creó el recurso.

Considere la posibilidad de asignar nombres a las etiquetas que usen todas minúsculas, con guiones separando las palabras y un prefijo que identifique el nombre de la organización o el nombre abreviado. Por ejemplo, para una compañía ficticia llamada AnyCompany, puede definir etiquetas como:

- `anycompany:cost-center` para identificar el código interno del centro de costes
- `anycompany:environment-type` para identificar si el entorno es de desarrollo, prueba o producción
- `anycompany:application-id` para identificar la aplicación para la que se creó el recurso

El prefijo garantiza que las etiquetas estén claramente identificadas como definidas por su organización y no por AWS o por una herramienta de terceros que usted pueda estar utilizando. Usar todas minúsculas con guiones para los separadores evita confusiones sobre cómo poner en mayúsculas el nombre de una etiqueta. Por ejemplo, `anycompany:project-id` es más fácil de recordar que `ANYCOMPANY:ProjectID`, `anycompany:projectID` o bien `Anycompany:ProjectId`.

Reglas para etiquetar en IAM y AWS STS

Una serie de convenciones rigen la creación y aplicación de etiquetas en IAM y AWS STS.

Asignación de nombres a etiquetas

Observe las siguientes convenciones al formular una convención de nomenclatura de etiquetas para recursos de IAM, sesiones de asumir rol de AWS STS y sesiones de usuarios federados de AWS STS:

Requisitos de caracteres: las claves y los valores de las etiquetas pueden incluir cualquier combinación de letras, números, espacios y los símbolos _ . : / = + - @.

Distinción de mayúsculas y minúsculas: la distinción de mayúsculas y minúsculas para las claves de etiqueta varía en función del tipo de recurso de IAM que se etiqueta. Los valores de clave de etiqueta para usuarios y roles de IAM no distinguen mayúsculas y minúsculas, pero se conservan.

Esto significa que no puede haber claves de etiqueta **`Department`** y **`department`** separadas. Si ha etiquetado un usuario con la etiqueta **`Department=finance`** y añade la etiqueta **`department=hr`**, sustituye la primera etiqueta. No se ha añadido una segunda etiqueta.

Para otros tipos de recursos de IAM, los valores de clave de etiqueta distinguen mayúsculas y minúsculas. Eso significa que puede tener claves de etiquetas **`Costcenter`** y **`costcenter`** distintas. Por ejemplo, si ha etiquetado una política administrada por el cliente con la etiqueta **`Costcenter = 1234`** y agrega la etiqueta **`costcenter = 5678`**, la política tendrá ambas claves de etiqueta, **`Costcenter`** y **`costcenter`**.

Como práctica recomendada, sugerimos que evite el uso de etiquetas similares con distinción de minúsculas y mayúsculas inconsistente. Le recomendamos que decida una estrategia de uso de mayúsculas y minúsculas en las etiquetas e implemente esa estrategia sistemáticamente en todos los tipos de recursos. Para obtener más información sobre las prácticas recomendadas para el etiquetado, consulte Etiquetado de recursos de AWS en Referencia general de AWS.

Las siguientes listas muestran las diferencias entre mayúsculas y minúsculas para las claves de etiqueta asociadas a los recursos de IAM.

Los valores de clave de etiqueta no distinguen mayúsculas y minúsculas:

- Roles de IAM
- Usuarios de IAM

Los valores de claves de etiqueta distinguen entre mayúsculas y minúsculas:

- Políticas administradas por el cliente
- Perfiles de instancias
- Proveedores de identidad de OpenID Connect
- Proveedores de identidad SAML
- Certificados de servidor
- Dispositivos MFA virtuales

Además, se aplican las siguientes reglas:

- No puede crear una clave o valor de etiqueta que comience con el texto **`aws:`**. Este prefijo de etiqueta se reserva para uso interno de AWS.

- Puede crear una etiqueta con un valor vacío como **`phoneNumber =`** . No se puede crear una clave de etiqueta vacía.
- No puede especificar varios valores en una etiqueta única, pero puede crear una estructura personalizada con varios valores en el único valor. Por ejemplo, supongamos que el usuario Zhang trabaja en el equipo de ingeniería y el equipo de control de calidad. Si asocia la etiqueta **`team = Engineering`** y, a continuación, asocia la etiqueta **`team = QA`**, cambie el valor de la etiqueta de **`Engineering`** a **`QA`**. En su lugar, puede incluir varios valores en una única etiqueta con un separador personalizado. En este ejemplo, puede asociar la etiqueta de **`team = Engineering:QA`** a Zhang.

> **Note**
>
> Para controlar el acceso a los ingenieros, en este ejemplo se utilizará la etiqueta **`team`**, debe crear una política que permita que cada configuración pueda incluir **`Engineering`**, incluida **`Engineering:QA`**. Para obtener más información sobre el uso de etiquetas en políticas, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

Aplicación y edición de etiquetas

Tenga en cuenta las siguientes convenciones al asociar etiquetas a los recursos de IAM:

- Puede etiquetar la mayoría de los recursos de IAM, pero no grupos, roles asumidos, informes de acceso, dispositivos basados en hardware o dispositivos MFA.
- No puede utilizar Tag Editor para etiquetar recursos de IAM. Tag Editor no es compatible con etiquetas de IAM. Para obtener más información acerca de cómo utilizar Tag Editor con otros servicios, consulte Working with Tag Editor (Uso de Tag Editor) en la Guía del usuario de AWS Resource Groups.
- Para etiquetar un recurso de IAM, debe tener permisos específicos. Para etiquetar o desetiquetar recursos, también debe tener permiso para enumerar etiquetas. Para obtener más información, consulte la lista de temas de cada recurso de IAM al final de esta página.
- El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.
- Puede aplicar la misma etiqueta a varios recursos de IAM. Por ejemplo, suponga que tiene un departamento denominado `AWS_Development` con 12 miembros. Puede tener 12 usuarios y

un rol con la clave de etiqueta **`department`** y un valor de **`awsDevelopment`** (**`department = awsDevelopment`**). También puede utilizar la misma etiqueta en recursos de otros servicios que admiten etiquetado.

- Las entidades de IAM (usuarios o roles) no pueden tener varias instancias de la misma clave de etiqueta. Por ejemplo, si tiene un usuario con el par de clave-valor de etiqueta **`costCenter = 1234`**, puede asociar el par de clave-valor de etiqueta a **`costCenter = 5678`**. IAM actualiza el valor de la etiqueta **`costCenter`** a **`5678`**.
- Para editar una etiqueta que se asocia a una entidad de IAM (usuario o rol), asocie una etiqueta a un nuevo valor para sobrescribir la etiqueta existente. Por ejemplo, supongamos que tiene un usuario con el par de clave-valor de etiqueta **`department = Engineering`**. Si necesita trasladar el usuario al departamento de control de calidad, puede asociar el par de clave-valor de etiqueta **`department = QA`** al usuario. El resultado es el valor **`Engineering`** de la clave de etiqueta **`department`** que se va a sustituir por el valor **`QA`**.

Temas

- Etiquetado de usuarios de IAM
- Etiquetado de un rol de IAM
- Etiquetado de políticas administradas por el cliente
- Etiquetado de proveedores de identidad de IAM
- Etiquetado de perfiles de instancia para roles de Amazon EC2
- Etiquetado de certificados de servidor
- Etiquetado de dispositivos MFA virtuales
- Transferencia de etiquetas de sesión en AWS STS

Etiquetado de usuarios de IAM

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a un usuario de IAM. Por ejemplo, para añadir información de ubicación a un usuario, puede agregar la clave de etiqueta **`location`** y el valor de etiqueta **`us_wa_seattle`**. O bien puede utilizar tres pares de clave-valor de ubicación independientes: **`loc-country = us`**, **`loc-state = wa`** y **`loc-city = seattle`**. Puede utilizar etiquetas para controlar el acceso de los usuarios a los recursos o para controlar qué etiquetas se pueden asociar a un usuario. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar usuarios de IAM

Debe configurar permisos para permitir que un usuario de IAM pueda etiquetar a otros usuarios. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Para permitir a un usuario de IAM agregar, enumerar o eliminar una etiqueta para un usuario específico

Agregue la siguiente instrucción a la política de permisos del usuario de IAM que necesita administrar etiquetas. Utilice el número de cuenta y reemplace *<username>* por el nombre del usuario que se debe administrar. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir a un usuario de IAM administrar las etiquetas

Añada la siguiente instrucción a la política de permisos de los usuarios para permitir a los usuarios administrar sus propias etiquetas. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::user/${aws:username}"
}
```

Para permitir a un usuario de IAM agregar una etiqueta a un usuario específico

Agregue la siguiente instrucción a la política de permisos del usuario de IAM que necesite agregar, pero no eliminar, etiquetas para un usuario específico.

> **Note**
>
> La acción `iam:TagUser` requiere que también incluya la acción `iam:ListUserTags`.

Para utilizar esta política, reemplace *<username>* por el nombre del usuario cuyas etiquetas se deben administrar. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administración de etiquetas en usuarios de IAM (consola)

Puede administrar etiquetas para usuarios de IAM desde la AWS Management Console.

Para administrar etiquetas en usuarios (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Roles (Roles) y, a continuación, elija el nombre del usuario que desea editar.
3. Elija la pestaña Tags (Etiquetas) y, a continuación, realice una de las siguientes acciones:
   - Elija Agregar etiqueta nueva si el usuario aún no tiene etiquetas.
   - Elija Manage tags (Administrar etiquetas) para administrar el conjunto de etiquetas existente.
4. Añada o elimine etiquetas para completar el conjunto de etiquetas. A continuación, elija Save changes (Guardar cambios).

Administrar etiquetas en usuarios de IAM (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para los usuarios de IAM. También puede utilizar la AWS CLI o la API de AWS para administrar etiquetas de usuarios de IAM.

Para obtener una lista de las etiquetas asociadas actualmente a un usuario de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam list-user-tags
- API de AWS: ListUserTags

Para asociar etiquetas a un usuario de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-user
- API de AWS: TagUser

Para eliminar etiquetas de un usuario de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-user
- API de AWS: UntagUser

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de un rol de IAM

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados atributo un rol de IAM. Por ejemplo, para agregar información de ubicación a un rol, puede agregar la clave de etiqueta **`location`** y el valor de etiqueta **`us_wa_seattle`**. O bien puede utilizar tres pares de clave-valor de ubicación independientes: **`loc-country = us`**, **`loc-state = wa`** y **`loc-city = seattle`**. Puede utilizar etiquetas para controlar el acceso de un rol a los recursos o para controlar qué etiquetas se pueden asociar a un rol. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar roles de IAM

Debe configurar permisos para permitir que una entidad de IAM (usuario o rol) pueda etiquetar otras entidades (usuarios o roles). Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListRoleTags`
- `iam:TagRole`
- `iam:UntagRole`
- `iam:ListUserTags`
- `iam:TagUser`
- `iam:UntagUser`

Para permitir que un rol de IAM agregue, enumere o elimine una etiqueta para un usuario específico

Agregue la siguiente instrucción a la política de permisos del rol de IAM cuyas etiquetas deben ser administradas. Utilice el número de cuenta y reemplace *`<username>`* por el nombre del usuario que se debe administrar. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser",
        "iam:UntagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir a un rol de IAM agregar una etiqueta a un usuario específico

Agregue la siguiente instrucción a la política de permisos del rol de IAM que necesite agregar, pero no eliminar, etiquetas para un usuario específico.

> **Note**
>
> La acción `iam:TagRole` requiere que también incluya la acción `iam:ListRoleTags`.

Para utilizar esta política, reemplace *<username>* por el nombre del usuario cuyas etiquetas se deben administrar. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListUserTags",
        "iam:TagUser"
    ],
    "Resource": "arn:aws:iam::<account-number>:user/<username>"
}
```

Para permitir a un rol de IAM agregar, enumerar o eliminar una etiqueta para un rol específico

Agregue la siguiente instrucción a la política de permisos del rol de IAM cuyas etiquetas deben ser administradas. Reemplace *<rolename>* por el nombre del rol que debe administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListRoleTags",
        "iam:TagRole",
        "iam:UntagRole"
    ],
    "Resource": "arn:aws:iam::<account-number>:role/<rolename>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en roles de IAM (consola)

Puede administrar etiquetas para roles de IAM desde la AWS Management Console.

Para administrar etiquetas en roles (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Roles (Roles) y, a continuación, elija el nombre de la entidad que desea editar.
3. Elija la pestaña Tags (Etiquetas) y, a continuación, realice una de las siguientes acciones:
   - Elija Add new tags (Agregar nuevas etiquetas) si el rol aún no tiene.
   - Elija Manage tags (Administrar etiquetas) para administrar el conjunto de etiquetas existente.
4. Añada o elimine etiquetas para completar el conjunto de etiquetas. A continuación, elija Save changes (Guardar cambios).

Administrar etiquetas en roles de IAM (AWS CLI o API de AWS)

Puede enumerar, asociar o eliminar etiquetas para usuarios y roles de IAM. También puede utilizar la AWS CLI o la API de AWS para administrar etiquetas de roles de IAM.

Para obtener una lista de las etiquetas asociadas actualmente a un rol de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam list-role-tags

- API de AWS: ListRoleTags

Para asociar etiquetas a un rol de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-role
- API de AWS: TagRole

Para eliminar etiquetas de un rol de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-role
- API de AWS: UntagRole

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de políticas administradas por el cliente

Puede utilizar pares clave-valor de etiqueta de IAM para agregar atributos personalizados a las políticas administradas por el cliente. Por ejemplo, para etiquetar una política con información del departamento, puede agregar la clave de etiqueta **`Department`** y el valor de la etiqueta **`eng`**. O bien, es posible que desee etiquetar políticas para indicar que son para un entorno específico, como por ejemplo **`Environment = lab`**. Puede utilizar etiquetas para controlar el acceso a los recursos o para controlar qué etiquetas se pueden asociar a un recurso. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar políticas administradas por el cliente

Debe configurar permisos para permitir que una entidad de IAM (usuarios o roles) etiquete políticas administradas por el cliente. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListPolicyTags`
- `iam:TagPolicy`
- `iam:UntagPolicy`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para una política administrada por el cliente

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice su número de cuenta y reemplace *<policyname>* por el nombre de la política cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy",
        "iam:UntagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a una política específica administrada por el cliente

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para una política específica.

> **Note**
>
> La acción `iam:TagPolicy` requiere que también incluya la acción `iam:ListPolicyTags`.

Para utilizar esta política, reemplace *<policyname>* por el nombre de la política cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListPolicyTags",
        "iam:TagPolicy"
    ],
    "Resource": "arn:aws:iam::<account-number>:policy/<policyname>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administración de etiquetas en políticas de IAM administradas por el cliente (consola)

Puede administrar etiquetas para políticas de IAM administradas por el cliente desde la AWS Management Console.

Para administrar etiquetas en políticas administradas por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Policies (Políticas) y, a continuación, elija el nombre de la política administrada por el cliente que desea editar.
3. Seleccione la pestaña Etiquetas y, a continuación, Administrar etiquetas.
4. Añada o elimine etiquetas para completar el conjunto de etiquetas. A continuación, elija Save changes (Guardar cambios).

Administrar etiquetas en políticas de IAM administradas por el cliente (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para políticas de IAM administradas por el cliente. Puede utilizar la AWS CLI o la API de AWS para administrar etiquetas para políticas de IAM administradas por el cliente.

Para enumerar las etiquetas actualmente asociadas a una política de IAM administrada por el cliente (AWS CLI o API de AWS)

- AWS CLI: aws iam list-policy-tags

- API de AWS: ListPolicyTags

Para asociar etiquetas a una política de IAM administrada por el cliente (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-policy
- API de AWS: TagPolicy

Para quitar etiquetas de una política de IAM administrada por el cliente (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-policy
- API de AWS: UntagPolicy

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de proveedores de identidad de IAM

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a proveedores de identidad (IDP) de IAM.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Para obtener información sobre cómo etiquetar a los proveedores de identidad en IAM, consulte las siguientes secciones:

Temas

- Etiquetado de proveedores de identidad OpenID Connect (OIDC)
- Etiquetado de proveedores de identidad SAML de IAM

Etiquetado de proveedores de identidad OpenID Connect (OIDC)

Puede utilizar la etiqueta clave-valores de IAM para agregar atributos personalizados a proveedores de identidad OpenID Connect (OIDC) de IAM. Por ejemplo, para identificar un proveedor de

identidades OIDC, puede agregar la clave de etiqueta **`google`** y el valor de la etiqueta **`oidc`**. Puede utilizar etiquetas para controlar el acceso a los recursos o para controlar qué etiquetas se pueden asociar a un objeto. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

Permisos necesarios para etiquetar proveedores de identidad OIDC de IAM

Debe configurar permisos para permitir que una entidad de IAM (usuario o rol) pueda etiquetar proveedores de identidad OIDC de IAM. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListOpenIDConnectProviderTags`
- `iam:TagOpenIDConnectProvider`
- `iam:UntagOpenIDConnectProvider`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para un proveedor de identidades OIDC de IAM

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice su número de cuenta y reemplace *<OIDCProviderName>* por el nombre del proveedor OIDC cuyas etiquetas deben ser administradas. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider",
        "iam:UntagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a un proveedor de identidad OIDC de IAM específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para un proveedor de identidad específico.

> **Note**
>
> La acción `iam:TagOpenIDConnectProvider` requiere que también incluya la acción `iam:ListOpenIDConnectProviderTags`.

Para utilizar esta política, reemplace *<OIDCProviderName>* por el nombre del proveedor OIDC cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListOpenIDConnectProviderTags",
        "iam:TagOpenIDConnectProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:oidc-provider/<OIDCProviderName>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en proveedores de identidad OIDC de IAM (consola)

Puede administrar etiquetas para proveedores de identidad OIDC de IAM desde la AWS Management Console.

> **Note**
>
> Solo puede administrar etiquetas a través de la nueva experiencia de consola de proveedores de identidad.

Para administrar etiquetas en proveedores de identidad OIDC (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola, elija Identity providers (Proveedores de identidad) y, a continuación, elija el nombre del proveedor de identidad que desea editar.

3. En la sección Tags (Etiquetas), elija Manage tags (Administrar etiquetas) y, a continuación, complete una de las siguientes acciones:
   - Elija Add tag (Agregar etiqueta) si el proveedor de identidad OIDC aún no tiene etiquetas o para agregar una nueva etiqueta.
   - Edite las claves y los valores de etiqueta existentes.
   - Elija Remove tag (Eliminar etiqueta) para eliminar una etiqueta.
4. A continuación, elija Save changes (Guardar cambios).

Administrar etiquetas en proveedores de identidad OIDC de IAM (AWS CLI o API de AWS)

Puede enumerar, adjuntar o quitar etiquetas para proveedores de identidad OIDC de IAM. Puede utilizar la AWS CLI o la API de AWS para administrar etiquetas para proveedores de identidad OIDC de IAM.

Para enumerar las etiquetas actualmente asociadas a un proveedor de identidad OIDC de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam list-open id-connect-provider-tags
- API de AWS: ListOpenIDConnectProviderTags

Para asociar etiquetas a un proveedor de identidad OIDC de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-open-id-connect-provider
- API de AWS: TagOpenIDConnectProvider

Para quitar etiquetas de un proveedor de identidad OIDC de IAM (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-open-id-connect-provider
- API de AWS: UntagOpenIDConnectProvider

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de proveedores de identidad SAML de IAM

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a proveedores de identidad SAML. Por ejemplo, para identificar un proveedor, puede agregar la clave de etiqueta **`okta`** y el valor de la etiqueta **`saml`**. Puede utilizar etiquetas para controlar el acceso a los recursos o para controlar qué etiquetas se pueden asociar a un objeto. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

Permisos necesarios para etiquetar proveedores de identidad SAML

Debe configurar permisos para permitir que una entidad de IAM (usuarios o roles) pueda etiquetar proveedores de identidad (IdP) basados en SAML 2.0. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListSAMLProviderTags`
- `iam:TagSAMLProvider`
- `iam:UntagSAMLProvider`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para un proveedor de identidad SAML

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice su número de cuenta y reemplace *<SAMLProviderName>* por el nombre del proveedor SAML cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider",
        "iam:UntagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a un proveedor de identidad SAML específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para un usuario específico.

> **Note**
>
> La acción `iam:TagSAMLProvider` requiere que también incluya la acción `iam:ListSAMLProviderTags`.

Para utilizar esta política, reemplace *<SAMLProviderName>* por el nombre del proveedor SAML cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListSAMLProviderTags",
        "iam:TagSAMLProvider"
    ],
    "Resource": "arn:aws:iam::<account-number>:saml-provider/<SAMLProviderName>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en proveedores de identidad SAML de IAM (consola)

Puede administrar etiquetas para proveedores de identidad SAML de IAM desde la AWS Management Console.

> **Note**
>
> Solo puede administrar etiquetas a través de la nueva experiencia de consola de proveedores de identidad.

Para administrar etiquetas en proveedores de identidad SAML (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Identity providers (Proveedores de identidad) y, a continuación, elija el nombre del proveedor de identidad SAML que desea actualizar.
3. En la sección Tags (Etiquetas), elija Manage tags (Administrar etiquetas) y, a continuación, complete una de las siguientes acciones:
   - Elija Add tag (Agregar etiqueta) si el proveedor de identidad SAML aún no tiene etiquetas o para agregar una nueva etiqueta.
   - Edite las claves y los valores de etiqueta existentes.
   - Elija Remove tag (Eliminar etiqueta) para eliminar una etiqueta.
4. Añada o elimine etiquetas para completar el conjunto de etiquetas. A continuación, elija Save changes (Guardar cambios).

Administrar etiquetas en proveedores de identidad SAML de IAM (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para proveedores de identidad SAML de IAM. Puede utilizar la AWS CLI o la API de AWS para administrar etiquetas para proveedores de identidad SAML de IAM.

Para enumerar las etiquetas actualmente asociadas a un proveedor de identidad SAML (AWS CLI o API de AWS)

- AWS CLI: aws iam list-saml-provider-tags
- API de AWS: ListSamlProviderTags

Para adjuntar etiquetas a un proveedor de identidades SAML (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-saml-provider
- API de AWS: TagSamlProvider

Para quitar etiquetas de un proveedor de identidad SAML (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-saml-provider

- API de AWS: UntagsAMLProvider

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de perfiles de instancia para roles de Amazon EC2

Cuando lanza una instancia de Amazon EC2, usted especifica un rol de IAM que se asocia a la instancia. Un perfil de instancia es un contenedor de una función de IAM que se puede utilizar para transferir información de la función a una instancia Amazon EC2 cuando la instancia se inicia. Puede etiquetar perfiles de instancia cuando utiliza la AWS CLI o la API de AWS.

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a un perfil de instancias. Por ejemplo, para agregar información de departamento a un perfil de instancia, puede agregar la clave de etiqueta **`access-team`** y el valor de etiqueta **`eng`**. Al hacer esto, las entidades principales con etiquetas coincidentes tienen acceso a perfiles de instancia con la misma etiqueta. Puede utilizar varios pares clave-valor de etiquetas para especificar un equipo y un proyecto: **`access-team = eng`** y **`project = peg`**. Puede utilizar etiquetas para controlar el acceso de los usuarios a los recursos o para controlar qué etiquetas se pueden asociar a un usuario. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar perfiles de instancia

Debe configurar permisos para permitir que una entidad de IAM (usuario o rol) pueda etiquetar otros perfiles de instancia. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListInstanceProfileTags`
- `iam:TagInstanceProfile`
- `iam:UntagInstanceProfile`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para un perfil de instancias

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice su número de cuenta y reemplace *<InstanceProfileName>* por el nombre del perfil de instancia cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
        "iam:TagInstanceProfile",
        "iam:UntagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a un perfil de instancias específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para un perfil de instancias específico.

> **Note**
>
> La acción `iam:TagInstanceProfile` requiere que también incluya la acción `iam:ListInstanceProfileTags`.

Para utilizar esta política, reemplace *<InstanceProfileName>* por el nombre del perfil de instancia cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListInstanceProfileTags",
```

```
        "iam:TagInstanceProfile"
    ],
    "Resource": "arn:aws:iam::<account-number>:instance-profile/<InstanceProfileName>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en perfiles de instancia (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para perfiles de instancia. También puede utilizar la AWS CLI o la API de AWS para administrar etiquetas de perfiles de instancia.

Para obtener una lista de las etiquetas asociadas actualmente a un perfil de instancia (AWS CLI o API de AWS)

- AWS CLI: aws iam list-instance-profile-tags
- API de AWS: ListInstanceProfileTags

Para asociar etiquetas a un perfil de instancia (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-instance-profile
- API de AWS: TagInstanceProfile

Para quitar etiquetas de un perfil de instancia (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-instance-profile
- API de AWS: UntagInstanceProfile

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de certificados de servidor

Si utiliza IAM para administrar certificados SSL/TLS, puede etiquetar certificados de servidor de IAM mediante la AWS CLI o la API de AWS. Si se trata de una región compatible con AWS Certificate

Manager (ACM), le recomendamos que utilice ACM en vez de IAM para aprovisionar, administrar e implementar los certificados de servidor. En las regiones no compatibles, debe utilizar IAM como Certificate Manager. Para saber qué regiones admite , consulte puntos finales y cuotas de AWS Certificate Manager en la Referencia general de AWS.

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a un certificado de servidor. Por ejemplo, para agregar información sobre el propietario o administrador de un certificado de servidor, agregue la clave de etiqueta **`owner`** y el valor de la etiqueta **`net-eng`**. También puede especificar un centro de costo al agregar la clave de etiqueta **`CostCenter`** y el valor de la etiqueta **`1234`**. Puede utilizar etiquetas para controlar el acceso a los recursos o para controlar qué etiquetas se pueden asociar a recursos. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar certificados de servidor

Debe configurar permisos para permitir que una entidad de IAM (usuario o rol) pueda etiquetar certificados de servidor. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListServerCertificateTags`
- `iam:TagServerCertificate`
- `iam:UntagServerCertificate`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para un certificado de servidor

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice su número de cuenta y reemplace *`<CertificateName>`* por el nombre del certificado del servidor cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
```

```
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate",
        "iam:UntagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a un certificado de servidor específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para un certificado de servidor específico.

> **Note**
>
> La acción `iam:TagServerCertificate` requiere que también incluya la acción `iam:ListServerCertificateTags`.

Para utilizar esta política, reemplace *<CertificateName>* por el nombre del certificado de servidor cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListServerCertificateTags",
        "iam:TagServerCertificate"
    ],
    "Resource": "arn:aws:iam::<account-number>:server-certificate/<CertificateName>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en certificados de servidor (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para certificados de servidor. También puede utilizar la AWS CLI o la API de AWS para administrar etiquetas de certificados de servidor.

Para enumerar las etiquetas actualmente asociadas a un certificado de servidor (AWS CLI o API de AWS)

- AWS CLI: aws iam list-server-certificate-tags
- API de AWS: ListServerCertificateTags

Para asociar etiquetas a un certificado de servidor (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-server-certificate
- API de AWS: TagServerCertificate

Para quitar etiquetas de un certificado de servidor (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-server-certificate
- API de AWS: UntagServerCertificate

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Etiquetado de dispositivos MFA virtuales

Puede utilizar pares clave-valor de etiquetas de IAM para agregar atributos personalizados a un dispositivo MFA virtual. Por ejemplo, para agregar información del centro de costo para el dispositivo MFA virtual de un usuario, puede agregar la clave de etiqueta **`CostCenter`** y el valor de la etiqueta **`1234`**. Puede utilizar etiquetas para controlar el acceso a los recursos o para controlar qué etiquetas se pueden asociar a un objeto. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

También puede utilizar etiquetas en AWS STS para añadir atributos personalizados cuando asuma un rol o federe un usuario. Para obtener más información, consulte Transferencia de etiquetas de sesión en AWS STS.

Permisos necesarios para etiquetar dispositivos MFA virtuales

Debe configurar permisos para permitir que una entidad de IAM (usuario o rol) pueda etiquetar otros dispositivos MFA virtuales. Puede especificar una o todas las acciones de etiqueta de IAM siguientes en una política de IAM:

- `iam:ListMFADeviceTags`
- `iam:TagMFADevice`
- `iam:UntagMFADevice`

Para permitir que una entidad de IAM (usuario o rol) agregue, enumere o elimine una etiqueta para un dispositivo MFA virtual

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesita administrar etiquetas. Utilice el número de cuenta y reemplace *<MFATokenID>* por el nombre del dispositivo MFA virtual cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called "Creación de políticas mediante el editor JSON".

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice",
        "iam:UntagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

Para permitir que una entidad de IAM (usuario o rol) agregue una etiqueta a un dispositivo MFA virtual específico

Agregue la siguiente instrucción a la política de permisos de la entidad de IAM que necesite agregar, pero no eliminar, etiquetas para un dispositivo MFA específico.

> **Note**
>
> La acción `iam:TagMFADevice` requiere que también incluya la acción `iam:ListMFADeviceTags`.

Para utilizar esta política, reemplace *<MFATokenID>* por el nombre del dispositivo MFA virtual cuyas etiquetas deben administrarse. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:ListMFADeviceTags",
        "iam:TagMFADevice"
    ],
    "Resource": "arn:aws:iam::<account-number>:mfa/<MFATokenID>"
}
```

También puede utilizar una política administrada de AWS, como IAMFullAccess para proporcionar acceso completo a IAM.

Administrar etiquetas en dispositivos MFA virtuales (AWS CLI o API de AWS)

Puede enumerar, asociar o quitar etiquetas para un dispositivo MFA virtual. Puede utilizar la AWS CLI o la API de AWS para administrar etiquetas de un dispositivo MFA virtual.

Para enumerar las etiquetas actualmente asociadas a un dispositivo MFA virtual (AWS CLI o API de AWS)

- AWS CLI: aws iam list-mfa-device-tags
- API de AWS: ListMFadeviceTags

Para asociar etiquetas a un dispositivo MFA virtual (AWS CLI o API de AWS)

- AWS CLI: aws iam tag-mfa-device
- API de AWS: TagmFadevice

Para quitar etiquetas de un dispositivo MFA virtual (AWS CLI o API de AWS)

- AWS CLI: aws iam untag-mfa-device
- API de AWS: UntagmFadevice

Para obtener información acerca de cómo asociar etiquetas a los recursos de otros servicios de AWS, consulte la documentación de dichos servicios.

Para obtener información sobre el uso de etiquetas para establecer permisos pormenorizados con políticas de permisos de IAM, consulte Elementos de la política de IAM: variables y etiquetas.

Transferencia de etiquetas de sesión en AWS STS

Las etiquetas de sesión son atributos de par clave-valor que se pasan al asumir un rol de IAM o federar un usuario en AWS STS. Para ello, realice una solicitud a la AWS CLI o API AWS a través de AWS STS o a través de su proveedor de identidad (IdP). Cuando se utiliza AWS STS para solicitar credenciales de seguridad temporales, se genera una sesión. Las sesiones caducan y tienen credenciales, como un par de claves de acceso y un token de sesión. Cuando utiliza las credenciales de sesión para realizar una solicitud posterior, el contexto de solicitud incluye la `aws:PrincipalTag` clave de contexto. Puede utilizar la clave `aws:PrincipalTag` del elemento `Condition` de sus políticas para permitir o denegar el acceso basándose en esas etiquetas.

Cuando utiliza credenciales temporales para realizar una solicitud, la entidad principal puede incluir un conjunto de etiquetas. Estas etiquetas provienen de las siguientes fuentes:

1. Etiquetas de sesión - Estas etiquetas se pasaron cuando asumió el rol o se federó al usuario mediante la AWS CLI o API de AWS. Para obtener más información sobre estas operaciones, consulte Operaciones de etiquetado de sesiones.
2. Etiquetas de sesión transitiva entrantes - Estas etiquetas se heredaron de una sesión anterior en una cadena de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión más adelante en este tema.
3. Etiquetas de IAM — Las etiquetas adjuntas a su rol asumido de IAM.

Temas

- Operaciones de etiquetado de sesiones
- Cosas que debe saber sobre las etiquetas de sesión
- Permisos necesarios para agregar etiquetas de sesión
- Traspaso de etiquetas de sesión mediante AssumeRole
- Traspaso de etiquetas de sesión mediante AssumeRoleWithSAML
- Traspaso de etiquetas de sesión mediante AssumeRoleWithWebIdentity
- Traspaso de etiquetas de sesión mediante GetFederationToken

- Encadenamiento de roles con etiquetas de sesión
- Uso de etiquetas de sesión para ABAC
- Ver las etiquetas de sesión en CloudTrail

Operaciones de etiquetado de sesiones

Puede pasar etiquetas de sesión utilizando las siguientes operaciones de las API de AWS CLI o AWS en AWS STS. La AWS Management Console tiene una función Cambiar rol que no permite pasar etiquetas de sesión.

También puede establecer las etiquetas de sesión como transitivas. Las etiquetas transitivas persisten durante el encadenamiento de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.

Comparación de métodos para pasar etiquetas de sesión

Operación	¿Quién puede asumir el rol?	Método para pasar etiquetas	Método para establecer etiquetas transitivas
Operación `assume-role` de la CLI u operación `AssumeRole` de la API	Usuario o sesión de IAM	Parámetro `Tags` de la API u opción `--tags` de CLI	Parámetro `TransitiveTagKeys` de la API u opción `--transitive-tag-keys` de CLI
Operación `assume-role-with-saml` de la CLI u operación `AssumeRoleWithSAML` de la API	Cualquier usuario autenticado con un proveedor de identidad de SAML	Atributo `PrincipalTag` de SAML	Atributo `TransitiveTagKeys` de SAML

Operación	¿Quién puede asumir el rol?	Método para pasar etiquetas	Método para establecer etiquetas transitivas
Operación `assume-role-with-web-identity` de la CLI u operación `AssumeRoleWithWebIdentity` de la API	Cualquier usuario autenticado con OIDC	`PrincipalTag` Símbolo OIDC	`TransitiveTagKeys` Símbolo OIDC
Operación `get-federation-token` de la CLI u operación `GetFederationToken` de la API	Usuario raíz o usuario de IAM	Parámetro `Tags` de la API u opción `--tags` de CLI	No admitido

Las operaciones que admiten el etiquetado de sesiones pueden fallar bajo alguna de las condiciones siguientes:

- Pasa más de 50 etiquetas de sesión.
- El texto sin formato de las claves de etiqueta de sesión supera los 128 caracteres.
- El texto sin formato de los valores de las etiquetas de sesión supera los 256 caracteres.
- El tamaño total del texto sin formato de las políticas de sesión supera los 2048 caracteres.
- El tamaño total del paquete de las políticas de sesión y las etiquetas combinadas es demasiado grande. Si la operación falla, el mensaje de error indica, por porcentaje, qué tan cerca están las políticas y etiquetas combinadas al límite de tamaño superior.

Cosas que debe saber sobre las etiquetas de sesión

Antes de utilizar las etiquetas de sesión, revise los siguientes detalles sobre las sesiones y las etiquetas.

- Cuando se utilizan etiquetas de sesión, las políticas de confianza para todos los roles conectados al proveedor de identidades (IdP) que pasa etiquetas deben tener el permiso `sts:TagSession`. En el caso de los roles que no tienen este permiso en la política de confianza, la operación `AssumeRole` fallará.
- Cuando solicita una sesión, puede especificar etiquetas principales como etiquetas de sesión. Las etiquetas se aplican a las solicitudes que realice con las credenciales de la sesión.
- Las etiquetas de sesión usan pares clave/valor. Por ejemplo, para agregar información de contacto a una sesión, puede agregar la clave de etiqueta de sesión `email` y el valor de etiqueta `johndoe@example.com`.
- Las etiquetas de sesión deben seguir las reglas para asignar nombres a las etiquetas en IAM y AWS STS. Este tema incluye información sobre la distinción entre mayúsculas y minúsculas y los prefijos restringidos que se aplican a las etiquetas de sesión.
- Las nuevas etiquetas de sesión anulan las etiquetas de usuario federado o de rol asumido existentes con la misma clave de etiqueta, independientemente de las mayúsculas y minúsculas.
- No puede pasar las etiquetas de sesión con la AWS Management Console.
- Las etiquetas de sesión son válidas solo para la sesión actual.
- Las etiquetas de sesión admiten el encadenamiento de roles. De forma predeterminada, AWS STS no pasa etiquetas a sesiones de rol posteriores. Sin embargo, puede establecer las etiquetas de sesión como transitivas. Las etiquetas transitivas persisten durante el encadenamiento de roles y reemplazan la coincidencia de los valores `ResourceTag` después de la evaluación de la política de confianza de rol. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.
- Puede utilizar etiquetas de sesión para controlar el acceso a los recursos o para controlar qué etiquetas se pueden pasar a una sesión posterior. Para obtener más información, consulte Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC.
- Puede ver las etiquetas principales de la sesión, incluidas sus etiquetas de sesión, en los registros de AWS CloudTrail. Para obtener más información, consulte Ver las etiquetas de sesión en CloudTrail.
- Debe pasar un solo valor para cada etiqueta de sesión. AWS STS no admite etiquetas de sesión de varios valores.

- Puede pasar un máximo de 50 etiquetas de sesión. El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.
- Una conversión de AWS comprime las políticas de sesión pasadas y las etiquetas de sesión combinadas en un formato binario empaquetado con un límite separado. Si supera este límite, el mensaje de error de la API de AWS CLI o AWS indica, por porcentaje, qué tan cerca están las políticas y etiquetas combinadas del límite de tamaño superior.

Permisos necesarios para agregar etiquetas de sesión

Además de la acción que coincide con la operación de la API, debe tener la siguiente acción de solo permisos en la política:

```
sts:TagSession
```

> **Important**
>
> Cuando se utilizan etiquetas de sesión, las políticas de confianza de roles para todos los roles conectados a un proveedor de identidades (IdP) deben tener el permiso `sts:TagSession`. La operación `AssumeRole` producirá un error para cualquier rol conectado a un proveedor de identidad que pase etiquetas de sesión sin este permiso. Si no desea actualizar la política de confianza de rol para cada rol, puede utilizar una instancia de proveedor de identidades independiente para pasar las etiquetas de sesión. A continuación, agregue el permiso `sts:TagSession` solo a los roles que están conectados al proveedor de identidades independiente.

Puede utilizar la acción `sts:TagSession` con las siguientes claves de condición.

- `aws:PrincipalTag` – Compara la etiqueta asociada a la entidad principal que realiza la solicitud con la etiqueta que especifique en la política. Por ejemplo, puede permitir que una entidad principal pase las etiquetas de sesión solo si la entidad principal que realiza la solicitud tiene las etiquetas especificadas.
- `aws:RequestTag` – Compara el par clave-valor de etiqueta que se transfirió en la solicitud con el par de etiquetas especificado en la política. Por ejemplo, puede permitir que la entidad principal pase las etiquetas de sesión especificadas, pero solo con los valores especificados.

- `aws:ResourceTag` – Compara el par clave-valor de etiqueta que especifique en la política con el par clave-valor asociado al recurso. Por ejemplo, puede permitir que la entidad principal pase las etiquetas de sesión solo si el rol que asume incluye las etiquetas especificadas.
- `aws:TagKeys` – Compara las claves de etiqueta de una solicitud con las claves que especifique en la política. Por ejemplo, puede permitir que la entidad principal pase solo las etiquetas de sesión con las claves de etiqueta especificadas. Esta clave de condición limita el conjunto máximo de etiquetas de sesión que se pueden pasar.
- `sts:TransitiveTagKeys` - Compara las claves de etiqueta de sesión transitiva de la solicitud con las especificadas en la política. Por ejemplo, puede escribir una política para permitir que una entidad principal establezca solo etiquetas específicas como transitivas. Las etiquetas transitivas persisten durante el encadenamiento de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.

Por ejemplo, la siguiente política de confianza de rol permite al usuario `test-session-tags` asumir el rol con la política asociada. Cuando ese usuario asume el rol, debe utilizar la API de AWS CLI o AWS para pasar las tres etiquetas de sesión requeridas y el ID externorequerido. Además, el usuario puede elegir establecer las etiquetas `Project` y `Department` como transitivas.

Example Ejemplo de política de confianza de rol para etiquetas de sesión

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowIamUserAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
              "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*",
                    "aws:RequestTag/Department": "*"
                },
                "StringEquals": {"sts:ExternalId": "Example987"}
            }
        },
        {
            "Sid": "AllowPassSessionTagsAndTransitive",
            "Effect": "Allow",
```

```
            "Action": "sts:TagSession",
            "Principal": {"AWS": "arn:aws:iam::123456789012:user/test-session-tags"},
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/Project": "*",
                    "aws:RequestTag/CostCenter": "*"
                },
                "StringEquals": {
                    "aws:RequestTag/Department": [
                        "Engineering",
                        "Marketing"
                    ]
                },
                "ForAllValues:StringEquals": {
                    "sts:TransitiveTagKeys": [
                        "Project",
                        "Department"
                    ]
                }
            }
        }
    ]
}
```

¿Qué hace esta política?

- La instrucción `AllowIamUserAssumeRole` permite al usuario `test-session-tags` asumir el rol con la política asociada. Cuando ese usuario asume el rol, debe pasar las etiquetas de sesión requeridas y el ID externo.
  - El primer bloque de condición de esta instrucción requiere que el usuario pase las etiquetas de sesión `Project`, `CostCenter` y `Department`. Los valores de etiqueta no importan en esta instrucción, por lo que puede utilizar comodines (*) para los valores de etiqueta. Este bloque garantiza que el usuario pase al menos estas tres etiquetas de sesión. De lo contrario, la operación no se llevará a cabo correctamente. El usuario puede pasar etiquetas adicionales.
  - El segundo bloque de condición requiere que el usuario pase un ID externo con el valor `Example987`.
- La instrucción `AllowPassSessionTagsAndTransitive` permite la acción `sts:TagSession` de solo permisos. Esta acción debe permitirse antes de que el usuario pueda pasar las etiquetas de sesión. Si la política incluye la primera instrucción sin la segunda sentencia, el usuario no puede asumir el rol.

- El primer bloque de condición de esta sentencia permite al usuario pasar cualquier valor para las etiquetas de sesión `CostCenter` y `Project`. Para ello, se utilizan comodines (*) para el valor de etiqueta en la política, lo que requiere que utilice el operador de condición StringLike .
- El segundo bloque de condición permite al usuario pasar solo el valor `Engineering` o `Marketing` para la etiqueta de sesión `Department`.
- El tercer bloque de condición enumera el conjunto máximo de etiquetas que se puede establecer como transitivo. El usuario puede elegir establecer un subconjunto o ninguna etiqueta como transitivo. No pueden establecer etiquetas adicionales como transitivas. Puede requerir que establezcan al menos una de las etiquetas como transitiva agregando otro bloque de condición que incluya `"Null":{"sts:TransitiveTagKeys":"false"}`.

Traspaso de etiquetas de sesión mediante AssumeRole

La operación `AssumeRole` devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Puede utilizar credenciales de usuario de IAM o credenciales de rol para llamar a `AssumeRole`. Para pasar etiquetas de sesión mientras asume un rol, utilice la opción `--tags` de AWS CLI o el parámetro `Tags` de la API de AWS.

Para definir las etiquetas como transitivas, utilice la opción `--transitive-tag-keys` de AWS CLI o el parámetro `TransitiveTagKeys` de la API de AWS. Las etiquetas transitivas persisten durante el encadenamiento de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.

En el ejemplo siguiente se muestra una solicitud de ejemplo que utiliza `AssumeRole`. En este ejemplo, cuando se asume el rol `my-role-example`, se crea una sesión denominada `my-session`. Agregue los pares clave-valor de etiqueta de sesión `Project` = `Automation`, `CostCenter` = `12345` y `Department` = `Engineering`. También puede establecer las etiquetas `Project` y `Department` como transitivas especificando sus claves.

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/my-role-example \
--role-session-name my-session \
--tags Key=Project,Value=Automation Key=CostCenter,Value=12345
 Key=Department,Value=Engineering \
--transitive-tag-keys Project Department \
--external-id Example987
```

Traspaso de etiquetas de sesión mediante AssumeRoleWithSAML

La operación `AssumeRoleWithSAML` se autentica mediante la federación basada en SAML. Esta operación devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Para obtener más información acerca del uso de la federación basada en SAML para el acceso a la AWS Management Console, consulte Concesión de acceso a la AWS Management Console a los usuarios federados SAML 2.0 . Para obtener información detallada sobre el acceso a la API de AWS CLI o AWS, consulte Federación SAML 2.0. Para obtener un tutorial sobre cómo configurar la federación de SAML para los usuarios de Active Directory, consulte Federated Authentication with Active Directory Federation Services (ADFS) de AWS en el blog de seguridad de AWS.

Como administrador, puede permitir que los miembros del directorio de su empresa se federen en AWS mediante la operación AWS STS `AssumeRoleWithSAML`. Para ello, debe completar las siguientes tareas:

1. Configurar la red como proveedor SAML para AWS
2. Crear un proveedor SAML en IAM
3. Configurar un rol y sus permisos en AWS para los usuarios federados
4. Finalizar la configuración del proveedor de identidad SAML y crear aserciones para la respuesta de autenticación SAML

AWS incluye proveedores de identidad que han certificado la experiencia integral para etiquetas de sesión con sus soluciones de identidad. Para obtener información sobre cómo utilizar estos proveedores de identidad para configurar etiquetas de sesión, consulte Integración de proveedores de soluciones SAML externos con AWS.

Para pasar los atributos de SAML como etiquetas de sesión, incluya el elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/PrincipalTag:`*`{TagKey}`*. Utilice el elemento `AttributeValue` para especificar el valor de la etiqueta. Incluya un elemento `Attribute` separado para cada etiqueta de sesión.

Por ejemplo, suponga que desea pasar los siguientes atributos de identidad como etiquetas de sesión:

- `Project:Automation`
- `CostCenter:12345`

- `Department:Engineering`

Para pasar estos atributos, incluya los siguientes elementos en su aserción de SAML.

Example Ejemplo de fragmento de una aserción de SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Project">
  <AttributeValue>Automation</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:CostCenter">
  <AttributeValue>12345</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:Department">
  <AttributeValue>Engineering</AttributeValue>
</Attribute>
```

Para establecer las etiquetas anteriores como transitivas, incluya otro elemento `Attribute` con el atributo `Name` establecido en `https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys`. Las etiquetas transitivas persisten durante el encadenamiento de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.

Para establecer las etiquetas `Project` y `Department` como transitivas, utilice el siguiente atributo multivalor:

Example Ejemplo de fragmento de una aserción de SAML

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/TransitiveTagKeys">
  <AttributeValue>Project</AttributeValue>
  <AttributeValue>Department</AttributeValue>
</Attribute>
```

Traspaso de etiquetas de sesión mediante AssumeRoleWithWebIdentity

Utilice la federación de OpenID Connect(OIDC)-compliant para autenticar la operación `AssumeRoleWithWebIdentity`. Esta operación devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Para obtener más información acerca del uso de la federación de identidades web para el acceso a la AWS Management Console, consulte Federación OIDC.

Para pasar las etiquetas de sesión desde OpenID Connect (OIDC), debe incluir las etiquetas de sesión en JSON Web Token (JWT). Incluya etiquetas de sesión en el espacio para el

nombre en el token de `https://aws.amazon.com/ tags` cuando envíe la solicitud `AssumeRoleWithWebIdentity`. Para obtener más información sobre los tokens y las notificaciones de OIDC, consulte Uso de tokens con grupos de usuarios en la Guía para desarrolladores de Amazon Cognito.

Por ejemplo, el siguiente JWT descodificado es un token que se utiliza para llamar a `AssumeRoleWithWebIdentity` con las etiquetas de sesión `Project`, `CostCenter`, y `Department`. El token también establece las etiquetas `Project` y `CostCenter` como transitivas. Las etiquetas transitivas persisten durante el encadenamiento de roles. Para obtener más información, consulte Encadenamiento de roles con etiquetas de sesión.

Example Ejemplo de token web JSON decodificado

```
{
    "sub": "johndoe",
    "aud": "ac_oic_client",
    "jti": "ZYUCeRMQVtqHypVPWAN3VB",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "Project": ["Automation"],
            "CostCenter": ["987654"],
            "Department": ["Engineering"]
        },
        "transitive_tag_keys": [
            "Project",
            "CostCenter"
        ]
    }
}
```

Traspaso de etiquetas de sesión mediante GetFederationToken

`GetFederationToken` le permite federar a su usuario. Esta operación devuelve un conjunto de credenciales temporales que puede utilizar para tener acceso a los recursos de AWS. Para agregar etiquetas a la sesión de usuario federado, utilice la opción `--tags` de AWS CLI o el parámetro `Tags` de la API de AWS. No se pueden establecer las etiquetas de sesión como transitivas cuando se

utiliza `GetFederationToken`, porque no puede utilizar las credenciales temporales para asumir un rol. No se puede utilizar el encadenamiento de roles en este caso.

El ejemplo siguiente es una respuesta de ejemplo utilizando `GetFederationToken`. En este ejemplo, cuando se solicita el token, se crea una sesión denominada `my-fed-user`. Agregue los pares clave-valor de etiqueta de sesión `Project` = `Automation` y `Department` = `Engineering`.

Example Ejemplo de solicitud de la CLI de GetFederationToken

```
aws sts get-federation-token \
--name my-fed-user \
--tags key=Project,value=Automation key=Department,value=Engineering
```

Cuando utiliza las credenciales temporales devueltas por la operación `GetFederationToken`, las etiquetas principales de la sesión incluyen las etiquetas del usuario y las etiquetas de sesión pasadas.

Encadenamiento de roles con etiquetas de sesión

Puede asumir un rol y, a continuación, utilizar las credenciales temporales para asumir otro rol. Puede continuar de una sesión a otra. Esto se llama encadenamiento de roles. Cuando pasa las etiquetas de sesión mientras asume un rol, puede establecer las claves como transitivas. Esto garantiza que esas etiquetas de sesión pasen a sesiones posteriores en una cadena de roles. No se pueden establecer etiquetas de rol como transitivas. Para pasar estas etiquetas a sesiones posteriores, especifíquelas como etiquetas de sesión.

> **Note**
>
> Las etiquetas transitivas persisten durante el encadenamiento de roles y reemplazan la coincidencia de los valores `ResourceTag` después de la evaluación de la política de confianza de rol.

El siguiente ejemplo muestra cómo AWS STS pasa las etiquetas de sesión, las etiquetas transitivas y las etiquetas de rol a sesiones posteriores en una cadena de roles.

En este escenario de encadenamiento de roles de ejemplo, utilice las claves de acceso de un usuario de IAM en el AWS CLI para asumir un rol denominado `Role1`. A continuación, utilice las credenciales de sesión resultantes para asumir un segundo rol denominado `Role2`. A continuación, puede utilizar las credenciales de la segunda sesión para asumir un tercer rol denominado `Role3`.

Estas solicitudes se producen como tres operaciones separadas. Cada rol ya está etiquetado en IAM. Y durante cada solicitud, se pasan etiquetas de sesión adicionales.

Al encadenar roles, puede asegurarse de que las etiquetas de una sesión anterior persisten en las sesiones posteriores. Para ello mediante el comando `assume-role` de CLI, debe pasar la etiqueta como una etiqueta de sesión y establecer la etiqueta como transitiva. Pase la etiqueta `Star = 1` como una etiqueta de sesión. El comando también adjunta la etiqueta `Heart = 1` al rol y se aplica como etiqueta principal cuando se utiliza la sesión. Sin embargo, también quiere que la etiqueta `Heart = 1` pase automáticamente a la segunda o tercera sesión. Para ello, debe incluirla manualmente como una etiqueta de sesión. Las etiquetas principales de sesión resultantes incluyen ambas etiquetas y las establece como transitivas.

Realice esta solicitud mediante el siguiente comando de AWS CLI:

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role1 \
--role-session-name Session1 \
--tags Key=Star,Value=1 Key=Heart,Value=1 \
--transitive-tag-keys Star Heart
```

A continuación, utilice las credenciales para esa sesión para asumir `Role2`. El comando adjunta la etiqueta `Sun = 2` al segundo rol y se aplica como etiqueta principal cuando se utiliza la segunda sesión. Las etiquetas `Heart` y `Star` heredan las etiquetas de sesión transitiva de la primera sesión. Las etiquetas principales resultantes de la segunda sesión son `Heart = 1`, `Star = 1` y `Sun = 2`. `Heart` y `Star` seguirán siendo transitorias. La etiqueta `Sun` que se adjuntó a `Role2` no está marcada como transitiva porque no es una etiqueta de sesión. Las sesiones futuras no heredan esta etiqueta.

Realice esta segunda solicitud utilizando el siguiente comando de AWS CLI:

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role2 \
--role-session-name Session2
```

A continuación, utilice las credenciales de la segunda sesión para asumir `Role3`. Las etiquetas principales de la tercera sesión provienen de cualquier etiqueta de sesión nueva, de las etiquetas

de sesión transitiva heredadas y de las etiquetas de rol. Las etiquetas `Heart = 1` y `Star = 1` de la segunda sesión se heredaron de la etiqueta de sesión transitiva de la primera sesión. Si intenta pasar la etiqueta de sesión `Sun = 2`, la operación fallará. La etiqueta de sesión heredada `Star = 1` anula la etiqueta rol `Star = 3` . En el encadenamiento de roles, el valor de una etiqueta transitiva anula el rol que coincide con el valor `ResourceTag` después de la evaluación de la política de confianza de rol. En este ejemplo, si `Role3` utiliza `Star` como `ResourceTag` en la política de confianza de rol, y establece `ResourceTag` al valor de etiqueta transitiva de la sesión de rol de llamada. La etiqueta del rol `Lightning` también se aplica a la tercera sesión y no se establece como transitiva.

Realice la tercera solicitud con el siguiente comando de AWS CLI:

Example Ejemplo de solicitud de la CLI de AssumeRole

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Role3 \
--role-session-name Session3
```

Uso de etiquetas de sesión para ABAC

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos basados en atributos de etiquetas.

Si su empresa utiliza un proveedor de identidad (IdP) basado en OIDC o SAML para administrar las identidades de usuario, puede configurar su aserción SAML para que pase las etiquetas de sesión a AWS. Por ejemplo, con las identidades de usuario corporativo, cuando sus empleados se federan en AWS, AWSaplica sus atributos a su entidad principal resultante. Entonces puede utilizar ABAC para permitir o denegar permisos basados en esos atributos. Para obtener más información, consulte Tutorial de IAM: utilizar etiquetas de sesión SAML para ABAC.

Para obtener más información acerca del uso de IAM Identity Center con ABAC, consulte Atributos para el control de acceso en la Guía del usuario de AWS IAM Identity Center.

Ver las etiquetas de sesión en CloudTrail

Puede utilizar AWS CloudTrail para ver las solicitudes hechas para asumir roles o federar usuarios. El archivo de registro de CloudTrail incluye información sobre las etiquetas principales para la sesión de usuario federado o de rol asumido. Para obtener más información, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail.

Por ejemplo, suponga que realiza una solicitud AWS STS `AssumeRoleWithSAML`, pasa las etiquetas de sesión y establece esas etiquetas como transitivas. Puede encontrar la siguiente información en su registro de CloudTrail.

Example Ejemplo de registro de CloudTrail de ASSumeroleWithSAML

```
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyRoleSessionName",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "durationSeconds": 3600,
        "roleArn": "arn:aws:iam::123456789012:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::123456789012:saml-provider/Shibboleth"
    },
```

Puede ver los registros de CloudTrail de ejemplo siguientes para ver los eventos que utilizan etiquetas de sesión.

- Ejemplo de evento de API de encadenamiento de roles AWS STS en el archivo de registro de CloudTrail
- Ejemplo de evento API de AWS STS SAML en el archivo de registros de CloudTrail
- Ejemplo de evento de OIDC de AWS STS API en el archivo de registros de CloudTrail

Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail

IAM y AWS STS se integran con AWS CloudTrail, un servicio que proporciona un registro de las medidas adoptadas por un usuario o un rol de IAM. CloudTrail captura todas las llamadas a la API de IAM y AWS STS como eventos, incluidas las llamadas procedentes de la consola y de llamadas a la API. Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon S3. Si no configura un registro de seguimiento, puede ver los eventos más recientes en la consola de CloudTrail en el Historial de eventos. Puede utilizar CloudTrail para

obtener información acerca de la solicitud que se realizó a IAM o AWS STS. Por ejemplo, puede ver la dirección IP desde la que se realizó la solicitud, quién realizó la solicitud, cuándo se realizó y otros detalles adicionales.

Para más información sobre CloudTrail, consulte la Guía del usuario de AWS CloudTrail.

Temas

Información de IAM y AWS STS en CloudTrail

CloudTrail se habilita en su Cuenta de AWS cuando se crea la cuenta. Cuando se produce actividad en IAM o AWS STS, esta se registra en un evento de CloudTrail junto con otros eventos de servicio de AWS en el Historial de eventos. Puede ver, buscar y descargar eventos recientes en su Cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos de la cuenta de Cuenta de AWS, incluidos los eventos de IAM y AWS STS, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para obtener más información, consulte:

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail

- Configurar notificaciones de Amazon SNS para CloudTrail
- Recepción de archivos de registro de CloudTrail de varias regiones y Recepción de archivos de registro de CloudTrail de varias cuentas

Todas las acciones de IAM y AWS STS las registra CloudTrail y se documentan en la Referencia de la API de IAM y la Referencia de API de AWS Security Token Service.

Registro de solicitudes de IAM y API de AWS STS

CloudTrail registra todas las solicitudes de API autenticadas (realizadas con credenciales) en IAM y operaciones de API de AWS STS. CloudTrail también registra solicitudes no autenticadas a las acciones de AWS STS, `AssumeRoleWithSAML` y `AssumeRoleWithWebIdentity`, y registra información proporcionada por el proveedor de identidad. Puede utilizar esta información para asignar llamadas realizadas por un usuario federado con un rol asumido al intermediario federado externo de origen. En el caso de `AssumeRole`, puede asignar llamadas al servicio de AWS de origen o a la cuenta del usuario de origen. La sección `userIdentity` de los datos JSON en la entrada de registro de CloudTrail incluye la información que necesita para asignar la solicitud AssumeRole* a un determinado usuario federado. Para obtener más información, consulte Elemento userIdentity de CloudTrail en la Guía del usuario de AWS CloudTrail.

Por ejemplo, las llamadas a `CreateUser`, `DeleteRole` y `ListGroups` de IAM y otras operaciones de API quedan registradas en CloudTrail.

Más adelante en este tema se incluyen ejemplos de este tipo de entrada de log.

Registro de solicitudes de API a otros servicios de AWS

Las solicitudes autenticadas a las operaciones de la API de otros servicios AWS son registradas por CloudTrail, y estas entradas de registro contienen información sobre quién generó la solicitud.

Por ejemplo, supongamos que ha realizado una solicitud para enumerar las instancias de Amazon EC2 o crear un grupo de implementaciones de AWS CodeDeploy. Los detalles de la persona o servicio que ha realizado la solicitud se incluyen en la entrada de registro de dicha solicitud. Esta información permite determinar si la solicitud se hizo mediante el Usuario raíz de la cuenta de AWS, un usuario de IAM, un rol u otro servicio de AWS.

Para obtener más información sobre la información de identidad del usuario en las entradas de registro de CloudTrail, consulte el Elemento userIdentity en la Guía del usuario de AWS CloudTrail.

Registro de eventos de inicio de sesión de usuarios

CloudTrail registra los eventos de inicio de sesión en la AWS Management Console, los foros de debate de AWS, y AWS Marketplace. CloudTrail registra los intentos de inicio de sesión correctos y fallidos de usuarios de IAM y usuarios federados.

Para ver ejemplos de eventos de CloudTrail para los inicios de sesión de usuario raíz correctos y fallidos, consulte Ejemplo de registros de eventos para usuarios raíz en la Guía del usuario de AWS CloudTrail.

Como práctica recomendada de seguridad, AWS no registra el texto del nombre de usuario de IAM introducido si el error de inicio de sesión se ha producido por un nombre de usuario incorrecto. El texto del nombre de usuario está enmascarado por el valor `HIDDEN_DUE_TO_SECURITY_REASONS`. Para obtener un ejemplo, consulte Ejemplo de evento de error de inicio de sesión provocado por un nombre de usuario incorrecto, más adelante en este tema. El texto del nombre de usuario está oculto, ya que estos errores pueden deberse a errores de usuario. El registro de estos errores podría exponer información potencialmente confidencial. Por ejemplo:

- Se escribe por error una contraseña en el cuadro de nombre de usuario.
- Elige el enlace de una página de inicio de sesión de una cuenta de Cuenta de AWS pero se escribe el número de una cuenta diferente de Cuenta de AWS.
- Se olvida la cuenta en la que se ha iniciado sesión y se escribe por error el nombre de una cuenta de correo electrónico personal, un identificador de inicio de sesión en un banco u otro ID privado.

Registro de eventos de inicio de sesión para credenciales temporales

Cuando una entidad principal solicita credenciales temporales, el tipo de entidad principal determina cómo CloudTrail registra el evento. Esto puede ser complicado cuando una entidad principal asume un rol en otra cuenta. Existen varias llamadas a la API para realizar operaciones relacionadas con las operaciones de roles entre cuentas. En primer lugar, la entidad principal llama a una API de AWS STS para recuperar las credenciales temporales. Dicha operación se registra en la cuenta que realiza la llamada y en la cuenta donde se realiza la operación de AWS STS. A continuación, la entidad principal utiliza el rol para realizar otras llamadas a la API en la cuenta del rol asumido.

Puede utilizar la clave de condición de `sts:SourceIdentity` en la política de confianza de rol para exigir a los usuarios que especifiquen una identidad cuando asuman un rol. Por ejemplo, puede requerir que los usuarios de IAM especifiquen su propio nombre de usuario como su

identidad de origen. Esto puede ayudarle a determinar qué usuario realizó una acción específica en AWS. Para obtener más información, consulte `sts:SourceIdentity`. También puede utilizar `sts:RoleSessionName` para exigir a los usuarios que especifiquen un nombre de sesión cuando asuman un rol. Esto puede ayudarle a diferenciar entre sesiones de rol para un rol que utilizan diferentes entidades principales cuando revisa los registros de AWS CloudTrail.

En la siguiente tabla se muestra cómo CloudTrail registra información de identidad de usuarios diferente para cada una de las API de AWS STS que generan credenciales temporales.

Tipo de elemento principal	API de STS	Identidad de usuario en el registro de CloudTrail para la cuenta del intermediario	Identidad de usuario en el registro de CloudTrail para la cuenta del rol asumido	Identidad de usuario en el registro de CloudTrail para llamadas a la API posteriores del rol
Credenciales de Usuario raíz de la cuenta de AWS	GetSessionToken	Identidad de usuario raíz	La cuenta propietaria del rol es la misma cuenta que realiza la llamada	Identidad de usuario raíz
Usuario de IAM	GetSessionToken	Identidad de usuario de IAM	La cuenta propietaria del rol es la misma cuenta que realiza la llamada	Identidad de usuario de IAM
Usuario de IAM	GetFederationToken	Identidad de usuario de IAM	La cuenta propietaria del rol es la misma cuenta que realiza la llamada	Identidad de usuario de IAM

Tipo de elemento principal	API de STS	Identidad de usuario en el registro de CloudTrail para la cuenta del intermediario	Identidad de usuario en el registro de CloudTrail para la cuenta del rol asumido	Identidad de usuario en el registro de CloudTrail para llamadas a la API posteriores del rol
Usuario de IAM	AssumeRole	Identidad de usuario de IAM	Número de cuenta e ID principal (si se trata de un usuario) o elemento principal del servicio de AWS	Solo identidad de rol (ningún usuario)
Usuario autenticado externamente	AssumeRoleWithSAML	n/a	Identidad de usuario de SAML	Solo identidad de rol (ningún usuario)
Usuario autenticado externamente	AssumeRoleWithWebIdentity	n/a	Identidad de usuario web o de OIDC	Solo identidad de rol (ningún usuario)

CloudTrail considera que una acción es de solo lectura si no tiene ningún efecto de cambio en un recurso. Al registrar un evento de solo lectura, CloudTrail omite la información de los `responseElements` en el registro. Cuando CloudTrail registra un evento que no es de solo lectura, muestra los `responseElements` completos en la entrada del registro. Sin embargo, para las API `AssumeRole`, `AssumeRoleWithSAML` y `AssumeRoleWithWebIdentity` de AWS STS aunque se registren como de solo lectura, CloudTrail incluirá los `responseElements` completos en el registro para estas API.

En la siguiente tabla se muestra cómo CloudTrail registra la información de `responseElements` y `readOnly` para cada una de las API de AWS STS que generan credenciales temporales.

API de STS	Información de elementos de respuesta	Solo lectura
AssumeRole	Incluido	true
AssumeRoleWithSAML	Incluido	true
AssumeRoleWithWebIdentity	Incluido	true
GetFederationToken	Incluido	false
GetSessionToken	Incluido	false

Ejemplo de eventos API de IAM en el registro de CloudTrail

Los archivos de registro de CloudTrail incluyen eventos con formato JSON. Un evento de API representa una solicitud de API única e incluye información sobre la entidad principal, la acción solicitada, cualquier parámetro y la fecha y la hora de la acción.

Ejemplo de evento API de IAM en el archivo de registros de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro de CloudTrail para una solicitud realizada para la acción IAM de `GetUserPolicy`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
```

```
  "eventTime": "2014-07-15T21:40:14Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "GetUserPolicy",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "signin.amazonaws.com",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "userName": "JaneDoe",
    "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
  },
  "responseElements": null,
  "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
  "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```

A partir de la información de este evento, puede determinar que la solicitud se realizó para obtener una política de usuario denominada `ReadOnlyAccess-JaneDoe-201407151307` para el usuario `JaneDoe`, tal y como se especifica en el elemento `requestParameters`. También puede ver que la solicitud fue realizada por un usuario de IAM denominado `JaneDoe` el 15 de julio de 2014 a las 21:40 h (UTC). En este caso, la solicitud se originó en la AWS Management Console, tal y como puede observar en el elemento `userAgent`.

Ejemplo de evento API de AWS STS en el archivo de registros de CloudTrail

Los archivos de registro de CloudTrail incluyen eventos con formato JSON. Un evento de API representa una solicitud de API única e incluye información sobre la entidad principal, la acción solicitada, cualquier parámetro y la fecha y la hora de la acción.

Ejemplo de eventos de API AWS STS entre cuentas en archivos de registro de CloudTrail

El usuario de IAM llamado `JohnDoe` en la cuenta 777788889999 llama a la acción AWS STS `AssumeRole` para asumir el rol `EC2-dev` en la cuenta 111122223333. El administrador de cuenta requiere que los usuarios establezcan una identidad de origen igual a su nombre de usuario cuando asuman el rol. El usuario pasa el valor de identidad de origen de `JohnDoe`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
```

```
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2023, 4:07:39 PM"
      },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
  "sourceIdentity": "JohnDoe"
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::111122223333:role/EC2-dev",
      "accountId": "111122223333",
      "type": "AWS::IAM::Role"
    }
  ],
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
```

```
}
```

En el segundo ejemplo se muestra la entrada de registro de CloudTrail (111122223333) de la cuenta del rol asumido para la misma solicitud.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AWSAccount",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "accountId": "777788889999"
  },
  "eventTime": "2014-07-18T15:07:39Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRole",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67",
  "requestParameters": {
    "roleArn": "arn:aws:iam::111122223333:role/EC2-dev",
    "roleSessionName": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "serialNumber": "arn:aws:iam::777788889999:mfa"
  },
  "responseElements": {
    "credentials": {
      "sessionToken": "<encoded session token blob>",
      "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
      "expiration": "Jul 18, 2014, 4:07:39 PM"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AIDAQRSTUVWXYZEXAMPLE:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
      },
  "sourceIdentity": "JohnDoe"
  },
  "requestID": "4EXAMPLE-0e8d-11e4-96e4-e55c0EXAMPLE",
  "sharedEventID": "bEXAMPLE-efea-4a70-b951-19a88EXAMPLE",
  "eventID": "dEXAMPLE-ac7f-466c-a608-4ac8dEXAMPLE"
}
```

Ejemplo de evento de API de encadenamiento de roles AWS STS en el archivo de registro de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro CloudTrail para una solicitud realizada por John Doe en la cuenta 111111111111. John utilizó previamente su usuario `JohnDoe` para asumir el rol `JohnRole1`. Para esta solicitud, utiliza las credenciales de ese rol para asumir el rol `JohnRole2`. Esto se conoce como encadenamiento de roles. La identidad de origen que estableció cuando asumió el rol `JohnDoe1` persiste en la solicitud de asumir `JohnRole2`. Si John intenta establecer una identidad de origen diferente al asumir el rol, se deniega la solicitud. John pasa dos etiquetas de sesión en la solicitud. Establece esas dos etiquetas como transitivas. La solicitud hereda la etiqueta `Department` como transitiva porque John la estableció como transitiva cuando asumió `JohnRole1`. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos. Para obtener más información acerca de las claves transitivas en las cadenas de roles, consulte Encadenamiento de roles con etiquetas de sesión.


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIN5ATK5U7KEXAMPLE:JohnRole1",
        "arn": "arn:aws:sts::111111111111:assumed-role/JohnDoe/JohnRole1",
        "accountId": "111111111111",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2019-10-02T21:50:54Z"
            },
                "sessionIssuer": {
                "type": "Role",
               "principalId": "AROAIN5ATK5U7KEXAMPLE",
                "arn": "arn:aws:iam::111111111111:role/JohnRole1",
                "accountId": "111111111111",
                "userName": "JohnDoe"
            },
            "sourceIdentity": "JohnDoe"
        }
    },
    "eventTime": "2019-10-02T22:12:29Z",
    "eventSource": "sts.amazonaws.com",
```

```
    "eventName": "AssumeRole",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "123.145.67.89",
    "userAgent": "aws-cli/1.16.248 Python/3.4.7
 Linux/4.9.184-0.1.ac.235.83.329.metal1.x86_64 botocore/1.12.239",
    "requestParameters": {
        "incomingTransitiveTags": {
            "Department": "Engineering"
        },
        "tags": [
            {
                "value": "johndoe@example.com",
                "key": "Email"
            },
            {
                "value": "12345",
                "key": "CostCenter"
            }
        ],
        "roleArn": "arn:aws:iam::111111111111:role/JohnRole2",
        "roleSessionName": "Role2WithTags",
        "sourceIdentity": "JohnDoe",
        "transitiveTagKeys": [
            "Email",
            "CostCenter"
        ],
        "durationSeconds": 3600
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
            "expiration": "Oct 2, 2019, 11:12:29 PM",
            "sessionToken": "AgoJb3JpZ2luX2VjEB4aCXVzLXdlc3QtMSJHMEXAMPLETOKEN
+//rJb8Lo30mFc5MlhFCEbubZvEj0wHB/mDMwIgSEe9gk/Zjr09tZV7F1HDTMhmEXAMPLETOKEN/iEJ/
rkqngII9//////////
ARABGgw0MjgzMDc4NjM5NjYiDLZjZFKwP4qxQG5sFCryASO4UPz5qE97wPPH1eLMvs7CgSDBSWfonmRTCfokm2FN1+hWUdQ
+C+WKFZb701eiv9J5La2EXAMPLETOKEN/c7S5Iro1WUJ0q3Cxuo/8HUoSxVhQHM7zF7mWWLhXLEQ52ivL
+F6q5dpXu4aTFedpMfnJa8JtkWwG9x1Axj0Ypy2ok8v5unpQGWych1vwdvj6ez1Dm8Xg1+qIzXILiEXAMPLETOKEN/
vQGqu8H+nxp3kabcrtOvTFTvxX6vsc8OGwUfHhzAfYGEXAMPLETOKEN/
L6v1yMM3B1OwFOrQBno1HEjf1oNI8RnQiMNFdUOtwYj7HUZIOCZmjfN8PPHq77N7GJl9lzvIZKQA0Owcjg
+mc78zHCj8y0siY8C96paEXAMPLETOKEN/
E3cpksxWdgs91HRzJWScjN2+r2LTGjYhyPqcmFzzo2mCE7mBNEXAMPLETOKEN/oJy
+2o83YNW5tOiDmczgDzJZ4UKR84yGYOMfSnF4XcEJrDgAJ3OJFwmTcTQICAlSwLEXAMPLETOKEN"
        },
```

```
        "assumedRoleUser": {
            "assumedRoleId": "AROAIFR7WHDTSOYQYHFUE:Role2WithTags",
            "arn": "arn:aws:sts::111111111111:assumed-role/test-role/Role2WithTags"
        },
    "sourceIdentity": "JohnDoe"
    },
    "requestID": "b96b0e4e-e561-11e9-8b3f-7b396EXAMPLE",
    "eventID": "1917948f-3042-46ec-98e2-62865EXAMPLE",
    "resources": [
        {
            "ARN": "arn:aws:iam::111111111111:role/JohnRole2",
            "accountId": "111111111111",
            "type": "AWS::IAM::Role"
        }
    ],
    "eventType": "AwsApiCall",
    "recipientAccountId": "111111111111"
}
```

Ejemplo de evento de API AWS STS de servicio de AWS en un archivo de registro de CloudTrail

En el siguiente ejemplo se muestra una entrada de registro de CloudTrail para una solicitud realizada por un servicio de AWS que llama otra API de servicio con los permisos de una función de servicio. Muestra la entrada de registro de CloudTrail de la solicitud realizada en la cuenta 777788889999.

```
{
  "eventVersion": "1.04",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAQRSTUVWXYZEXAMPLE:devdsk",
    "arn": "arn:aws:sts::777788889999:assumed-role/AssumeNothing/devdsk",
    "accountId": "777788889999",
    "accessKeyId": "ASIAI44QH8DHBEXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2016-11-14T17:25:26Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAQRSTUVWXYZEXAMPLE",
        "arn": "arn:aws:iam::777788889999:role/AssumeNothing",
```

```
        "accountId": "777788889999",
        "userName": "AssumeNothing"
      }
    }
  },
  "eventTime": "2016-11-14T17:25:45Z",
  "eventSource": "s3.amazonaws.com",
  "eventName": "DeleteBucket",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "[aws-cli/1.11.10 Python/2.7.8
 Linux/3.2.45-0.6.wd.865.49.315.metal1.x86_64 botocore/1.4.67]",
  "requestParameters": {
    "bucketName": "my-test-bucket-cross-account"
  },
  "responseElements": null,
  "requestID": "EXAMPLE463D56D4C",
  "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "777788889999"
}
```

Ejemplo de evento API de AWS STS SAML en el archivo de registros de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro de CloudTrail para una solicitud realizada para la acción AWS STSde `AssumeRoleWithSAML`. La solicitud incluye los atributos de SAML `CostCenter` y `Project` que se pasan a través de la aserción SAML como etiquetas de sesión. Esas etiquetas se establecen como transitivas para que persistan en escenarios de encadenamiento de roles. La solicitud incluye el parámetro de API opcional `DurationSeconds`, representado como `durationSeconds` en el registro de CloudTrail, y configurado en `1800` segundos. La solicitud también incluye el atributo SAML `sourceIdentity`, que se pasa en la aserción SAML. Si alguien utiliza las credenciales de sesión de rol resultantes para asumir otro rol, esta identidad de origen persiste.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "SAMLUser",
        "principalId": "SampleUkh1i4+ExamplexL/jEvs=:SamlExample",
        "userName": "SamlExample",
        "identityProvider": "bdGOnTesti4+ExamplexL/jEvs="
    },
```

```
    "eventTime": "2023-08-28T18:30:58Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRoleWithSAML",
    "awsRegion": "us-east-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "aws-internal/3 aws-sdk-java/1.12.479
 Linux/5.10.186-157.751.amzn2int.x86_64 OpenJDK_64-Bit_Server_VM/17.0.7+11 java/17.0.7
 kotlin/1.3.72 vendor/Amazon.com_Inc. cfg/retry-mode/standard",
    "requestParameters": {
        "sAMLAssertionID": "_c0046cEXAMPLEb9d4b8eEXAMPLE2619aEXAMPLE",
        "roleSessionName": "MyAssignedRoleSessionName",
        "sourceIdentity": "MySAMLUser",
        "principalTags": {
            "CostCenter": "987654",
            "Project": "Unicorn",
            "Department": "Engineering"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
        "roleArn": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth",
        "principalArn": "arn:aws:iam::444455556666:saml-provider/Shibboleth",
        "durationSeconds": 1800
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "<encoded session token blob>",
            "expiration": "Aug 28, 2023, 7:00:58 PM"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROAD35QRSTUVWEXAMPLE:MyAssignedRoleSessionName",
            "arn": "arn:aws:sts::444455556666:assumed-role/SAMLTestRoleShibboleth/
MyAssignedRoleSessionName"
        },
        "packedPolicySize": 1,
        "subject": "SamlExample",
        "subjectType": "transient",
        "issuer": "https://server.example.com/idp/shibboleth",
        "audience": "https://signin.aws.amazon.com/saml",
        "nameQualifier": "bdGOnTesti4+ExamplexL/jEvs=",
        "sourceIdentity": "MySAMLUser"
    },
```

```
    "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
    "eventID": "dEXAMPLE-265a-41e0-9352-4401bEXAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::Role",
            "ARN": "arn:aws:iam::444455556666:role/SAMLTestRoleShibboleth"
        },
        {
            "accountId": "444455556666",
            "type": "AWS::IAM::SAMLProvider",
            "ARN": "arn:aws:iam::444455556666:saml-provider/test-saml-provider"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sts.us-east-2.amazonaws.com"
    }
}
```

Ejemplo de evento de OIDC de AWS STS API en el archivo de registros de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro de CloudTrail para una solicitud realizada para la acción AWS STSde `AssumeRoleWithWebIdentity`. La solicitud incluye los atributos `CostCenter` y `Project` que se pasan a través del token de proveedor de identidad como etiquetas de sesión. Esas etiquetas se establecen como transitivas para que persistan en encadenamiento de roles. La solicitud incluye el atributo `sourceIdentity` del token del proveedor de identidades. Si alguien utiliza las credenciales de sesión de rol resultantes para asumir otro rol, esta identidad de origen persiste.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:<id-of-
application>.apps.googleusercontent.com:<id-of-user>",
```

```
    "userName": "<id of user>",
    "identityProvider": "accounts.google.com"
  },
  "eventTime": "2016-03-23T01:39:51Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
  "requestParameters": {
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "MyAssignedRoleSessionName"
        "principalTags": {
            "CostCenter": "24680",
            "Project": "Pegasus"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
  },
  "responseElements": {
    "provider": "accounts.google.com",
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "audience": "<id of application>.apps.googleusercontent.com",
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "expiration": "Mar 23, 2016, 2:39:51 AM",
      "sessionToken": "<encoded session token blob>"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/
MyAssignedRoleSessionName"
    }
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
      "accountId": "444455556666",
      "type": "AWS::IAM::Role"
```

```
    }
  ],
  "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
  "eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "444455556666"
}
```

Ejemplo de eventos de inicio de sesión en el registro de CloudTrail

Los archivos de registro de CloudTrail incluyen eventos con formato JSON. Un evento de inicio de sesión representa una única solicitud de inicio de sesión e incluye información sobre el principal de inicio de sesión, la región y la fecha y la hora de la acción.

Ejemplo de evento de inicio de sesión correcto en un archivo de registro de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro de CloudTrail para un evento de inicio de sesión correcto.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "eventTime": "2014-07-16T15:49:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.110",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Success"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/s3/  ",
    "MFAUsed": "No"
```

```
  },
  "eventID": "3fcfb182-98f8-4744-bd45-10a395ab61cb"
}
```

Para obtener más información sobre la información incluida en los archivos de sesión de CloudTrail, consulte Referencia de eventos de CloudTrail en la Guía del usuario de AWS CloudTrail.

Ejemplo de evento de error de inicio de sesión en un archivo de registro de CloudTrail

En el ejemplo siguiente se muestra una entrada de registro de CloudTrail para un evento de inicio de sesión incorrecto.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::111122223333:user/JaneDoe",
    "accountId": "111122223333",
    "userName": "JaneDoe"
  },
  "eventTime": "2014-07-08T17:35:27Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.100",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "Failed authentication",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "MobileVersion": "No",
    "LoginTo": "https://console.aws.amazon.com/sns",
    "MFAUsed": "No"
  },
  "eventID": "11ea990b-4678-4bcd-8fbe-62509088b7cf"
}
```

A partir de esta información, puede determinar que el intento de inicio de sesión fue realizado por un usuario de IAM denominado `JaneDoe`, tal y como se muestra en el elemento `userIdentity`.

También puede ver que se ha producido un error en el intento de inicio de sesión, tal y como se muestra en el elemento `responseElements`. Puede ver que `JaneDoe` intentó iniciar sesión en la consola de Amazon SNS el 8 de julio de 2014 a las 17:35 h (UTC).

Ejemplo de evento de error de inicio de sesión provocado por un nombre de usuario incorrecto

En el siguiente ejemplo se muestra una entrada de registro de CloudTrail para un evento de inicio de sesión incorrecto que el usuario ha provocado al introducir un nombre de usuario incorrecto. AWS enmascara el texto de `userName` con `HIDDEN_DUE_TO_SECURITY_REASONS` para evitar la revelación de información potencialmente confidencial.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "HIDDEN_DUE_TO_SECURITY_REASONS"
  },
  "eventTime": "2015-03-31T22:20:42Z",
  "eventSource": "signin.amazonaws.com",
  "eventName": "ConsoleLogin",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "Mozilla/5.0 (Windows NT 6.1; WOW64; rv:24.0) Gecko/20100101
 Firefox/24.0",
  "errorMessage": "No username found in supplied account",
  "requestParameters": null,
  "responseElements": {
    "ConsoleLogin": "Failure"
  },
  "additionalEventData": {
    "LoginTo": "https://console.aws.amazon.com/console/home?state=hashArgs
%23&isauthcode=true",
    "MobileVersion": "No",
    "MFAUsed": "No"
  },
  "eventID": "a7654656-0417-45c6-9386-ea8231385051",
  "eventType": "AwsConsoleSignin",
  "recipientAccountId": "123456789012"
}
```

Política de confianza del rol de IAM

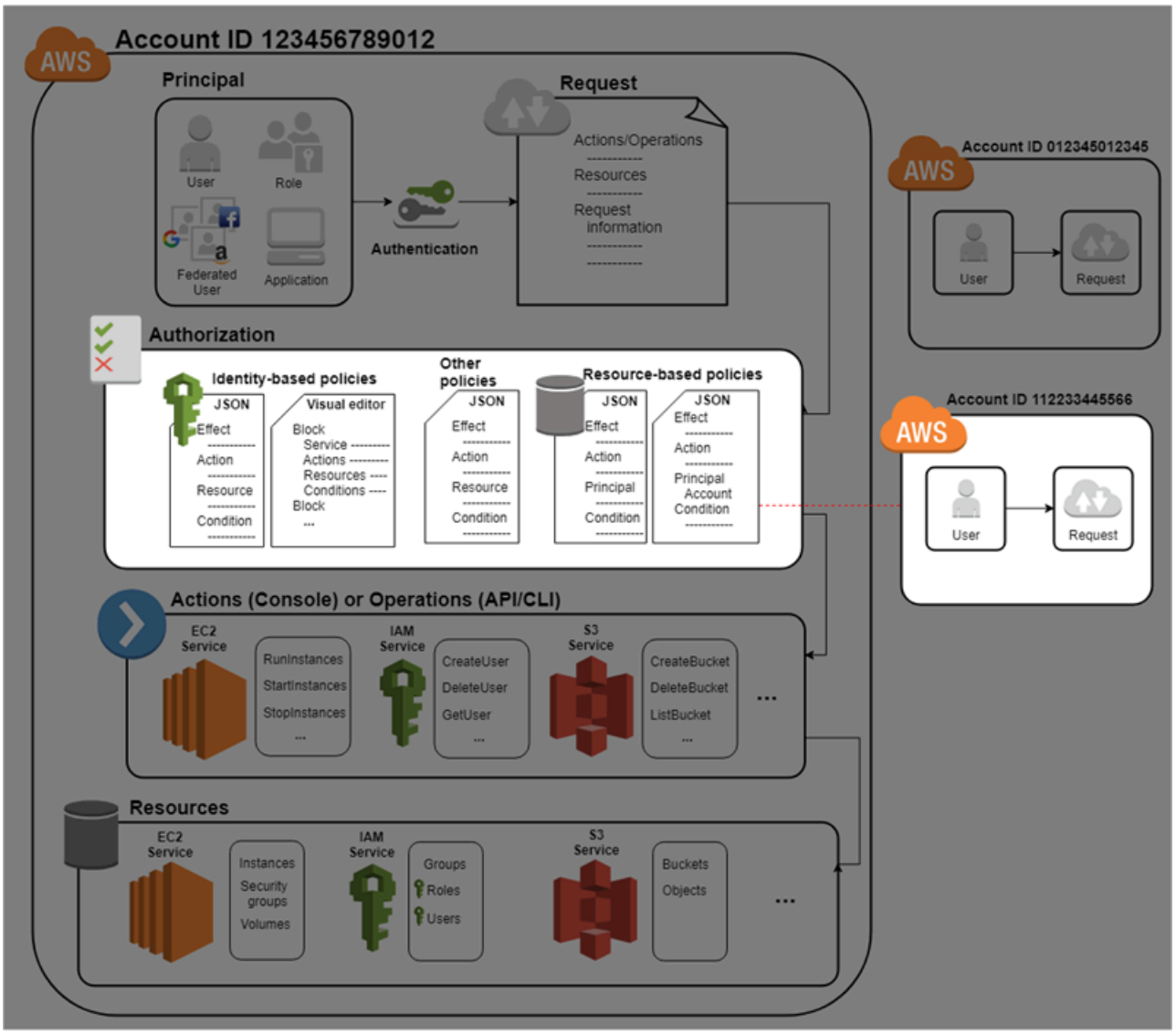
El 21 de septiembre de 2022, AWS introdujo cambios en el comportamiento de la política de confianza de roles de la IAM para exigir que se permita explícitamente incluir en una política de confianza de roles cuando un rol se asume a sí mismo. Los roles de IAM de la lista heredada de comportamientos permitidos tienen un campo de additionalEventData para explicitTrustGrant para eventos `AssumeRole`. El valor de `explicitTrustGrant` es falso cuando un rol de la lista heredada de permitidos asume que utiliza el comportamiento heredado. Cuando un rol de la lista heredada de roles permitidos se asume a sí mismo, pero el comportamiento de la política de confianza del rol se actualizó para permitir explícitamente que el rol se asuma por sí mismo, el valor de `explicitTrustGrant` es verdadero.

Solo un número muy reducido de funciones de IAM figuran en la lista de funciones permitidas para el comportamiento heredado y este campo solo está presente en los registros de CloudTrail de estas funciones cuando se asumen por sí mismas. En la mayoría de los casos, no es necesario que un rol de IAM se asuma a sí mismo. AWS recomienda actualizar los procesos, los códigos o las configuraciones para eliminar este comportamiento o actualizar las políticas de confianza de rol para permitir este comportamiento explícitamente. Para más información, consulte Anuncio de una actualización del comportamiento de la política de confianza en los roles de IAM.

Recursos de AWS para administración de acceso

AWS Identity and Access Management (IAM) es un servicio web que lo ayuda a controlar de forma segura el acceso a los recursos de AWS. Cuando una entidad principal realiza una solicitud en AWS, el código de aplicación de AWS comprueba si la entidad principal está autenticada (ha iniciado sesión) y autorizada (tiene permisos). Para administrar el acceso en AWS cree políticas y asócielas a identidades de IAM o recursos de AWS. Las políticas son documentos JSON de AWS que, cuando se asocian a una identidad o un recurso, definen sus permisos. Para obtener más información sobre los tipos de políticas y sus usos, consulte Políticas y permisos en IAM.

Para obtener más información sobre el resto del proceso de autenticación y autorización, consulte Cómo funciona IAM.



Durante la autorización, el código de aplicación de AWS utiliza los valores del contexto de la solicitud para buscar políticas coincidentes y determinar si se debe permitir o denegar la solicitud.

AWS comprueba cada política que se aplica al contexto de la solicitud. Si una sola política deniega la solicitud, AWS deniega toda la solicitud y deja de evaluar las políticas. Esto se denomina una denegación explícita. Dado que las solicitudes se deniegan de forma predeterminada, IAM autoriza una solicitud únicamente si las políticas aplicables permiten todas las partes de la solicitud. La lógica de evaluación de una solicitud para una cuenta individual se rige por las siguientes normas:

- De forma predeterminada, todas las solicitudes se deniegan implícitamente. (Como alternativa, de forma predeterminada, Usuario raíz de la cuenta de AWS tiene acceso completo).
- Un permiso explícito en una política basada en identidad o en recursos anula esta opción predeterminada.
- Si existe un límite de permisos, una SCP de Organizations o una política de sesión, es posible que anule el permiso con una denegación implícita.
- Una denegación explícita en cualquier política invalida cualquier permiso concedido.

Una vez que la solicitud se ha autenticado y se ha autorizado, AWS aprueba la solicitud. Si necesita realizar una solicitud en otra cuenta, una política de la otra cuenta debe permitirle el acceso al recurso. Además, la entidad de IAM que utilice para realizar la solicitud debe tener una política basada en identidad que permita la solicitud.

Recursos de administración de acceso

Para obtener más información acerca de los permisos y la creación de políticas, consulte los recursos siguientes:

Las siguientes entradas en el blog de seguridad de AWS tratan las formas habituales de escribir políticas para obtener acceso a buckets y objetos de Amazon S3.

- Escritura de políticas de IAM: Cómo conceder acceso a un bucket de Amazon S3
- Escritura de políticas de IAM: Conceder acceso a carpetas específicas de usuarios en un bucket de Amazon S3
- Políticas de IAM y políticas de bucket y ACLs! Oh My! (Control del acceso a los recursos de S3)
- A Primer on RDS Resource-Level Permissions
- Demystifying EC2 Resource-Level Permissions

Políticas y permisos en IAM

Puede administrar el acceso en AWS creando políticas y asignándoselas a identidades de IAM (usuarios, grupos de usuarios o roles) o a recursos de AWS. Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal de IAM (usuario o rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de políticas se almacenan en AWS como documentos JSON. AWS admite seis tipos de políticas: basadas en identidad, basadas en recursos, límites de permisos, SCP de Organizations, ACL y políticas de sesión.

Las políticas de IAM definen permisos para una acción independientemente del método que se utilice para realizar la operación. Por ejemplo, si una política permite la acción GetUser, un usuario con dicha política puede obtener información de los usuarios desde la AWS Management Console, la AWS CLI o la API de AWS. Cuando se crea un usuario de IAM, se le puede permitir el acceso a la consola o el acceso mediante programación. Si se permite el acceso a la consola, el usuario de IAM puede iniciar sesión en la consola con sus credenciales de inicio de sesión. Si se permite el acceso programático, el usuario puede utilizar claves de acceso para trabajar con la CLI o la API.

Tipos de políticas

Los tipos de políticas siguientes, que se muestran por orden desde los que se utilizan con más frecuencia hasta los que se utilizan con menos frecuencia, están disponibles para su uso en AWS. Para obtener más información, consulte las secciones siguientes para cada tipo de política.

- Identity-based policies (Políticas basadas en identidad): asocie políticas administradas e insertadas a identidades de IAM (usuarios, grupos a los que pertenecen los usuarios o roles). Las políticas basadas en identidad, conceder permisos a una identidad.
- Políticas basadas en recursos – Asocie políticas insertadas a los recursos. Los ejemplos más comunes de políticas basadas en recursos son las políticas de bucket de Amazon S3 y las políticas de confianza de roles de IAM. Las políticas basadas en recursos conceden permisos a la entidad principal que se especifica en la política. Las entidades principales pueden estar en la misma cuenta que el recurso o en cuentas distintas.
- Límites de permisos – Puede utilizar políticas administradas para definir el límite de permisos de una entidad de IAM (usuario o rol). Esa política define los permisos máximos que las políticas basadas en identidad pueden conceder a una entidad, pero no concede permisos por sí misma. Los límites de permisos no definen los permisos máximos que una política basada en recursos puede conceder a una entidad.

- SCP de Organizations— Utilice una política de control de servicio (SCP) de AWS Organizations para definir los permisos máximos de para los miembros de cuentas de una organización o unidad organizativa (OU). Las SCP limitan los permisos que las políticas basadas en identidad o en recursos conceden a las entidades (usuarios o roles) dentro de la cuenta, pero no conceden permisos por sí mismas.
- Listas de control de acceso (ACL) - Utilice ACL para controlar qué entidades principales de otras cuentas pueden tener acceso al recurso al que la ACL está asociada. Las ACL son similares a las políticas basadas en recursos, aunque son el único tipo de política que no utiliza la estructura de los documentos de política JSON. Las ACL son políticas de permisos para varias cuentas que conceden permisos a la entidad principal especificada. Las ACL no pueden conceder permisos a entidades dentro de una misma cuenta.
- Políticas de sesión — Pase las políticas de sesión avanzadas cuando utilice el AWS CLI o la API de AWS para asumir un rol o un usuario federado. Las políticas de sesión limitan los permisos que las políticas basadas en identidad aplicadas al rol o al usuario conceden a la sesión. Las políticas de sesión limitan los permisos para una sesión creada, pero no conceden permisos por sí mismas. Para obtener más información, consulte Políticas de sesión.

Políticas basadas en identidad

Las políticas basadas en identidad son documentos de políticas de permisos JSON que controlan qué acciones puede realizar una identidad (usuarios, grupos de usuarios y roles), en qué recursos y en qué condiciones. Las políticas basadas en la identidad pueden clasificarse así:

- Políticas administradas: políticas independientes basadas en la identidad que puede adjuntar a varios usuarios, grupos y funciones en su Cuenta de AWS. Existen dos tipos de políticas administradas:
  - Políticas administradas de AWS – Políticas administradas creadas y administradas por AWS.
  - Políticas administradas por el cliente: políticas administradas que crea y administra en su Cuenta de AWS. Las políticas administradas por el cliente ofrecen un control más preciso sobre las políticas que las políticas administradas por AWS.
- Políticas insertadas: políticas que agrega directamente a un único usuario, grupo o rol. Las políticas insertadas mantienen una relación estricta de uno a uno entre una política y una identidad. Se eliminan cuando se elimina la identidad.

Para obtener información sobre cómo elegir entre una política administrada o una insertada, consulte Elegir entre políticas administradas y políticas insertadas.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de política JSON que puede asociar a un recurso como, por ejemplo, un bucket de Amazon S3. Estas políticas conceden a la entidad principal especificada permiso para ejecutar acciones concretas en el recurso y definen en qué condiciones son aplicables. Las políticas basadas en recursos son políticas insertadas. No existen políticas basadas en recursos que sean administradas.

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política basada en recursos. Añadir a una política basada en recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando la entidad principal y el recurso se encuentran en cuentas de Cuentas de AWS distintas, también debe utilizar una política basada en identidades para conceder a la entidad principal el acceso al recurso. Sin embargo, si la política basada en recursos concede el acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para obtener instrucciones paso a paso para conceder acceso entre servicios, consulte Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.

El servicio IAM solo admite un tipo de política basada en recursos, el llamado política de confianza de rol, que se asocia a un rol de IAM. Un rol de IAM es tanto una identidad como un recurso que admite políticas basadas en recursos. Por este motivo, debe asociar una política de confianza y una política basada en identidades al rol de IAM. Las políticas de confianza definen qué entidades principales (cuentas, usuarios, roles y usuarios federados) puede asumir el rol. Para obtener información sobre cómo difieren los roles de IAM con respecto a otras políticas basadas en recursos, consulte Acceso a recursos entre cuentas en IAM.

Para saber qué otros servicios admiten políticas basadas en recursos, consulte Servicios de AWS que funcionan con IAM. Para obtener más información sobre las políticas basadas en recursos, consulte Políticas basadas en identidad y políticas basadas en recursos. Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

Límites de permisos de IAM

Un límite de permisos es una característica avanzada que le permite definir los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM. Al establecer un límite

de permisos para una entidad, esta solo puede realizar las acciones que le permitan tanto sus políticas basadas en identidad como sus límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol como entidad principal no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de permisos, consulte Límites de permisos para las entidades de IAM.

Políticas de control de servicios (SCP)

AWS Organizations es un servicio que le permite agrupar y administrar de forma centralizada las cuentas de Cuentas de AWS que posee su negocio. Si habilita todas las características en una organización, entonces podrá aplicar políticas de control de servicio (SCP) a una o todas sus cuentas. Las SCP son políticas JSON que especifican los permisos máximos para una organización o unidad organizativa (OU). Una SCP limita los permisos para las entidades de las cuentas de miembros, incluido cada Usuario raíz de la cuenta de AWS. Una denegación explícita en cualquiera de estas políticas anulará el permiso.

Para obtener más información acerca de Organizations y las SCP, consulte Funcionamiento de las SCP en la Guía del usuario de AWS Organizations.

Listas de control de acceso (ACL)

Las listas de control de acceso (ACL) son políticas de servicio que le permiten controlar qué entidades principales de otra cuenta pueden obtener acceso a un recurso. Las ACL no se pueden utilizar para controlar el acceso de una entidad principal de la misma cuenta. Las ACL son similares a las políticas basadas en recursos, aunque son el único tipo de política que no utiliza el formato de documento de política JSON. Amazon S3, AWS WAF y Amazon VPC son ejemplos de servicios que admiten las ACL. Para obtener más información sobre las ACL, consulte Información general de Lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Políticas de sesión

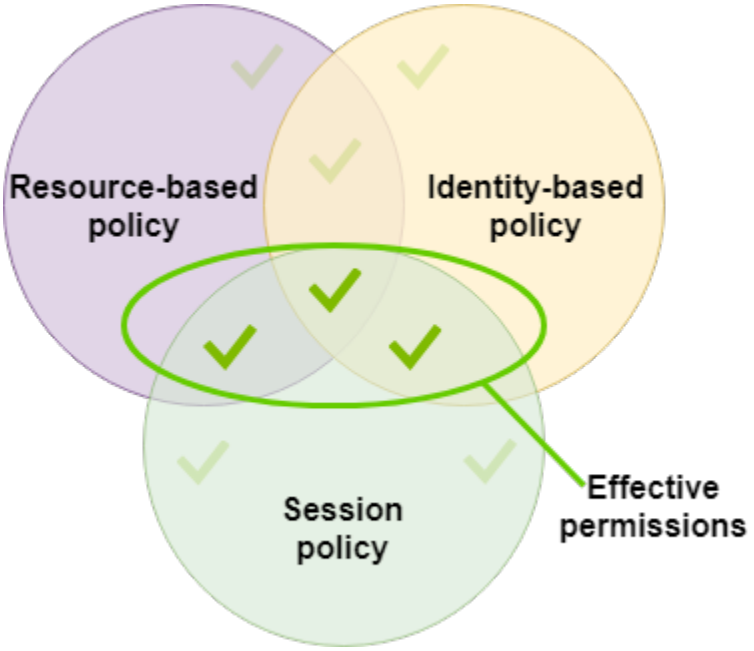
Las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de una sesión son la intersección de las políticas basadas en identidades aplicadas a la entidad de IAM (usuario o rol) utilizada para crear la sesión y las políticas de sesión. Los permisos también pueden proceder de una política basada en recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso.

Puede crear una sesión de rol y pasar políticas de sesión mediante programación con las operaciones de API `AssumeRole`, `AssumeRoleWithSAML` o `AssumeRoleWithWebIdentity`.

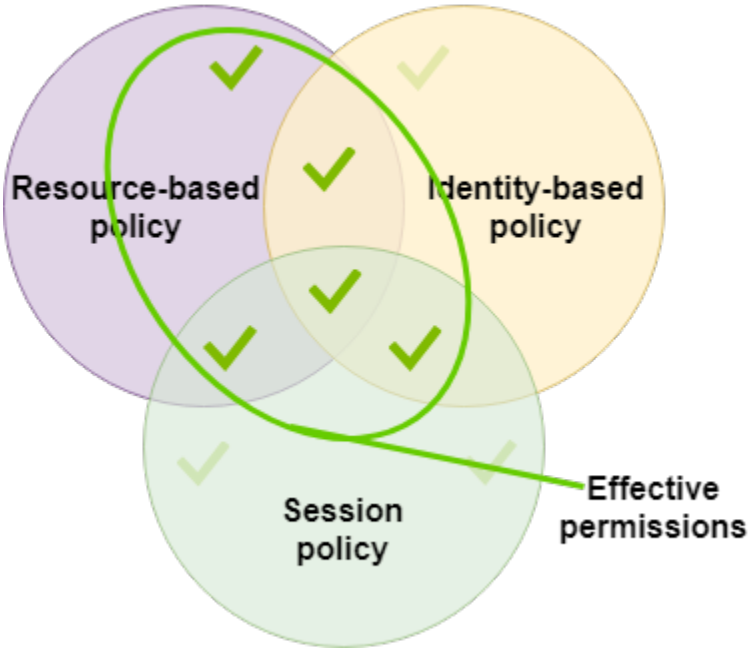
Puede transferir un único documento de política de sesión insertada JSON utilizando el parámetro `Policy`. Puede utilizar el parámetro `PolicyArns` para especificar hasta 10 políticas de sesión administrada. Para obtener más información sobre cómo crear una sesión de un rol, consulte Solicitud de credenciales de seguridad temporales.

Al crear una sesión de un usuario federado, se usan las claves de acceso del usuario de IAM para llamar de manera programática a la operación de API `GetFederationToken`. Asimismo, debe transferir las políticas de sesión. Los permisos de la sesión resultantes son la intersección de la política basada en identidades y la política de sesión. Para obtener más información sobre cómo crear una sesión de un usuario federado, consulte GetFederationToken: federación a través de un agente de identidades personalizadas.

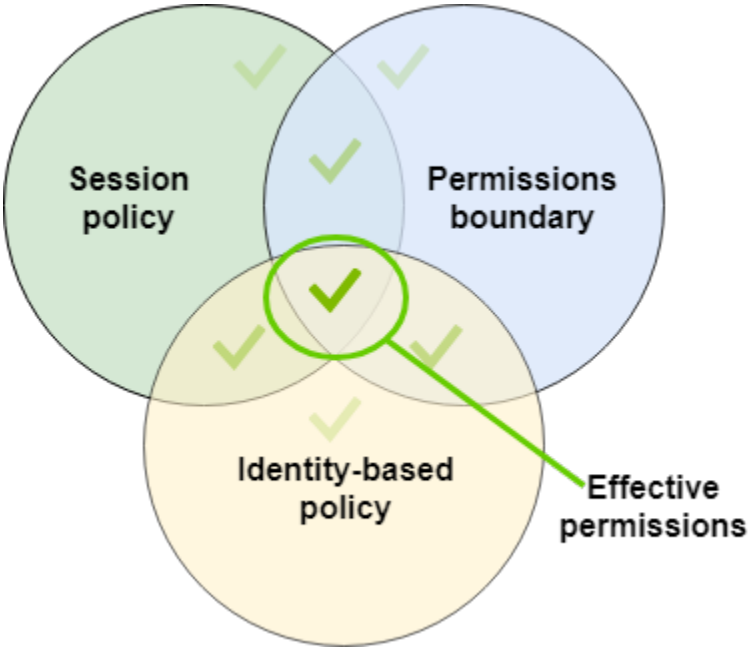
Una política basada en recursos puede especificar el ARN del usuario o el rol como una entidad principal. En ese caso, los permisos de la política basada en recursos se añaden a la política basada en identidades del usuario o rol antes de crear la sesión. La política de sesión limita los permisos totales concedidos por la política basada en recursos y la política basada en identidad. Los permisos de la sesión resultantes son la intersección de las políticas de sesión y las políticas basadas en recursos más la intersección de las políticas de sesión y las políticas basadas en identidades.



Una política basada en recursos puede especificar el ARN de la sesión como una entidad principal. En ese caso, los permisos de la política basada en recursos se añaden después de crear la sesión. Los permisos de la política basada en recursos no están limitados por la política de sesión. La sesión resultante tiene todos los permisos de la política basada en recursos más la intersección de la política basada en identidades y la política de sesión.



Un límite de permisos puede establecer el número de permisos máximo de un usuario o un rol que se utiliza para crear una sesión. En ese caso, los permisos de la sesión resultantes son la intersección de la política de sesión, el límite de permisos y la política basada en identidades. Sin embargo, un límite de permisos no restringe los permisos concedidos por una política basada en recursos que especifica el ARN de la sesión resultante.



Las políticas y el usuario raíz

A la Usuario raíz de la cuenta de AWS le afectan algunos tipos de políticas, pero no todos. No se pueden asociar políticas basadas en identidad al usuario raíz así como tampoco establecer el límite de permisos para el mismo. Sin embargo, es posible especificar el usuario raíz como la entidad principal en un política basada en identidad o una ACL. Un usuario raíz aún sigue siendo miembro de una cuenta. Si una cuenta es miembro de una organización en AWS Organizations, el usuario raíz se ve afectado por las SCP definidas para la cuenta.

Información general de políticas de JSON

Las mayoría de las políticas se almacenan en AWS como documentos JSON. Las políticas basadas en identidad y las políticas que se utilizan para establecer límites de permisos son documentos de política JSON que se asocian a un usuario o un rol. Las políticas basadas en recursos son documentos de política JSON que se adjuntan a un recurso. Las SCP son documentos de política JSON con sintaxis restringida que se asocian a una unidad organizativa de (OU) de AWS Organizations. Las ACL también se asocian a un recurso, pero se debe utilizar una sintaxis diferente. Las políticas de sesión son políticas JSON que se proporcionan al asumir una sesión de rol o usuario federado.

No es necesario que comprenda la sintaxis JSON. Puede utilizar el editor visual en la AWS Management Console para crear y editar políticas administradas por el cliente sin utilizar JSON. No obstante, si decide utilizar las políticas insertadas para grupos o las políticas complejas, seguirá siendo necesario crear y editar esas políticas en el editor de JSON mediante la consola. Para obtener información sobre cómo utilizar el editor visual, consulte Crear políticas de IAM y Edición de políticas de IAM.

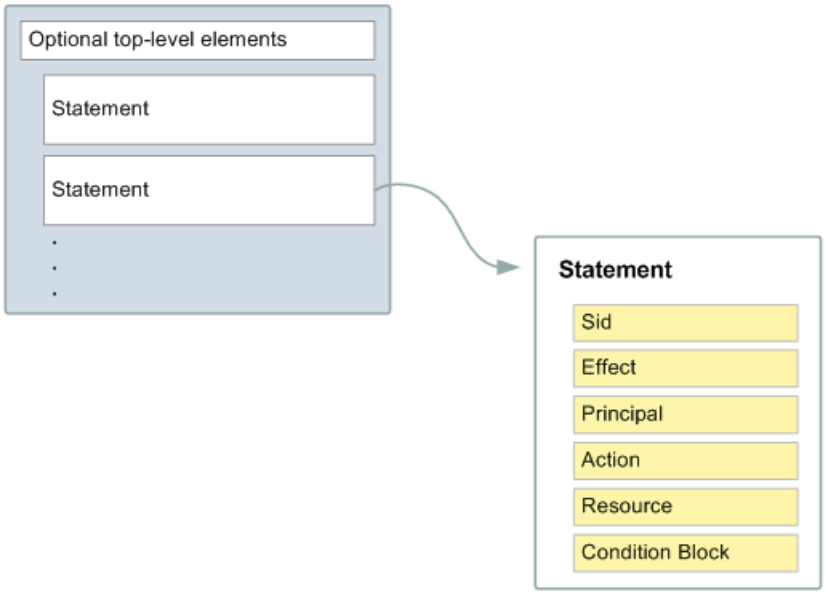
Cuando usted crea o edita una política JSON, IAM puede realizar la validación de políticas para ayudarle a crear una política eficaz. IAM identifica errores de sintaxis JSON, mientras que IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más las políticas. Para obtener más información acerca la validación de políticas, consulte Validación de políticas de IAM. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer.

Estructura de los documentos de política JSON

Tal y como se muestra en la siguiente figura, un documento de política JSON incluye estos elementos:

- Información opcional aplicable a toda la política en la parte superior del documento
- Una o varias instrucciones individuales

Cada instrucción incluye información sobre un único permiso. Si una política incluye varias instrucciones, AWS aplica un `OR` lógico a todas las instrucciones al evaluarlas. Si varias políticas son aplicables a una solicitud, AWS aplica un `OR` lógico a todas esas políticas al evaluarlas.



La información de una instrucción se incluye en una serie de elementos.

- Version – Especifica la versión del idioma de la política que desea utilizar. Le recomendamos que utilice la última versión de `2012-10-17`. Para obtener más información, consultar Elementos de política JSON de IAM: Version
- Statement – Utilice este elemento de política principal como contenedor de los siguientes elementos. Puede incluir varias instrucciones en una política.
- Sid (Opcional) – Incluye un ID de instrucción opcional para diferenciar entre las instrucciones.
- Effect – Utilice `Allow` o `Deny` para indicar si la política permite o deniega el acceso.
- Principal (Obligatorio únicamente en algunas circunstancias): si crea una política basada en recursos, debe indicar la cuenta, el usuario, el rol o el usuario federado al que desea permitir o denegar el acceso. Si va a crear una política de permisos de IAM para asociarla a un usuario o un rol, no puede incluir este elemento. La entidad principal está implícita como ese usuario o rol.
- Action – Incluye una lista de acciones que la política permite o deniega.

- Resource (Obligatorio solo en algunas circunstancias): si crea una política de permisos de IAM, debe especificar una lista de recursos a los que se aplican las acciones. Si crea una política basada en recursos, este elemento es opcional. Si no incluye este elemento, el recurso al que se aplica la acción es el recurso al que está asociada la política.
- Condition (Opcional) - Especifica las circunstancias bajo las cuales la política concede permisos.

Para obtener más información sobre estos y otros elementos más avanzados de las políticas, consulte Referencia de los elementos de las políticas de JSON de IAM.

Varias instrucciones y varias políticas

Si desea definir varios permisos para una entidad principal (usuario o rol), puede utilizar varias instrucciones en una única política. También puede asociar varias políticas. Si intenta definir varios permisos en una única instrucción, es posible que la política no conceda el acceso previsto. Le recomendamos que separe las políticas por tipo de recurso.

Debido al tamaño limitado de las políticas, podría ser necesario utilizar varias políticas para permisos más complejos. También es buena idea crear agrupaciones funcionales de permisos en una política independiente administrada por el cliente. Por ejemplo, crear una política para la administración de usuarios de IAM, una para la autoadministración y otra para la administración de buckets de S3. Independientemente de la combinación de varias instrucciones y varias políticas, AWS evalúa las políticas de la misma manera.

Por ejemplo, la siguiente política tiene tres instrucciones, cada una de las cuales define un conjunto de permisos independiente dentro de una única cuenta. Las instrucciones definen lo siguiente:

- La primera instrucción, con un `Sid` (ID de instrucción) `FirstStatement`, permite al usuario que tiene la política asociada cambiar su propia contraseña. El elemento `Resource` de esta instrucción es "`*`" (lo que significa "todos los recursos"). Sin embargo, en la práctica, la operación de la API `ChangePassword` (o el comando de la CLI equivalente `change-password`) solo afecta a la contraseña del usuario que realiza la solicitud.
- La segunda instrucción permite al usuario obtener una lista de todos los buckets de Amazon S3 en su cuenta de Cuenta de AWS. El elemento `Resource` de esta instrucción es "`*`" (lo que significa "todos los recursos"). Pero debido a que las políticas no conceden acceso a los recursos de otras cuentas, el usuario puede obtener únicamente la lista de los buckets de su propia cuenta de Cuenta de AWS.

- La tercera instrucción permite al usuario enumerar y recuperar cualquier objeto que se encuentre en un bucket denominado `confidential-data`, pero solo cuando el usuario se autentica con la autenticación multifactor (MFA). El elemento `Condition` de la política aplica la autenticación MFA.

  Si la instrucción de una política incluye un elemento `Condition`, la instrucción solo entra en vigor cuando el elemento `Condition` se evalúa como true. En este caso, la `Condition` se evalúa como true cuando el usuario se autentica mediante MFA. Si el usuario no se autentica mediante MFA, esta `Condition` se evalúa como false. En ese caso, la tercera instrucción de esta política no se aplica y el usuario no tendrá acceso al bucket `confidential-data`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::confidential-data",
        "arn:aws:s3:::confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Ejemplos de sintaxis de las políticas JSON

La política basada en identidad siguiente permite a la entidad principal enumerar un único bucket de Amazon S3 denominado `example_bucket`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket"
  }
}
```

La política basada en recursos siguiente se puede asociar a un bucket de Amazon S3. La política permite a los miembros de una cuenta de Cuenta de AWS específica realizar cualquier acción de Amazon S3 en el bucket denominado `mybucket`. Permite realizar cualquier acción que pueda llevarse a cabo en un bucket o en los objetos que contiene. (dado que la política concede confianza únicamente a la cuenta, se debe seguir concediendo permisos a los usuarios individuales de la cuenta para realizar las acciones especificadas de Amazon S3).

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

Para ver políticas de ejemplo que contemplan situaciones comunes, consulte Ejemplos de políticas basadas en identidad de IAM.

Conceder privilegios mínimos

Al crear políticas de IAM, siga los consejos de seguridad estándar de concesión de privilegios mínimos o garantizando solo los permisos necesarios para realizar una tarea. Determine las tareas que tienen que realizar los usuarios y roles, y luego elabore políticas que les permitan realizar solo esas tareas.

Comience con un conjunto mínimo de permisos y conceda permisos adicionales según sea necesario. Por lo general, es más seguro que comenzar con permisos que son demasiado tolerantes e intentar hacerlos más estrictos más adelante.

Como alternativa a los privilegios mínimos, puede utilizar las políticas administradas de AWS o las políticas con comodín de permisos de * para empezar a utilizar políticas. Considere el riesgo de seguridad de conceder a sus entidades principales más permisos de los que necesitan para realizar su trabajo. Supervise esas entidades principales para saber qué permisos están utilizando. A continuación, escriba políticas de privilegios mínimos.

IAM proporciona varias opciones para ayudarle a refinar los permisos que concede.

- Comprender las agrupaciones a nivel de acceso - Puede utilizar las agrupaciones de nivel de acceso para conocer el nivel de acceso que concede una política. Las acciones de política se clasifican como `List`, `Read`, `Write`, `Permissions management` o `Tagging`. Por ejemplo, puede elegir acciones de los niveles de acceso `List` y `Read` para a conceder acceso de solo lectura a los usuarios. Para obtener información sobre cómo utilizar los resúmenes de políticas para entender los permisos de nivel de acceso, consulte Descripción de los niveles de acceso en los resúmenes de políticas.
- Valide las políticas: puede realizar la validación de políticas mediante IAM Access Analyzer cuando cree y edite políticas JSON. Le recomendamos que revise y valide todas las políticas existentes. IAM Access Analyzer proporciona más de 100 verificaciones de políticas para validar sus políticas. Genera advertencias de seguridad cuando una declaración de su política permite el acceso que consideramos excesivamente permisivo. Puede utilizar las recomendaciones procesables que se proporcionan a través de las advertencias de seguridad mientras trabaja para conceder privilegios mínimos. Para obtener más información sobre las verificaciones de políticas proporcionadas por IAM Access Analyzer, consulte Validación de políticas de IAM Access Analyzer.
- Generar una política basada en la actividad de acceso - Para ayudarle a refinar los permisos que concede, puede generar una política de IAM que esté basada en la actividad de acceso de una entidad de IAM (usuario o rol). El analizador de acceso de IAM revisa los registros de AWS CloudTrail y genera una plantilla de política que contiene los permisos que ha utilizado la

entidad en el intervalo de tiempo especificado. Puede utilizar la plantilla para crear una política administrada con permisos detallados y, a continuación, adjuntarla a la entidad de IAM. De esta forma, solo concede los permisos que el usuario o rol necesita para interactuar con los recursos de AWS para su caso de uso específico. Para obtener más información, consulte Generar políticas basadas en la actividad de acceso.

- Utilizar la información de acceso reciente — Otra característica que puede ayudarle con menos privilegios es Información de acceso reciente. Encontrará esta información en la pestaña Asesor de acceso de la página de detalles de la consola de IAM de un usuario, grupo, rol o política de IAM. La información del último acceso también incluye información sobre algunas acciones a las que se accedió por última vez para algunos servicios como Amazon EC2, IAM, Lambda y Amazon S3. Si inicia sesión con las credenciales de cuenta administración de AWS Organizations, podrá ver la información de acceso reciente del servicio en la sección AWS Organizations de la consola de IAM. También puede utilizar AWS CLI o la API de AWS para recuperar un informe con información de acceso reciente de las entidades o políticas de IAM o de Organizations. Puede utilizar esta información para identificar permisos innecesarios, de modo que pueda perfeccionar sus políticas de IAM o de Organizations para que cumplan mejor con el principio de privilegio mínimo. Para obtener más información, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.
- Revisar eventos de cuenta en AWS CloudTrail: para reducir aún más los permisos, puede observar los eventos de su cuenta en el Historial de eventos de AWS CloudTrail. Los registros de eventos de CloudTrail de incluyen información detallada que usted puede utilizar para reducir los permisos de la política. Los registros solo incluyen las acciones y los recursos que sus entidades de IAM necesitan. Para obtener más información, consulte Ver eventos de CloudTrail en la consola de CloudTrail en la Guía del usuario de AWS CloudTrail.

Para obtener más información, consulte los siguientes temas de políticas para servicios individuales, en los que se ofrecen ejemplos sobre cómo redactar políticas para recursos específicos de servicios.

- Autenticación y control de acceso para Amazon DynamoDB en la Guía para desarrolladores de Amazon DynamoDB
- Uso de políticas de bucket y políticas de usuario en la Guía del usuario de Amazon Simple Storage Service.
- Información general de las listas de control de acceso en la Guía del usuario de Amazon Simple Storage Service.

Políticas administradas y políticas insertadas

Cuando establezca los permisos para una identidad en IAM, debe decidir si desea utilizar una política administrada por AWS, una política administrada por el cliente o una política insertada. En las siguientes secciones, se brinda más información sobre cada uno de los tipos de políticas basadas en identidad y cuándo utilizarlas.

Temas

Políticas administradas de AWS

Una política administrada por AWS es una política independiente creada y administrada por AWS. Política independiente significa que la política tiene su propio Nombre de recurso de Amazon(ARN) que incluye el nombre de la política. Por ejemplo, `arn:aws:iam::aws:policy/IAMReadOnlyAccess` es una política administrada por AWS. Para obtener más información sobre los ARN, consulte ARN de IAM. Para obtener una lista de políticas administradas de AWS para los Servicios de AWS, consulte Políticas administradas de AWS.

Las políticas administradas por AWS le permiten asignar los permisos adecuados a los usuarios, grupos y roles. Es más rápido que escribir las políticas uno mismo e incluye permisos para muchos casos de uso comunes.

No puede cambiar los permisos definidos en las políticas administradas por AWS. De vez en cuando, AWS actualiza los permisos definidos en una política administrada por AWS. Cuando AWS hace esto, la actualización afecta a todas las entidades principales (usuarios, grupos y roles) a las que se asocia la política. Es más probable que AWS actualice una política administrada por AWS cuando se lanza un nuevo servicio de AWS o hay nuevas llamadas a la API para los servicios existentes. Por ejemplo, la política administrada por AWS denominada ReadOnlyAccess ofrece acceso de solo lectura a todos los servicios y recursos de AWS. Cuando AWS lanza un nuevo servicio, AWS

actualiza la política ReadOnlyAccess para añadir permisos de solo lectura para el nuevo servicio. Los permisos actualizados se aplican a todas las entidades principales a las que la política está asociada.

Las políticas administradas por AWS de acceso completo definen los permisos para los administradores de servicios al otorgar acceso completo a un servicio.

- AmazonDynamoDBFullAccess
- IAMFullAccess

Las políticas administradas por AWS de usuarios avanzados proporcionan acceso completo a los servicios y recursos de AWS, pero no permiten administrar usuarios y grupos.

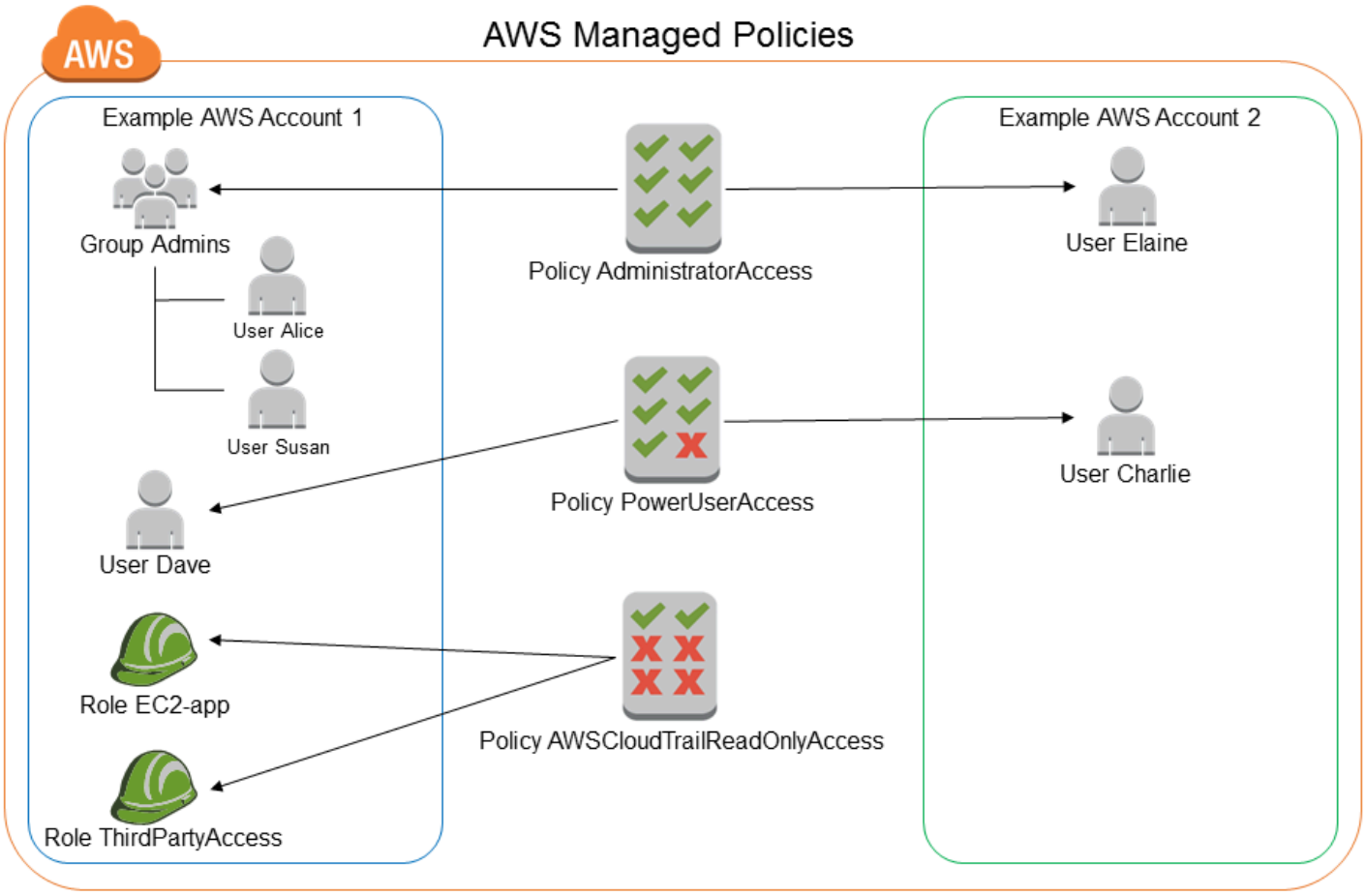
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

Las políticas administradas por AWS de acceso parcial proporcionan niveles específicos de acceso a los servicios de AWS sin los permisos de nivel de acceso de administración de permisos.

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

Una categoría particularmente útil de las políticas administradas por AWS son las concebidas para funciones. Estas políticas están en consonancia con funciones de trabajo típicamente empleadas en la industria de TI y facilitan conceder permisos para estas funciones de trabajo. Una ventaja clave del uso de las políticas de la función de trabajo es que se mantienen y actualizan AWS como nuevos servicios y operaciones del API. Por ejemplo, la función de trabajo AdministratorAccess proporciona acceso completo y delegación de permisos a cada servicio y recurso de AWS. Le recomendamos que utilice esta política únicamente para el administrador de la cuenta. Para los usuarios avanzados que requieran acceso completo a todos los servicios excepto acceso limitado a IAM y Organizations, utilice la función de trabajo PowerUserAccess. Para obtener una lista y las descripciones de la función de políticas, consulte Managed Policies de AWS para funciones de trabajo.

El siguiente diagrama ilustra las políticas administradas por AWS. Este diagrama muestra tres políticas administradas de AWS: AdministratorAccess, PowerUserAccess, y AWSCloudTrailReadOnlyAccess. Tenga en cuenta que una única política administrada por AWS puede asociarse a las entidades principales de diferentes cuentas de Cuentas de AWS y a diferentes entidades principales de una única cuenta de Cuenta de AWS.



Políticas administradas por el cliente

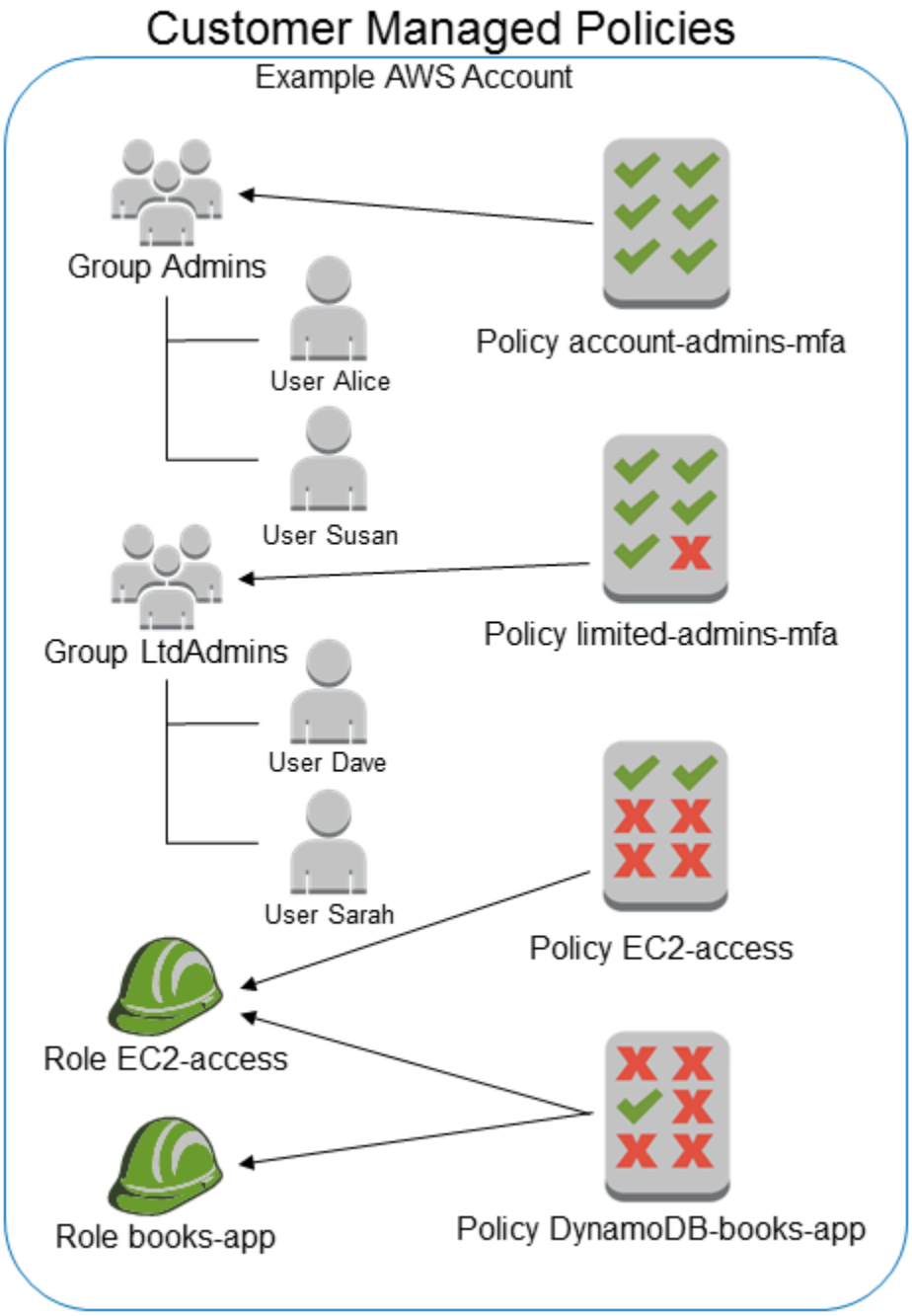
Puede crear políticas independientes en su propia Cuenta de AWS que puede asociar a las entidades principales (usuarios, grupos y roles). Puede crear estas políticas administradas por el cliente para sus casos de uso específicos y puede cambiarlas y actualizarlas con la frecuencia que desee. Al igual que con las políticas administradas de AWS, al asociar una política a una entidad principal, concederá a la entidad los permisos que están definidos en la política. Al actualizar permisos en la política, los cambios se aplican a todas las entidades principales a las que la política está asociada.

Una forma ideal para crear una política administrada por el cliente es comenzar copiando una política administrada por AWS existente. De esta forma sabrá que la política es correcta desde el principio y lo único que necesita hacer es personalizarla según su entorno.

El siguiente diagrama ilustra las políticas administradas por el cliente. Cada política es una entidad de IAM con su propio Nombre de recurso de Amazon (ARN) que incluye el nombre de la política.

Tenga en cuenta que la misma política puede asociarse a varias entidades principales por ejemplo, la misma política DynamoDB-books-app se asocia a dos roles diferentes de IAM.

Para obtener más información, consulte Crear políticas de IAM

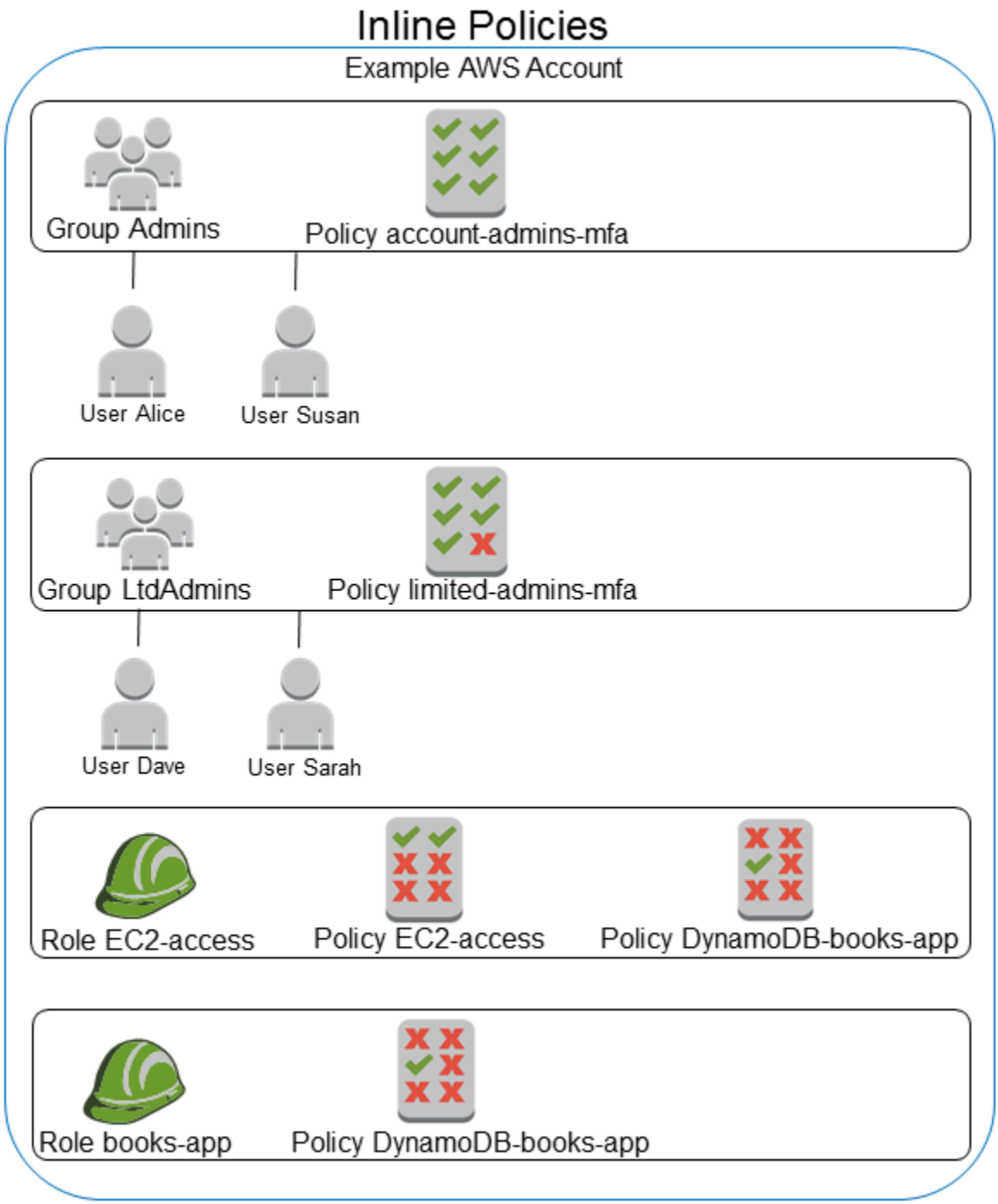


Políticas insertadas

Una política insertada es una política creada para una única identidad de IAM (un usuario, grupo o rol). Las políticas insertadas mantienen una relación estricta de uno a uno entre una política y una

identidad. Se eliminan cuando se elimina la identidad. Puede crear una política e incluirla en una identidad, ya sea al momento de crear la identidad o posteriormente. Si una política puede aplicarse a más de una entidad, es mejor utilizar una política administrada.

El siguiente diagrama ilustra las políticas insertadas. Cada política forma parte integrante del usuario, grupo o rol. Observe que dos roles incluyen la misma política (la política DynamoDB-books-app), pero no la comparten. Cada rol tiene su propia copia de la política.



Elegir entre políticas administradas y políticas insertadas

Tenga en cuenta sus casos de uso al decidir entre políticas insertadas o políticas administradas. En la mayoría de los casos, le recomendamos que utilice políticas administradas en lugar de políticas insertadas.

> **Note**
>
> Puede usar políticas administradas e insertadas juntas para definir permisos comunes y únicos para una entidad principal.

Las políticas administradas proporcionan las siguientes características:

Poder reutilizarlas

Una única política administrada puede asociarse a varias entidades principales (usuarios, grupos y roles). Puede crear una biblioteca de políticas que definan los permisos útiles para su cuenta de Cuenta de AWS y, a continuación, asociar dichas políticas a las entidades principales según sea necesario.

Administración centralizada de los cambios

Al cambiar una política administrada, el cambio se aplica a todas las entidades principales a las que la política está asociada. Por ejemplo, si desea añadir un permiso para una nueva API de AWS, puede actualizar la política administrada por el cliente o asociar una política administrada de AWS para añadir el permiso. Si utiliza una política administrada por AWS, AWS actualiza la política. Al actualizar una política administrada, los cambios se aplican a todas las entidades principales a las que la política está asociada. En cambio, para cambiar una política insertada, se debe editar individualmente cada identidad insertada que incluya la política. Por ejemplo, si un grupo y un rol incluyen la misma política insertada, debe editar individualmente ambas entidades principales para poder cambiar dicha política.

Control de versiones y restauración

Al cambiar una política administrada por el cliente, la política cambiada no sobrescribe la política existente. En cambio, IAM crea una nueva versión de la política administrada. IAM almacena hasta cinco versiones de las políticas administradas por el cliente. Puede utilizar las versiones de políticas para revertir una política a una versión anterior en caso de que sea necesario.

> **Note**
>
> Una versión de política es diferente de un elemento de política `Version`. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Para obtener más información sobre las versiones de política, consulte the section called “Control de versiones de políticas de IAM”. Para obtener más información sobre el elemento de política `Version`, consulte Elementos de política JSON de IAM: Version.

Delegar la administración de permisos

Puede permitir a los usuarios de su cuenta de Cuenta de AWS asociar y desasociar políticas a la vez que mantiene el control de los permisos definidos en dichas políticas. Para ello, designe algunos usuarios como administradores completos, es decir, administradores que pueden crear, actualizar y eliminar políticas. A continuación, puede designar otros usuarios como administradores limitados. Esos administradores limitados pueden asociar políticas a otras entidades principales, pero solo las políticas que les ha permitido asociar.

Para obtener más información acerca de cómo delegar la administración de permisos, consulte Control del acceso a políticas.

Límites de caracteres de política más amplios

El límite máximo de tamaño de caracteres para las políticas administradas es superior al límite de caracteres para las políticas en línea. Si alcanza el límite de tamaño de caracteres de la política integrada, puede crear más grupos de IAM y adjuntar la política administrada al grupo.

Para obtener más información sobre las cuotas y los límites, consulte IAM y cuotas de AWS STS.

Actualizaciones automáticas para las políticas administradas por AWS

AWS mantiene políticas administradas por AWS y las actualiza según sea necesario, por ejemplo, para agregar permisos para nuevos servicios de AWS, sin que usted tenga que realizar cambios. Las actualizaciones se aplican automáticamente a las entidades principales a las que haya asociado la política administrada por AWS.

Uso de políticas insertadas

Las políticas insertadas son útiles si desea mantener una relación estricta de uno a uno entre una política y la identidad a la cual se aplica. Por ejemplo, si desea asegurarse de que los permisos en

una política no se asignen por error a una identidad que no sea la prevista. Al utilizar una política insertada, los permisos de la política no se pueden asociar por error a la identidad incorrecta. Además, si utiliza la AWS Management Console para eliminar dicha identidad, las políticas insertadas en la identidad también se eliminan, ya que son parte de la entidad principal.

Introducción a las políticas administradas

Recomendamos utilizar políticas que otorguen el menor privilegio, o que concedan solo los permisos necesarios para realizar una tarea. La forma más segura de conceder un privilegio mínimo es redactar una política administrada por el cliente que conceda únicamente los permisos que necesite el equipo. Debe crear un proceso para permitir que su equipo solicite más permisos cuando sea necesario. Se necesita tiempo y experiencia para crear políticas de IAM administradas por el cliente que proporcionen a su equipo solo los permisos necesarios.

Para comenzar a agregar permisos a las identidades de IAM (usuarios, grupos de usuarios y roles), puede utilizar Políticas administradas de AWS. Las políticas administradas de AWS no conceden permisos de privilegios mínimos. Considere el riesgo de seguridad de conceder a sus entidades principales más permisos de los que necesitan para realizar su trabajo.

Puede adjuntar políticas administradas de AWS, incluidas las funciones de trabajo, a cualquier identidad de IAM. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

Para cambiar a permisos de privilegios mínimos, puede ejecutar AWS Identity and Access Management Access Analyzer para supervisar las entidades principales con las políticas administradas de AWS. Después de saber qué permisos están utilizando, puede escribir o generar una política administrada por el cliente con solo los permisos necesarios para su equipo. Esto es menos seguro, pero proporciona más flexibilidad a medida que aprende cómo usa su equipo AWS. Para obtener más información, consulte Generación de políticas del Analizador de acceso de IAM.

Las políticas administradas por AWS se han concebido para ofrecer permisos para muchos casos de uso comunes. Para obtener más información acerca de las políticas administradas AWS que están diseñadas para funciones de trabajo específicas, consulte Managed Policies de AWS para funciones de trabajo.

Para obtener una lista de políticas administradas por AWS, consulte la Guía de referencia de políticas administradas por AWS.

Cómo convertir una política insertada en una política administrada

Si tiene políticas insertadas en su cuenta, puede convertirlas en políticas administradas. Para ello, copie la política en una nueva política administrada. A continuación, asocie la nueva política a la identidad que tiene la política insertada. A continuación, elimine la política insertada.

Para convertir una política insertada en una política administrada

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios, Usuarios o Roles.
3. En la lista, elija el nombre del grupo de usuarios, usuario o rol que tiene la política que desea quitar.
4. Elija la pestaña Permissions (Permisos).
5. Para grupos de usuarios, seleccione el nombre de la política insertada que desea eliminar. Para usuarios y roles, seleccione Mostrar ***n*** más, si es necesario, y luego amplíe la política insertada que desee eliminar.
6. Seleccione Copiar para copiar el documento de política de JSON correspondiente a la política.
7. En el panel de navegación, seleccione Policies (Políticas).
8. Seleccione Crear política y, a continuación, la opción JSON.
9. Reemplace el texto existente con el texto de la política de JSON, y luego seleccione Siguiente.
10. Ingrese un nombre y una descripción opcional para la política y, a continuación, seleccione Crear política.
11. En el panel de navegación, seleccione Grupos, Usuarios o Roles y vuelva a seleccionar el nombre del grupo de usuario, usuario o rol que tenga la política que desea quitar.
12. Seleccione la pestaña Permisos y, a continuación, Agregar permisos.
13. Para grupos de usuarios, seleccione la casilla que se encuentra junto al nombre de la nueva política, elija Agregar permisos y, a continuación, elija Adjuntar política. Para usuarios o roles, elija Add permissions (Agregar permisos). En la página siguiente, seleccione Asociar políticas existentes directamente, después la casilla de verificación situada junto al nombre de la nueva política, a continuación Siguiente, y por último Agregar permisos.

    Volverá a la página Resumen de su usuario, grupo de usuarios o rol.
14. Seleccione la casilla de verificación situada junto a la política insertada que desee eliminar, y luego Eliminar.

Políticas obsoletas administradas por AWS

Para simplificar la asignación de permisos, AWS proporciona políticas administradas, es decir, políticas predefinidas listas para asociarlas a usuarios, grupos y roles de IAM.

A veces, AWS necesita añadir un permiso nuevo a una política existente, como, por ejemplo, cuando se introduce un servicio nuevo. Añadir un permiso nuevo a una política no altera ni elimina ninguna característica o capacidad.

Sin embargo, AWS puede optar por crear una política nueva cuando los cambios necesarios pueden repercutir sobre los clientes si se aplican a una política ya existente. Por ejemplo, eliminar permisos de una política ya existente podría acabar con los permisos de cualquier entidad o aplicación de IAM que dependiera de dicha política e incluso podría llegar a interrumpir una operación de importancia vital.

Por lo tanto, cuando se requiere un cambio de este tipo, AWS crea una política totalmente nueva con los cambios necesarios y la pone a disposición de los clientes. Después, la política antigua se marca como descartada. Una política administrada obsoleta se muestra con un icono de advertencia a su lado en la lista Políticas de la consola de IAM.

Una política descartada tiene las siguientes características:

- Sigue funcionando para todos los usuarios, grupos y roles asociados actualmente. Ningún elemento deja de funcionar.
- No puede asociarse a ningún usuario, grupo o rol nuevo. Si la separa de una entidad actual, no puede volver a acoplarla.
- Después de separarla de todas las entidades actuales, ya no es visible y deja de poder utilizarse.

Si un usuario, grupo o rol requiere la política, deberá asociar la nueva política en su lugar. Cuando reciba una notificación de que una política se ha descartado, le recomendamos que planee inmediatamente asociar todos los usuarios, grupos y roles a la política de sustitución y que los desconecte de la política descartada. Si sigue utilizando la política descartada, corre riesgos que solo se pueden mitigar pasando a utilizar la política de sustitución.

Límites de permisos para las entidades de IAM

AWS admite límites de permisos para las entidades de IAM (usuarios o roles). Un límite de permisos es una característica avanzada para utilizar una política administrada con el fin de definir los

permisos máximos que una política basada en identidad puede conceder a una entidad de IAM. Un límite de permisos para una entidad le posibilita realizar las acciones que le permitan tanto sus políticas basadas en identidad como sus límites de permisos.

Para obtener más información acerca de los tipos de políticas, consulte Tipos de políticas.

> **Important**
>
> No utilice instrucciones de política basadas en recursos que incluyan un elemento de política `NotPrincipal` con un efecto `Deny` para los usuarios o roles de IAM que tengan una política de límite de permisos adjunta. El elemento `NotPrincipal` con efecto `Deny` siempre denegará cualquier entidad principal de IAM que tenga una política de límite de permisos adjunta, independientemente de los valores especificados en el elemento `NotPrincipal`. Esto provoca que algunos usuarios o roles de IAM que de otro modo tendrían acceso al recurso pierdan dicho acceso. Recomendamos cambiar las instrucciones de política basadas en recursos y utilizar el operador de condición `ArnNotEquals` con la clave de contexto `aws:PrincipalArn` para limitar el acceso en lugar del elemento `NotPrincipal`. Para obtener más información sobre el elemento `NotPrincipal`, consulte Elemento de la política de JSON de AWS: NotPrincipal.

Puede utilizar una política administrada de AWS o una política administrada por el cliente para configurar el límite para una entidad de IAM (usuario o rol). Esa política limita los permisos que puede tener el usuario o rol como máximo.

Por ejemplo, suponga que el usuario de IAM llamado `ShirleyRodriguez` debe poder administrar únicamente Amazon S3, Amazon CloudWatch y Amazon EC2. Para aplicar esta regla, puede utilizar la siguiente política para configurar el límite de permisos para la usuaria `ShirleyRodriguez`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

Cuando se utiliza una política para configurar el límite de permisos para un usuario, limita los permisos del usuario, pero no proporciona permisos por sí misma. En este ejemplo, la política establece como permisos máximos de `ShirleyRodriguez` todas las operaciones en Amazon S3, CloudWatch y Amazon EC2. Shirley nunca podrá realizar operaciones en ningún otro servicio, tampoco IAM, aunque tenga una política de permisos que lo permita. Por ejemplo, puede añadir la siguiente política a la usuaria `ShirleyRodriguez`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

Esta política permite crear un usuario en IAM. Si asocia esta política de permisos a la usuaria `ShirleyRodriguez` y Shirley intenta crear un usuario, se produce un error en la operación. Se produce un error porque el límite de permisos no permite la operación `iam:CreateUser`. Dadas estas dos políticas, Shirley no tiene permisos para realizar ninguna operación en AWS. Debe agregar una política de permisos diferente para permitir acciones en otros servicios, como Amazon S3. También puede actualizar el límite de permisos para que pueda crear un usuario en IAM.

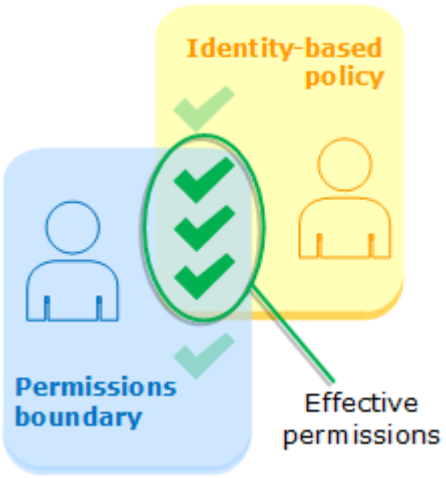
Evaluación de los permisos efectivos cuando se usan límites

El límite de permisos de una entidad de IAM (usuario o rol) establece los permisos máximos que esa entidad puede tener. Esto puede cambiar los permisos efectivos para ese usuario o rol. Los permisos efectivos de una entidad son los permisos que le conceden todas las políticas que afectan al usuario o rol. Dentro de una cuenta, los permisos de una entidad pueden verse afectados por políticas basadas en identidad, políticas basadas en recursos, los límites de permisos, SCP de Organizations o políticas de sesión. Para obtener más información sobre los distintos tipos de políticas, consulte Políticas y permisos en IAM.

Si cualquiera de estos tipos de políticas deniega de forma explícita el acceso para realizar una operación, la solicitud se deniega. Los permisos concedidos a una entidad por varios tipos de

permisos son más complejos. Para obtener más información sobre el modo en que AWS evalúa las políticas, consulte Lógica de evaluación de políticas.

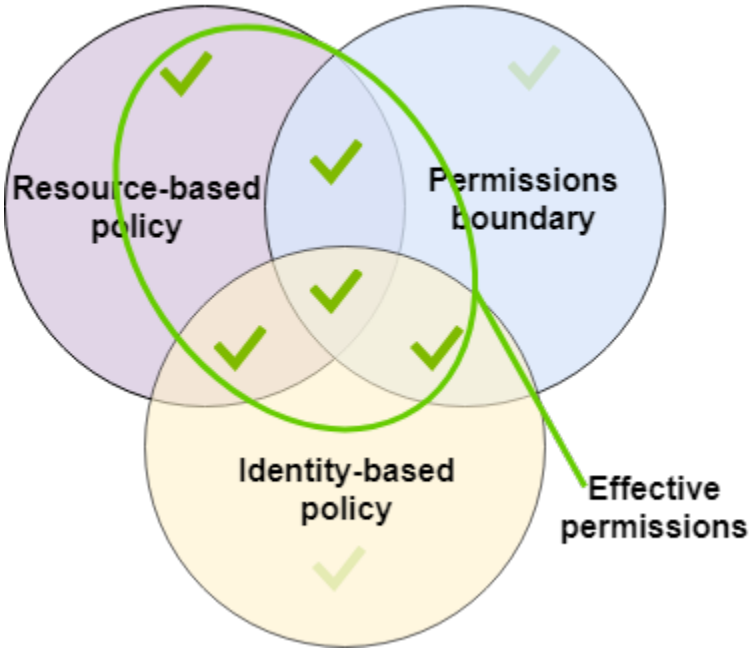
Políticas basadas en identidad con límites: las políticas basadas en identidad son políticas insertadas o administradas asociadas a un usuario, a un grupo de usuarios o a un rol. Las políticas basadas en identidad conceden permisos a la entidad, mientras que los límites de permisos restringen esos permisos. Los permisos efectivos son la intersección de ambos tipos de políticas. Una denegación explícita en una de estas políticas anulará el permiso.



Políticas basadas en recursos: las políticas basadas en recursos controlan la forma en que la entidad principal especificada tiene acceso al recurso al que la política está asociada.

Políticas basadas en recursos para los usuarios de IAM

> Dentro de la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de usuario de IAM (que no es una sesión de usuario federado) no están limitadas por una denegación implícita en una política basada en identidad o en un límite de permisos.



Políticas basadas en recursos para los roles de IAM

Rol de IAM: las políticas basadas en recursos que otorgan permisos a un ARN de rol de IAM están limitadas por una denegación implícita en un límite de permisos o una política de sesión.

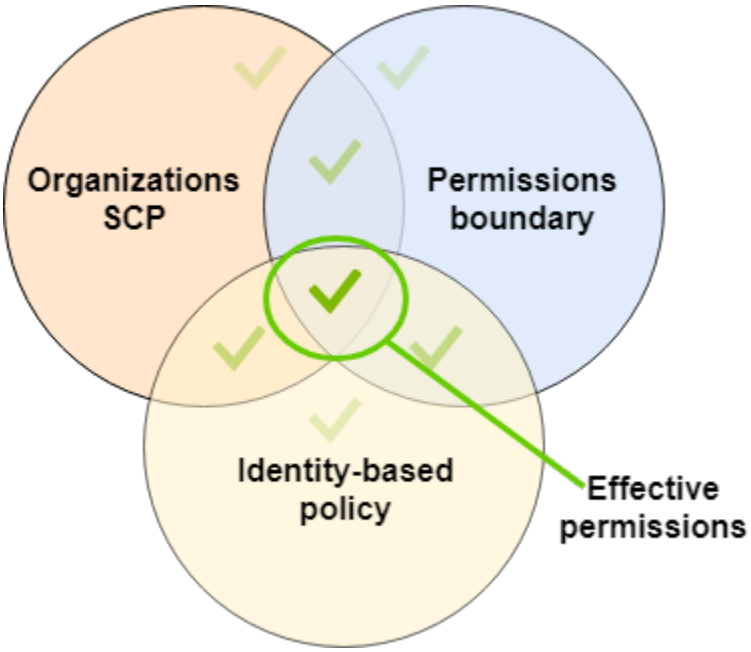
Sesión de rol de IAM: dentro de la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de sesión de rol de IAM otorgan permisos directamente a la sesión de rol asumida. Los permisos otorgados directamente a una sesión no están limitados por una denegación implícita en una política basada en la identidad, un límite de permisos ni una política de sesión. Cuando asume un rol y realiza una solicitud, la entidad principal que realiza la solicitud es el ARN de sesión de rol de IAM y no el ARN del rol en sí.

Políticas basadas en recursos para las sesiones de usuarios federados de IAM

Sesiones de usuarios federados de IAM: una sesión de usuario federado de IAM es una sesión creada mediante la llamada a `GetFederationToken`. Cuando un usuario federado realiza una solicitud, la entidad principal que realiza la solicitud es el ARN de usuario federado y no el ARN del usuario de IAM que se federó. En la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de usuario federado otorgan permisos directamente a la sesión. Los permisos otorgados directamente a una sesión no están limitados por una denegación implícita en una política basada en la identidad, un límite de permisos ni una política de sesión.

Sin embargo, si una política basada en recursos concede permiso al ARN del usuario de IAM que se federó, las solicitudes realizadas por el usuario federado durante la sesión están limitadas por una denegación implícita en un límite de permisos o una política de sesión.

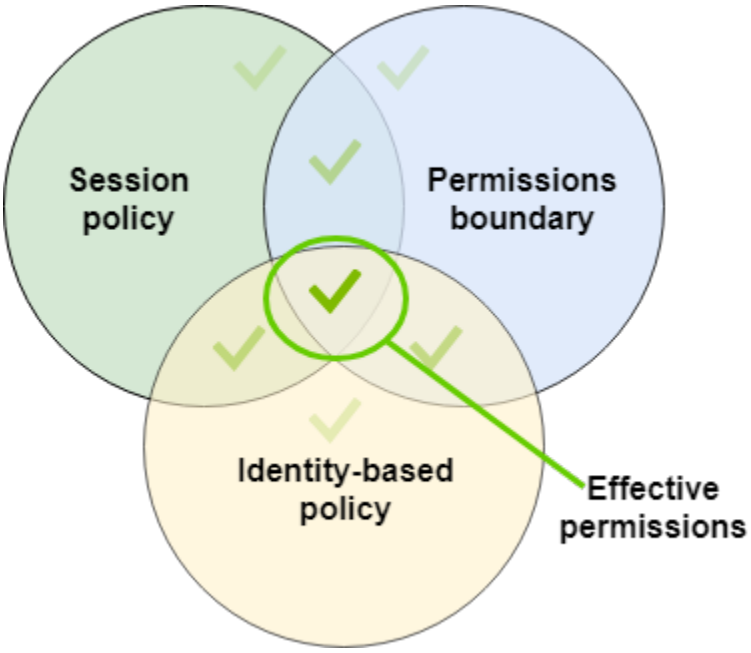
SCP de Organizaciones: las SCP se aplican a toda una cuenta de Cuenta de AWS. Limitan los permisos de todas las solicitudes que realice alguna entidad principal dentro de la cuenta. Una entidad de IAM (usuario o rol) puede realizar una solicitud que se ve afectada por una SCP, un límite de permisos y una política basada en identidad. En este caso, la solicitud se permite solo si los tres tipos de políticas lo permiten. Los permisos efectivos son la intersección de los tres tipos de políticas. Una denegación explícita en cualquiera de estas políticas anulará el permiso.



Puede saber si su cuenta es miembro de una organización en AWS Organizations. Los miembros de la organización podrían verse afectados por una SCP. Para ver estos datos a través del comando AWS CLI u operación de la API de AWS, debe tener permisos para la acción `organizations:DescribeOrganization` para su entidad de Organizations. Debe tener permisos adicionales para realizar la operación en la consola Organizations. Para saber si una SCP deniega el acceso a una solicitud específica o para cambiar los permisos efectivos, póngase en contacto con su administrador de AWS Organizations.

Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de una sesión proceden de la entidad de IAM (usuario o rol) usada para crear la sesión y de la política de sesión. Los permisos que concede la política basada en identidad de la entidad

están limitados por la política de sesión y por el límite de permisos. Los permisos efectivos para este conjunto de tipos de políticas son la intersección de los tres tipos de políticas. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre las políticas de sesión, consulte Políticas de sesión.



Delegación de responsabilidades en otras personas mediante el uso de límites de permisos

Puede utilizar los límites de permisos para delegar tareas de administración de permisos, como, por ejemplo, la creación de usuarios, en los usuarios de IAM de su cuenta. Esto permite que otros puedan realizar tareas en su nombre dentro de un límite de permisos específico.

Por ejemplo, supongamos que María es la administradora de la cuenta de Cuenta de AWS de la empresa X. María quiere delegar las tareas de creación de usuarios en Zhang. Sin embargo, debe asegurarse de que Zhang crea los usuarios ateniéndose a las siguientes reglas de la empresa:

- Los usuarios no pueden utilizar IAM para crear ni administrar usuarios, grupos, roles ni políticas.
- A los usuarios se les deniega el acceso al bucket `logs` de Amazon S3 y no pueden tener acceso a la instancia de Amazon EC2 `i-1234567890abcdef0`.
- Los usuarios no pueden eliminar sus propias políticas de límites.

Para aplicar estas reglas, María realiza las tareas siguientes, cuyos detalles se incluyen a continuación:

1. María crea la política administrada `XCompanyBoundaries` para utilizarla como límite de permisos para todos los usuarios nuevos de la cuenta.
2. María crea la política administrada `DelegatedUserBoundary` y se la asigna a Zhang como límite de permisos. Maria toma nota del ARN del usuario administrador de y lo usa en la política para evitar que Zhang obtenga acceso a él.
3. María crea la política administrada `DelegatedUserPermissions` y se la asocia a Zhang como política de permisos.
4. María informa a Zhang sobre sus nuevas responsabilidades y limitaciones.

Tarea 1: María primero debe crear una política administrada para definir los límites para los nuevos usuarios. María permitirá a Zhang que proporcione a los usuarios las políticas de permisos que necesitan, pero desea que los usuarios estén restringidos. Para ello, crea la siguiente política administrada por el cliente denominada `XCompanyBoundaries`. Esta política hace lo siguiente:

- Otorga a los usuarios acceso completo a varios servicios
- Permite acceso de autoadministración limitado en la consola de IAM. Esto significa que pueden cambiar su contraseña después de iniciar sesión en la consola. No pueden establecer su contraseña inicial. Si desea permitirlo, añada la acción `"*LoginProfile"` a la instrucción `AllowManageOwnPasswordAndAccessKeys`.
- Deniega a los usuarios el acceso al bucket de registros de Amazon S3 o a la Instancia de Amazon EC2 de `i-1234567890abcdef0`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ServiceBoundaries",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*",
                "dynamodb:*"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

Cada instrucción tiene una finalidad específica:

1. La instrucción `ServiceBoundaries` de esta política permite acceso completo a los servicios de AWS especificados. Esto significa que las acciones de un usuario nuevo en estos servicios solamente están limitadas por las políticas de permisos que se han asociado al usuario.

2. La instrucción `AllowIAMConsoleForCredentials` permite el acceso para obtener una lista de todos los usuarios de IAM. Este acceso es necesario para recorrer la página Users (Usuarios) de la AWS Management Console. También permite ver los requisitos de la contraseña de la cuenta, lo que es necesario para cambiar su propia contraseña.
3. La instrucción `AllowManageOwnPasswordAndAccessKeys` les permite a los usuarios administrar únicamente su propia contraseña de la consola y las claves de acceso mediante programación. Esto es importante si Zhang u otro administrador le asigna a un usuario nuevo una política de permisos con acceso completo a IAM. En tal caso, este usuario podría cambiar sus propios permisos o los de otros usuarios. Esta instrucción impide que eso ocurra.
4. La instrucción `DenyS3Logs` deniega explícitamente el acceso al bucket `logs`.
5. La instrucción `DenyEC2Production` deniega explícitamente el acceso a la instancia `i-1234567890abcdef0`.

Tarea 2: María desea permitir que Zhang cree todos los usuarios de X-Company, pero solo con el límite de permisos `XCompanyBoundaries`. Crea la siguiente política administrada por el cliente denominada `DelegatedUserBoundary`. Esta política define los permisos que Zhang puede tener como máximo.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateOrChangeOnlyWithBoundary",
            "Effect": "Allow",
            "Action": [
                "iam:AttachUserPolicy",
                "iam:CreateUser",
                "iam:DeleteUserPolicy",
                "iam:DetachUserPolicy",
                "iam:PutUserPermissionsBoundary",
                "iam:PutUserPolicy"
            ],
            "Resource": "*",
            "Condition": {
               "StringEquals": {
                 "iam:PermissionsBoundary": "arn:aws:iam::123456789012:policy/
XCompanyBoundaries"
                }
            }
```

```
        },
        {
            "Sid": "CloudWatchAndOtherIAMTasks",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:*",
              "iam:CreateAccessKey",
              "iam:CreateGroup",
              "iam:CreateLoginProfile",
              "iam:CreatePolicy",
              "iam:DeleteGroup",
              "iam:DeletePolicy",
              "iam:DeletePolicyVersion",
              "iam:DeleteUser",
              "iam:GetAccountPasswordPolicy",
              "iam:GetGroup",
              "iam:GetLoginProfile",
              "iam:GetPolicy",
              "iam:GetPolicyVersion",
              "iam:GetRolePolicy",
              "iam:GetUser",
              "iam:GetUserPolicy",
              "iam:ListAccessKeys",
              "iam:ListAttachedRolePolicies",
              "iam:ListAttachedUserPolicies",
              "iam:ListEntitiesForPolicy",
              "iam:ListGroups",
              "iam:ListGroupsForUser",
              "iam:ListMFADevices",
              "iam:ListPolicies",
              "iam:ListPolicyVersions",
              "iam:ListRolePolicies",
              "iam:ListSSHPublicKeys",
              "iam:ListServiceSpecificCredentials",
              "iam:ListSigningCertificates",
              "iam:ListUserPolicies",
              "iam:ListUsers",
              "iam:SetDefaultPolicyVersion",
              "iam:SimulateCustomPolicy",
              "iam:SimulatePrincipalPolicy",
              "iam:UpdateGroup",
              "iam:UpdateLoginProfile",
              "iam:UpdateUser"
            ],
```

```
            "NotResource": "arn:aws:iam::123456789012:user/Maria"
        },
        {
            "Sid": "NoBoundaryPolicyEdit",
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:DeletePolicy",
                "iam:DeletePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:policy/XCompanyBoundaries",
                "arn:aws:iam::123456789012:policy/DelegatedUserBoundary"
            ]
        },
        {
            "Sid": "NoBoundaryUserDelete",
            "Effect": "Deny",
            "Action": "iam:DeleteUserPermissionsBoundary",
            "Resource": "*"
        }
    ]
}
```

Cada instrucción tiene una finalidad específica:

1. La instrucción `CreateOrChangeOnlyWithBoundary` permite a Zhang crear usuarios de IAM pero solo si utiliza la política `XCompanyBoundaries` para establecer el límite de permisos. Esta instrucción también le permite configurar el límite de permisos de los usuarios existentes, pero únicamente si utiliza esa misma política. Por último, esta instrucción permite a Zhang administrar las políticas de permisos de los usuarios que tienen configurado este límite de permisos.
2. La instrucción `CloudWatchAndOtherIAMTasks` permite a Zhang realizar otras tareas de administración de usuarios, grupos y políticas. Él tiene permisos para restablecer contraseñas y crear claves de acceso para cualquier usuario de IAM que no aparezca en el elemento de política `NotResource`. Esto le permite ayudar a los usuarios con problemas de inicio de sesión.
3. La instrucción `NoBoundaryPolicyEdit` deniega a Zhang el acceso para actualizar la política `XCompanyBoundaries`. No se le permite modificar ninguna política que se utilice para configurar el límite de permisos para sí mismo ni para los demás usuarios.

4. La instrucción `NoBoundaryUserDelete` deniega a Zhang acceso de eliminación del límite de permisos para sí mismo y para los demás usuarios.

A continuación, María asigna la política `DelegatedUserBoundary` como límite de permisos para el usuario `Zhang`.

Tarea 3: debido a que el límite de permisos limita los permisos máximos, pero no concede acceso por sí solo, María debe crear una política de permisos para Zhang. Crea la siguiente política denominada `DelegatedUserPermissions`. Esta política define las operaciones que Zhang puede realizar, dentro del límite definido.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "IAM",
            "Effect": "Allow",
            "Action": "iam:*",
            "Resource": "*"
        },
        {
            "Sid": "CloudWatchLimited",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetDashboard",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3BucketContents",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::ZhangBucket"
        }
    ]
}
```

Cada instrucción tiene una finalidad específica:

1. La instrucción `IAM` de la política permite a Zhang acceso completo a IAM. No obstante, puesto que su límite de permisos permite únicamente algunas operaciones de IAM, sus permisos efectivos de IAM solamente se ven limitados por dicho límite de permisos.
2. La instrucción `CloudWatchLimited` permite a Zhang realizar cinco acciones en CloudWatch. Su límite de permisos permite todas las acciones de CloudWatch, por lo que sus permisos efectivos de CloudWatch solamente se ven limitados por su política de permisos.
3. La instrucción `S3BucketContents` permite a Zhang mostrar el bucket de Amazon S3 `ZhangBucket`. Sin embargo, como su límite de permisos no permite realizar ninguna acción de Amazon S3, no puede realizar operaciones de S3, independientemente de su política de permisos.

> **Note**
>
> Las políticas de Zhang le permiten crear un usuario que pueda obtener acceso a recursos de Amazon S3 a los que él no tiene acceso. Al delegar estas acciones administrativas, Maria establece una relación de confianza con Zhang con acceso a Amazon S3.

A continuación, María asocia la política `DelegatedUserPermissions` como política de permisos para el usuario `Zhang`.

Tarea 4: proporciona a Zhang instrucciones para crear usuarios. Le indica que puede crear usuarios con los permisos que necesiten, pero que debe asignarles la política `XCompanyBoundaries` como límite de permisos.

Zhang realiza las tareas siguientes:

1. Zhang crea un usuario con la AWS Management Console. Escribe el nombre de usuario `Nikhil` y le concede acceso a la consola. Desmarque la casilla de verificación situada junto a Requiere restablecimiento de contraseña, ya que las políticas anteriores solo permiten cambiar las contraseñas a los usuarios después de haber iniciado sesión en la consola de IAM.
2. En la página Set permissions (Establecer permisos), Zhang elige las políticas de permisos IAMFullAccess y AmazonS3ReadOnlyAccess que permiten a Nikhil hacer su trabajo.
3. Zhang omite la sección Set permissions boundary (Configurar límite de permisos), olvidando las instrucciones de María.
4. Zhang revisa los detalles del usuario y elige Create user (Crear usuario).

Se produce un error en la operación y se deniega el acceso. El límite de permisos `DelegatedUserBoundary` de Zhang requiere que cualquier usuario que cree utilice la política `XCompanyBoundaries` como límite de permisos.

5. Zhang vuelve a la página anterior. En la sección, Set permissions boundary (Configurar límite de permisos) elige la política `XCompanyBoundaries`.
6. Zhang revisa los detalles del usuario y elige Create user (Crear usuario).

   El usuario se crea.

Cuando Nikhil inicia sesión, tiene acceso a IAM y a Amazon S3, salvo para las operaciones denegadas por el límite de permisos. Por ejemplo, puede cambiar su propia contraseña en IAM, pero no puede crear otro usuario ni editar sus políticas. Nikhil tiene acceso de solo lectura a Amazon S3.

Si alguien añade una política basada en recursos al bucket de `logs` que permite que Nikhil coloque un objeto en el bucket, aun así no puede obtener acceso al bucket. El motivo es que el límite de permisos deniega explícitamente cualquier acción sobre el bucket de `logs`. Una denegación explícita en cualquier tipo de política hace que la solicitud se deniegue. Sin embargo, si una política basada en recursos asociada a un secreto de Secrets Manager permite a Nikhil ejecutar la acción `secretsmanager:GetSecretValue`, entonces Nikhil podrá obtener y descifrar el secreto. Esto se debe a que el límite de permisos no deniega explícitamente las operaciones de Secrets Manager, y las denegaciones implícitas de los límites de permisos no restringen las políticas basadas en recursos.

Políticas basadas en identidad y políticas basadas en recursos

Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. Al crear una política de permisos para restringir el acceso a un recurso, puede elegir una política basada en identidad o una política basada en recursos.

Las políticas basadas en identidad se asocian a un usuario, grupo o rol de IAM. Estas políticas le permiten especificar lo que esa identidad puede hacer (sus permisos). Por ejemplo, puede asociar la política a un usuario de IAM llamado John, indicando que tiene permiso para utilizar la acción `RunInstances` de Amazon EC2. La política podría indicar también que John tiene permiso para obtener elementos de una tabla de Amazon DynamoDB denominada `MyCompany`. También puede permitir a John administrar sus propias credenciales de seguridad de IAM. Las políticas basadas en la identidad pueden ser administradas o insertadas.

Las políticas basadas en recursos se asocian a un recurso. Por ejemplo, puede asociar políticas basadas en recursos a buckets de Amazon S3, colas de Amazon SQS, puntos de conexión de VPC, claves de cifrado de AWS Key Management Service y tablas y secuencias de Amazon DynamoDB. Para obtener una lista de las políticas que admiten los permisos basados en recursos, consulte Servicios de AWS que funcionan con IAM.

Con las políticas basadas en recursos, puede especificar quién tiene acceso al recurso y qué acciones puede realizar en él. Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles. Las políticas basadas en recursos solo son insertadas, no se administran.

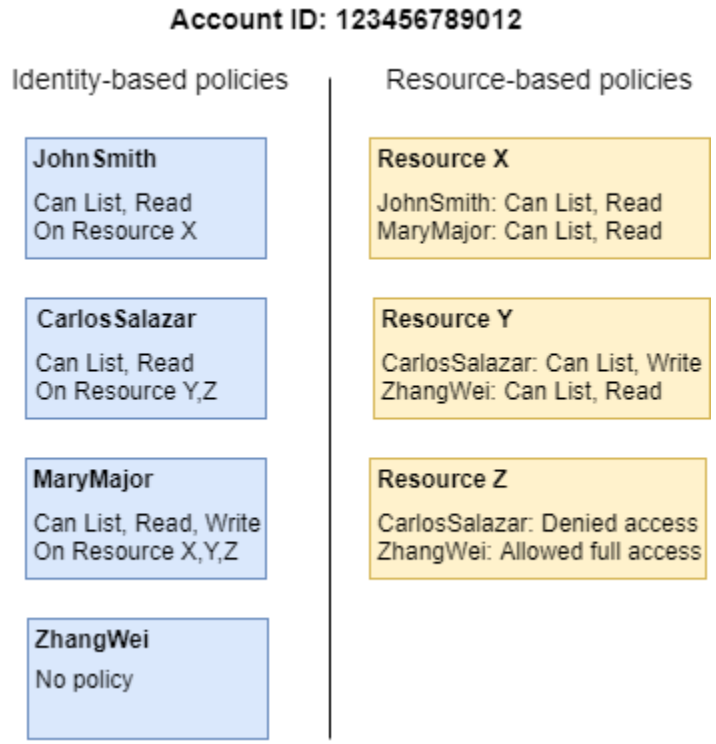
> **Note**
>
> Las políticas basadas en recursos difieren de los permisos en el nivel de recursos. Puede asociar políticas basadas en recursos directamente a un recurso, tal y como se describe en este tema. Los permisos de nivel de recursos se refieren a la capacidad de utilizar ARN para especificar recursos individuales en una política. Las políticas basadas en recursos solo se admiten en algunos servicios de AWS. Para obtener una lista de los servicios que admiten las políticas basadas en recursos y de nivel de recursos, consulte Servicios de AWS que funcionan con IAM.

Para obtener información sobre cómo interactúan las políticas basadas en identidad y las políticas basadas en recursos dentro de la misma cuenta, consulte Evaluación de políticas dentro de una misma cuenta.

Para obtener información sobre cómo interactúan las políticas entre las cuentas, consulte Lógica de evaluación de políticas entre cuentas.

Para comprender mejor estos conceptos, vea la siguiente ilustración. El administrador de la cuenta `123456789012` asociada a las políticas basadas en identidad a los usuarios `JohnSmith`, `CarlosSalazar` y `MaryMajor`. Algunas de las acciones de estas políticas pueden realizarse en recursos específicos. Por ejemplo, el usuario `JohnSmith` puede realizar algunas acciones en `Resource X`. Se trata de un permiso de nivel de recursos en una política basada en la identidad. El administrador también ha añadido políticas basadas en recursos a `Resource X`, `Resource Y` y `Resource Z`. Las políticas basadas en recursos le permiten especificar quién puede acceder a ese

recurso. Por ejemplo, la política basada en recursos en `Resource X` permite a la lista de usuarios `JohnSmith` y `MaryMajor` acceso de lectura al recurso.



El ejemplo de cuenta `123456789012` permite a los siguientes usuarios realizar las acciones indicadas:

- JohnSmith – John puede realizar acciones de lista y lectura en `Resource X`. Se le concede este permiso mediante la política basada en la identidad en su usuario y la política basada en recursos en `Resource X`.
- CarlosSalazar – Carlos puede realizar acciones de lista, lectura y escritura en `Resource Y`, pero se deniega el acceso a `Resource Z`. La política basada en la identidad en Carlos le permite realizar acciones de lista y lectura en `Resource Y`. La política basada en recursos `Resource Y` le permite también permisos de escritura. Sin embargo, aunque su política basada en la identidad le permite el acceso a `Resource Z`, la política basada en recursos `Resource Z` deniega ese tipo de acceso. Una denegación `Deny` explícita anula un permiso `Allow`, lo que le impide el acceso a `Resource Z`. Para obtener más información, consulte Lógica de evaluación de políticas.
- MaryMajor – Mary puede realizar operaciones de lista, lectura y escritura en `Resource X`, `Resource Y` y `Resource Z`. Su política basada en la identidad le permite más acciones en más recursos de las políticas basadas en recursos, pero ninguno de ellos deniega el acceso.

- ZhangWei – Zhang tiene acceso total a `Resource Z`. Zhang no tiene políticas basadas en la identidad, pero la política basada en recursos `Resource Z` le permite el acceso completo al recurso. Zhang también puede realizar acciones de lista y lectura en `Resource Y`.

Las políticas basadas en la identidad y las políticas basadas en recursos son políticas de permisos y se evalúan juntas. Para una solicitud a la que se aplican solo políticas de permisos, AWS en primer lugar comprueba todas las políticas de `Deny`. Si existe, se deniega la solicitud. A continuación, AWS comprueba cada permiso `Allow`. Si al menos una instrucción de la política permite la acción en la solicitud, la solicitud se permite. No importa si el permiso `Allow` se encuentra en la política basada en identidad o en la política basada en recursos.

> **Important**
>
> Esta lógica solo se aplica cuando la solicitud se realiza dentro de una cuenta única de Cuenta de AWS. Para las solicitudes realizadas de una cuenta a otra, el solicitante de la cuenta `Account A` debe tener una política basada en identidad que le permita realizar una solicitud al recurso en la cuenta `Account B`. Además, la política basada en recursos en `Account B` debe permitir al solicitante en `Account A` obtener acceso al recurso. Debe haber políticas en ambas cuentas que permitan la operación; de lo contrario, la solicitud producirá un error. Para obtener más información acerca del uso de políticas basadas en recursos para acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM.

Un usuario que tenga permisos específicos puede solicitar un recurso que también tenga una política de permisos asociada. En ese caso, AWS evalúa ambos conjuntos de permisos al determinar si debe conceder acceso al recurso. Para obtener información sobre cómo se evalúan las políticas, consulte Lógica de evaluación de políticas.

> **Note**
>
> Amazon S3 admite las políticas basadas en identidad y las políticas basadas en recursos (a las que se denomina políticas de bucket). Además, Amazon S3 es compatible con un mecanismo de permiso conocido como una lista de control de acceso (ACL) independiente de las políticas y los permisos de IAM. Puede utilizar políticas de IAM combinadas con ACL de Amazon S3. Para obtener más información, consulte Control de acceso en la Guía del usuario de Amazon Simple Storage Service.

Controlar el acceso a los recursos de AWS mediante políticas.

Puede utilizar una política para controlar el acceso a los recursos de IAM o de todo AWS.

Para utilizar una política para controlar el acceso en AWS, debe entender cómo AWS otorga acceso. AWS se compone de colecciones de recursos. Un usuario de IAM es un recurso. Un bucket de Amazon S3 es un recurso. Cuando se utiliza la API de AWS, la AWS CLI o la AWS Management Console para realizar una operación (como, por ejemplo, crear un usuario), se envía una solicitud para dicha operación. La solicitud especifica una acción, un recurso, una entidad principal (usuario o rol), una cuenta principal y toda la información de la solicitud necesaria. Toda esta información proporciona el contexto.

AWS comprueba entonces que usted (la entidad principal) se ha autenticado (ha iniciado sesión) y está autorizado (tiene permiso) para realizar la acción especificada en el recurso especificado. Durante la autorización, AWS comprueba todas las políticas aplicables al contexto de la solicitud. La mayoría de las políticas se almacenan en AWS como documentos JSON y especifican los permisos de las entidades principales. Para obtener más información sobre los tipos de políticas y sus usos, consulte Políticas y permisos en IAM.

AWS autoriza la solicitud únicamente si cada parte de la solicitud está permitida por las políticas. Para ver un diagrama de este proceso, consulte Cómo funciona IAM. Para obtener información detallada acerca de cómo AWS determina si una solicitud está permitida, consulte Lógica de evaluación de políticas.

Cuando crea una política de IAM, puede controlar el acceso a lo siguiente:

- Entidades principales – Controle qué puede hacer la persona que realiza la solicitud (la entidad principal).
- Identidades de IAM – Controle a qué identidades de IAM (grupos de usuarios, usuarios y roles) se puede tener acceso y cómo.
- Políticas de IAM – Controle quién puede crear, editar y eliminar políticas administradas por el cliente y quién puede asociar y desasociar todas las políticas administradas.
- Recursos de AWS – Controle quién tiene acceso a los recursos a través de una política basada en identidad o una política basada en recursos.
- Cuentas de AWS – Controle si una solicitud se permite únicamente para los miembros de una cuenta determinada.

Las políticas le permiten especificar quién tiene acceso a recursos de AWS y qué acciones pueden realizar en dichos recursos. Todos los usuarios de IAM empiezan sin permisos. En otras palabras, de forma predeterminada, los usuarios no pueden hacer nada, ni siquiera consultar sus propias claves de acceso. Para proporcionar a un usuario permiso para hacer algo, puede añadir el permiso al usuario (es decir, asociar una política al usuario). También puede agregar el usuario a un grupo de usuarios que tenga el permiso necesario.

Por ejemplo, puede conceder a un usuario permiso para generar una lista de sus propias claves de acceso. También puede ampliar ese permiso y permitir que cada usuario cree, actualice y elimine sus propias claves.

Cuando concede permisos a un grupo de usuarios, todos los usuarios de ese grupo de usuarios obtienen los permisos. Por ejemplo, puede dar permiso al grupo de usuarios Administrators para que realice cualquiera de las acciones de IAM con cualquiera de los recursos de la cuenta de Cuenta de AWS. Otro ejemplo: puede dar permiso al grupo de usuarios de administradores para describir las instancias de Amazon EC2 de la cuenta de Cuenta de AWS.

Para obtener información sobre cómo delegar permisos básicos a sus usuarios, grupos de usuarios y roles, consulte Permisos obligatorios para obtener acceso a recursos de IAM. Para ver ejemplos de políticas adicionales que ilustran estos permisos básicos, consulte Ejemplos de políticas para administrar recursos de IAM.

Control del acceso para entidades principales de

Puede utilizar políticas para controlar qué puede hacer la persona que realiza la solicitud (la entidad principal). Para ello, debe asociar una política basada en identidad a la identidad de esa persona (usuario, grupo de usuarios o rol). También puede utilizar un límite de permisos que establezca los permisos máximos que una entidad (usuario o rol) puede tener.

Por ejemplo, suponga que desea que el usuario Zhang Wei tenga acceso completo a CloudWatch, Amazon DynamoDB, Amazon EC2, y Amazon S3. Puede crear dos políticas diferentes para poder dividirlas más adelante si necesita un conjunto de permisos para un usuario distinto. O puede incluir los dos permisos en una única política y, a continuación, asociar dicha política al usuario de IAM llamado Zhang Wei. También podría asociar una política a un grupo de usuarios al que pertenezca Zhang o a un rol que Zhang pueda asumir. Como resultado, cuando Zhang vea el contenido de un bucket de S3, se permitirán sus solicitudes. Si intenta crear un usuario de IAM, se denegará su solicitud porque no tiene permiso.

Puede utilizar un límite de permisos con Zhang para asegurarse de que nunca se le concede acceso al bucket de S3 *DOC-EXAMPLE-BUCKET1*. Para ello, determine los permisos que desea que Zhang

tenga como máximo. En este caso, puede controlar lo que hace con sus políticas de permisos. Aquí, solo le importa que no tenga acceso al bucket confidencial. Por lo tanto, utilice la siguiente política para definir el límite de Zhang de forma que se le permitan todas las acciones de AWS para Amazon S3 y algunos otros servicios, pero se le deniegue el acceso al bucket de S3 *DOC-EXAMPLE-BUCKET1*. Debido a que el límite de permisos no permite ninguna acción de IAM, evita que Zhang elimine su límite (o el de cualquier otra persona).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsBoundarySomeServices",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "dynamodb:*",
                "ec2:*",
                "s3:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsBoundaryNoConfidentialBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
    ]
}
```

Cuando asigne una política de este tipo como un límite de permisos para un usuario, recuerde que no concede ningún permiso. Solo establece los permisos máximos que una política basada en la identidad puede conceder a una entidad de IAM. Para obtener más información sobre los límites de permisos, consulte Límites de permisos para las entidades de IAM.

Para obtener información detallada sobre los procedimientos mencionados anteriormente, consulte estos recursos:

- Para obtener más información sobre cómo crear una política de IAM que pueda asociar a una entidad principal, consulte Crear políticas de IAM.
- Para obtener información sobre cómo asociar una política de IAM a una entidad principal, consulte Adición y eliminación de permisos de identidad de IAM.
- Para ver una política de ejemplo que concede acceso completo a EC2, consulte Amazon EC2: permite el acceso completo a EC2 en una región determinada, mediante programación y en la consola.
- Para permitir el acceso de solo lectura a un bucket de S3, utilice las dos primeras instrucciones de la siguiente política de ejemplo: Amazon S3: permite el acceso de lectura y escritura a objetos en un bucket de S3 mediante programación y en la consola.
- Para ver una política de ejemplo a fin de permitirles a los usuarios definir sus credenciales, como la contraseña de la consola, las claves de acceso mediante programación y los dispositivos MFA, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

Control del acceso a identidades

Puede utilizar políticas de IAM para controlar lo que los usuarios pueden hacer con una identidad creando una política y asociándola a todos los usuarios mediante un grupo de usuarios. Para ello, cree una política que limite lo que se puede hacer en una identidad o quién puede tener acceso a ella.

Por ejemplo, puede crear un grupo de usuarios llamado AllUsers y, a continuación, asociar ese grupo de usuarios a todos los usuarios. Cuando crea el grupo de usuarios, puede otorgarle acceso a todos los usuarios para que definan sus credenciales como se describe en la sección anterior. A continuación, puede crear una política que deniegue el acceso para cambiar el grupo de usuarios a menos que el nombre de usuario se incluya en la condición de la política. Sin embargo, esa parte de la política solo deniega el acceso a todos los usuarios menos a los indicados. También debe incluir permisos para permitir todas las acciones de administración del grupo de usuarios a todos los miembros del grupo de usuarios. Por último, debe asociar esta política al grupo de usuarios de forma que se aplique a todos los usuarios. Como resultado, cuando un usuario no especificado en la política intente realizar cambios en el grupo de usuarios, se denegará la solicitud.

Para crear esta política con el editor visual

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación de la izquierda, elija Policies (Políticas).

   Si es la primera vez que elige Policies (Políticas), aparecerá la página Welcome to Managed Policies (Bienvenido a políticas administradas). Elija Get Started (Comenzar).
3. Elija Create Policy (Crear política).
4. En la sección Editor de políticas, seleccione la opción Visual.
5. En Seleccionar un servicio, seleccione IAM.
6. En Acciones permitidas, escriba **`group`** en el cuadro de búsqueda. El editor visual muestra todas las acciones de IAM que contienen la palabra `group`. Seleccione todas las casillas de verificación.
7. Elija Resources (Recursos) para especificar recursos para su política. En función de las acciones que haya elegido, debería ver los tipos de recursos grupo y usuario.

   - grupo: seleccione Agregar ARN. En Recurso en, seleccione la opción Cualquier cuenta. Seleccione la casilla de verificación Cualquier nombre de grupo con ruta y, a continuación, escriba el nombre de grupo de usuarios **`AllUsers`**. A continuación, seleccione Agregar ARN.
   - usuario: seleccione la casilla de verificación situada junto a Cualquiera de esta cuenta.

   Una de las acciones que ha elegido, `ListGroups`, no permite el uso de recursos específicos. No tiene que elegir All resources (Todos los recursos) para esa acción. Cuando guarde la política o la visualice en el editor JSON, podrá ver que IAM crea automáticamente un nuevo bloque de permisos que conceden a esta acción permiso en todos los recursos.
8. Para agregar otro bloque de permisos, seleccione Agregar más permisos.
9. Seleccione Seleccionar un servicio, y luego IAM.
10. Seleccione Acciones permitidas, y después Cambiar a denegar permisos. Cuando termine, se utilizará todo el bloque para denegar permisos.
11. Escriba **`group`** en el cuadro de búsqueda. El editor visual muestra todas las acciones de IAM que contienen la palabra `group`. Seleccione las casillas situadas junto a las siguientes acciones:

    - CreateGroup
    - DeleteGroup
    - RemoveUserFromGroup
    - AttachGroupPolicy
    - DeleteGroupPolicy

- DetachGroupPolicy
- PutGroupPolicy
- UpdateGroup

12. Elija Resources (Recursos) para especificar los recursos para su política. En función de las acciones que ha elegido, debería ver el tipo de recurso group (grupo). Seleccione Agregar ARN. En Recurso en, seleccione la opción Cualquier cuenta. En Cualquier nombre de grupo con ruta, escriba el nombre de grupo de usuarios **`AllUsers`**. A continuación, seleccione Agregar ARN.
13. Seleccione Solicitar condiciones: opcional, y luego Agregar otra condición. Complete el formulario con los siguientes valores:

    - Clave de condición: seleccione aws:username
    - Calificador - Elija Predeterminado
    - Operador - Elija StringNotEquals
    - Valor: escriba **`srodriguez`** y después seleccione Agregar para agregar otro valor. Escriba **`mjackson`** y después seleccione Agregar para agregar otro valor. Escriba **`adesai`** y después seleccione Agregar condición.

    Esta condición garantiza que se denegará el acceso a las acciones de administración del grupo de usuarios especificadas cuando el usuario que realice la llamada no esté incluido en la lista. Dado que este permiso deniega el permiso de forma explícita, invalida el bloque anterior que permitía a los usuarios llamar a las acciones. A los usuarios de la lista no se les denegará el acceso y se les concederá permiso en el primer bloque de permisos, por lo que podrán administrar totalmente el grupo de usuarios.
14. Cuando haya terminado, elija Next (Siguiente).

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

15. En la página Revisar y crear, en Nombre de la política, escriba **`LimitAllUserGroupManagement`**. En Description (Descripción), escriba **`Allows all users read-only access to a specific user group, and allows only`**

`specific users access to make changes to the user group`. Revise los Permisos definidos en esta política para asegurarse de que ha concedido los permisos deseados. A continuación, seleccione Create policy (Crear política) para guardar la nueva política.

16. Asocie la política al grupo de usuarios. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

También puede crear la misma política utilizando este documento de política JSON de ejemplo. Para consultar esta política de JSON, visite IAM: permite que usuarios específicos de IAM administren un grupo, mediante programación y en la consola. Para obtener instrucciones detalladas para crear una política utilizando un documento JSON, consulte the section called “Creación de políticas mediante el editor JSON”.

Control del acceso a políticas

Puede controlar la forma en que los usuarios pueden aplicar políticas administradas por AWS. Para ello, asocie esta política a todos los usuarios. Lo ideal sería hacer esto mediante un grupo de usuarios.

Por ejemplo, podría crear una política que permitiera a los usuarios asociar solo las políticas IAMUserChangePassword y PowerUserAccess administradas por AWS a un nuevo usuario, grupo de usuarios o rol de IAM.

Para las políticas administradas por el cliente, puede controlar quién puede crear, actualizar y eliminar estas políticas. Puede controlar quién puede asociar y desvincular políticas de entidades principales (grupos de usuarios, usuarios y roles). También puede controlar las políticas que un usuario puede asociar a determinadas entidades, o bien desasociarlas.

Por ejemplo, puede conceder permisos a un administrador de la cuenta para crear, actualizar y eliminar políticas. A continuación, puede conceder permisos a un jefe de equipo o a otro administrador con permisos limitados para asociar estas políticas a entidades principales, o bien desvincularlas, que dicho administrador administra.

Para obtener más información, consulte estos recursos:

- Para obtener más información sobre cómo crear una política de IAM que pueda asociar a una entidad principal, consulte Crear políticas de IAM.
- Para obtener información sobre cómo asociar una política de IAM a una entidad principal, consulte Adición y eliminación de permisos de identidad de IAM.

- Para ver un ejemplo de política para limitar el uso de las políticas administradas, consulte IAM: limita las políticas administradas que pueden aplicarse a un usuario, grupo o rol de .

Control de permisos para crear, actualizar y eliminar políticas administradas por el cliente

Puede utilizar las políticas de IAM para controlar quién puede crear, actualizar y eliminar las políticas administradas por el cliente en su cuenta de Cuenta de AWS. En la siguiente lista se incluyen las operaciones de API relacionadas directamente con la creación, actualización y eliminación de políticas o versiones de políticas:

- CreatePolicy
- CreatePolicyVersion
- DeletePolicy
- DeletePolicyVersion
- SetDefaultPolicyVersion

Las operaciones de la API de la lista anterior corresponden a acciones que puede permitir o denegar es decir, permisos que puede conceder con una política de IAM.

Considere la política de ejemplo siguiente. Permite a un usuario crear, actualizar (es decir, crear una nueva versión de la política), eliminar y configurar una versión predeterminada para todas las políticas administradas por el cliente de una cuenta de Cuenta de AWS. El ejemplo de política también permite a los usuarios enumerar políticas y obtener políticas. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

Example Ejemplo de política que permite crear, actualizar, eliminar, enumerar, obtener y configurar la versión predeterminada de todas las políticas

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:CreatePolicy",
      "iam:CreatePolicyVersion",
      "iam:DeletePolicy",
      "iam:DeletePolicyVersion",
```

```
      "iam:GetPolicy",
      "iam:GetPolicyVersion",
      "iam:ListPolicies",
      "iam:ListPolicyVersions",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "*"
  }
}
```

Puede crear políticas que limiten el uso de estas operaciones de API de modo que solo afecten a las políticas administradas que especifique. Por ejemplo, es posible que quiera permitir a un usuario configurar la versión predeterminada y eliminar las versiones de políticas, pero solo para determinadas políticas administradas por el cliente. Para ello, especifique el ARN de la política en el elemento `Resource` de la política que concede dichos permisos.

En el siguiente ejemplo, se muestra una política que permite a un usuario eliminar versiones de políticas y configurar la versión predeterminada. Sin embargo, estas acciones solo están permitidas para las políticas administradas por el cliente que incluyan la ruta /TEAM-A. El ARN de la política administrada por el cliente se especifica en el elemento `Resource` de la política. (En este ejemplo, el ARN incluye una ruta y un carácter comodín y, por tanto, coincide con todas las políticas administradas por el cliente que incluyan la ruta /TEAM-A/). Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

Para obtener más información sobre cómo utilizar las rutas de acceso en los nombres de las políticas administradas por el cliente, consulte Nombres fáciles de recordar y rutas.

Example Ejemplo de política que permite eliminar las versiones de políticas y configurar la versión predeterminada únicamente para políticas específicas

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id:policy/TEAM-A/*"
  }
```

```
}
```

Control de permisos para asociar políticas administradas y desasociarlas

También puede utilizar políticas de IAM para permitir a los usuarios trabajar solo con determinadas políticas administradas. De hecho, puede controlar los permisos que un usuario puede conceder a otras entidades principales.

En la siguiente lista se muestran las operaciones de API relacionadas directamente con la asociación de políticas administradas a entidades principales y su desvinculación:

- AttachGroupPolicy
- AttachRolePolicy
- AttachUserPolicy
- DetachGroupPolicy
- DetachRolePolicy
- DetachUserPolicy

Puede crear políticas que limiten el uso de estas operaciones de API de forma que solo afecten a las políticas administradas o entidades principales especificadas. Por ejemplo, es posible que quiera permitir a un usuario asociar políticas administradas, pero solo las que usted especifique. O bien, es posible que quiera permitir a un usuario asociar políticas administradas, pero solo a las entidades principales que usted especifique.

En el siguiente ejemplo de política se permite a un usuario asociar políticas administradas a únicamente los grupos de usuarios y roles que incluyan la ruta de acceso /TEAM-A/. Los ARN del grupo de usuarios y del rol se especifican en el elemento `Resource` de la política. (En este ejemplo, los ARN incluyen una ruta y un carácter comodín y, por lo tanto, coinciden con todos los grupos de usuarios y roles que contienen la ruta /TEAM-A/). Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

Example Ejemplo de política que permite asociar políticas administradas únicamente a determinados grupos de usuarios o roles

```
{
  "Version": "2012-10-17",
```

```
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ]
  }
}
```

Puede limitar incluso más las acciones del ejemplo anterior para que afecten únicamente a determinadas políticas. Es decir, puede controlar los permisos que un usuario puede asociar a otras entidades principales mediante la adición de una condición a la política.

En el siguiente ejemplo, la condición garantiza que los permisos `AttachGroupPolicy` y `AttachRolePolicy` solo están permitidos cuando la política que se asocia coincide con una de las políticas especificadas. La condición utiliza la `iam:PolicyARN` clave de condición para determinar la política o políticas que pueden asociarse. En la política de ejemplo siguiente, se amplía el ejemplo anterior. Se permite a un usuario asociar únicamente las políticas administradas que incluyen la ruta /TEAM-A/ únicamente a los grupos de usuarios y roles que incluyan la ruta /TEAM-A/. Para obtener información sobre cómo crear una política mediante este documento de política JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id:policy/TEAM-A/*"}
    }
  }
```

```
}
```

Esta política utiliza el operador de condición `ArnLike`, pero también puede utilizar el operador de condición `ArnEquals` porque estos dos operadores de condición se comportan de forma idéntica. Para obtener más información sobre los tipos de condición `ArnLike` y `ArnEquals`, consulte Operadores de condición de nombre de recurso de Amazon (ARN) en la sección Tipos de condición de la Referencia de los elementos de la política.

Por ejemplo, puede limitar el uso de estas acciones para involucrar únicamente a las políticas administradas que especifique. Para ello, especifique el ARN de la política en el elemento `Condition` de la política que concede dichos permisos. Por ejemplo, para especificar el ARN de una política administrada por el cliente:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::123456789012:policy/POLICY-NAME"}
}
```

También puede especificar el ARN de una política administrada por AWS en el elemento `Condition` de una política. El ARN de una política administrada por AWS utiliza el alias especial `aws` en el ARN de la política, en lugar de un ID de cuenta, tal y como se indica en este ejemplo:

```
"Condition": {"ArnEquals":
  {"iam:PolicyARN": "arn:aws:iam::aws:policy/AmazonEC2FullAccess"}
}
```

Control del acceso a los recursos

Puede controlar el acceso a los recursos a través de una política basada en la identidad o una política basada en recursos. En una política basada en la identidad, la política se asocia a una identidad y se especifica a qué recursos tiene acceso dicha identidad. En una política basada en recursos, se asocia una política al recurso que desea controlar. En la política, especifica las entidades principales que pueden tener acceso a dicho recurso. Para obtener más información sobre ambos tipos de políticas, consulte Políticas basadas en identidad y políticas basadas en recursos.

Para obtener más información, consulte estos recursos:

- Para obtener más información sobre cómo crear una política de IAM que pueda asociar a una entidad principal, consulte Crear políticas de IAM.

- Para obtener información sobre cómo asociar una política de IAM a una entidad principal, consulte Adición y eliminación de permisos de identidad de IAM.
- Amazon S3 admite el uso de políticas basadas en recursos en sus buckets. Para obtener más información, consulte Ejemplos de política de bucket.

Los creadores de recursos no tienen automáticamente permisos

Si inicia sesión con las credenciales de Usuario raíz de la cuenta de AWS, tendrá permiso para realizar cualquier acción en los recursos que pertenecen a la cuenta. Sin embargo, esto no es cierto en el caso de los usuarios de IAM. Puede que a un usuario de IAM se le haya concedido permiso para obtener acceso a un recurso, pero los permisos de usuario, incluso sobre ese recurso, se limitan a lo que se ha concedido explícitamente. Esto significa que solo por el hecho de crear un recurso, como un rol de IAM, no se le concede automáticamente permiso para editar o eliminar dicho rol. Además, el propietario de la cuenta u otro usuario al que se haya concedido acceso para administrar sus permisos pueden revocar su permiso en cualquier momento.

Control del acceso a entidades principales en una cuenta específica

Puede conceder directamente a los usuarios de IAM de su propia cuenta acceso a los recursos. Si hay usuarios de otra cuenta que necesitan tener acceso a sus recursos, puede crear un rol de IAM. Un rol es una entidad que incluye permisos, pero que no está asociada a un usuario concreto. Los usuarios de otras cuentas pueden asumir el rol y obtener acceso a recursos en función de los permisos que haya asignado al rol. Para obtener más información, consulte Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS propia.

> **Note**
>
> Algunos servicios de admiten políticas basadas en recursos como se describe en Políticas basadas en identidad y políticas basadas en recursos (como Amazon S3, Amazon SNS y Amazon SQS). Para esos servicios, una alternativa al uso de roles es adjuntar una política al recurso (bucket, tema o cola) que desea compartir. La política basada en recursos puede especificar la cuenta de AWS que tenga permisos para obtener acceso al recurso.

Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas

Utilice la información de la siguiente sección para controlar quién puede obtener acceso a los usuarios y roles de IAM y a qué recursos pueden acceder. Para obtener información general y

ejemplos sobre el control de acceso a otros recursos de AWS, incluidos otros recursos de IAM, consulte Etiquetado de recursos de IAM.

> **Note**
>
> Para obtener más información sobre la distinción entre mayúsculas y minúsculas en las claves de etiqueta y los valores de las claves de etiqueta, consulte Case sensitivity.

Las etiquetas se pueden asociar al recurso de IAM, pasar en la solicitud o asociar a la entidad principal que realiza la solicitud. Un usuario o rol de IAM puede ser tanto un recurso como una entidad principal. Por ejemplo, puede escribir una política que permita a un usuario enumerar los grupos de un usuario. Esta operación solo se permite si el usuario que realiza la solicitud (entidad principal) tiene la misma etiqueta `project=blue` que el usuario que está intentando ver. En este ejemplo, el usuario puede ver la pertenencia a grupos de cualquier usuario, incluidos ellos mismos, siempre y cuando trabajen en el mismo proyecto.

Para controlar el acceso según las etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política. Al crear una política de IAM, puede utilizar las etiquetas de IAM y la clave de condición de etiqueta asociada para controlar el acceso a cualquiera de las siguientes operaciones:

- Recurso – Permite controlar el acceso a los recursos de usuario o rol en función de sus etiquetas. Para ello, utilice la clave de condición aws:ResourceTag/***key-name*** para especificar qué par de clave-valor de etiqueta debe adjuntarse al recurso. Para obtener más información, consulte Control del acceso a los recursos de AWS.
- Solicitud – Controla las etiquetas que se pueden pasar en una solicitud de IAM. Para ello, utilice la clave de condición aws:RequestTag/***key-name*** para especificar qué etiquetas se pueden agregar, cambiar o eliminar de un usuario o rol de IAM. Esta clave se utiliza de la misma forma para los recursos de IAM y otros recursos de AWS. Para obtener más información, consulte Control del acceso durante solicitudes de AWS.
- Entidad principal - Permite controlar qué puede hacer la persona que realiza la solicitud (entidad principal) en función de las etiquetas que se asocian al usuario o rol de IAM. Para ello, utilice la clave de condición aws:PrincipalTag/***key-name*** para especificar qué etiquetas se deben asociar al usuario o rol de IAM antes de que se admita la solicitud.
- Cualquier parte del proceso de autorización: utilice la clave de condición aws:TagKeys para controlar si se pueden incluir claves de etiqueta específicas en una solicitud o por una entidad

principal. En este caso, el valor de clave no importa. Esta clave funciona de manera similar para IAM y otros servicios de AWS. Sin embargo, al etiquetar a un usuario en IAM, esto también controla si la entidad principal puede realizar la solicitud a cualquier servicio. Para obtener más información, consulte Control del acceso en función de las claves de etiqueta.

Puede crear una política de IAM utilizando el editor visual, con JSON o importando una política administrada existente. Para obtener más información, consulte Crear políticas de IAM.

> **Note**
>
> También puede pasar etiquetas de sesión al asumir un rol de IAM o federar un usuario. Son válidas solo para la duración de la sesión.

Control del acceso para entidades principales de IAM

Puede controlar lo que puede hacer la entidad principal en función de las etiquetas asociadas a la identidad de esa persona.

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a cualquier usuario de esta cuenta ver la pertenencia al grupo de cualquier usuario, incluido él mismo, siempre que estén trabajando en el mismo proyecto. Esta operación solo se permite cuando la etiqueta del recurso del usuario y la etiqueta de la entidad principal tienen el mismo valor para la clave de la etiqueta `project`. Para utilizar esta política, sustituya el *`texto en cursiva del marcador`* de la política de ejemplo con su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "iam:ListGroupsForUser",
            "Resource": "arn:aws:iam::111222333444:user/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/project":
 "${aws:PrincipalTag/project}"}
            }
```

```
        }]
}
```

Control del acceso en función de las claves de etiqueta

Puede utilizar etiquetas en sus políticas de IAM para controlar si una solicitud o una entidad principal puede usar determinadas claves de etiqueta.

Este ejemplo muestra cómo podría crear una política basada en identidad que permita eliminar solo la etiqueta con la clave `temporary` de los usuarios. Para utilizar esta política, sustituya el *`texto en cursiva del marcador`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:UntagUser",
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": ["temporary"]}}
    }]
}
```

Control de acceso a los recursos de AWS mediante etiquetas

Puede utilizar etiquetas para controlar el acceso a los recursos de AWS que admiten el etiquetado, incluidos los recursos de IAM. Puede etiquetar usuarios y roles de IAM para controlar a qué pueden tener acceso. Para obtener información sobre cómo etiquetar usuarios y roles de IAM, consulte Etiquetado de recursos de IAM. Además, puede controlar el acceso a los siguientes recursos de IAM: políticas administradas por el cliente, proveedores de identidad de IAM, perfiles de instancia, certificados de servidor y dispositivos MFA virtuales. Para ver un tutorial sobre cómo crear y probar una política que permita a los roles de IAM con etiquetas de entidades principales obtener acceso a los recursos con etiquetas coincidentes, consulte Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas. Utilice la información de la siguiente sección para controlar el acceso a otros recursos de AWS, incluidos los recursos de IAM, sin etiquetar usuarios o roles de IAM.

Para utilizar etiquetas para controlar el acceso a sus recursos de AWS, debe entender cómo AWS otorga el acceso. AWS se compone de colecciones de recursos. Una instancia de Amazon EC2 es

un recurso. Un bucket de Amazon S3 es un recurso. Puede utilizar la API de AWS, la AWS CLI o la AWS Management Console para realizar una operación, como crear un bucket en Amazon S3. Cuando lo haga, envíe una solicitud para dicha operación. La solicitud especifica una acción, un recurso, una entidad principal (usuario o rol), una cuenta principal y toda la información de la solicitud necesaria. Toda esta información proporciona el contexto.

AWS comprueba entonces que usted (la entidad principal) se ha autenticado (ha iniciado sesión) y está autorizado (tiene permiso) para realizar la acción especificada en el recurso especificado. Durante la autorización, AWS comprueba todas las políticas aplicables al contexto de la solicitud. La mayoría de las políticas se almacenan en AWS como documentos JSON y especifican los permisos de las entidades principales. Para obtener más información sobre los tipos de políticas y sus usos, consulte Políticas y permisos en IAM.

AWS autoriza la solicitud únicamente si cada parte de la solicitud está permitida por las políticas. Para ver un diagrama y obtener más información sobre la infraestructura de IAM, consulte Cómo funciona IAM. Para obtener información detallada acerca de cómo IAM determina si una solicitud está permitida, consulte Lógica de evaluación de políticas.

Las etiquetas son otro elemento que se deben considerar, ya que se pueden adjuntar al recurso o pasarse en la solicitud a servicios que admitan etiquetado. Para controlar el acceso según las etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política. Para saber si un servicio de AWS admite el acceso de control utilizando etiquetas, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Sí en la columna Autorización basada en etiquetas. Elija el nombre del servicio para ver la documentación sobre la autorización y el control de acceso para dicho servicio.

A continuación, puede crear una política de IAM que permita o deniegue el acceso a un recurso en función de la etiqueta de dicho recurso. En dicha política, puede utilizar las claves de condición de etiqueta para controlar el acceso a cualquiera de las siguientes operaciones:

- Recurso – Controla el acceso a los recursos de servicio de AWS basado en las etiquetas de esos recursos. Para ello, utilice la clave de condición ResourceTag/***nombre-clave*** para determinar si se permite o no el acceso al recurso en función de las etiquetas adjuntas al recurso.
- Solicitud – Controla las etiquetas que se pueden pasar en una solicitud. Para ello, utilice la clave de condición aws:RequestTag/***nombre-clave*** a fin de especificar qué pares de clave-valor de etiqueta se pueden aprobar en una solicitud para etiquetar un recurso de AWS.
- Cualquier parte del proceso de autorización: utilice la clave de condición aws:TagKeys para controlar si claves de etiqueta específicas pueden estar en una solicitud.

Puede crear visualmente una política de IAM utilizando JSON o importando una política administrada existente. Para obtener más información, consulte Crear políticas de IAM.

> **Note**
>
> Algunos servicios permiten que los usuarios especifiquen etiquetas cuando crean el recurso si tienen permisos para utilizar la acción que crea el recurso.

Control del acceso a los recursos de AWS

Puede utilizar las condiciones de sus políticas de IAM para controlar el acceso a los recursos de AWS en función de las etiquetas de ese recurso. Puede hacerlo a través de la clave de condición global `aws:ResourceTag/tag-key` o de una clave específica del servicio. Algunos servicios solo admiten la versión específica del servicio de esta clave y no la versión global.

> **Warning**
>
> No intente controlar quién puede pasar un rol etiquetando el rol y, a continuación, utilizando la clave de condición `ResourceTag` en una política con la acción `iam:PassRole`. Este planteamiento no da resultados fiables. Para obtener más información acerca de los permisos necesarios para transferir una función a un servicio, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS.

Este ejemplo muestra cómo podría crear una política basada en identidad que permita iniciar o detener instancias de Amazon EC2. Estas operaciones solo se permiten si la etiqueta de instancia `Owner` tiene el valor del nombre de dicho usuario. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
```

```
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/Owner": "${aws:username}"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

También puede asociar esta política al usuario de IAM en su cuenta. Si un usuario llamado `richard-roe` intenta iniciar una instancia de Amazon EC2, la instancia debe tener una etiqueta `Owner=richard-roe` o `owner=richard-roe`. De lo contrario, se le denegará el acceso. La clave de la etiqueta `Owner` coincide con los nombres de las claves de condición `Owner` y `owner` porque no distinguen entre mayúsculas y minúsculas. Para obtener más información, consulte Elementos de política JSON de IAM: Condition.

En el siguiente ejemplo, se muestra cómo se puede crear una política basada en identidad que utilice la etiqueta principal `team` en el ARN del recurso. La política concede permiso para eliminar colas de Amazon Simple Queue Service, pero solo si el nombre de la cola comienza con el nombre del equipo seguido de `-queue`. Por ejemplo, `qa-queue` si `qa` es el nombre del equipo para la etiqueta principal `team`.

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2::${aws:PrincipalTag/team}-queue"
      }
}
```

Control del acceso durante solicitudes de AWS

Puede utilizar condiciones en sus políticas de IAM para controlar qué pares de clave y valor de etiqueta pueden incluirse en una solicitud que aplica etiquetas a un recurso de AWS.

Este ejemplo muestra cómo podría crear una política basada en identidad que permite utilizar la acción `CreateTags` de Amazon EC2 para asociar etiquetas a una instancia. Solo puede asociar etiquetas si la etiqueta contiene la clave `environment` y los valores `production` o `preprod`. Si lo desea, puede utilizar el modificador `ForAllValues` con la clave de condición `aws:TagKeys` para indicar que solo se permite la clave `environment` en la solicitud. Esto impide a los usuarios que incluyan otras claves, por ejemplo, utilizar accidentalmente `Environment` en lugar de `environment`.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "ec2:CreateTags",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": [
                    "preprod",
                    "production"
                ]
            },
            "ForAllValues:StringEquals": {"aws:TagKeys": "environment"}
        }
    }
}
```

Control del acceso en función de las claves de etiqueta

Puede utilizar una condición en sus políticas de IAM para controlar si se pueden utilizar claves de etiqueta específicas en una solicitud.

Le recomendamos que cuando utilice políticas para controlar el acceso mediante etiquetas, utilice la clave de condiciónaws:TagKeys. Puede que los servicios de AWS que admiten etiquetas le permitan crear varios nombres de clave de etiqueta que se diferencien únicamente por el uso de mayúsculas y minúsculas, como puede ser el etiquetado de una instancia de Amazon EC2 con `stack=production` y `Stack=test`. Los nombres de claves no distinguen entre mayúsculas y minúsculas en las condiciones de políticas. Esto significa que si especifica `"aws:ResourceTag/TagKey1": "Value1"` en el elemento de condición de su política, la condición coincidirá con una clave de etiqueta de recurso denominada `TagKey1` o `tagkey1`, pero no con ambas. Para evitar etiquetas duplicadas con una clave que varía únicamente por las mayúsculas y minúsculas, utilice la

condición `aws:TagKeys` para definir las claves de etiqueta que los usuarios pueden aplicar, o bien utilice políticas de etiquetas, disponibles con AWS Organizations. Para obtener más información, consulte Políticas de etiquetas en la Guía del usuario de Organizations.

Este ejemplo muestra cómo se puede crear una política basada en identidad que permita crear y etiquetar un secreto de Secrets Manager, pero solo con las claves de etiqueta `environment` o `cost-center`. La condición `Null` garantiza que la condición se evalúe como `false` si no hay etiquetas en la solicitud.

```
{
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "*",
        "Condition": {
            "Null": {
                "aws:TagKeys": "false"
            },
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "environment",
                    "cost-center"
                ]
            }
        }
}
```

Acceso a recursos entre cuentas en IAM

Para algunos servicios de AWS, puede conceder acceso entre cuentas a los recursos mediante el uso de IAM. Para hacerlo, puede asociar una política de recursos directamente al recurso que desea compartir o utilizar un rol como proxy.

Para compartir el recurso directamente, el recurso que desea compartir debe admitir políticas basadas en recursos. A diferencia de una política basada en identidad para un rol, una política basada en recursos especifica quién (qué entidad principal) puede acceder a ese recurso.

Utilice un rol como proxy cuando desee acceder a los recursos en otra cuenta que no admiten políticas basadas en recursos.

Para obtener más información sobre las diferencias entre estos tipos de políticas, consulte Políticas basadas en identidad y políticas basadas en recursos.

> **Note**
>
> Los roles de IAM y las políticas basadas en recursos delegan el acceso entre cuentas solo dentro de una única partición. Por ejemplo, tiene una cuenta en Oeste de EE. UU. (Norte de California) en la partición estándar `aws`. También tiene una cuenta en China en la partición `aws-cn`. No puede usar una política basada en recursos en su cuenta en China para permitir el acceso a los usuarios en su cuenta estándar de AWS.

Acceso entre cuentas mediante roles

No todos los servicios de AWS admiten políticas basadas en recursos. Para estos servicios, puede utilizar roles de IAM entre cuentas para centralizar la administración de permisos al proporcionar acceso entre cuentas a varios servicios. Un rol de IAM entre cuentas es un rol de IAM que incluye una política de confianza que permite a las entidades principales de IAM en otra cuenta de AWS asumir el rol. En pocas palabras, puede crear un rol en una cuenta de AWS que delega permisos específicos a otra cuenta de AWS.

Para obtener información sobre cómo asociar una política a una identidad de IAM, consulte Administración de políticas de IAM.

> **Note**
>
> Cuando una entidad principal cambia a un rol para usar temporalmente los permisos del rol, renuncia a sus permisos originales y asume los permisos asignados al rol que ha asumido.

Analicemos el proceso general en lo que respecta al software de socios de APN que necesita acceder a una cuenta de cliente.

1. El cliente crea un rol de IAM en su propia cuenta con una política que permite acceder a los recursos de Amazon S3 que necesita el socio de APN. En este ejemplo, el nombre del rol es `APNPartner`.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

2. A continuación, el cliente especifica que la cuenta de AWS del socio puede asumir el rol mediante el ID de la Cuenta de AWS del socio de APN en la política de confianza para el rol de `APNPartner`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::APN-account-ID:role/APN-user-name"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. El cliente proporciona el Nombre de recurso de Amazon (ARN) del rol al socio de APN. El ARN es el nombre completo del rol.

```
arn:aws:iam::APN-ACCOUNT-ID:role/APNPartner
```

> **Note**
>
> Se recomienda utilizar un identificador externo en situaciones con varios inquilinos. Para obtener más información, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.

4. Cuando el software de socios de APN necesita acceder a la cuenta del cliente, el software llama a la API AssumeRole en AWS Security Token Service junto con el ARN del rol en la cuenta del

cliente. STS devuelve una credencial de AWS temporal que permite que el software haga su trabajo.

Para ver otro ejemplo de cómo conceder acceso entre cuentas mediante roles, consulte Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS propia. También puede consultar Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM.

Concesión de acceso entre cuentas con políticas de recursos

Cuando una cuenta accede a un recurose a través de otra cuenta mediante una política basada en recursos, la entidad principal sigue trabajando en la cuenta de confianza y no tiene que renunciar a sus permisos para recibir los permisos de rol. Dicho de otro modo, la entidad principal sigue teniendo acceso a los recursos en la cuenta de confianza y también al recurso en la cuenta que confía. Esto es útil para tareas como copiar información en o desde el recurso compartido en la otra cuenta.

Los principales que puede especificar en una política basada en recursos incluyen cuentas, usuarios de IAM, usuarios federados, roles de IAM, sesiones de rol asumido o servicios de AWS. Para obtener más información, consulte Especificación de una entidad principal.

Para obtener información sobre si las entidades principales en las cuentas que no se encuentran en su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles, consulte Identifying resources shared with an external entity.

La lista siguiente incluye algunos de los servicios de AWS que admiten políticas basadas en recursos. Para obtener una lista completa del número creciente de servicios de AWS que admiten la asociación de políticas de permisos a recursos en lugar de asociarlas a elementos principales, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Sí en la columna basadas en recursos.

- Buckets de Amazon S3: la política se asocia al bucket, pero la política controla el acceso tanto al bucket como a los objetos que hay en él. Para obtener más información, consulte Control de acceso en la Guía del usuario de Amazon Simple Storage Service. En algunos casos, puede ser mejor utilizar roles para el acceso entre cuentas a Amazon S3. Para obtener más información, consulte la explicación de ejemplo en la Guía del usuario de Amazon Simple Storage Service.
- Temas de Amazon Simple Notification Service (Amazon SNS): para obtener más información, consulte Ejemplos de casos de contol de acceso con Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service.

- Colas de Amazon Simple Queue Service (Amazon SQS) - Para obtener más información, consulte Apéndice: El lenguaje de la política de acceso en la Guía para desarrolladores de Amazon Simple Queue Service (Amazon SQS).

Cómo delegar permisos de AWS en una política basada en recursos

Si un recurso concede permisos a los principales de la cuenta, puede delegar esos permisos a identidades de IAM específicas. Las identidades son usuarios, grupos de usuarios o roles de la cuenta. Para delegar permisos debe asociar una política a la identidad. Puede otorgar el número máximo de permisos permitidos por la cuenta propietaria de los recursos.

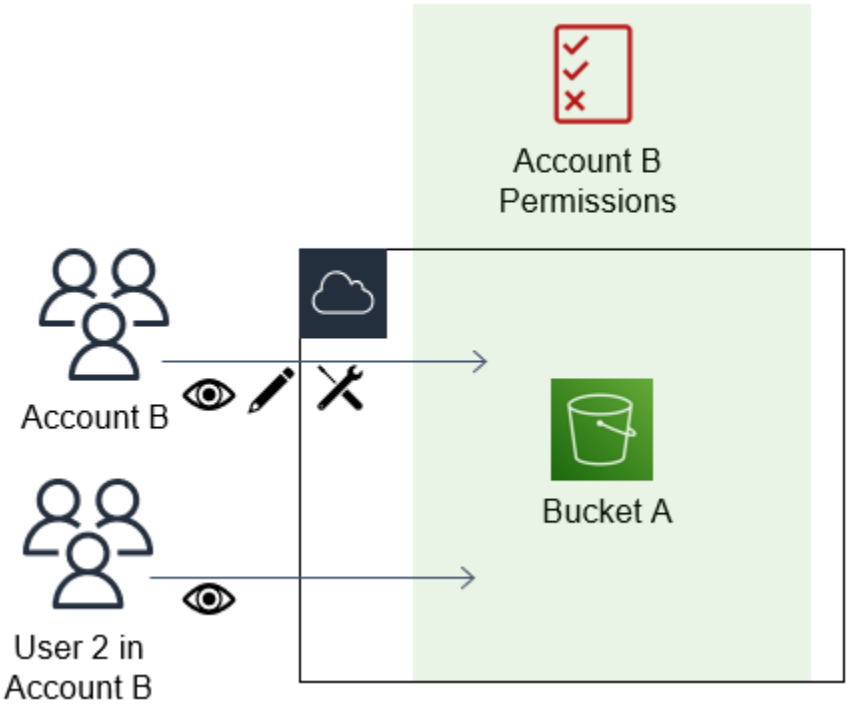
> **Important**
>
> En el acceso entre cuentas, una entidad principal necesita una `Allow` en la política de identidad y en la política basada en recursos.

Suponga que una política basada en recursos permite a todos los principales de la cuenta acceso administrativo completo a un recurso. A continuación, puede delegar acceso completo, acceso de solo lectura o cualquier otro acceso parcial a entidades principales de la cuenta de AWS. Alternativamente, si la política basada en recursos solo permite permisos de lista, entonces solo podrá delegar el acceso a la lista. Si intenta delegar más permisos que los que tiene la cuenta, los principales seguirán teniendo acceso únicamente a las listas.

Para obtener más información sobre cómo se toman estas decisiones, consulte Cómo determinar si una solicitud se permite o se deniega dentro de una cuenta.

> **Note**
>
> Los roles de IAM y las políticas basadas en recursos delegan el acceso entre cuentas solo dentro de una única partición. Por ejemplo, no puede añadir acceso entre cuentas entre una cuenta en la partición `aws` estándar y una cuenta en la partición `aws-cn`.

Por ejemplo, suponga que administra `AccountA` y `AccountB`. En AccountA, hay un bucket de Amazon S3 denominado `BucketA`.



1. Se asocia una política basada en recursos a `BucketA` que les permite a todas las entidades principales en AccountB acceder de manera completa a los objetos en el bucket. De este modo podrán crear, leer o eliminar cualquier objeto de ese bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::AccountB:root"},
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::BucketA/*"
            }
        ]
}
```

   AccountA otorga a AccountB acceso completo a BucketA al designar a AccountB como entidad principal en la política basada en recursos. Como resultado, AccountB está autorizada a realizar cualquier acción en BucketA, y el administrador de AccountB puede delegar el acceso a los usuarios en AccountB.

El usuario raíz de AccountB tiene todos los permisos que se conceden a la cuenta. Por lo tanto, el usuario raíz tiene acceso completo a BucketA.

2. En AccountB, asocie una política al usuario de IAM llamado User2. Esa política le permite al usuario acceso de solo lectura a los objetos en BucketA. Esto significa que User2 puede ver los objetos, pero no crearlos, editarlos ni eliminarlos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Action" : [
                "s3:Get*",
                "s3:List*" ],
                "Resource" : "arn:aws:s3:::BucketA/*"
        }
    ]
}
```

El nivel máximo de acceso que AccountB puede delegar es el nivel de acceso que se concede a la cuenta. En este caso, la política basada en recursos ha concedido acceso completo a AccountB, pero User2 solo tiene acceso de solo lectura.

El administrador de AccountB no concede acceso a User1. De forma predeterminada, los usuarios no tienen permisos, salvo aquellos que se concedan de forma explícita, por lo que User1 no tiene acceso a BucketA.

IAM evalúa los permisos del principal en el momento en que el principal realiza una solicitud. Si utiliza comodines (*) para darles a los usuarios acceso completo a los recursos, las entidades principales pueden acceder a todos los recursos a los que tenga acceso la cuenta de AWS. Esto es cierto incluso para los recursos a los que agrega o para los que obtiene acceso después de crear la política del usuario.

En el ejemplo anterior, si AccountB le hubiera asociado una política a User2 que le permita acceso completo a todos los recursos en todas las cuentas, User2 tendría acceso de manera automática a todos los recursos que AccountB tiene acceso. Esto incluye el acceso a BucketA y el acceso a cualquier otro recurso otorgado por las políticas basadas en recursos en AccountA.

Para obtener más información sobre los usos complejos de los roles, como la concesión de acceso a aplicaciones y servicios, consulte Situaciones habituales con los roles: usuarios, aplicaciones y servicios.

> **Important**
>
> Conceda acceso únicamente a las entidades en las que confíe y conceda el nivel mínimo de acceso necesario. Siempre que la entidad de confianza sea otra cuenta de AWS, se le puede conceder acceso a su recurso a cualquier entidad principal de IAM. La cuenta de confianza de AWS puede delegar acceso únicamente en la medida en que se ha concedido acceso a dicha cuenta; no puede delegar más acceso que el que la propia cuenta tiene.

Para obtener información sobre los permisos, las políticas y el lenguaje de la política de permisos que debe utilizar para escribir políticas, consulte Recursos de AWS para administración de acceso.

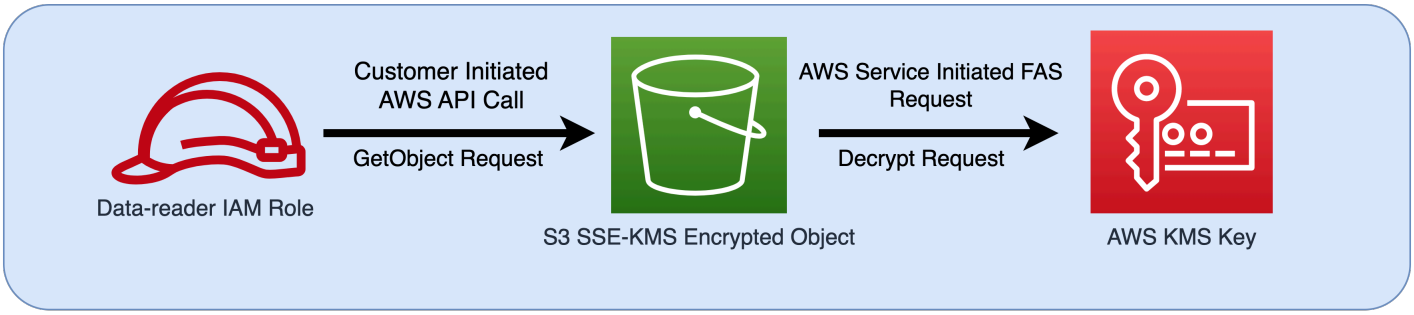
Sesiones de acceso directo

Forward access sessions (FAS) es una tecnología IAM utilizada por los servicios de AWS para proporcionar su identidad, permisos y atributos de sesión cuando un servicio de AWS realiza una solicitud en su nombre. FAS utiliza los permisos de la identidad que llama a un servicio de AWS, combinados con la identidad de un servicio de AWS para realizar peticiones a servicios posteriores. Las solicitudes FAS solo se realizan a servicios de AWS en nombre de una entidad principal de IAM después de que un servicio haya recibido una solicitud que requiera interacciones con otros servicios o recursos de AWS para completarse. Cuando se realiza una solicitud FAS:

- El servicio que recibe la solicitud inicial de una entidad principal de IAM comprueba los permisos de la entidad principal de IAM.
- El servicio que recibe una solicitud FAS posterior también comprueba los permisos de la misma entidad principal de IAM.

Por ejemplo, Amazon S3 utiliza FAS para realizar llamadas a AWS Key Management Service para descifrar un objeto cuando se ha utilizado SSE-KMS para cifrarlo. Al descargar un objeto SSE-KMS cifrado, un rol denominado data-reader llama a GetObject en el objeto en Amazon S3, y no llama a AWS KMS directamente. Tras recibir la solicitud GetObject y autorizar al lector de datos, Amazon S3 realiza una solicitud FAS a AWS KMS para descifrar el objeto de Amazon S3. Cuando KMS recibe la solicitud FAS, comprueba los permisos del rol y solo autoriza la solicitud de descifrado si el lector

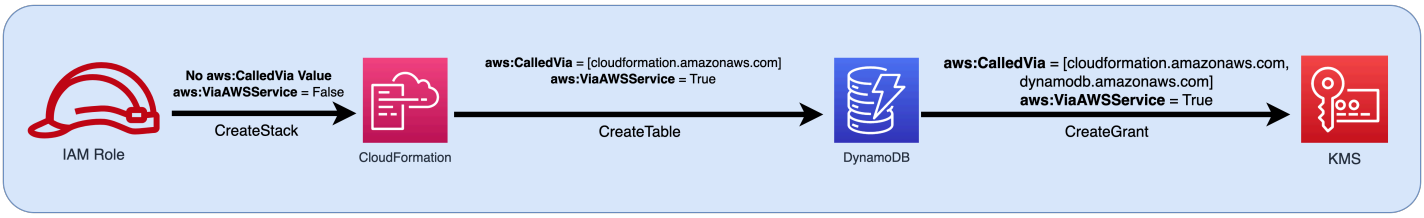
de datos tiene los permisos correctos sobre la clave KMS. Las solicitudes tanto a Amazon S3 como a AWS KMS se autorizan utilizando los permisos del rol y solo se ejecutan correctamente si data-reader tiene permisos tanto para el objeto de Amazon S3 como para la clave KMS de AWS.



> **Note**
>
> Los servicios que han recibido una solicitud FAS pueden realizar solicitudes FAS adicionales. En estos casos, la entidad principal solicitante debe tener permisos para todos los servicios a los que llama FAS.

Solicitudes FAS y condiciones de la política de IAM

Cuando se realizan solicitudes FAS, las claves de condición aws:CalledVia, aws:CalledViaFirst y aws:CalledViaLast se rellenan con la entidad principal del servicio que inició la llamada FAS. El valor de la clave de condición aws:ViaAWSService se establece en `true` cada vez que se realiza una solicitud FAS. En el siguiente diagrama, la solicitud directa a CloudFormation no tiene ninguna clave de condición `aws:CalledVia` o `aws:ViaAWSService` establecida. Cuando CloudFormation y DynamoDB realizan solicitudes FAS descendentes en nombre del rol, se rellenan los valores de estas claves de condición.



Para permitir que se realice una solicitud FAS cuando, de otra forma, sería denegada por una declaración de política de denegación con una clave de condición que compruebe las direcciones IP de origen o las VPC de origen, debe utilizar claves de condición para proporcionar una excepción para las solicitudes FAS en su política de denegación. Esto puede hacerse para todas las solicitudes

FAS utilizando la tecla de condición `aws:ViaAWSService`. Para permitir que solo servicios de AWS específicos realicen solicitudes FAS, utilice `aws:CalledVia`.

> **Important**
>
> Cuando se realiza una solicitud FAS después de una solicitud inicial a través de un punto de conexión de VPC, los valores de clave de condición para `aws:SourceVpce`, `aws:SourceVpc` y `aws:VpcSourceIp` de la solicitud inicial no se utilizan en las solicitudes FAS. Cuando redacte políticas utilizando `aws:VPCSourceIP` o `aws:SourceVPCE` para conceder acceso de manera condicional, también debe utilizar `aws:ViaAWSService` o `aws:CalledVia` para permitir solicitudes FAS. Cuando se realiza una solicitud FAS después de que un punto de conexión público de un servicio de AWS reciba una solicitud inicial, las solicitudes FAS posteriores se realizarán con el mismo valor de clave de condición `aws:SourceIP`.

Ejemplo: autorización de acceso a Amazon S3 desde una VPC o mediante FAS

En el siguiente ejemplo de política de IAM, las solicitudes GetObject de Amazon S3 y Athena solo se permiten si se originan en puntos de conexión de VPC adjuntos a *`example_vpc`*, o si la solicitud es una solicitud FAS realizada por Athena.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "OnlyAllowMyIPs",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*",
        "athena:StartQueryExecution",
        "athena:GetQueryResults",
        "athena:GetWorkGroup",
        "athena:StopQueryExecution",
        "athena:GetQueryExecution"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceVPC": [
            "example_vpc"
```

```
          ]
        }
      }
    },
    {
      "Sid": "OnlyAllowFAS",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "athena.amazonaws.com"
        }
      }
    }
  ]
}
```

Para ver ejemplos adicionales sobre el uso de claves de condición para permitir el acceso a FAS, consulte el repositorio de políticas de ejemplo para perímetros de datos.

Ejemplos de políticas basadas en identidad de IAM

Una política es un objeto de AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal de IAM (usuario o rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de políticas se almacenan en AWS como documentos JSON que se asocian a una identidad de IAM (usuario, grupo de usuarios o rol). Las políticas basadas en la identidad incluyen políticas administradas por AWS, políticas administradas por el cliente y políticas insertadas. Para obtener información sobre cómo crear una política de IAM mediante estos documentos de políticas JSON de ejemplo, consulte the section called “Creación de políticas mediante el editor JSON”.

De forma predeterminada, todas las solicitudes se deniegan, por lo que debe proporcionar acceso a los servicios, acciones y recursos que piensa utilizar para la identidad de acceso. Si también quiere permitir el acceso para completar las acciones especificadas en la consola de IAM, debe proporcionar permisos adicionales.

La siguiente biblioteca de políticas puede serle útil para definir permisos para las identidades de IAM. Después de encontrar la política que necesita, seleccione View this policy (Ver esta política) para ver

el JSON de la política. Puede utilizar el documento de política JSON como plantilla de sus propias políticas.

> **Note**
>
> Si desea enviar una política para que se incluya en esta guía de referencia, utilice el botón Feedback (Comentarios) de la parte inferior de esta página.

Ejemplos de políticas: AWS

- Permite el acceso durante un intervalo de fechas específico. (Ver esta política).
- Permite la habilitación y deshabilitación de regiones de AWS. (Ver esta política).
- Permite a los usuarios autenticados por MFA administrar las credenciales propias en la página Mis credenciales de seguridad. (Ver esta política).
- Permite el acceso específico cuando se usa MFA en un intervalo específico de fechas. (Ver esta política).
- Permite a los usuarios administrar las credenciales propias en la página Mis credenciales de seguridad. (Ver esta política).
- Permite a los usuarios administrar los dispositivos MFA propios en la página Mis credenciales de seguridad. (Ver esta política).
- Permite a los usuarios administrar las contraseñas propias en la página Mis credenciales de seguridad. (Ver esta política).
- Permite a los usuarios administrar las contraseñas, las claves de acceso y las claves públicas SSH propias en la página Mis credenciales de seguridad. (Ver esta política).
- Deniega el acceso a AWS en función de la región solicitada. (Ver esta política).
- Deniega el acceso a AWS en función de la dirección IP de origen. (Ver esta política).

Política de ejemplo: AWS Data Exchange

- Deniega el acceso a los recursos de Amazon S3 fuera de su cuenta, excepto AWS Data Exchange. (Ver esta política).

Ejemplos de políticas: AWS Data Pipeline

- Deniega el acceso a canalizaciones que el usuario no ha creado (ver esta política).

Políticas de ejemplo: Amazon DynamoDB

- Permite el acceso a una tabla de Amazon DynamoDB específica (ver esta política).
- Permite el acceso a atributos de Amazon DynamoDB específicos (ver esta política).
- Permite el acceso de elementos a Amazon DynamoDB en función del ID de Amazon Cognito (ver esta política).

Políticas de ejemplo: Amazon EC2

- Permite asociar o separar volúmenes de Amazon EBS a instancias de Amazon EC2 en función de las etiquetas (ver esta política).
- Permite lanzar instancias de Amazon EC2 en una determinada subred, mediante programación, y en la consola (ver esta política)
- Permite administrar grupos de seguridad de Amazon EC2 asociados a una VPC específica, mediante programación y en la consola (ver esta política).
- Permite iniciar o detener instancias de Amazon EC2 que un usuario haya etiquetado, mediante programación y en la consola (ver esta política).
- Permite iniciar o detener instancias de Amazon EC2 en función de las etiquetas de recursos y entidades principales, mediante programación y en la consola (ver esta política).
- Permite iniciar o detener instancias de Amazon EC2 cuando las etiquetas de recursos y entidades principales coincidan (ver esta política).
- Permite el acceso completo a Amazon EC2 en una región determinada, mediante programación y en la consola. (Ver esta política).
- Permite iniciar o detener una instancia de Amazon EC2 concreta y modificar un grupo de seguridad específico, mediante programación y en la consola (ver esta política)
- Deniega el acceso a operaciones de Amazon EC2 específicas sin MFA (ver esta política).
- Limita la terminación de instancias de Amazon EC2 a un rango específico de direcciones IP (ver esta política)

Ejemplos de políticas: AWS Identity and Access Management (IAM)

- Permite el acceso a la API del simulador de políticas (ver esta política).
- Permite el acceso a la consola del simulador de políticas (ver esta política).
- Permite asumir cualquier rol que tenga una etiqueta específica, mediante programación y en la consola (ver esta política).
- Permite y deniega el acceso a varios servicios, mediante programación y en la consola (ver esta política).
- Permite agregar una etiqueta específica a un usuario de IAM con otra etiqueta específica, mediante programación y en la consola (ver esta política).
- Permite agregar una etiqueta específica a cualquier usuario o rol de IAM, mediante programación y en la consola (ver esta política).
- Permite crear un nuevo usuario solo con etiquetas específicas (ver esta política).
- Permite generar y recuperar informes de credenciales de IAM (ver esta política).
- Permite administrar la pertenencia a un grupo, mediante programación y en la consola (ver esta política).
- Permite administrar una etiqueta específica (ver esta política).
- Permite transferir un rol de IAM a un servicio específico (ver esta política).
- Permite acceso de solo lectura a la consola de IAM sin informes (ver esta política).
- Permite acceso de solo lectura a la consola de IAM (ver esta política).
- Permite que usuarios específicos administren un grupo, mediante programación y en la consola (ver esta política).
- Permite establecer los requisitos de contraseña de la cuenta, mediante programación y en la consola (ver esta política).
- Permite utilizar la API del simulador de políticas para los usuarios con una ruta específica (ver esta política).
- Permite utilizar la consola del simulador de políticas para los usuarios con una ruta específica (ver esta política).
- Permite a los usuarios de IAM administrar ellos mismos un dispositivo MFA. (Ver esta política).
- Permite a los usuarios de IAM definir sus propias credenciales, mediante programación y en la consola. (Ver esta política).
- Permite ver la información de acceso reciente de una política de AWS Organizations en la consola de IAM. (Ver esta política).

- Limita las políticas administradas que pueden aplicarse a un usuario, grupo o rol de IAM (ver esta política).
- Permite el acceso a las políticas de IAM solo en su cuenta (Ver esta política).

Ejemplos de políticas: AWS Lambda

- Permite que la función de AWS Lambda obtenga acceso a una tabla de Amazon DynamoDB (ver esta política).

Políticas de ejemplo: Amazon RDS

- Permite acceso completo a la base de datos de Amazon RDS dentro de una región específica. (Ver esta política).
- Permite restaurar bases de datos de Amazon RDS, mediante programación y en la consola (ver esta política)
- Permite a los propietarios de etiquetas el acceso completo a los recursos de Amazon RDS que han etiquetado (ver esta política)

Políticas de ejemplo: Amazon S3

- Permite que un usuario de Amazon Cognito obtenga acceso a los objetos en su propio bucket de Amazon S3 (ver esta política)
- Permite que los usuarios federados obtengan acceso a su propio directorio principal en Amazon S3, mediante programación y en la consola (ver esta política).
- Permite acceso S3 completo, pero deniega de forma explícita el acceso al bucket de producción en caso de que el administrador no haya iniciado sesión utilizando MFA en los últimos treinta minutos (ver esta política).
- Permite que los usuarios de IAM obtengan acceso a su propio directorio de inicio en Amazon S3, mediante programación y en la consola (ver esta política)
- Permite que un usuario administre un único bucket de Amazon S3 y deniega todas las demás acciones y recursos de AWS (ver esta política).
- Permite el acceso de `Read` y `Write` a un bucket de Amazon S3 específico (ver esta política)
- Permite el acceso de `Read` y `Write` a un bucket de Amazon S3 específico, mediante programación, y en la consola (ver esta política)

AWS: permite el acceso en función de la fecha y la hora

En este ejemplo se muestra cómo crear una política basada en identidad que permite el acceso a acciones según la fecha y hora. Esta política restringe el acceso a las acciones que se produzcan entre el 1 de abril de 2020 y el 30 de junio de 2020 (UTC), ambos inclusive. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para obtener información acerca de la utilización de varias condiciones dentro del bloque `Condition` de una política de IAM, consulte Múltiples valores en un elemento Condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "service-prefix:action-name",
            "Resource": "*",
            "Condition": {
                "DateGreaterThan": {"aws:CurrentTime": "2020-04-01T00:00:00Z"},
                "DateLessThan": {"aws:CurrentTime": "2020-06-30T23:59:59Z"}
            }
        }
    ]
}
```

> **Note**
>
> No puede utilizar una variable de política con el operador de condición de fecha. Para obtener más información, consulte Elemento de condición

AWS: permite habilitar y deshabilitar regiones de AWS

Este ejemplo muestra cómo puede crear una política basada en identidad que permita a un administrador activar y desactivar la Región Asia-Pacífico (Hong Kong) (ap-east-1). Esta política define los permisos para el acceso programático y a la consola. Esta configuración aparece en la página Account settings (Configuración de la cuenta) en la AWS Management Console. Esta

página incluye información de nivel de cuenta sensible que debe visualizar y administrar únicamente los administradores de cuentas. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

> **Important**
>
> No puede habilitar ni deshabilitar las regiones habilitadas de forma predeterminada. Solo puede incluir las regiones que están deshabilitadas de forma predeterminada. Para obtener más información, consulte Administración de las regiones de AWS en la Referencia general de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnableDisableHongKong",
            "Effect": "Allow",
            "Action": [
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"account:TargetRegion": "ap-east-1"}
            }
        },
        {
            "Sid": "ViewConsole",
            "Effect": "Allow",
            "Action": [
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad

En este ejemplo, se muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM autenticados mediante autenticación multifactor (MFA) administrar sus propias credenciales en la página Credenciales de seguridad. En esta página de la AWS Management Console, se muestra información de la cuenta, como el ID de cuenta y el ID de usuario canónico. Los usuarios también pueden ver y editar sus propias contraseñas, claves de acceso, dispositivos MFA, certificados X.509, claves SSH y credenciales de Git. Esta política de ejemplo incluye los permisos necesarios para ver y editar toda la información de la página. También requiere que el usuario se configure y autentique con MFA para poder realizar cualquier otra operación en AWS. Para permitir a los usuarios administrar sus propias credenciales sin utilizar MFA, consulte AWS: permite a los usuarios de IAM administrar sus propias credenciales en la página Credenciales de seguridad.

Para información sobre el acceso a la página Credenciales de seguridad, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

> **Note**
>
> - Este ejemplo de política no permite a los usuarios restablecer una contraseña al iniciar sesión en la AWS Management Console por primera vez. Recomendamos que no conceda permisos a los nuevos usuarios hasta después de que inicien sesión. Para obtener más información, consulte ¿Cómo puedo crear usuarios de IAM de forma segura?. Esto también impide que los usuarios con una contraseña vencida restablezcan su contraseña durante el inicio de sesión. Puede permitir esto añadiendo `iam:ChangePassword` y `iam:GetAccountPasswordPolicy` a la instrucción `DenyAllExceptListedIfNoMFA`. No obstante, no lo recomendamos porque permitir a los usuarios cambiar su contraseña sin MFA puede ser un riesgo para la seguridad.
> - Si tiene la intención de utilizar esta política para el acceso programático, debe llamar al `GetSessionToken` para autenticarse con MFA. Para obtener más información, consulte Configuración del acceso a una API protegido por MFA.

¿Qué hace esta política?

- La instrucción `AllowViewAccountInfo` permite al usuario ver la información de nivel de cuenta. Estos permisos deben estar en su propia instrucción, ya que no admiten o no requieren un ARN de

recurso. En lugar de ello, los permisos especifican `"Resource" : "*"`. Esta instrucción incluye las siguientes acciones que permiten al usuario ver información específica:

- `GetAccountPasswordPolicy`: ver los requisitos de contraseña de la cuenta y cambiar la contraseña de su propio usuario de IAM.
- `ListVirtualMFADevices`: ver información detallada de un dispositivo MFA virtual que está habilitado para el usuario.

- La instrucción `AllowManageOwnPasswords` permite al usuario cambiar su propia contraseña. Esta instrucción incluye también la acción `GetUser`, que es necesaria para ver la mayor parte de la información de la página My security credentials (Mis credenciales de seguridad).
- La instrucción `AllowManageOwnAccessKeys` permite al usuario crear, actualizar y eliminar sus propias claves de acceso. El usuario también puede recuperar información acerca de cuándo se utilizó por última vez la clave de acceso especificada.
- La instrucción `AllowManageOwnSigningCertificates` permite al usuario cargar, actualizar y eliminar sus propios certificados de firma.
- La instrucción `AllowManageOwnSSHPublicKeys` permite al usuario cargar, actualizar y eliminar sus propias claves públicas SSH de CodeCommit.
- La instrucción `AllowManageOwnGitCredentials` permite al usuario cargar, crear y eliminar sus propias credenciales de Git de CodeCommit.
- La declaración `AllowManageOwnVirtualMFADevice` permite al usuario crear su propio dispositivo MFA virtual. El recurso de ARN de esta declaración permite al usuario crear un dispositivo MFA con cualquier nombre, pero las otras declaraciones en la política solo permiten al usuario adjuntar el dispositivo al usuario actualmente conectado.
- La instrucción `AllowManageOwnUserMFA` permite al usuario ver o administrar el dispositivo MFA o U2F virtual o físico de su propio usuario. El ARN de recurso de esta instrucción permite el acceso únicamente al propio usuario de IAM. Los usuarios no pueden ver ni administrar el dispositivo MFA de otros usuarios.
- La instrucción `DenyAllExceptListedIfNoMFA` deniega el acceso a todas las acciones de todos los servicios de AWS, salvo algunas acciones indicadas, pero solo si el usuario no ha iniciado sesión con MFA. La instrucción utiliza una combinación de `"Deny"` y `"NotAction"` para denegar explícitamente el acceso a las acciones que no se indican en la lista. Esta instrucción no deniega ni permite los elementos enumerados. Son otras instrucciones de la política las que permiten las acciones. Para obtener más información acerca de la lógica de esta instrucción, consulte NotAction con Deny. Si el usuario ha iniciado sesión con MFA, la prueba `Condition` no se cumple y esta

instrucción no deniega ninguna acción. En este caso, otras políticas o instrucciones determinan los permisos del usuario.

Esta instrucción garantiza que cuando el usuario no ha iniciado sesión con MFA únicamente pueda realizar las acciones que se muestran. Además, puede realizar las acciones mostradas solo si otra instrucción o política permite el acceso a estas acciones. Esto no permite a un usuario crear una contraseña durante el inicio de sesión, ya que la acción `iam:ChangePassword` no debe permitirse sin autorización de MFA.

La versión `...IfExists` del operador `Bool` garantiza que si falta la clave `aws:MultiFactorAuthPresent`, la condición devuelve el valor verdadero. Esto significa que a un usuario que accede a una API con credenciales a largo plazo, como una clave de acceso, se le deniega el acceso a las operaciones de la API que no son de IAM.

Esta política no permite a los usuarios ver la página Usuarios de la consola de IAM ni utilizar esa página para obtener acceso a su propia información de usuario. Para permitir esto, añada la acción `iam:ListUsers` a la instrucción `AllowViewAccountInfo` y a la instrucción `DenyAllExceptListedIfNoMFA`. Tampoco permite a los usuarios cambiar su contraseña en su propia página de usuario. Si desea permitirlo, agregue las acciones `iam:GetLoginProfile` e `iam:UpdateLoginProfile` a la instrucción `AllowManageOwnPasswords`. Para permitir también que un usuario cambie su contraseña desde su propia página de usuario sin iniciar sesión con MFA, añada la acción `iam:UpdateLoginProfile` a la instrucción `DenyAllExceptListedIfNoMFA`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
```

```
            "iam:GetUser"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnAccessKeys",
        "Effect": "Allow",
        "Action": [
            "iam:CreateAccessKey",
            "iam:DeleteAccessKey",
            "iam:ListAccessKeys",
            "iam:UpdateAccessKey",
            "iam:GetAccessKeyLastUsed"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSigningCertificates",
        "Effect": "Allow",
        "Action": [
            "iam:DeleteSigningCertificate",
            "iam:ListSigningCertificates",
            "iam:UpdateSigningCertificate",
            "iam:UploadSigningCertificate"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnSSHPublicKeys",
        "Effect": "Allow",
        "Action": [
            "iam:DeleteSSHPublicKey",
            "iam:GetSSHPublicKey",
            "iam:ListSSHPublicKeys",
            "iam:UpdateSSHPublicKey",
            "iam:UploadSSHPublicKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
        "Sid": "AllowManageOwnGitCredentials",
        "Effect": "Allow",
        "Action": [
            "iam:CreateServiceSpecificCredential",
```

```
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
```

```
            }
        }
    ]
}
```

AWS: permite un acceso específico mediante MFA en unas fechas específicas

En este ejemplo se muestra cómo crear una política basada en identidad que utilice varias condiciones, que se evalúan al utilizar una lógica `AND`. Permite acceso total al servicio denominado `SERVICE-NAME-1` y acceso a las acciones `ACTION-NAME-A` y `ACTION-NAME-B` en el servicio denominado `SERVICE-NAME-2`. Estas acciones están permitidas solo cuando el usuario está autenticado mediante la autenticación multifactor (MFA). El acceso está restringido a acciones que se producen entre el 1 de julio de 2017 y el 31 de diciembre de 2017 (UTC), ambos incluidos. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para obtener información acerca de la utilización de varias condiciones dentro del bloque `Condition` de una política de IAM, consulte Múltiples valores en un elemento Condition.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "service-prefix-1:*",
            "service-prefix-2:action-name-a",
            "service-prefix-2:action-name-b"
        ],
        "Resource": "*",
        "Condition": {
            "Bool": {"aws:MultiFactorAuthPresent": true},
            "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
            "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
        }
    }
}
```

AWS: permite a los usuarios de IAM administrar sus propias credenciales en la página Credenciales de seguridad

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM administrar sus propias credenciales en la página Credenciales de seguridad. En esta página de la AWS Management Console, se muestra información de la cuenta, como el ID de cuenta y el ID de usuario canónico. Los usuarios también pueden ver y editar sus propias contraseñas, claves de acceso, certificados X.509, claves SSH y credenciales de Git. Esta política de ejemplo incluye los permisos necesarios para ver y editar toda la información de la página, salvo el dispositivo MFA del usuario. Para permitir a los usuarios administrar sus propias credenciales con MFA, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

Para información sobre el acceso a la página Credenciales de seguridad, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

¿Qué hace esta política?

- La instrucción `AllowViewAccountInfo` permite al usuario ver la información de nivel de cuenta. Estos permisos deben estar en su propia instrucción, ya que no admiten o no requieren un ARN de recurso. En lugar de ello, los permisos especifican `"Resource" : "*"`. Esta instrucción incluye las siguientes acciones que permiten al usuario ver información específica:
  - `GetAccountPasswordPolicy`: ver los requisitos de contraseña de la cuenta y cambiar la contraseña de su propio usuario de IAM.
  - `GetAccountSummary`: ver el ID de cuenta y el ID de usuario canónico de la cuenta.
- La instrucción `AllowManageOwnPasswords` permite al usuario cambiar su propia contraseña. Esta instrucción incluye también la acción `GetUser`, que es necesaria para ver la mayor parte de la información de la página My security credentials (Mis credenciales de seguridad).
- La instrucción `AllowManageOwnAccessKeys` permite al usuario crear, actualizar y eliminar sus propias claves de acceso. El usuario también puede recuperar información acerca de cuándo se utilizó por última vez la clave de acceso especificada.
- La instrucción `AllowManageOwnSigningCertificates` permite al usuario cargar, actualizar y eliminar sus propios certificados de firma.
- La instrucción `AllowManageOwnSSHPublicKeys` permite al usuario cargar, actualizar y eliminar sus propias claves públicas SSH de CodeCommit.

- La instrucción `AllowManageOwnGitCredentials` permite al usuario cargar, crear y eliminar sus propias credenciales de Git de CodeCommit.

Esta política no permite a los usuarios ver ni administrar sus propios dispositivos MFA. Tampoco pueden ver la página Usuarios de la consola de IAM ni utilizar esa página para obtener acceso a su propia información de usuario. Si desea permitirlo, añada la acción `iam:ListUsers` a la instrucción `AllowViewAccountInfo`. Tampoco permite a los usuarios cambiar su contraseña en su propia página de usuario. Si desea permitirlo, añada las acciones `iam:CreateLoginProfile`, `iam:DeleteLoginProfile`, `iam:GetLoginProfile` e `iam:UpdateLoginProfile` a la instrucción `AllowManageOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
```

```
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSigningCertificates",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSigningCertificate",
                "iam:ListSigningCertificates",
                "iam:UpdateSigningCertificate",
                "iam:UploadSigningCertificate"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnGitCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceSpecificCredential",
                "iam:DeleteServiceSpecificCredential",
                "iam:ListServiceSpecificCredentials",
                "iam:ResetServiceSpecificCredential",
                "iam:UpdateServiceSpecificCredential"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permite a los usuarios de IAM autenticados por MFA administrar su propio dispositivo MFA en la página Credenciales de seguridad

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM autenticados mediante autenticación multifactor (MFA) para administrar su propio dispositivo MFA en la página Credenciales de seguridad. En esta página de la AWS Management Console se muestra información de la cuenta y del usuario, pero el usuario solo puede ver y editar su propio dispositivo MFA. Para permitir a los usuarios administrar todas sus credenciales con MFA, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

> **Note**
>
> Si un usuario de IAM con esta política no está autenticado por MFA, esta política deniega el acceso a todas las acciones de AWS excepto las necesarias para autenticarse mediante MFA. Para utilizar la AWS CLI y la API de AWS, los usuarios de IAM primero deben recuperar su token de MFA mediante la operación GetSessionToken de AWS STS y, a continuación, utilizar dicho token para autenticar la operación deseada. Otras políticas, como las políticas basadas en recursos o identidad, pueden permitir acciones en otros servicios. Esta política denegará ese acceso si el usuario de IAM no se autentica mediante MFA.

Para información sobre el acceso a la página Credenciales de seguridad, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

¿Qué hace esta política?

- La instrucción `AllowViewAccountInfo` permite al usuario ver los detalles de un dispositivo MFA virtual que está habilitado para el usuario. Este permiso debe estar en su propia instrucción, ya que no es posible especificar el ARN de un recurso. En su lugar, debe especificar `"Resource" : "*"`.
- La declaración `AllowManageOwnVirtualMFADevice` permite al usuario crear su propio dispositivo MFA virtual. El recurso de ARN de esta declaración permite al usuario crear un dispositivo MFA con cualquier nombre, pero las otras declaraciones en la política solo permiten al usuario adjuntar el dispositivo al usuario actualmente conectado.
- La instrucción `AllowManageOwnUserMFA` permite al usuario ver o administrar su propio dispositivo MFA o U2F virtual o físico. El ARN de recurso de esta instrucción permite el acceso

únicamente al propio usuario de IAM. Los usuarios no pueden ver ni administrar el dispositivo MFA de otros usuarios.

- La instrucción `DenyAllExceptListedIfNoMFA` deniega el acceso a todas las acciones de todos los servicios de AWS, salvo algunas acciones indicadas, pero solo si el usuario no ha iniciado sesión con MFA. La instrucción utiliza una combinación de `"Deny"` y `"NotAction"` para denegar explícitamente el acceso a las acciones que no se indican en la lista. Esta instrucción no deniega ni permite los elementos enumerados. Son otras instrucciones de la política las que permiten las acciones. Para obtener más información acerca de la lógica de esta instrucción, consulte NotAction con Deny. Si el usuario ha iniciado sesión con MFA, la prueba `Condition` no se cumple y esta instrucción no deniega ninguna acción. En este caso, otras políticas o instrucciones determinan los permisos del usuario.

  Esta instrucción garantiza que cuando el usuario no ha iniciado sesión con MFA únicamente pueda realizar las acciones que se muestran. Además, puede realizar las acciones mostradas solo si otra instrucción o política permite el acceso a estas acciones.

  La versión `...IfExists` del operador `Bool` garantiza que si falta la clave `aws:MultiFactorAuthPresent`, la condición devuelve el valor verdadero. Esto significa que a un usuario que obtiene acceso a una operación de la API con credenciales a largo plazo, como una clave de acceso, se le deniega el acceso a las operaciones de la API que no son de IAM.

Esta política no permite a los usuarios ver la página Usuarios de la consola de IAM ni utilizar esa página para obtener acceso a su propia información de usuario. Para permitir esto, añada la acción `iam:ListUsers` a la instrucción `AllowViewAccountInfo` y a la instrucción `DenyAllExceptListedIfNoMFA`.

> **Warning**
>
> No agregue el permiso para eliminar un dispositivo MFA sin la autenticación MFA. Los usuarios con esta política es posible que intenten asignarse a sí mismos un dispositivo MFA virtual y reciban un error que indica que no están autorizados para realizar `iam:DeleteVirtualMFADevice`. Si esto ocurre, no agregue ese permiso a la declaración `DenyAllExceptListedIfNoMFA`. A los usuarios que no se han autenticado con MFA nunca se les debe permitir eliminar su dispositivo MFA. Los usuarios pueden ver este error si han empezado anteriormente la asignación de un dispositivo MFA virtual a su usuario y cancelado el proceso. Para solucionar este problema, usted u otro administrador debe eliminar el dispositivo MFA virtual existente del usuario con la AWS CLI o la API de AWS.

Para obtener más información, consulte No tengo autorización para realizar la operación iam:DeleteVirtualMFADevice.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": "iam:ListVirtualMFADevices",
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowManageOwnUserMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "DenyAllExceptListedIfNoMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:GetUser",
                "iam:ListMFADevices",
                "iam:ListVirtualMFADevices",
```

```
                "iam:ResyncMFADevice",
                "sts:GetSessionToken"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}
            }
        }
    ]
}
```

AWS: permite a los usuarios de IAM cambiar su propia contraseña de consola en la página Credenciales de seguridad

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM cambiar su propia contraseña de la AWS Management Console en la página Credenciales de seguridad. En esta página de la AWS Management Console se muestra información de la cuenta y del usuario, pero el usuario solo tiene acceso a su propia contraseña. Para permitir a los usuarios administrar todas sus credenciales con MFA, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad. Para permitir a los usuarios administrar sus propias credenciales sin utilizar MFA, consulte AWS: permite a los usuarios de IAM administrar sus propias credenciales en la página Credenciales de seguridad.

Para información sobre el acceso a la página Credenciales de seguridad, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

¿Qué hace esta política?

- La instrucción `ViewAccountPasswordRequirements` permite al usuario ver los requisitos de contraseña de la cuenta y cambiar la contraseña de su propio usuario de IAM.
- La instrucción `ChangeOwnPassword` permite al usuario cambiar su propia contraseña. Esta instrucción incluye también la acción `GetUser`, que es necesaria para ver la mayor parte de la información de la página My security credentials (Mis credenciales de seguridad).

Esta política no permite a los usuarios ver la página Usuarios de la consola de IAM ni utilizar esa página para obtener acceso a su propia información de usuario. Si desea permitirlo, añada la acción `iam:ListUsers` a la instrucción `ViewAccountPasswordRequirements`. Tampoco permite a los usuarios cambiar su contraseña en su propia página de usuario. Si desea permitirlo,

agregue las acciones `iam:GetLoginProfile` e `iam:UpdateLoginProfile` a la instrucción `ChangeOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAccountPasswordRequirements",
            "Effect": "Allow",
            "Action": "iam:GetAccountPasswordPolicy",
            "Resource": "*"
        },
        {
            "Sid": "ChangeOwnPassword",
            "Effect": "Allow",
            "Action": [
                "iam:GetUser",
                "iam:ChangePassword"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: permite a los usuarios de IAM administrar su propia contraseña, sus claves de acceso y sus claves públicas SSH en la página Credenciales de seguridad

Este ejemplo muestra cómo podría crear una política de IAM que permita a los usuarios de IAM administrar su propia contraseña, claves de acceso y certificados X.509 en la página Credenciales de seguridad. En esta página de la AWS Management Console, se muestra información de la cuenta, como el ID de cuenta y el ID de usuario canónico. Los usuarios también pueden ver y editar sus propias contraseñas, claves de acceso, dispositivos MFA, certificados X.509, claves SSH y credenciales de Git. Esta política de ejemplo incluye los permisos que son necesarios únicamente para ver y editar la contraseña, las claves de acceso y el certificado X.509. Para permitir a los usuarios administrar todas sus credenciales con MFA, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad. Para permitir a los usuarios administrar sus propias credenciales sin utilizar MFA, consulte AWS: permite a los usuarios de IAM administrar sus propias credenciales en la página Credenciales de seguridad.

Para información sobre el acceso a la página Credenciales de seguridad, consulte Cómo cambian los usuarios de IAM su propia contraseña (consola).

¿Qué hace esta política?

- La instrucción `AllowViewAccountInfo` permite al usuario ver la información de nivel de cuenta. Estos permisos deben estar en su propia instrucción, ya que no admiten o no requieren un ARN de recurso. En lugar de ello, los permisos especifican `"Resource" : "*"`. Esta instrucción incluye las siguientes acciones que permiten al usuario ver información específica:
  - `GetAccountPasswordPolicy`: ver los requisitos de contraseña de la cuenta y cambiar la contraseña de su propio usuario de IAM.
  - `GetAccountSummary`: ver el ID de cuenta y el ID de usuario canónico de la cuenta.
- La instrucción `AllowManageOwnPasswords` permite al usuario cambiar su propia contraseña. Esta instrucción incluye también la acción `GetUser`, que es necesaria para ver la mayor parte de la información de la página My security credentials (Mis credenciales de seguridad).
- La instrucción `AllowManageOwnAccessKeys` permite al usuario crear, actualizar y eliminar sus propias claves de acceso. El usuario también puede recuperar información acerca de cuándo se utilizó por última vez la clave de acceso especificada.
- La instrucción `AllowManageOwnSSHPublicKeys` permite al usuario cargar, actualizar y eliminar sus propias claves públicas SSH de CodeCommit.

Esta política no permite a los usuarios ver ni administrar sus propios dispositivos MFA. Tampoco pueden ver la página Usuarios de la consola de IAM ni utilizar esa página para obtener acceso a su propia información de usuario. Si desea permitirlo, añada la acción `iam:ListUsers` a la instrucción `AllowViewAccountInfo`. Tampoco permite a los usuarios cambiar su contraseña en su propia página de usuario. Si desea permitirlo, agregue las acciones `iam:GetLoginProfile` e `iam:UpdateLoginProfile` a la instrucción `AllowManageOwnPasswords`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:CreateAccessKey",
                "iam:DeleteAccessKey",
                "iam:ListAccessKeys",
                "iam:UpdateAccessKey",
                "iam:GetAccessKeyLastUsed"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowManageOwnSSHPublicKeys",
            "Effect": "Allow",
            "Action": [
                "iam:DeleteSSHPublicKey",
                "iam:GetSSHPublicKey",
                "iam:ListSSHPublicKeys",
                "iam:UpdateSSHPublicKey",
                "iam:UploadSSHPublicKey"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

AWS: deniega el acceso a AWS en función de la región solicitada.

En este ejemplo se muestra cómo crear una política basada en identidad que deniegue el acceso a cualquier acción fuera de las regiones especificadas mediante la clave de condición `aws:RequestedRegion`, a excepción de las acciones en los servicios especificados al utilizar

`NotAction`. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Esta política utiliza el elemento `NotAction` con el efecto `Deny`, que deniega explícitamente el acceso a todas las acciones que no figuran en la instrucción. Las acciones de los servicios CloudFront, IAM, Route 53 y AWS Support no deben ser denegados porque son servicios globales populares de AWS con un único punto de enlace que se encuentra físicamente en la región `us-east-1`. Dado que todas las solicitudes a estos servicios se realizan a la región `us-east-1`, las solicitudes se denegaría sin el elemento `NotAction`. Edite este elemento para incluir acciones para otros servicios globales de AWS que utilice, como, por ejemplo, `budgets`, `globalaccelerator`, `importexport`, `organizations`, o `waf`. Algunos otros servicios globales, como AWS Chatbot y AWS Device Farm, son servicios globales con puntos de enlace ubicados físicamente en la región `us-west-2`. Para obtener más información acerca de todos los servicios que tienen un único punto de enlace global, consulte Regiones y puntos de enlace de AWS en la Referencia general de AWS. Para obtener más información acerca de cómo utilizar el elemento `NotAction` con el efecto `Deny`, consulte Elementos de política JSON de IAM: NotAction.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "route53:*",
                "support:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:RequestedRegion": [
```

```
                    "eu-central-1",
                    "eu-west-1",
                    "eu-west-2",
                    "eu-west-3"
                ]
            }
        }
    }
  ]
}
```

AWS: Deniega acceso a AWS en función de la dirección IP de origen

Este ejemplo muestra cómo puede crear una política basada en identidad que deniegue el acceso a todas las acciones de AWS en la cuenta cuando la solicitud proviene de entidades principales que están fuera del intervalo de direcciones IP especificadas. La política es útil cuando las direcciones IP de su empresa están dentro de los rangos especificados. En este ejemplo, la solicitud se deniega a menos que se origine en el rango de CIDR 192.0.2.0/24 o 203.0.113.0/24. La política no deniega las solicitudes realizadas por los servicios de AWS que utilizan Sesiones de acceso directo mientras la dirección IP del solicitante original se conserva.

Tenga cuidado con el uso de condiciones negativas en la misma instrucción de política que `"Effect": "Deny"`. Cuando lo haga, las acciones especificadas en la instrucción de política se deniegan explícitamente en todas las condiciones excepto en las especificadas.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

Cuando otras políticas permiten acciones, las entidades principales pueden realizar solicitudes desde dentro del intervalo de direcciones IP. Un servicio de AWS también puede realizar solicitudes utilizando las credenciales de la entidad principal. Cuando una entidad principal realiza una solicitud desde fuera del intervalo de direcciones IP, la solicitud se deniega.

Para obtener más información acerca del uso de las claves de condición `aws:SourceIp`, incluida información acerca de cuándo `aws:SourceIp` puede no funcionar en su política, consulte Claves de contexto de condición globales de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Deny",
        "Action": "*",
        "Resource": "*",
        "Condition": {
            "NotIpAddress": {
                "aws:SourceIp": [
                    "192.0.2.0/24",
                    "203.0.113.0/24"
                ]
            }
        }
    }
}
```

AWS: denegar el acceso a los recursos de Amazon S3 fuera de su cuenta, excepto AWS Data Exchange

El siguiente ejemplo muestra cómo crear una política basada en identidad que deniegue el acceso a todos los recursos de AWS que no pertenezcan a su cuenta, excepto los recursos que AWS Data Exchange requiere para un funcionamiento normal. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Puede crear una política similar para restringir el acceso a los recursos dentro de una organización o una unidad organizativa y, al mismo tiempo, contabilizar los recursos propios de AWS Data Exchange mediante las claves de condición `aws:ResourceOrgPaths` y `aws:ResourceOrgID`.

Si usa AWS Data Exchange en su entorno, el servicio crea recursos como los buckets de Amazon S3 que son propiedad de la cuenta de servicio e interactúa con ellos. Por ejemplo, AWS Data Exchange envía solicitudes a los buckets de Amazon S3 propiedad del servicio AWS Data Exchange en nombre de la entidad principal de IAM (usuario o rol) que invoca las API de AWS Data Exchange. En ese caso, el uso de `aws:ResourceAccount`, `aws:ResourceOrgPaths` o `aws:ResourceOrgID` en una política, sin tener en cuenta los recursos propiedad de AWS Data Exchange, deniega el acceso a los buckets propiedad de la cuenta de servicio.

- La declaración, `DenyAllAwsResourcesOutsideAccountExceptS3`, utiliza el elemento `NotAction` con el efecto Deny (Denegar) que deniega explícitamente el acceso a todas las

acciones que no figuren en la declaración y que tampoco pertenezcan a la cuenta incluida en la lista. El elemento `NotAction` indica las excepciones a esta declaración. Estas acciones son la excepción a esta declaración porque, si las acciones se llevan a cabo en los recursos creados por AWS Data Exchange, la política las deniega.

- La declaración, `DenyAllS3ResoucesOutsideAccountExceptDataExchange`, utiliza una combinación de las condiciones `ResourceAccount` y `CalledVia` para denegar el acceso a las tres acciones de Amazon S3 excluidas en la declaración anterior. La declaración deniega las acciones si los recursos no pertenecen a la cuenta enumerada y si el servicio de llamadas no es AWS Data Exchange. La declaración no deniega las acciones si el recurso pertenece a la cuenta enumerada o si la entidad principal de servicio enumerada, `dataexchange.amazonaws.com`, lleva a cabo las operaciones.

> **Important**
>
> Esta política no permite ninguna acción. Utiliza el efecto `Deny` que deniega el acceso a todos los recursos enumerados en la declaración que no pertenecen a la cuenta que se indica. Utilice esta política en combinación con otras políticas que permiten acceder a acciones específicas.

El siguiente ejemplo muestra cómo puede configurar la política para permitir el acceso a los buckets de Amazon S3 necesarios.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllAwsReourcesOutsideAccountExceptAmazonS3",
      "Effect": "Deny",
      "NotAction": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
```

```
          ]
        }
      }
    },
    {
      "Sid": "DenyAllS3ResourcesOutsideAccountExceptDataExchange",
      "Effect": "Deny",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:PutObjectAcl"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        },
        "ForAllValues:StringNotEquals": {
          "aws:CalledVia": [
            "dataexchange.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Data Pipeline: deniega el acceso a canalizaciones DataPipeline que el usuario no ha creado

En este ejemplo se muestra cómo crear una política basada en identidad que deniegue el acceso a canalizaciones que el usuario no haya creado. Si el valor del campo `PipelineCreator` coincide con el nombre de usuario de IAM, no se denegarán las acciones especificadas. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExplicitDenyIfNotTheOwner",
            "Effect": "Deny",
            "Action": [
                "datapipeline:ActivatePipeline",
                "datapipeline:AddTags",
                "datapipeline:DeactivatePipeline",
                "datapipeline:DeletePipeline",
                "datapipeline:DescribeObjects",
                "datapipeline:EvaluateExpression",
                "datapipeline:GetPipelineDefinition",
                "datapipeline:PollForTask",
                "datapipeline:PutPipelineDefinition",
                "datapipeline:QueryObjects",
                "datapipeline:RemoveTags",
                "datapipeline:ReportTaskProgress",
                "datapipeline:ReportTaskRunnerHeartbeat",
                "datapipeline:SetStatus",
                "datapipeline:SetTaskStatus",
                "datapipeline:ValidatePipelineDefinition"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringNotEquals": {"datapipeline:PipelineCreator": "${aws:userid}"}
            }
        }
    ]
}
```

Amazon DynamoDB: permite el acceso a una determinada tabla

En este ejemplo se muestra cómo crear una política basada en identidad que permita el acceso completo a la tabla `MyTable` de DynamoDB. Esta política concede los permisos necesarios para

llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

> **Important**
>
> Esta política permite todas las acciones que pueden llevarse a cabo en una tabla de DynamoDB. Para revisar estas acciones, consulte Permisos de la API de DynamoDB: referencia sobre acciones, recursos y condiciones en la Guía para desarrolladores de Amazon DynamoDB. Puede proporcionar los mismos permisos si enumera cada acción individual. Sin embargo, si utiliza el comodín (*) en el elemento `Action`, como `"dynamodb:List*"`, no tendrá que actualizar la política si DynamoDB agrega una nueva acción List.

Esta política solo permite realizar acciones en las tablas de DynamoDB que existen con el nombre especificado. Para permitir a los usuarios obtener acceso de `Read` a todos los elementos de DynamoDB, también puede asociar la política administrada por AmazonDynamoDBReadOnlyAccess de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAndDescribe",
            "Effect": "Allow",
            "Action": [
                "dynamodb:List*",
                "dynamodb:DescribeReservedCapacity*",
                "dynamodb:DescribeLimits",
                "dynamodb:DescribeTimeToLive"
            ],
            "Resource": "*"
        },
        {
            "Sid": "SpecificTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGet*",
                "dynamodb:DescribeStream",
                "dynamodb:DescribeTable",
```

```
                "dynamodb:Get*",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWrite*",
                "dynamodb:CreateTable",
                "dynamodb:Delete*",
                "dynamodb:Update*",
                "dynamodb:PutItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/MyTable"
        }
    ]
}
```

Amazon DynamoDB: permite el acceso a atributos específicos

En este ejemplo se muestra cómo crear una política basada en identidad que permita el acceso a los atributos específicos de DynamoDB. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

El requisito `dynamodb:Select` impide que la acción de la API devuelva cualquier atributo no permitido, como una proyección de índice. Para obtener más información sobre las claves de condición de DynamoDB, consulte Especificación de condiciones: uso de claves de condiciones en la Guía para desarrolladores de Amazon DynamoDB. Para obtener información acerca de la utilización de varias condiciones o de varias claves de condición dentro del bloque `Condition` de una política de IAM, consulte Múltiples valores en un elemento Condition.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:BatchGetItem",
                "dynamodb:Query",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:DeleteItem",
                "dynamodb:BatchWriteItem"
```

```
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/table-name"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:Attributes": [
                        "column-name-1",
                        "column-name-2",
                        "column-name-3"
                    ]
                },
                "StringEqualsIfExists": {"dynamodb:Select": "SPECIFIC_ATTRIBUTES"}
            }
        }
    ]
}
```

Amazon DynamoDB: permite el acceso en el nivel de elemento a DynamoDB en función de un ID de Amazon Cognito

En este ejemplo, se muestra cómo crear una política basada en identidad que permita el acceso de todo el elemento a la tabla de DynamoDB `MyTable` según el ID de usuario del grupo de identidades de Amazon Cognito. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para utilizar esta política, debe estructurar la tabla de DynamoDB, de modo que el ID de usuario del grupo de identidades de Amazon Cognito sea la clave de partición. Para obtener más información, consulte Crear una tabla en la Guía para desarrolladores de Amazon DynamoDB.

Para obtener más información sobre las claves de condición de DynamoDB, consulte Especificación de condiciones: uso de claves de condiciones en la Guía para desarrolladores de Amazon DynamoDB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
```

```
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:UpdateItem"
            ],
            "Resource": ["arn:aws:dynamodb:*:*:table/MyTable"],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": ["${cognito-identity.amazonaws.com:sub}"]
                }
            }
        }
    ]
}
```

Amazon EC2: asociar volúmenes de Amazon EBS a instancias EC2 en función de las etiquetas, o bien desasociarlos

Este ejemplo muestra cómo puede crear una política basada en identidad que permite a los propietarios de volúmenes de EBS asociar o desasociar sus volúmenes definidos mediante la etiqueta `VolumeUser` a instancias de EC2 etiquetadas como instancias de desarrollo (`Department=Development`). Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para obtener más información sobre la creación de políticas de IAM para controlar el acceso a los recursos de Amazon EC2, consulte Control de acceso a recursos de Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
```

```
                "StringEquals": {"aws:ResourceTag/Department": "Development"}
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:*:*:volume/*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/VolumeUser": "${aws:username}"}
            }
        }
    ]
}
```

Amazon EC2: permite lanzar instancias EC2 en una determinada subred, de forma programática y en la consola

En este ejemplo se muestra cómo crear una política basada en identidad que permita enumerar información de todos los objetos de EC2 y lanzar instancias de EC2 en una subred específica. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:GetConsole*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:subnet/subnet-subnet-id",
```

```
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Amazon EC2: permite administrar grupos de seguridad de EC2 con un par de etiquetas de valor de clave específico, mediante programación y en la consola

En este ejemplo se muestra cómo crear una política basada en identidad que otorga a los usuarios permiso a fin de tomar ciertas acciones para grupos de seguridad que tienen la misma etiqueta. Esta política concede permisos para ver grupos de seguridad en la consola de Amazon EC2, así como para agregar y quitar reglas de entrada y salida y enumerar y modificar las descripciones de las reglas de los grupos de seguridad existentes que tengan la etiqueta `Department=Test`. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
```

```
            "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
            "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:ResourceTag/Department": "Test"
            }
        }
      },
      {
        "Effect": "Allow",
        "Action": [
            "ec2:ModifySecurityGroupRules"
        ],
        "Resource": [
            "arn:aws:ec2:region:111122223333:security-group-rule/*"
        ]
      }
    ]
}
```

Amazon EC2: permite iniciar o detener instancias EC2 que un usuario haya etiquetado, mediante programación y en la consola

En este ejemplo se muestra cómo crear una política basada en identidad que permita a un usuario de IAM iniciar o detener instancias de EC2, pero solo si la etiqueta `Owner` de la instancia tiene el valor del nombre de usuario de dicho usuario. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
```

```
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeInstances",
            "Resource": "*"
        }
    ]
}
```

EC2: iniciar o detener instancias en función de las etiquetas

Este ejemplo muestra cómo podría crear una política basada en identidad que permita iniciar o detener instancias con el par de valor de clave de la etiqueta `Project = DataAnalytics`, pero solo por las entidades principales con el par de valor de clave de la etiqueta `Department = Data`. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La condición de la política se cumple si ambas partes de la condición son ciertas. La instancia debe tener la etiqueta `Project=DataAnalytics`. Además, la entidad principal de IAM (usuario o rol) que realiza la solicitud debe tener la etiqueta `Department=Data`.

> **Note**
>
> Como práctica recomendada, asocie políticas con la clave de condición `aws:PrincipalTag` para grupos de IAM en el caso en el que algunos usuarios pudieran tener la etiqueta especificada y otros no.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
```

```
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Project": "DataAnalytics",
                    "aws:PrincipalTag/Department": "Data"
                }
            }
        }
    ]
}
```

EC2: iniciar o detener instancias basándose en etiquetas de recursos y principal coincidentes

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a una entidad principal iniciar o detener una instancia de Amazon EC2 cuando la etiqueta de recurso de la instancia y la etiqueta de la entidad de seguridad tengan el mismo valor para la clave de etiqueta `CostCenter`. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

> **Note**
>
> Como práctica recomendada, asocie políticas con la clave de condición `aws:PrincipalTag` para grupos de IAM en el caso en el que algunos usuarios pudieran tener la etiqueta especificada y otros no.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "ec2:startInstances",
            "ec2:stopInstances"
```

```
        ],
        "Resource": "*",
        "Condition": {"StringEquals":
            {"aws:ResourceTag/CostCenter": "${aws:PrincipalTag/CostCenter}"}}
    }
}
```

Amazon EC2: permite el acceso completo a EC2 en una región determinada, mediante programación y en la consola

En este ejemplo se muestra cómo crear una política basada en identidad que permita el acceso completo a EC2 dentro de una región específica. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política. Para obtener una lista de códigos de región, consulte Regiones disponibles en la Guía del usuario de Amazon EC2.

Como alternativa, puede utilizar la clave de condición global `aws:RequestedRegion`, que es admitida por todas las acciones de la API de Amazon EC2. Para obtener más información, consulte Ejemplo: Limitar el acceso a una región específica en la Guía del usuario de Amazon EC2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ec2:*",
            "Resource": "*",
            "Effect": "Allow",
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2"
                }
            }
        }
    ]
}
```

Amazon EC2: permite iniciar o detener una instancia EC2 y modificar un grupo de seguridad mediante programación y en la consola

En este ejemplo se muestra cómo crear una política basada en identidad que permita iniciar o detener una determinada instancia de EC2 y modificar un grupo de seguridad específico. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeSecurityGroupReferences",
        "ec2:DescribeStaleSecurityGroups"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:StartInstances",
        "ec2:StopInstances"
      ],
      "Resource": [
        "arn:aws:ec2:*:*:instance/i-instance-id",
        "arn:aws:ec2:*:*:security-group/sg-security-group-id"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Amazon EC2: requiere MFA (GetSessionToken) para operaciones EC2 específicas

En este ejemplo se muestra cómo crear una política basada en identidad que permita el acceso completo a todas las operaciones de API de AWS en Amazon EC2. Sin embargo, deniega de forma explícita el acceso a las operaciones de API `StopInstances` y `TerminateInstances` si el usuario no está autenticado mediante la Multi-Factor Authentication (MFA). Para hacer esto mediante programación, el usuario debe incluir los valores opcionales `TokenCode` y `SerialNumber` al llamar a la operación `GetSessionToken`. Esta operación devuelve las credenciales temporales que se hayan autenticado con MFA. Para obtener más información acerca del GetSessionToken, consulte GetSessionToken: credenciales temporales para usuarios de entornos que no son de confianza.

¿Qué hace esta política?

- La instrucción `AllowAllActionsForEC2` permite todas las acciones de Amazon EC2.
- La declaración `DenyStopAndTerminateWhenMFAIsNotPresent` rechaza las acciones `TerminateInstances` y `StopInstances` cuando falta el contexto de MFA. Esto significa que las acciones se deniegan cuando falta el contexto de la autenticación multifactor (lo que indica que no se ha utilizado MFA). Una denegación anula el permiso.

> **Note**
>
> La verificación de la condición de `MultiFactorAuthPresent` en la instrucción `Deny` no debe ser `{"Bool":{"aws:MultiFactorAuthPresent":false}}` ya que dicha clave no está presente y no puede evaluarse cuando no se utiliza MFA. Por lo tanto, utilice la verificación `BoolIfExists` para ver si la clave está presente antes de comprobar el valor. Para obtener más información, consulte Operadores de condición …IfExists.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllActionsForEC2",
            "Effect": "Allow",
            "Action": "ec2:*",
            "Resource": "*"
        },
        {
```

```
            "Sid": "DenyStopAndTerminateWhenMFAIsNotPresent",
            "Effect": "Deny",
            "Action": [
                "ec2:StopInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {"aws:MultiFactorAuthPresent": false}
            }
        }
    ]
}
```

Amazon EC2: limita el término de instancias EC2 en un rango de direcciones IP

Este ejemplo muestra cómo puede crear una política basada en identidad que limita instancias de EC2 al permitir la acción, pero denegar explícitamente el acceso cuando la solicitud procede de una dirección IP fuera del rango especificado. La política es útil cuando las direcciones IP de su empresa están dentro de los rangos especificados. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Si esta política se utiliza en combinación con otras políticas que permiten la acción `ec2:TerminateInstances` (como la política administrada por AWS AmazonEC2FullAccess), se deniega el acceso. Esto se debe a que una instrucción de denegación explícita prevalece sobre las instrucciones de permiso. Para obtener más información, consulte the section called “Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta”.

> **Important**
>
> La clave de condición `aws:SourceIp` deniega el acceso a un servicio de AWS, como AWS CloudFormation, que realiza llamadas en su nombre. Para obtener más información sobre el uso de la clave de condición `aws:SourceIp`, consulte Claves de contexto de condición globales de AWS.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

IAM: acceso a la API del simulador de políticas

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita utilizar la API del simulador de políticas para las políticas asociadas a un usuario, grupo o rol de la Cuenta de AWS actual. Esta política también permite el acceso para simular políticas menos sensibles transferidas a la API como cadenas. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForCustomPolicy",
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulateCustomPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
```

```
    ]
}
```

> **Note**
>
> Para permitir el acceso de un usuario a la consola del simulador de políticas para simular políticas asociadas a un usuario, grupo o rol en la Cuenta de AWS actual, consulte IAM: permite el acceso a la consola del simulador de políticas.

IAM: permite el acceso a la consola del simulador de políticas

Este ejemplo muestra cómo podría crear una política basada en identidad que permita utilizar la consola del simulador de políticas para las políticas asociadas a un usuario, grupo o rol de la Cuenta de AWS actual. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

Puede obtener acceso a la consola del simulador de políticas de IAM en: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListRolePolicies",
                "iam:ListRoles",
```

```
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

IAM: asumir funciones que tienen una etiqueta específica

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a un usuario de IAM asumir roles con el par de valor de clave de etiqueta `Project = ExampleCorpABC`. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Si una función con esta etiqueta existe en la misma cuenta que el usuario, este puede asumir esa función. Si una función con esta etiqueta existe en una cuenta que no sea la del usuario, requiere permisos adicionales. La política de confianza de la función entre cuentas también debe permitir al usuario o a todos los miembros de la cuenta del usuario para que asuman la función. Para obtener más información acerca de cómo utilizar las funciones para el acceso entre cuentas, consulte Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS propia.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AssumeTaggedRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:ResourceTag/Project": "ExampleCorpABC"}
            }
        }
    ]
}
```

IAM: permite y deniega el acceso a varios servicios mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita acceso completo a varios servicios y acceso de autoadministración limitado en IAM. También deniega a los usuarios el acceso al bucket de `logs` de Amazon S3 o a la instancia de Amazon EC2 de `i-1234567890abcdef0` Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

> **Warning**
>
> Esta política permite acceso completo a cada acción y recurso en varios servicios. Esta política debe aplicarse únicamente a los administradores de confianza.

Puede utilizar esta política como límite de permisos para definir los permisos máximos que una política basada en identidad puede conceder a un usuario de IAM. Para obtener más información, consulte Delegación de responsabilidades en otras personas mediante el uso de límites de permisos. Cuando la política se utiliza como un límite de permisos para un usuario, las declaraciones definen los siguientes límites:

- La declaración `AllowServices` permite acceso completo a los servicios de AWS especificados. Esto significa que las acciones del usuario en estos servicios solamente están limitadas por las políticas de permisos que se han asociado al usuario.
- La instrucción `AllowIAMConsoleForCredentials` permite el acceso para obtener una lista de todos los usuarios de IAM. Este acceso es necesario para recorrer la página Users (Usuarios) de la AWS Management Console. También permite ver los requisitos de la contraseña de la cuenta, para que el usuario pueda cambiar su propia contraseña.
- La instrucción `AllowManageOwnPasswordAndAccessKeys` permite a los usuarios administrar únicamente su propia contraseña de la consola y sus claves de acceso mediante programación. Esto es importante porque si otra política brinda al usuario acceso completo a IAM, este podría cambiar sus propios permisos o los de otros usuarios. Esta instrucción impide que eso ocurra.
- La instrucción `DenyS3Logs` deniega explícitamente el acceso al bucket `logs`. Esta política aplica las restricciones de la empresa sobre el usuario.
- La instrucción `DenyEC2Production` deniega explícitamente el acceso a la instancia `i-1234567890abcdef0`.

Esta política no permite el acceso a otros servicios o acciones. Cuando la política se utiliza como un límite de permisos en un usuario, aunque otras políticas asociadas al usuario permitan esas acciones, AWS deniega la solicitud.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudwatch:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowIAMConsoleForCredentials",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswordAndAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*LoginProfile*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::logs",
```

```
                "arn:aws:s3:::logs/*"
            ]
        },
        {
            "Sid": "DenyEC2Production",
            "Effect": "Deny",
            "Action": "ec2:*",
            "Resource": "arn:aws:ec2:*:*:instance/i-1234567890abcdef0"
        }
    ]
}
```

IAM: agregar una etiqueta específica a un usuario con una etiqueta específica

Este ejemplo muestra cómo podría crear una política basada en identidad que permita agregar la clave de etiqueta `Department` con los valores de etiqueta `Marketing`, `Development` o `QualityAssurance` a un usuario de IAM. El usuario ya debe incluir el par clave-valor `JobFunction = manager`. Puede utilizar esta política para exigir que un administrador solo pertenezca a uno de tres departamentos. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La instrucción `ListTagsForAllUsers` permite ver las etiquetas de todos los usuarios de la cuenta.

La primera condición de la instrucción `TagManagerWithSpecificDepartment` utiliza el operador de condición `StringEquals`. La condición se cumple si ambas partes de la condición son ciertas. El usuario que se etiqueta ya debe tener la etiqueta `JobFunction=Manager`. La solicitud debe incluir la clave de etiqueta `Department` con uno de los valores de etiqueta que se indican.

La segunda condición utiliza el operador de condición `ForAllValues:StringEquals`. La condición se cumple si todas las claves de etiqueta de la solicitud coinciden con la clave de la política. Esto significa que la única clave de etiqueta de la solicitud debe ser `Department`. Para obtener más información acerca del uso de `ForAllValues`, consulte Claves de contexto multivalor.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListTagsForAllUsers",
            "Effect": "Allow",
```

```
            "Action": [
                "iam:ListUserTags",
                "iam:ListUsers"
            ],
            "Resource": "*"
        },
        {
            "Sid": "TagManagerWithSpecificDepartment",
            "Effect": "Allow",
            "Action": "iam:TagUser",
            "Resource": "*",
            "Condition": {"StringEquals": {
                "iam:ResourceTag/JobFunction": "Manager",
                "aws:RequestTag/Department": [
                    "Marketing",
                    "Development",
                    "QualityAssurance"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "Department"}
            }
        }
    ]
}
```

IAM: agregar una etiqueta específica con valores específicos

Este ejemplo muestra cómo podría crear una política basada en identidad que permita agregar solo la clave de la etiqueta `CostCenter` y el valor de la etiqueta `A-123` o el valor de la etiqueta `B-456` a cualquier rol o usuario de IAM. Puede utilizar esta política para limitar el etiquetado a una clave de etiqueta y un conjunto de valores de etiqueta específicos. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La instrucción `ConsoleDisplay` permite ver las etiquetas de todos los usuarios y funciones de la cuenta.

La primera condición de la instrucción `AddTag` utiliza el operador de condición `StringEquals`. La condición se cumple si la solicitud incluye la clave de etiqueta `CostCenter` con uno de los valores de etiqueta que se indican.

La segunda condición utiliza el operador de condición `ForAllValues:StringEquals`. La condición se cumple si todas las claves de etiqueta de la solicitud coinciden con la clave de la política. Esto significa que la única clave de etiqueta de la solicitud debe ser `CostCenter`. Para obtener más información acerca del uso de `ForAllValues`, consulte Claves de contexto multivalor.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConsoleDisplay",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:GetUser",
                "iam:ListRoles",
                "iam:ListRoleTags",
                "iam:ListUsers",
                "iam:ListUserTags"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AddTag",
            "Effect": "Allow",
            "Action": [
                "iam:TagUser",
                "iam:TagRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/CostCenter": [
                        "A-123",
                        "B-456"
                    ]
                },
                "ForAllValues:StringEquals": {"aws:TagKeys": "CostCenter"}
            }
        }
    ]
}
```

IAM: crear nuevos usuarios solo con etiquetas específicas

Este ejemplo muestra cómo podría crear una política basada en identidad que permita la creación de usuarios de IAM, pero solo con una o ambas claves de etiqueta `Department` y `JobFunction`. La clave de etiqueta `Department` debe tener el valor de etiqueta `Development` o `QualityAssurance`. La clave de etiqueta `JobFunction` debe tener el valor de etiqueta `Employee`. Puede utilizar esta política para exigir que los nuevos usuarios tengan una función de trabajo y un departamento específicos. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La primera condición de la instrucción utiliza el operador de condición `StringEqualsIfExists`. Si hay una etiqueta con la clave `JobFunction` o `Department` en la solicitud, deberá tener el valor especificado. Si no hay ninguna clave, la evaluación considera que la condición se cumple. La única manera de la evaluación considere que la condición no se cumple es que una de las claves de condición especificadas se encuentre en la solicitud, pero con un valor distinto de los permitidos. Para obtener más información acerca del uso de `IfExists`, consulte Operadores de condición … IfExists.

La segunda condición utiliza el operador de condición `ForAllValues:StringEquals`. La condición se cumple si hay una coincidencia entre todas las claves de etiqueta especificadas en la solicitud con al menos uno de los valores de la política. Esto significa que todas las etiquetas de la solicitud deben encontrarse en esta lista. Sin embargo, la solicitud solo puede incluir una de las etiquetas de la lista. Por ejemplo, puede crear un usuario de IAM que solo tenga la etiqueta `Department=QualityAssurance`. Sin embargo, no puede crear un usuario de IAM con las etiquetas `JobFunction=employee` y `Project=core`. Para obtener más información acerca del uso de `ForAllValues`, consulte Claves de contexto multivalor.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagUsersWithOnlyTheseTags",
            "Effect": "Allow",
            "Action": [
                "iam:CreateUser",
                "iam:TagUser"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "aws:RequestTag/Department": [
                        "Development",
                        "QualityAssurance"
                    ],
                    "aws:RequestTag/JobFunction": "Employee"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "Department",
                        "JobFunction"
                    ]
                }
            }
        }
    ]
}
```

IAM: generar y recuperar de informes de credenciales de IAM

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios generar y descargar un informe que contenga una lista de todos los usuarios de IAM en su Cuenta de AWS. El informe muestra el estado de las credenciales del usuario, tales como las contraseñas, las claves de acceso, los dispositivos MFA y los certificados de firma. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

Para obtener más información sobre los informes de credenciales, consulte Obtención de informes de credenciales para su cuenta de Cuenta de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateCredentialReport",
            "iam:GetCredentialReport"
        ],
        "Resource": "*"
    }
```

```
}
```

IAM: permite administrar la pertenencia a un grupo mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita actualizar la pertenencia del grupo denominado `MarketingTeam`. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

¿Qué hace esta política?

- La declaración `ViewGroups` permite que el usuario genere una lista de todos los usuarios y grupos en el AWS Management Console. También permite al usuario ver información básica acerca de los usuarios de la cuenta. Estos permisos deben estar en su propia instrucción, ya que no admiten o no requieren un ARN de recurso. En lugar de ello, los permisos especifican `"Resource" : "*"`.
- La declaración `ViewEditThisGroup` permite al usuario ver información sobre el grupo `MarketingTeam` y añadir y eliminar usuarios de ese grupo.

Esta política no permite al usuario ver o editar los permisos de los usuarios o el grupo `MarketingTeam`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewGroups",
            "Effect": "Allow",
            "Action": [
                "iam:ListGroups",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:ListGroupsForUser"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "ViewEditThisGroup",
            "Effect": "Allow",
            "Action": [
                "iam:AddUserToGroup",
                "iam:RemoveUserFromGroup",
                "iam:GetGroup"
            ],
            "Resource": "arn:aws:iam::*:group/MarketingTeam"
        }
    ]
}
```

IAM: administrar una etiqueta específica

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita eliminar solo la etiqueta de IAM con la clave de etiqueta `Department` de las entidades (usuarios y roles). Esta política no limita el valor de la etiqueta `Department`. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:TagUser",
            "iam:TagRole",
            "iam:UntagUser",
            "iam:UntagRole"

        ],
        "Resource": "*",
        "Condition": {"ForAllValues:StringEquals": {"aws:TagKeys": "Department"}}
    }
}
```

IAM: pasar una función de IAM a un servicio de AWS específico

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita transferir cualquier rol de servicio de IAM al servicio de Amazon CloudWatch. Esta política concede

los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Una función de servicio es una función de IAM que especifica un servicio de AWS como la entidad principal que puede asumir la función. Esta permite que el servicio asuma la función y obtenga acceso a los recursos de otros servicios en su nombre. Para permitir que Amazon CloudWatch asuma el rol que esté pasando, debe especificar la entidad principal de servicio `cloudwatch.amazonaws.com` como la entidad principal de la política de confianza de su rol. El servicio define la entidad principal de servicio. Para conocer la entidad principal de un servicio, consulte la documentación correspondiente a dicho servicio. En algunos servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios para los que se indique Sí en la columna Roles vinculados a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión. Busque `amazonaws.com` para ver el principal del servicio.

Para obtener más información acerca de cómo pasar una función de servicio a un servicio, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:PassedToService": "cloudwatch.amazonaws.com"}
            }
        }
    ]
}
```

IAM: permite acceso de solo lectura a la consola de IAM sin informes

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM hacer cualquier acción de IAM que comience con la cadena `Get` o `List`. A medida que los usuarios trabajan con la consola, esta realiza solicitudes a IAM para obtener listas de grupos, usuarios, funciones y políticas, y para generar informes sobre esos recursos.

El asterisco actúa como un carácter comodín. Si utiliza `iam:Get*` en una política, los permisos resultantes incluyen todas las acciones de IAM que comienzan con `Get`, por ejemplo, `GetUser` y `GetRole`. Los comodines son útiles si se añaden nuevos tipos de entidades a IAM en el futuro. En ese caso, los permisos concedidos por la política permiten automáticamente a los usuarios generar listas y obtener los detalles acerca de esas nuevas entidades.

Esta política no se puede utilizar para generar informes o detalles de servicio al que se ha accedido por última vez. Para obtener una política diferente que lo permita, consulte IAM: ermite el acceso de solo lectura a la consola de IAM.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
}
```

IAM: ermite el acceso de solo lectura a la consola de IAM

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM hacer cualquier acción de IAM que comience con la cadena `Get`, `List`, o `Generate`. A medida que los usuarios trabajan con la consola, esta realiza solicitudes a IAM para obtener listas de grupos, usuarios, funciones y políticas, y para generar informes sobre esos recursos.

El asterisco actúa como un carácter comodín. Si utiliza `iam:Get*` en una política, los permisos resultantes incluyen todas las acciones de IAM que comienzan con `Get`, por ejemplo, `GetUser` y `GetRole`. Utilizar un comodín resulta beneficioso, especialmente si se añaden nuevos tipos de entidades a IAM en el futuro. En ese caso, los permisos concedidos por la política permiten automáticamente a los usuarios generar listas y obtener los detalles acerca de esas nuevas entidades.

Utilice esta política para el acceso a la consola que incluye permisos para generar informes o detalles de servicio al que se accede por última vez. Para obtener una política diferente que no

permite generar acciones, consulte IAM: permite acceso de solo lectura a la consola de IAM sin informes.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:Get*",
            "iam:List*",
            "iam:Generate*"
        ],
        "Resource": "*"
    }
}
```

IAM: permite que usuarios específicos de IAM administren un grupo, mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a usuarios de IAM específicos administrar el grupo `AllUsers`. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

¿Qué hace esta política?

- La declaración `AllowAllUsersToListAllGroups` permite generar listas de todos los grupos. Esto es necesario para acceder a la consola. Este permiso debe estar en su propia declaración, ya que no admite el ARN de un recurso. En lugar de ello, los permisos especifican `"Resource" : "*"`.
- La declaración `AllowAllUsersToViewAndManageThisGroup` permite que se realicen todas las acciones de grupo que pueden ejecutarse en el tipo de recurso de grupo. No permite la acción `ListGroupsForUser`, que puede llevarse a cabo en un tipo de recurso de usuario y no en un tipo de recurso de grupo. Para obtener más información acerca de los tipos de recurso que puede especificar para una acción de IAM, consulte Claves de acciones, recursos y condición de AWS Identity and Access Management.
- La declaración `LimitGroupManagementAccessToSpecificUsers` deniega a los usuarios con los nombres especificados el acceso a escribir y las acciones de grupo de administración

de permisos. Cuando un usuario especificado en la política intenta realizar cambios en el grupo, esta declaración no deniega la solicitud. Esta solicitud la permite la declaración `AllowAllUsersToViewAndManageThisGroup`. Si otros usuarios intentan realizar estas operaciones, se deniega la solicitud. Puede ver las acciones de IAM que se definen con los niveles de acceso de Escritura o Administración de permisos mientras crea esta política en la consola de IAM. Para ello, cambie de la pestaña JSON a la pestaña Visual editor (Editor visual). Para obtener más información acerca de los niveles de acceso, consulte Claves de condición, recursos y acciones de AWS Identity and Access Management.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAllUsersToListAllGroups",
            "Effect": "Allow",
            "Action": "iam:ListGroups",
            "Resource": "*"
        },
        {
            "Sid": "AllowAllUsersToViewAndManageThisGroup",
            "Effect": "Allow",
            "Action": "iam:*Group*",
            "Resource": "arn:aws:iam::*:group/AllUsers"
        },
        {
            "Sid": "LimitGroupManagementAccessToSpecificUsers",
            "Effect": "Deny",
            "Action": [
                "iam:AddUserToGroup",
                "iam:CreateGroup",
                "iam:RemoveUserFromGroup",
                "iam:DeleteGroup",
                "iam:AttachGroupPolicy",
                "iam:UpdateGroup",
                "iam:DetachGroupPolicy",
                "iam:DeleteGroupPolicy",
                "iam:PutGroupPolicy"
            ],
            "Resource": "arn:aws:iam::*:group/AllUsers",
            "Condition": {
                "StringNotEquals": {
```

```
                    "aws:username": [
                        "srodriguez",
                        "mjackson",
                        "adesai"
                    ]
                }
            }
        }
    ]
}
```

IAM: permite establecer los requisitos de contraseña de la cuenta, mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a un usuario ver y actualizar los requisitos de contraseña de su cuenta. Los requisitos de contraseña especifican los requisitos de complejidad y periodos de rotación obligatorios para las contraseñas de los miembros de la cuenta. Esta política define los permisos para el acceso programático y a la consola.

Para obtener información sobre cómo configurar la política de requisitos de contraseñas para su cuenta, consulte Configuración de una política de contraseñas de la cuenta para usuarios de IAM.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GetAccountPasswordPolicy",
            "iam:UpdateAccountPasswordPolicy"
        ],
        "Resource": "*"
    }
}
```

IAM: obtiene acceso a la API del simulador de políticas en función de la ruta de acceso del usuario

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita utilizar la API del simulador de políticas únicamente a aquellos usuarios que tengan la ruta `Department/Development`. Esta política concede los permisos necesarios para llevar a cabo esta acción

mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetContextKeysForPrincipalPolicy",
                "iam:SimulatePrincipalPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

> **Note**
>
> Para crear una política que permita utilizar la consola del simulador de políticas a aquellos usuarios que tengan la ruta `Department/Development`, consulte IAM: permite el acceso a la consola de simulador de políticas en función de la ruta de acceso del usuario.

IAM: permite el acceso a la consola de simulador de políticas en función de la ruta de acceso del usuario

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita utilizar la consola del simulador de políticas únicamente a aquellos usuarios que tengan la ruta `Department/Development`. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Puede obtener acceso a la consola del simulador de políticas de IAM en: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Action": [
                "iam:GetPolicy",
                "iam:GetUserPolicy"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "iam:GetUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroupsForUser",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:user/Department/Development/*"
        }
    ]
}
```

IAM: permite a los usuarios de IAM administrar ellos mismos un dispositivo MFA

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM administrar automáticamente su dispositivo de autenticación multifactor (MFA). Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

> **Note**
>
> Si un usuario de IAM con esta política no está autenticado por MFA, esta política deniega el acceso a todas las acciones de AWS excepto las necesarias para autenticarse mediante MFA. Si añade estos permisos a un usuario que haya iniciado sesión en AWS, es posible que tenga que cerrar sesión y volver a iniciarla para ver estos cambios.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "AllowListActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:ListVirtualMFADevices"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUserToCreateVirtualMFADevice",
            "Effect": "Allow",
            "Action": [
                "iam:CreateVirtualMFADevice"
            ],
            "Resource": "arn:aws:iam::*:mfa/*"
        },
        {
            "Sid": "AllowUserToManageTheirOwnMFA",
            "Effect": "Allow",
            "Action": [
                "iam:EnableMFADevice",
                "iam:GetMFADevice",
                "iam:ListMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
            "Sid": "AllowUserToDeactivateTheirOwnMFAOnlyWhenUsingMFA",
            "Effect": "Allow",
            "Action": [
                "iam:DeactivateMFADevice"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ],
            "Condition": {
                "Bool": {
                    "aws:MultiFactorAuthPresent": "true"
                }
            }
        },
        {
```

```
            "Sid": "BlockMostAccessUnlessSignedInWithMFA",
            "Effect": "Deny",
            "NotAction": [
                "iam:CreateVirtualMFADevice",
                "iam:EnableMFADevice",
                "iam:ListMFADevices",
                "iam:ListUsers",
                "iam:ListVirtualMFADevices",
                "iam:ResyncMFADevice"
            ],
            "Resource": "*",
            "Condition": {
                "BoolIfExists": {
                    "aws:MultiFactorAuthPresent": "false"
                }
            }
        }
    ]
}
```

IAM: permite a los usuarios de IAM actualizar sus propias credenciales mediante programación en la consola

En este ejemplo, se muestra cómo podría crear una política basada en identidad que les permita a los usuarios de IAM actualizar sus propias claves de acceso, certificados de firma, credenciales específicas de servicio y contraseñas. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListUsers",
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iam:*AccessKey*",
                "iam:ChangePassword",
                "iam:GetUser",
                "iam:*ServiceSpecificCredential*",
                "iam:*SigningCertificate*"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        }
    ]
}
```

Para descubrir cómo un usuario puede cambiar su propia contraseña en la consola, consulte the section called “Cómo un usuario de IAM cambia su propia contraseña”.

IAM: ver la información del último acceso al servicio para una política de Organizaciones

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita visualizar la información de acceso reciente al servicio para una determinada política de Organizations. Esta política permite recuperar datos para la política de control de servicios (SCP) con el ID `p-policy123`. La persona que genera y ve el informe debe autenticarse con credenciales de cuenta de administración de AWS Organizations. Esta política permite al solicitante recuperar los datos de cualquier entidad de Organizations en su organización. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para obtener información importante sobre la información de acceso reciente, como los permisos necesarios, la solución de problemas y las regiones admitidas, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowOrgsReadOnlyAndIamGetReport",
            "Effect": "Allow",
            "Action": [
                "iam:GetOrganizationsAccessReport",
                "organizations:Describe*",
```

```
                "organizations:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowGenerateReportOnlyForThePolicy",
            "Effect": "Allow",
            "Action": "iam:GenerateOrganizationsAccessReport",
            "Resource": "*",
            "Condition": {
                "StringEquals": {"iam:OrganizationsPolicyId": "p-policy123"}
            }
        }
    ]
}
```

IAM: limita las políticas administradas que pueden aplicarse a un usuario, grupo o rol de

En este ejemplo se muestra cómo podría crear una política basada en identidad que limite la administración del cliente y las políticas administradas de AWS que pueden aplicarse a un rol, grupo o usuario de IAM. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:AttachUserPolicy",
            "iam:DetachUserPolicy"
        ],
        "Resource": "*",
        "Condition": {
            "ArnEquals": {
                "iam:PolicyARN": [
                    "arn:aws:iam::*:policy/policy-name-1",
                    "arn:aws:iam::*:policy/policy-name-2"
                ]
            }
```

```
        }
    }
}
```

AWS: denegar el acceso a recursos que están fuera de su cuenta, excepto a las políticas de IAM administradas por AWS

El uso de `aws:ResourceAccount` en sus políticas basadas en identidad puede afectar al usuario o a la capacidad del rol para utilizar algunos servicios que requieren la interacción con los recursos de las cuentas que son propiedad de un servicio.

Puede crear una política con una excepción para permitir la política de IAM administrada por AWS. Una cuenta administrada por un servicio fuera de AWS Organizations es propietaria de la política de IAM administrada. Hay cuatro acciones de IAM que enumeran y recuperan las políticas administradas por AWS. Utilice estas acciones en el elemento `NotAction` de la declaración. `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1` en la política.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToResourcesInSpecificAccountsAndSpecificService1",
      "Effect": "Deny",
      "NotAction": [
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListEntitiesForPolicy",
        "iam:ListPolicies"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "111122223333"
          ]
        }
      }
    }
  ]
}
```

AWS Lambda: permite que una función Lambda acceda a una tabla de Amazon DynamoDB

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita el acceso de escritura y lectura a una tabla de Amazon DynamoDB específica. La política también permite escribir en archivos de registro en CloudWatch Logs. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Para utilizar esta política, adjunte la política a una función de servicio de Lambda. Una función de servicio es una función que usted crea en su cuenta para permitir que un servicio realice acciones en su nombre. Esta función de servicio debe incluir AWS Lambda como la entidad principal en la política de confianza. Para obtener más información acerca de cómo utilizar esta política, consulte Cómo crear una política de IAM AWS para conceder acceso AWS Lambda a una tabla de Amazon DynamoDB en Security Blog AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadWriteTable",
            "Effect": "Allow",
            "Action": [
                "dynamodb:BatchGetItem",
                "dynamodb:GetItem",
                "dynamodb:Query",
                "dynamodb:Scan",
                "dynamodb:BatchWriteItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable"
        },
        {
            "Sid": "GetStreamRecords",
            "Effect": "Allow",
            "Action": "dynamodb:GetRecords",
            "Resource": "arn:aws:dynamodb:*:*:table/SampleTable/stream/* "
        },
        {
            "Sid": "WriteLogStreamsAndGroups",
```

```
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateLogGroup",
            "Effect": "Allow",
            "Action": "logs:CreateLogGroup",
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: permite el acceso completo a la base de datos de RDS dentro de una región específica

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita el acceso completo a la base de datos de RDS dentro de una región específica. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "Effect": "Allow",
            "Action": ["rds:Describe*"],
            "Resource": ["*"]
        }
    ]
}
```

Amazon RDS: permite restaurar bases de datos de RDS, mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita restaurar las bases de datos de RDS. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "rds:CreateDBParameterGroup",
                "rds:CreateDBSnapshot",
                "rds:DeleteDBSnapshot",
                "rds:Describe*",
                "rds:DownloadDBLogFilePortion",
                "rds:List*",
                "rds:ModifyDBInstance",
                "rds:ModifyDBParameterGroup",
                "rds:ModifyOptionGroup",
                "rds:RebootDBInstance",
                "rds:RestoreDBInstanceFromDBSnapshot",
                "rds:RestoreDBInstanceToPointInTime"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon RDS: permite a los propietarios de etiquetas el acceso completo a los recursos de RDS que han etiquetado

Este ejemplo muestra cómo podría crear una política basada en identidad que permita a los propietarios de las etiquetas obtener acceso completo a los recursos de RDS que tengan etiquetados. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "rds:Describe*",
                "rds:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "rds:DeleteDBInstance",
                "rds:RebootDBInstance",
                "rds:ModifyDBInstance"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:db-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyOptionGroup",
                "rds:DeleteOptionGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:og-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBParameterGroup",
                "rds:ResetDBParameterGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:pg-tag/Owner": "${aws:username}"}
            }
        },
```

```
        {
            "Action": [
                "rds:AuthorizeDBSecurityGroupIngress",
                "rds:RevokeDBSecurityGroupIngress",
                "rds:DeleteDBSecurityGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:secgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:DeleteDBSnapshot",
                "rds:RestoreDBInstanceFromDBSnapshot"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:snapshot-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyDBSubnetGroup",
                "rds:DeleteDBSubnetGroup"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEqualsIgnoreCase": {"rds:subgrp-tag/Owner": "${aws:username}"}
            }
        },
        {
            "Action": [
                "rds:ModifyEventSubscription",
                "rds:AddSourceIdentifierToSubscription",
                "rds:RemoveSourceIdentifierFromSubscription",
                "rds:DeleteEventSubscription"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
```

```
                "StringEqualsIgnoreCase": {"rds:es-tag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Amazon S3: permite a los usuarios de Amazon Cognito obtener acceso a los objetos de su bucket

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios de Amazon Cognito acceder a objetos de un bucket de S3 específico. Esta política permite el acceso únicamente a los objetos cuyo nombre incluya `cognito`, el nombre de la aplicación y el ID del usuario federado, representados por la variable ${cognito-identity.amazonaws.com:sub} Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

> **Note**
>
> El valor “sub” utilizado en la clave de objeto no es el subvalor del usuario en el grupo de usuarios. Se trata del ID de identidad asociado al usuario en el grupo de identidades.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListYourObjects",
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": [
        "arn:aws:s3:::bucket-name"
      ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
            "cognito/application-name/${cognito-identity.amazonaws.com:sub}/*"
          ]
        }
```

```
      }
    },
    {
      "Sid": "ReadWriteDeleteYourObjects",
      "Effect": "Allow",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::bucket-name/cognito/application-name/${cognito-
identity.amazonaws.com:sub}/*"
      ]
    }
  ]
}
```

Amazon Cognito ofrece autenticación, autorización y administración de usuarios para sus aplicaciones móviles y web. Los usuarios pueden iniciar sesión directamente con un nombre de usuario y una contraseña o a través de un tercero como Facebook, Amazon o Google.

Los dos componentes principales de Amazon Cognito son los grupos de usuarios y los grupos de identidades. Los grupos de usuarios son directorios de usuarios que proporcionan a los usuarios de las aplicaciones opciones para inscribirse e iniciar sesión. Los grupos de identidades permiten conceder a los usuarios acceso a otros servicios de AWS. Puede utilizar los grupos de identidades y los grupos de usuarios juntos o por separado.

Para obtener más información sobre Amazon Cognito, consulte la Guía del usuario de Amazon Cognito.

Amazon S3: permite a los usuarios federados obtener acceso a su directorio principal de S3, mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política de IAM que permita a los usuarios federados acceder a su propio objeto de bucket de directorio principal en S3. El directorio principal es un bucket que incluye una carpeta home y carpetas para usuarios federados individuales. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La variable `${aws:userid}` de esta política se resuelve como `role-id:specified-name`. La parte `role-id` del ID de usuario federado es un identificador exclusivo que se asigna al rol del usuario federado durante la creación. Para obtener más información, consulte Identificadores únicos. El `specified-name` es el parámetro RoleSessionName que se pasa a la solicitud `AssumeRoleWithWebIdentity` cuando el usuario federado asume su rol.

Puede ver el ID de rol con el comando `aws iam get-role --role-name` *`specified-name`* de la AWS CLI. Por ejemplo, imagine que especifica el nombre fácil de recordar `John` y que la CLI devuelve el ID de rol `AROAXXT2NJT7D3SIQN7Z6`. En este caso, el ID de usuario federado es `AROAXXT2NJT7D3SIQN7Z6:John`. En este caso, la política permite que el usuario federado John tenga acceso al bucket de Amazon S3 con el prefijo `AROAXXT2NJT7D3SIQN7Z6:John`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:userid}",
                "arn:aws:s3:::bucket-name/home/${aws:userid}/*"
            ]
        }
    ]
}
```

Amazon S3: acceso al bucket de S3, pero bucket de producción denegado sin MFA reciente

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a un administrador de Amazon S3 obtener acceso a cualquier bucket, incluida la actualización, incorporación y eliminación de objetos. Sin embargo, deniega explícitamente el acceso al bucket de `Production` si el usuario no ha iniciado sesión con la autenticación multifactor (MFA) en los últimos treinta minutos. Esta política concede los permisos necesarios para realizar esta acción en la consola o mediante programación a través de la AWS CLI o la API de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Esta política nunca permite acceso mediante programación al bucket `Production` con claves de acceso de usuario a largo plazo. Esto se logra con la clave de condición `aws:MultiFactorAuthAge` con el operador de condición `NumericGreaterThanIfExists`. Esta condición de la política devuelve `true` si el MFA no está presente o si la edad del MFA es superior a 30 minutos. En esas situaciones, se deniega el acceso. Para acceder al bucket `Production` mediante programación, el administrador de S3 debe utilizar credenciales temporales que se generaron en los últimos 30 minutos mediante la operación de API GetSessionToken.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListAllS3Buckets",
            "Effect": "Allow",
            "Action": ["s3:ListAllMyBuckets"],
```

```
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketLevelActions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowBucketObjectActions",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:DeleteObject"
            ],
            "Resource": "arn:aws:s3:::*/*"
        },
        {
            "Sid": "RequireMFAForProductionBucket",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::Production/*",
                "arn:aws:s3:::Production"
            ],
            "Condition": {
                "NumericGreaterThanIfExists": {"aws:MultiFactorAuthAge": "1800"}
            }
        }
    ]
}
```

Amazon S3: permite a los usuarios de IAM obtener acceso a su directorio principal de S3, mediante programación y en la consola.

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a los usuarios de IAM obtener acceso a su propio objeto de bucket de directorio principal en S3. El

directorio principal es un bucket que incluye una carpeta home y carpetas para usuarios individuales. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Esta política no funcionará cuando se utilicen los roles de IAM porque la variable `aws:username` no está disponible cuando se usan los roles de IAM. Para obtener información acerca de los valores de clave principales, consulte Valores clave principales.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
```

```
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:username}",
                "arn:aws:s3:::bucket-name/home/${aws:username}/*"
            ]
        }
    ]
}
```

Amazon S3: restringir la administración a un bucket de S3 específico

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita restringir la administración de un bucket de Amazon S3 a ese bucket específico. Esta política concede permiso para llevar a cabo todas las acciones de Amazon S3, pero deniega el acceso a cada Servicio de AWS, excepto Amazon S3. Consulte el siguiente ejemplo, . De acuerdo con esta política, solo puede acceder a las acciones de Amazon S3 que pueda hacer en un bucket de S3 o un recurso de objeto de S3. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *texto en cursiva* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

Si esta política se utiliza en combinación con otras políticas (como las políticas administradas por AmazonS3FullAccess o AmazonEC2FullAccess de AWS) que permiten acciones denegadas por esta política, el acceso se denegará. Esto se debe a que una instrucción de denegación explícita prevalece sobre las instrucciones de permiso. Para obtener más información, consulte the section called “Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta”.

> **Warning**
>
> `NotAction` y `NotResource` son elementos avanzados de política que deben utilizarse con precaución. Esta política deniega el acceso a cada servicio de AWS, salvo en Amazon S3. Si asocia esta política a un usuario, cualquier otra política que conceda permisos a otros servicios se pasará por alto y el acceso se denegará.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        },
        {
            "Effect": "Deny",
            "NotAction": "s3:*",
            "NotResource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

Amazon S3: permite el acceso de lectura y escritura a objetos en un bucket de S3

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a `Read` y a `Write` acceder a objetos de un bucket de S3 específico. Esta política concede los permisos necesarios para llevar a cabo esta acción mediante programación desde la API o la AWS CLI de AWS. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La acción `s3:*Object` utiliza un comodín como parte del nombre de la acción. La instrucción `AllObjectActions` permite `GetObject`, `DeleteObject`, `PutObject` y cualquier otra acción de Amazon S3 que termine con la palabra "Object".

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
```

```
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

> **Note**
>
> Para permitir el acceso a `Read` y `Write` a un objeto en un bucket de Amazon S3 y también incluir permisos adicionales para el acceso a la consola, consulte Amazon S3: permite el acceso de lectura y escritura a objetos en un bucket de S3 mediante programación y en la consola.

Amazon S3: permite el acceso de lectura y escritura a objetos en un bucket de S3 mediante programación y en la consola

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita a `Read` y a `Write` acceder a objetos de un bucket de S3 específico. Esta política define los permisos para el acceso programático y a la consola. Para utilizar esta política, sustituya el *`texto en cursiva`* de la política de ejemplo por su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

La acción `s3:*Object` utiliza un comodín como parte del nombre de la acción. La instrucción `AllObjectActions` permite `GetObject`, `DeleteObject`, `PutObject` y cualquier otra acción de Amazon S3 que termine con la palabra "Object".

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3ConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicyStatus",
                "s3:GetBucketPublicAccessBlock",
                "s3:ListAccessPoints",
```

```
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": ["arn:aws:s3:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
    ]
}
```

Administración de políticas de IAM

IAM le ofrece las herramientas para crear y administrar todos los tipos de políticas de IAM (políticas administradas y políticas insertadas). Para agregar permisos a una identidad de IAM (usuario, grupo o rol de IAM), cree una política, valide la política, y, a continuación, adjunte la política a la identidad. Puede asociar varias políticas a una identidad y cada política puede incluir varios permisos.

Consulte estos recursos para obtener más información:

- Para obtener más información sobre los distintos tipos de políticas de IAM, consulte Políticas y permisos en IAM.
- Para obtener información general sobre el uso de políticas en IAM, consulte Recursos de AWS para administración de acceso.
- Para obtener información sobre cómo los permisos se evalúan cuando varias políticas están en vigor para una determinada identidad de IAM, consulte Lógica de evaluación de políticas.
- El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Temas

- Crear políticas de IAM

- Validación de políticas de IAM
- Generar políticas basadas en la actividad de acceso
- Probar las políticas de IAM con el simulador de políticas de IAM.
- Adición y eliminación de permisos de identidad de IAM
- Control de versiones de políticas de IAM
- Edición de políticas de IAM
- Eliminación de políticas de IAM
- Perfeccionar los permisos con la información sobre los últimos accesos en AWS

Crear políticas de IAM

Una política es una entidad que, cuando se asocia a una identidad o recurso, define sus permisos. Puede utilizar la AWS Management Console, la AWS CLI o la API de AWS para crear políticas administradas por el cliente en IAM. Las políticas administradas por el cliente son políticas independientes que usted administra en su propia cuenta de Cuenta de AWS. De este modo, puede asociar las políticas a identidades (usuarios, grupos y roles) de su cuenta de Cuenta de AWS.

Una política que está asociada a una identidad en IAM recibe el nombre de política basada en identidad. Las políticas basadas en identidad pueden incluir políticas administradas por AWS, políticas administradas por el cliente y políticas insertadas. Las políticas administradas por AWS las crea y administra AWS. Puede utilizarlas, pero no puede administrarlas. Un política insertada es aquella que se crea e inserta directamente en un grupo, usuario o rol de IAM. Las políticas insertadas no pueden reutilizarse en otras identidades ni administrarse fuera de la identidad donde existen. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

Utilice políticas administradas por el cliente en lugar de las políticas insertadas. También es mejor utilizar políticas administradas por el cliente en lugar de políticas administradas por AWS. AWS proporciona normalmente permisos administrativos amplios o de solo lectura. Para mayor seguridad, conceda privilegios mínimos, es decir, conceda solo los permisos necesarios para realizar tareas específicas.

Al crear o editar políticas de IAM, AWS puede realizar automáticamente la validación de políticas para ayudarle a crear una política eficaz con el menor privilegio en mente. En AWS Management Console, IAM identifica errores de sintaxis JSON, mientras que IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más las políticas. Para obtener más información acerca la validación de políticas, consulte Validación

de políticas de IAM. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer.

Puede utilizar la AWS Management Console, la AWS CLI o la API de AWS para crear políticas administradas por el cliente en IAM. Para más información sobre el uso de plantillas AWS CloudFormation para agregar o actualizar políticas, consulte la referencia del tipo de recurso de AWS Identity and Access Management en la Guía del usuario de AWS CloudFormation.

Temas

- Creación de políticas de IAM (Consola)
- Crear políticas de IAM (AWS CLI)
- Crear políticas de IAM (API de AWS)

Creación de políticas de IAM (Consola)

Una política es una entidad que, cuando se asocia a una identidad o recurso, define sus permisos. Puede utilizar la AWS Management Console para crear políticas administradas por el cliente en IAM. Las políticas administradas por el cliente son políticas independientes que usted administra en su propia cuenta de Cuenta de AWS. De este modo, puede asociar las políticas a identidades (usuarios, grupos y roles) de su cuenta de Cuenta de AWS.

Temas

- Crear políticas de IAM
- Creación de políticas mediante el editor JSON
- Creación de políticas con el editor visual
- Importación de políticas administradas existentes

Crear políticas de IAM

Puede crear una política administrada por el cliente en la AWS Management Console mediante uno de los métodos siguientes:

- JSON — Pegue y personalice una política basada en identidad de ejemplo publicada.
- Editor visual - Cree una nueva política desde cero en el editor visual. Si utiliza el editor visual, no tiene que conocer la sintaxis JSON.

- Importar - Importe y personalice una política administrada desde su cuenta. Puede importar una política administrada por AWS o una política administrada por el cliente que haya creado anteriormente.

El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Creación de políticas mediante el editor JSON

Puede escribir o pegar políticas en JSON seleccionando la opción JSON. Este método es útil para copiar una política de ejemplo para utilizarla en su cuenta. O bien, puede escribir su propio documento de política de JSON en el editor de JSON. También puede utilizar la opción JSON para alternar entre el editor visual y JSON con el fin de comparar las vistas.

Cuando crea o edita una política en el editor JSON, IAM realiza la validación de políticas para ayudarle a crear una política eficaz. IAM identifica errores de sintaxis JSON, mientras que IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones procesables para ayudarle a perfeccionar aún más la política.

Un documento de política de JSON consta de una o más instrucciones. Cada instrucción debe contener todas las acciones que comparten el mismo efecto (`Allow` o `Deny`) y admitir los mismos recursos y condiciones. Si una acción requiere que especifique todos los recursos (`"*"`) y otra acción admite el nombre de recurso de Amazon (ARN) de un recurso específico, deben estar en dos instrucciones JSON independientes. Para obtener más información sobre los formatos ARN, consulte Nombre de recurso de Amazon (ARN) en la Guía de Referencia general de AWS. Para obtener información general sobre las políticas de IAM, consulte Políticas y permisos en IAM. Para obtener información sobre el lenguaje de políticas de IAM, consulte Referencia de políticas JSON de IAM.

Para utilizar el editor de política de JSON para crear una política

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Policies (Políticas).
3. Elija Create Policy (Crear política).
4. En la sección Editor de políticas, seleccione la opción JSON.
5. Escriba o pegue un documento de política de JSON. Para obtener más información sobre el lenguaje de políticas de IAM, consulte Referencia de políticas JSON de IAM.

6. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.

   > **Note**
   >
   > Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

7. (Opcional) Al crear o editar una política en la AWS Management Console, se puede generar una plantilla de política JSON o YAML que se puede utilizar en plantillas de AWS CloudFormation.

   Para ello, en el Editor de políticas, seleccione Acciones y, a continuación, Generar plantilla de CloudFormation. Para obtener más información sobre AWS CloudFormation, consulte la Referencia de tipos de recursos de AWS Identity and Access Management en la Guía del usuario de AWS CloudFormation.

8. Cuando haya terminado de agregar permisos a la política, seleccione Siguiente.

9. En la página Revisar y crear, escriba el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política.

10. (Opcional) Agregar metadatos a la política al adjuntar las etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.

11. Elija Create Policy (Crear política) para guardar la nueva política.

Después de crear una política, puede asociarla a sus grupos, usuarios o roles. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

Creación de políticas con el editor visual

El editor visual de la consola de IAM le guía a través de la creación de una política sin tener que escribir sintaxis JSON. Para ver un ejemplo de cómo utilizar el editor para crear una política, consulte the section called “Control del acceso a identidades”.

Para utilizar el editor visual para crear una política

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Policies (Políticas).
3. Elija Create Policy (Crear política).
4. En la sección Editor de políticas, busque la sección Seleccionar un servicio y, a continuación, seleccione un servicio de AWS. Puede utilizar el cuadro de búsqueda en la parte superior para limitar los resultados en la lista de servicios. Puede elegir solo un servicio dentro de un bloque de permisos de editor visual. Para conceder acceso a más de un servicio, agregue varios bloques de permisos seleccionando Agregar más permisos.
5. En Acciones permitidas, seleccione las acciones que desee agregar a la política. Puede elegir acciones de una de las siguientes formas:

   - Active la casilla de verificación para todas las acciones.
   - Elija Agregar acciones para escribir el nombre de una acción específica. Puede utilizar comodines (*) para especificar varias acciones.
   - Seleccione uno de los grupos de niveles de acceso para elegir todas las acciones del nivel de acceso (por ejemplo, Leer, Escribir, o Lista).
   - Amplíe cada uno de los grupos Access level (Nivel de acceso) para elegir acciones individuales.

   De forma predeterminada, la política que está creando permite las acciones que usted elija. Para denegar las acciones elegidas, seleccione Switch to deny permissions (Cambiar a denegar permisos). Dado que IAM deniega de forma predeterminada, por motivos de seguridad recomendamos que permita solo aquellas acciones y recursos a los que un usuario necesita acceso. Debe crear una instrucción JSON para denegar permisos únicamente si desea invalidar un permiso que otra instrucción o política permite. Le recomendamos que limite al mínimo el número de operaciones de denegación de permisos, ya que pueden aumentar la dificultad de solucionar problemas con los permisos.
6. Para Recursos, si el servicio y las acciones que seleccionó en los pasos anteriores no admiten la elección de recursos específicos, todos los recursos están permitidos y no puede editar esta sección.

Si eligió una o más acciones que admiten permisos en el nivel de recursos, el editor visual enumera dichos recursos. A continuación, puede elegir Resources (Recursos) para especificar los recursos para su política.

Puede especificar recursos de las siguientes maneras:

- Seleccione Agregar ARN para especificar recursos por su nombre de recurso de Amazon (ARN). Puede utilizar el editor ARN visual o enumerar ARN manualmente. Para obtener más información acerca de la sintaxis de ARN, consulte Amazon Resource Name (ARN) en laReferencia general de AWSGuía. Para obtener información sobre el uso de ARN en el elemento `Resource` de una política, consulte Elementos de política JSON de IAM: Resource.
- Seleccione Cualquiera de esta cuenta junto a un recurso para conceder permisos a cualquier recurso de ese tipo.
- Seleccione Todos para seleccionar todos los recursos para el servicio.

7. (Opcional) Seleccione Solicitar condiciones: opcional para agregar condiciones a la política que está creando. Las condiciones limitan el efecto de una instrucción de política de JSON. Por ejemplo, puede especificar que a un usuario se le permite realizar las acciones en los recursos solo cuando la solicitud de dicho usuario se produce dentro de un intervalo de tiempo determinado. También puede utilizar condiciones de uso común para limitar si un usuario debe autenticarse con un dispositivo de autenticación multifactor (MFA). O bien puede exigir que la solicitud se origine dentro de un determinado rango de direcciones IP. Para obtener listas de todas las claves de contexto que puede utilizar en una condición de política, consulte Acciones, recursos y claves de condición para AWS en la Referencia de autorizaciones de servicio.

   Puede elegir las condiciones de una de las siguientes formas:

   - Utilice las casillas de verificación para seleccionar condiciones de uso común.
   - Seleccione Agregar otra condición para especificar otras condiciones. Elija los valores Condition Key (Clave de condición), Qualifier (Calificador) y Operator (Operador) de la condición y, a continuación, escriba un Value (Valor). Para agregar más de un valor, seleccione Agregar. Puede considerar que los valores están conectados mediante un operador lógico "OR". Cuando haya terminado, seleccione Agregar condición.

   Para agregar más de una condición, vuelva a seleccionar Agregar condición. Repita este procedimiento según sea necesario. Cada condición se aplica únicamente a este bloque de permisos del editor visual. Todas las condiciones deben ser "true" para que el bloque de

permisos se considere una coincidencia. En otras palabras, considere que las condiciones están conectadas mediante un operador lógico "AND".

Para obtener más información sobre el elemento Condition (Condición), consulte Elementos de política JSON de IAM: Condition en la Referencia de políticas JSON de IAM.

8. Para agregar más bloques de permisos, seleccione Agregar más permisos. Para cada bloque, repita los pasos 2 a 5.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

9. (Opcional) Al crear o editar una política en la AWS Management Console, se puede generar una plantilla de política JSON o YAML que se puede utilizar en plantillas de AWS CloudFormation.

   Para ello, en el Editor de políticas, seleccione Acciones y, a continuación, Generar plantilla de CloudFormation. Para obtener más información sobre AWS CloudFormation, consulte la Referencia de tipos de recursos de AWS Identity and Access Management en la Guía del usuario de AWS CloudFormation.

10. Cuando haya terminado de agregar permisos a la política, seleccione Siguiente.
11. En la página Revisar y crear, escriba el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para asegurarse de que ha concedido los permisos deseados.
12. (Opcional) Agregar metadatos a la política al adjuntar las etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM.
13. Elija Create Policy (Crear política) para guardar la nueva política.

Después de crear una política, puede asociarla a sus grupos, usuarios o roles. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

Importación de políticas administradas existentes

Una forma sencilla de crear una nueva política consiste en importar una política administrada existente en la cuenta que tenga al menos alguno de los permisos que necesita. A continuación, puede personalizarla para adaptarse a los nuevos requisitos.

No puede importar una política insertada. Para obtener más información acerca de la diferencia entre las políticas gestionadas e insertadas, consulte Políticas administradas y políticas insertadas.

Para importar una política administrada existente en el editor visual

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Policies (Políticas).
3. Elija Create Policy (Crear política).
4. En el Editor de políticas, seleccione Visual; después, en la parte derecha de la página, seleccione Acciones, y luego Importar política.
5. En la ventana Importar política, seleccione las políticas administradas que mejor coincidan con la política que desea incluir en su nueva política. Puede utilizar el cuadro de búsqueda de la parte superior para limitar los resultados en la lista de políticas.
6. Seleccione Importar política.

   Las políticas importadas se añaden en nuevos bloques de permisos en la parte inferior de su política.
7. Utilice el Visual editor (Editor visual) o elija JSON para personalizar su política. A continuación, elija Next.

   > **Note**
   >
   > Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.
8. En la página Revisar y crear, escriba el Nombre de la política y la Descripción (opcional) para la política que está creando. No puede editar estos ajustes más tarde. Revise los Permisos definidos en esta política y, a continuación, seleccione Crear política para guardar su trabajo.

Para importar una política administrada existente en el editor JSON

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Policies (Políticas).
3. Elija Create Policy (Crear política).
4. En la sección Editor de políticas, seleccione la opción JSON; después, en la parte derecha de la página, seleccione Acciones, y luego Importar política.
5. En la ventana Importar política, seleccione las políticas administradas que mejor coincidan con la política que desea incluir en su nueva política. Puede utilizar el cuadro de búsqueda de la parte superior para limitar los resultados en la lista de políticas.
6. Seleccione Importar política.

   Las instrucciones de las políticas importadas se añaden a la parte inferior de su política de JSON.
7. Personalice su política en JSON. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next. O, personalice su política en JSON o elija Editor visual. A continuación, elija Next.

   > **Note**
   >
   > Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.
8. En la página Revisar y crear, escriba el Nombre de la política y la Descripción (opcional) para la política que está creando. No puede editarlos más tarde. Revise los Permisos definidos en esta política correspondientes a la política y, a continuación, seleccione Crear política para guardar su trabajo.

Después de crear una política, puede asociarla a sus grupos, usuarios o roles. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.

Crear políticas de IAM (AWS CLI)

Una política es una entidad que, cuando se asocia a una identidad o recurso, define sus permisos. Puede utilizar la AWS CLI para crear políticas administradas por el cliente en IAM. Las políticas administradas por el cliente son políticas independientes que usted administra en su propia cuenta de Cuenta de AWS. Como práctica recomendada, le sugerimos utilizar IAM Access Analyzer para validar sus políticas de IAM y así garantizar la seguridad y funcionalidad de los permisos. Al validar sus políticas, puede abordar cualquier error o recomendación antes de asociar las políticas a las identidades (usuarios, grupos y roles) de su cuenta de Cuenta de AWS.

El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Crear políticas de IAM (AWS CLI)

Puede crear una política de IAM administrada por el cliente o una política insertada mediante la AWS Command Line Interface (AWS CLI).

Para crear una política administrada por el cliente (AWS CLI)

Para ello, utilice el siguiente comando:

- create-policy

Para crear una política insertada para una identidad de IAM (grupo, usuario o rol) (AWS CLI)

Utilice uno de los siguientes comandos:

- put-group-policy
- put-role-policy
- put-user-policy

> **Note**
>
> No se puede utilizar IAM para incrustar una política insertada para un rol vinculado al servicio.

Para validar una política administrada por el cliente (AWS CLI)

Utilice el siguiente comando de IAM Access Analyzer:

- validate-policy

Crear políticas de IAM (API de AWS)

Una política es una entidad que, cuando se asocia a una identidad o recurso, define sus permisos. Puede utilizar la API de AWS para crear políticas administradas por el cliente en IAM. Las políticas administradas por el cliente son políticas independientes que usted administra en su propia cuenta de Cuenta de AWS. Como práctica recomendada, le sugerimos utilizar IAM Access Analyzer para validar sus políticas de IAM y así garantizar la seguridad y funcionalidad de los permisos. Al validar sus políticas, puede abordar cualquier error o recomendación antes de asociar las políticas a las identidades (usuarios, grupos y roles) de su cuenta de Cuenta de AWS.

El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Crear políticas de IAM (API de AWS)

Puede crear una política administrada por el cliente o una política insertada de IAM mediante la API de AWS.

Para crear una política administrada por el cliente (API de AWS)

Llame a la operación siguiente:

- CreatePolicy

Para crear una política insertada para una identidad de IAM (grupo, usuario o rol) (API de AWS)

Llame a una de las siguientes operaciones:

- PutGroupPolicy
- PutRolePolicy
- PutUserPolicy

> **Note**
>
> No se puede utilizar IAM para incrustar una política insertada para un rol vinculado al servicio.

Para validar una política administrada por el cliente (API de AWS)

Llame a la siguiente operación de IAM Access Analyzer:

- ValidatePolicy

Validación de políticas de IAM

Una política es un documento JSON escrito con la política gramatical de IAM. Cuando se asocia una política a una entidad de IAM, como un usuario, un grupo o un rol, concede permisos a esa entidad.

Cuando crea o edita políticas de control de acceso de IAM mediante AWS Management Console, AWS las examina automáticamente para asegurarse de que cumplan con la política gramatical de IAM. Si AWS determina que una política no cumple la gramática, se le pedirá que corrija la política.

IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más la política. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer. Para ver una lista de advertencias, errores y sugerencias que devuelve el analizador de acceso de IAM, consulte Referencia de comprobación de políticas del analizador de acceso de IAM.

Ámbito de validación

AWS comprueba la gramática y la sintaxis de la política JSON. También comprueba que los ARN tengan un formato correcto y que los nombres de acción y las claves de condición sean correctos.

Acceso a la validación de políticas

Las políticas se validan automáticamente cuando se crea una política JSON o se edita una política existente en el AWS Management Console. Si la sintaxis de la política no es válida, recibe una notificación y deberá solucionar el problema para poder continuar. Los resultados de la validación de políticas del Analizador de acceso de IAM se devuelven automáticamente en el AWS Management

Console si tiene permisos para `access-analyzer:ValidatePolicy`. También puede validar políticas mediante la API de AWS o AWS CLI.

Políticas existentes

Es posible que tenga políticas existentes que no sean válidas porque se crearon o guardaron por última vez antes de las últimas actualizaciones del motor de políticas. Como práctica recomendada, le sugerimos utilizar IAM Access Analyzer para validar sus políticas de IAM y así garantizar la seguridad y funcionalidad de los permisos. Le recomendamos que abra las políticas existentes y revise los resultados de validación de políticas que se generan. No puede editar y guardar las políticas existentes sin corregir ningún error de sintaxis de política.

Generar políticas basadas en la actividad de acceso

Como administrador o desarrollador, puede conceder permisos a entidades (usuarios o roles) de IAM más allá de lo que requieren. IAM proporciona varias opciones para ayudarle a refinar los permisos que concede. Una opción es generar una política de IAM basada en la actividad de acceso de una entidad. El analizador de acceso de IAM revisa los registros de AWS CloudTrail y genera una plantilla de política que contiene los permisos que la entidad ha utilizado en su intervalo de fechas especificado. Puede utilizar la plantilla para crear una política con permisos detallados que otorguen solo los permisos necesarios para admitir su caso de uso específico.

Por ejemplo, imagine que usted es un desarrollador y que su equipo de ingeniería ha estado trabajando en un proyecto para crear una nueva aplicación. Para fomentar la experimentación y permitir que su equipo se mueva rápidamente, ha configurado un rol con amplios permisos mientras la aplicación está en desarrollo. Ahora la aplicación está lista para la producción. Antes de que la aplicación pueda iniciarse en la cuenta de producción, querrá identificar solo los permisos que necesita el rol para que la aplicación funcione. Esto le ayudará a cumplir mejor las prácticas recomendadas para conceder privilegios mínimos. Puede generar una política basada en la actividad de acceso del rol que ha estado utilizando para la aplicación en la cuenta de desarrollo. Puede refinar aún más la política generada y, a continuación, asociarla a una entidad en su cuenta de producción.

Para obtener más información acerca de la generación de políticas de IAM Access Analyzer, consulte Generación de políticas de IAM Access Analyzer.

Probar las políticas de IAM con el simulador de políticas de IAM.

Para obtener más información sobre cómo y por qué utilizar las políticas de IAM, consulte Políticas y permisos en IAM.

Puede obtener acceso a la consola del simulador de políticas de IAM en: https://policysim.aws.amazon.com/

> **Important**
>
> Los resultados del simulador de política pueden diferir de los de su entorno de AWS real. Le recomendamos que compare sus políticas con su entorno de AWS real después de llevar a cabo las pruebas con el simulador de política para confirmar que obtiene los resultados deseados. Para obtener más información, consulte Cómo funciona el simulador de políticas de IAM.

Introducción al simulador de política de IAM

Con el simulador de política de IAM, puede probar y solucionar problemas de políticas basadas en identidad y límites de permisos de IAM. A continuación se enumeran algunas acciones habituales que puede hacer con el simulador de políticas:

- Pruebe las políticas basadas en la identidad que se adjuntan a los usuarios de IAM, grupos de usuarios o roles en su Cuenta de AWS. Si hay más de una política asociada al usuario, grupo de usuarios o rol, puede probarlas todas o seleccionarlas individualmente para probarlas. Puede probar las acciones permitidas o denegadas por las políticas seleccionadas para determinados recursos.
- Probar y solucionar problemas del efecto de los límites de permisos en las entidades de IAM. Solo puede simular un límite de permisos a la vez.
- Pruebe los efectos de las políticas basadas en recursos en los usuarios de IAM que están asociados a recursos de AWS, como buckets de Amazon S3, colas de Amazon SQS, temas de Amazon SNS o almacenes de Amazon S3 Glacier. Para utilizar una política basada en recursos en el simulador de política para usuarios de IAM, debe incluir el recurso en la simulación. También debe activar la casilla para incluir la política de ese recurso en la simulación.

  > **Note**
  >
  > Los roles de IAM no admiten la simulación de políticas basadas en recursos.

- Si la cuenta de Cuenta de AWS forma parte de una organización en AWS Organizations, puede probar el impacto de las políticas de control de servicios (SCP) en sus políticas basadas en identidades.

> **Note**
>
> El simulador de política no evalúa las SCP que tienen cualquier condición.

- Pruebe nuevas políticas basadas en identidades que aún no estén asociadas a un usuario, grupo de usuarios o rol. Para ello, escríbalas o cópielas en el simulador de política. Estas solo se utilizan en la simulación y no se guardan. No puede escribir ni copiar una política basada en recursos en el simulador de política.
- Pruebe las políticas basadas en identidades con servicios, acciones y recursos seleccionados. Por ejemplo, puede realizar una prueba para garantizar que la política permita que una entidad realice las acciones `ListAllMyBuckets`, `CreateBucket` y `DeleteBucket` en el servicio de Amazon S3 de un determinado bucket.
- Simule escenarios del mundo real proporcionando claves de contexto, como una dirección IP o fecha, que se incluyan en los elementos `Condition` de las políticas que se estén probando.

> **Note**
>
> El simulador de política no simula las etiquetas proporcionadas como entrada si la política basada en identidades de la simulación no tiene un elemento `Condition` que compruebe explícitamente las etiquetas.

- Identifique qué declaración específica de la política basada en identidades tiene como resultado permitir o denegar el acceso a un recurso o acción concretos.

Temas

Cómo funciona el simulador de políticas de IAM

El simulador de política evalúa las declaraciones de la política basada en la identidad y las entradas que el usuario proporciona durante la simulación. Los resultados del simulador de política pueden diferir de los de su entorno de AWS real. Le recomendamos que compare sus políticas con su

entorno de AWS real después de llevar a cabo las pruebas con el simulador de política para confirmar que obtiene los resultados deseados.

El simulador de política difiere del entorno de AWS real en los siguientes aspectos:

- El simulador de política no lleva a cabo ninguna solicitud real al servicio de AWS, de modo que puede probar de forma segura solicitudes que podrían hacer cambios no deseados en el entorno real de AWS. El simulador de política no considera los valores clave del contexto real en la producción.
- Dado que el simulador de política no simula la ejecución de las acciones seleccionadas, no se puede informar de ninguna respuesta a la solicitud simulada. El único resultado que se devuelve es si la acción solicitada se permitiría o se denegaría.
- Si edita una política en el simulador de política, estos cambios solo afectan al simulador de política. La correspondiente política de la Cuenta de AWS permanece sin cambios.
- No puede probar las políticas de control de servicio (SCP) con cualquier condición.
- El simulador de política no admite la simulación de roles y usuarios de IAM para el acceso entre cuentas.

> **Note**
>
> El simulador de política de IAM no determina qué servicios admiten claves de condiciones globales para la autorización. Por ejemplo, el simulador de política no identifica si un servicio no es compatible con `aws:TagKeys`.

Permisos necesarios para utilizar el simulador de políticas de IAM

Puede utilizar la consola del simulador de políticas o la API del simulador de políticas para probar políticas. De forma predeterminada, los usuarios de la consola pueden probar las políticas que aún no están asociadas a un usuario, grupo de usuarios o rol. Para ello, deben escribir o copiar dichas políticas en el simulador de política. Estas políticas se utilizan únicamente en la simulación y no revelan información confidencial. Los usuarios de API deben tener permisos para probar las políticas no asociadas. Puede permitir que los usuarios de la consola o de la API prueben políticas asociadas a usuarios de IAM, grupos de usuarios o roles en su cuenta de Cuenta de AWS. Para ello, debe proporcionar permiso para recuperar esas políticas. Para poder probar políticas basadas en recursos, los usuarios deben tener permiso para recuperar la política del recurso.

Para obtener ejemplos de las políticas de la consola y API que permiten a un usuario simular políticas, consulte the section called “Ejemplos de políticas: AWS Identity and Access Management (IAM)”.

Permisos necesarios para utilizar la consola del simulador de políticas

Puede permitir que los usuarios prueben políticas asociadas a usuarios de IAM, grupos de usuarios o roles en su cuenta de Cuenta de AWS. Para ello, debe proporcionar a los usuarios permisos para recuperar esas políticas. Para poder probar políticas basadas en recursos, los usuarios deben tener permiso para recuperar la política del recurso.

Para ver un ejemplo de política que permite utilizar la consola del simulador de políticas para las políticas asociadas a un usuario, grupo de usuarios o rol, consulte IAM: permite el acceso a la consola del simulador de políticas.

Para ver un ejemplo de política que permite utilizar la consola del simulador de políticas únicamente para aquellos usuarios con una ruta de acceso determinada, consulte IAM: permite el acceso a la consola de simulador de políticas en función de la ruta de acceso del usuario.

Para crear una política que permita utilizar la consola del simulador de políticas para únicamente un tipo de entidad, utilice los siguientes procedimientos.

Para permitir a los usuarios de la consola simular políticas para los usuarios

Incluya las siguientes acciones en la política:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetUser`
- `iam:GetUserPolicy`
- `iam:ListAttachedUserPolicies`
- `iam:ListGroupsForUser`
- `iam:ListGroupPolicies`
- `iam:ListUserPolicies`
- `iam:ListUsers`

Para permitir a los usuarios de la consola simular políticas para los grupos de usuarios

Incluya las siguientes acciones en la política:

- `iam:GetGroup`
- `iam:GetGroupPolicy`
- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:ListAttachedGroupPolicies`
- `iam:ListGroupPolicies`
- `iam:ListGroups`

Para permitir a los usuarios de la consola simular políticas para los roles

Incluya las siguientes acciones en la política:

- `iam:GetPolicy`
- `iam:GetPolicyVersion`
- `iam:GetRole`
- `iam:GetRolePolicy`
- `iam:ListAttachedRolePolicies`
- `iam:ListRolePolicies`
- `iam:ListRoles`

Para probar políticas basadas en recursos, los usuarios deben tener permiso para recuperar la política del recurso.

Para permitir a los usuarios de la consola probar políticas basadas en recursos en un bucket de Amazon S3

Incluya la siguiente acción en la política:

- `s3:GetBucketPolicy`

Por ejemplo, la siguiente política utiliza esta acción para permitir a los usuarios de la consola simular una política basada en recursos en un determinado bucket de Amazon S3.

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetBucketPolicy",
            "Resource":"arn:aws:s3:::bucket-name/*"
          }
        ]
      }
```

Permisos necesarios para utilizar la API el simulador de políticas

Las operaciones de la API del simulador de políticas GetContextKeyForCustomPolicy y SimulateCustomPolicy le permiten probar políticas que aún no asociadas a un usuario, grupo de usuarios o rol. Para probar dichas políticas, debe pasarlas como cadenas a la API. Estas políticas se utilizan únicamente en la simulación y no revelan información confidencial. También puede utilizar la API para probar políticas asociadas a usuarios de IAM, grupos o roles en su cuenta de Cuenta de AWS. Para ello, debe proporcionar a los usuarios permisos para llamar a GetContextKeyForPrincipalPolicy y SimulatePrincipalPolicy.

Para ver una política de ejemplo que permita utilizar la API del simulador de políticas para políticas asociadas y no asociadas en la cuenta de Cuenta de AWS actual, consulte IAM: acceso a la API del simulador de políticas.

Para crear una política que permita utilizar la API del simulador de políticas para únicamente un tipo de política, utilice los siguientes procedimientos.

Para permitir a los usuarios de la API simular políticas transferidas directamente a la API como cadenas

Incluya las siguientes acciones en la política:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

Para permitir a los usuarios de la API simular políticas asociadas a los usuarios, grupos de usuarios, roles o recursos de IAM

Incluya las siguientes acciones en la política:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

Por ejemplo, para conceder a un usuario llamado Bob permiso para simular una política asignada a un usuario llamado Alice, debe conceder acceso a Bob al siguiente recurso: `arn:aws:iam::777788889999:user/alice`.

Para ver un ejemplo de política que permite utilizar la API del simulador de políticas únicamente para aquellos usuarios con una ruta de acceso determinada, consulte IAM: obtiene acceso a la API del simulador de políticas en función de la ruta de acceso del usuario.

Uso del simulador de políticas de IAM (Consola)

De forma predeterminada, los usuarios pueden probar las políticas que aún no están asociadas a un usuario, grupo de usuarios o rol escribiéndolas o copiándolas en la consola del simulador de políticas. Estas políticas se utilizan únicamente en la simulación y no revelan información confidencial.

Para probar una política no asociada a un usuario, grupo de usuarios o rol (consola)

1. Abra la consola del simulador de políticas de IAM en: https://policysim.aws.amazon.com/.
2. En el menú Mode: (Modo:) situado en la parte superior de la página, elija New Policy (Nueva política).
3. En la opción Policy Sandbox (Entorno de pruebas de política), elija Create New Policy (Crear nueva política).
4. Escriba o copie una política en el simulador de política y utilícelo como se describe en los pasos siguientes.

Una vez que tenga permiso para utilizar la consola del simulador de política de IAM, podrá utilizar el simulador de política para probar una política de usuarios, grupos de usuarios, roles o recursos de IAM.

Para probar una política asociada a un usuario, grupo de usuarios o rol (consola)

1. Abra la consola del simulador de políticas de IAM en: https://policysim.aws.amazon.com/.

> **Note**
>
> Para iniciar sesión en el simulador de políticas como usuario de IAM, utilice la URL de inicio de sesión único en la AWS Management Console. A continuación, diríjase a https://policysim.aws.amazon.com/. Para obtener más información sobre cómo iniciar sesión como usuario de IAM, consulte Cómo inician sesión los usuarios de IAM en AWS.

El simulador de política se abrirá en el modo Existing Policies (Políticas existentes) y enumerará los usuarios de IAM de la cuenta en Users, Groups, and Roles (Usuarios, grupos y roles).

2. Elija la opción que sea apropiada para su tarea:

Para probarlo:	Haga lo siguiente:
Una política asociada a un usuario	Elija Users (Usuarios) en la lista Users, Groups, and Roles (Usuarios, grupos y roles). A continuación, elija el usuario.
Una política asociada a un grupo de usuarios	Elija Groups (Grupos) en la lista Users, Groups, and Roles (Usuarios, grupos y roles). A continuación, elija el grupo de usuarios.
Una política asociada a un rol	Elija Roles en la lista Users, Groups, and Roles (Usuarios, grupos y roles). A continuación, elija el rol.
Una política asociada a un recurso	Consulte Step 9.
Una política personalizada para un usuario, grupo de usuarios o rol	Elija Create new policy (Crear nueva política). En el nuevo panel Policies (Políticas) escriba o pegue una directiva y, a continuación, elija Apply (Aplicar).

> **Sugerencia**
>
> Para probar una política asociada a un grupo de usuarios, puede lanzar el simulador de políticas de IAM directamente desde la consola de IAM: en el panel de navegación, elija Grupos de usuarios. Elija el nombre del grupo en el que desea probar una política y, a continuación, elija la pestaña Permissions (Permisos). Seleccionar Simular.
> Para probar una política administrada por el cliente que está asociada a un usuario: en el panel de navegación, elija Users (Usuarios). Elija el nombre del usuario en el que desea probar una política. A continuación, elija la pestaña Permissions (Permisos) y amplíe la política que desea probar. En el extremo derecho, elija Simulate policy (Simular política). Se abrirá el Simulador de políticas de IAM en una nueva ventana y mostrará la política seleccionada en el panel Políticas.

3. (Opcional) Si su cuenta es miembro de una organización en AWS Organizations, a continuación, seleccione la casilla situada junto a SCP de AWS Organizations para incluir la SCP en su evaluación simulada. Las SCP son políticas JSON que especifican los permisos máximos para una organización o unidad organizativa (OU). Una SCP limita los permisos para las entidades de las cuentas de miembros. Si una SCP bloquea un servicio o acción, ninguna entidad de dicha cuenta puede obtener acceso a dicho servicio ni realizar la acción en cuestión. Esto es válido incluso si un administrador concede explícitamente permisos a dicho servicio o acción mediante una política de recursos o IAM.

   Si la cuenta no forma parte de una organización, la casilla de verificación no aparece.

4. (Opcional) Puede probar una política establecida como límite de permisos para una entidad de IAM (usuario o rol), pero no para grupos de usuarios. Si actualmente se establece una política de límite de permisos para la entidad, aparecerá en el panel Policies (Políticas). Solo puede establecer un límite de permisos para una identidad. Para probar otro límite de permisos, puede crear un límite de permisos personalizado. Para ello, elija Create new policy (Crear nueva política). Se abre un nuevo panel Policies (Políticas). En el menú, elija Custom IAM Permissions Boundary Policy (Política de límite de permisos de IAM personalizada). Escriba un nombre para la nueva política y escriba o copie una política en el espacio que hay debajo. Seleccione Apply (Aplicar) para guardar la política. A continuación, elija Back (Atrás) para volver al panel Policies (Políticas) original. Después, seleccione la casilla situada junto al límite de permisos que desea utilizar para la simulación.

5. (Opcional) Solo puede probar un subconjunto de políticas asociadas a un usuario, grupo de usuarios o rol. Para ello, en el panel Policies (Políticas) desactive la casilla situada junto a cada política que desee excluir.
6. En Policy Simulator (Simulador de políticas), elija Select service (Seleccionar servicio) y, a continuación, elija el servicio que desea probar. A continuación, elija Select actions (Seleccionar acciones) y seleccione una o varias acciones para probar. Aunque los menús muestran las selecciones disponibles únicamente para un servicio a la vez, todos los servicios y acciones que haya seleccionado aparecen en Action Settings and Results (Configuración y resultados de la acción).
7. (Opcionalmente) Si alguna de las políticas que seleccionó en Step 2 y Step 5 incluyen condiciones con claves de condición global de AWS, proporcione valores para esas claves. Puede hacerlo, ampliando la sección Global Settings (Configuración global) y escribir los valores para los nombres de las claves mostrados.

   > **Warning**
   >
   > Si deja en blanco el valor para una clave de condición, dicha clave se pasa por alto durante la simulación. En algunos casos, esto produce un error y la simulación no puede ejecutarse. En otros casos, se ejecuta la simulación, pero los resultados podrían no ser fiables. En tales casos, la simulación no coincide con las condiciones reales que incluyen un valor para la variable o clave de condición.

8. De manera opcional, cada acción seleccionada aparecerá en la lista Action Settings and Results (Configuración y resultados de la acción) con la opción Not simulated (No simulado) mostrada en la columna Permission (Permiso) hasta que ejecute la simulación. Antes de ejecutar la simulación, puede configurar cada acción con un recurso. Para configurar acciones individuales para un determinado escenario, elija la flecha para ampliar la fila de la acción. Si la acción admite permisos de nivel de recursos, puede escribir el Nombre de recurso de Amazon (ARN) del recurso específico cuyo acceso desea probar. De forma predeterminada, cada recurso está establecido en un carácter comodín (*). También puede especificar un valor para las claves de contexto de condición. Como se ha mencionado anteriormente, las claves con valores vacíos se pasan por alto, lo que puede provocar errores en la simulación o resultados no fiables.

   a. Elija la flecha junto al nombre de la acción para ampliar cada fila y configurar cualquier información adicional necesaria para simular de forma precisa la acción en su escenario. Si la acción exige permisos de nivel de recursos, puede escribir el Nombre de recurso

de Amazon (ARN) del recurso específico en el que desea simular el acceso. De forma predeterminada, cada recurso está establecido en un carácter comodín (*).

b. Si la acción admite permisos de nivel de recursos, pero no los necesita, puede elegir Add Resource (Añadir recurso) para seleccionar el tipo de recurso que desea agregar a la simulación.

c. Si cualquiera de las políticas seleccionadas incluyen un elemento `Condition` que haga referencia a una clave de contexto para el servicio de esta acción, el nombre de la clave aparecerá en la acción. Puede especificar el valor que desea utilizar durante la simulación de dicha acción para el recurso especificado.

Acciones que exigen grupos distintos de tipos de recursos

Algunas acciones exigen diferentes tipos de recursos en diferentes circunstancias. Cada grupo de tipos de recursos se asocia a un escenario. Si alguno de estos casos se aplica a su simulación, selecciónelo y el simulador de política exigirá los tipos de recursos adecuados para dicho escenario. En la siguiente lista se enumeran cada una de las opciones de escenarios admitidas y los recursos que debe definir para ejecutar la simulación.

Cada uno de los siguientes escenarios de Amazon EC2 exige que especifique los recursos `instance`, `image` y `security-group`. Si en su escenario se incluye un volumen de EBS, debe especificar que `volume` es un recurso. Si en el escenario de Amazon EC2 se incluye una Virtual Private Cloud (VPC), debe proporcionar el recurso `network-interface`. Si se incluye una subred IP, debe especificar el recurso `subnet`. Para obtener más información sobre las opciones de escenarios de Amazon EC2, consulte Plataformas compatibles en la Guía del usuario de Amazon EC2.

- EC2-VPC-InstanceStore

  instancia, imagen, grupo de seguridad, interfaz de red

- EC2-VPC-InstanceStore-Subnet

  instancia, imagen, grupo de seguridad, interfaz de red, subred

- EC2-VPC-EBS

  instancia, imagen, grupo de seguridad, interfaz de red, volumen

- EC2-VPC-EBS-Subnet

instancia, imagen, grupo de seguridad, interfaz de red, subred, volumen

9. (Opcional) Si desea incluir una política basada en recursos en la simulación, debe primero seleccionar las acciones que desea simular en dicho recurso en Step 6. Amplíe las filas de las acciones seleccionadas y escriba el ARN del recurso con una política que desea simular. A continuación, seleccione Include Resource Policy (Incluir política de recurso) junto al cuadro de texto ARN. El simulador de políticas de IAM admite actualmente políticas basadas en recursos de únicamente los siguientes servicios: Amazon S3 (solo políticas basadas en recursos; las ACL no son actualmente compatibles), Amazon SQS, Amazon SNS y almacenes desbloqueados de S3 Glacier (los almacenes bloqueados no son actualmente compatibles).
10. Elija Run Simulation (Ejecutar simulación) en la esquina superior derecha.

    La columna Permission (Permiso) de cada fila de Action Settings and Results (Configuración y resultados de la acción) muestra el resultado de la simulación de cada acción en el recurso especificado.
11. Para ver la instrucción de una política que permite o deniega explícitamente una acción, elija el enlace ***N*** matching statement(s) (Instrucción coincidente N) en la columna Permissions (Permisos) para ampliar la fila. A continuación, elija el enlace Show statement (Mostrar instrucción). El panel Policies (Políticas) muestra la correspondiente política con la instrucción resaltada que afectó al resultado de la simulación.

> **Note**
>
> Si una acción se deniega implícitamente es decir, si la acción se deniega únicamente porque no se permite explícitamente las opciones Enumerar y Mostrar instrucción no se muestran.

Solución de problemas de los mensajes de la consola del simulador de políticas de IAM

En la siguiente tabla se muestran los mensajes informativos y de advertencia que puede encontrar al utilizar el simulador de políticas de IAM. La tabla también indica los pasos que puede seguir para resolverlos.

Mensaje	Pasos para resolver el problema
Esta política se ha editado. Los cambios no se guardarán en su cuenta.	No hay que hacer nada. Este mensaje es informativo. Si edita una política existente en el simulador de políticas de IAM, el cambio no afecta a su cuenta de Cuenta de AWS. El simulador de política le permite hacer cambios en las políticas con fines de prueba únicamente.
No se puede obtener la política de recursos. Motivo: *mensaje de error detallado*	El simulador de política no puede obtener acceso a una política basada en un recurso especificado. Asegúrese de que el ARN del recurso especificado sea correcto y que el usuario que ejecuta la simulación tiene permisos para leer el recurso de la política.
Una o varias políticas exigen valores en la configuración de la simulación. Podría producirse un error en la simulación sin estos valores.	Este mensaje aparece si la política que está probando incluye variables o claves de condición, pero no ha proporcionado ningún valor para estas claves o variables en Simulation Settings (Configuración de simulación). Para que desaparezca este mensaje, elija Simulation Settings (Configuración de simulación) y, a continuación, escriba un valor para cada variable o clave de condición.
Ha cambiado las políticas. Estos resultados ya no son válidos.	Este mensaje aparece si ha cambiado la política seleccionada mientras los resultados aparecen en el panel Results (Resultados). Los resultados que se muestran en el panel Results (Resultados) no se actualizan dinámicamente. Para que desaparezca este mensaje, vuelva a elegir Run Simulation (Ejecutar simulación) para mostrar nuevos resultados de la simulació

Mensaje	Pasos para resolver el problema
	n basados en los cambios realizados en el panel Policies (Políticas).
El recurso que ha introducido para esta simulación no coincide con este servicio.	Este mensaje aparece si ha introducido un Nombre de recurso de Amazon (ARN) en el panel Simulation Settings (Configuración de simulación) que no coincide con el servicio que ha elegido para la simulación actual. Por ejemplo, este mensaje aparece si especifica un ARN de un recurso de Amazon DynamoDB pero ha elegido Amazon Redshift como el servicio que desea simular. Para que desaparezca este mensaje, realice uno de los siguientes pasos: • Elimine el ARN del cuadro del panel Simulation Settings (Configuración de simulación). • Elija el servicio que coincida con el ARN que ha especificado en Simulation Settings (Configuración de simulación).
Esta acción pertenece a un servicio que admite mecanismos de control de acceso especiales, además de políticas basadas en recursos, como las ACL de Amazon S3 o las políticas de bloqueo de almacenes de S3 Glacier. El simulador de políticas no admite estos mecanismos, de modo que los resultados pueden diferir de su entorno de producción.	No hay que hacer nada. Este mensaje es informativo. En la versión actual, el simulador de política evalúa políticas asociadas a usuarios y grupos de usuarios y puede evaluar políticas basadas en recursos para Amazon S3, Amazon SQS, Amazon SNS, y S3 Glacier. El simulador de políticas no admite todos los mecanismos de control de acceso compatibles con otros servicios de AWS.

Mensaje	Pasos para resolver el problema
DynamoDB FGAC no se admite actualmente.	No hay que hacer nada. Este mensaje informativo hace referencia al control de acceso detallado. El control de acceso detallado es la capacidad de utilizar condiciones de políticas de IAM para determina r quién puede obtener acceso a los elementos y atributos de los datos individuales en las tablas e índices de DynamoDB. También se refiere a las acciones que se pueden realizar en estas tablas e índices. La versión actual del simulador de políticas de IAM no admite este tipo de condición de política. Para obtener más información sobre el control de acceso detallado de DynamoDB, consulte Control de acceso detallado para DynamoDB.
Tiene políticas que no cumplen con la sintaxis de la política. Puede utilizar el validador de políticas para revisar las actualizaciones recomendadas para sus políticas.	Este mensaje aparece en la parte superior de la lista de políticas si tiene políticas que no cumplen con la gramática de políticas de IAM. Para simular estas políticas, consulte las opciones de validación de políticas en Validació n de políticas de IAM para identificar y corregir estas políticas.
Esta política debe actualizarse para cumplir con las últimas reglas de sintaxis de la política.	Este mensaje aparece si tiene políticas que no cumplen con la gramática de políticas de IAM. Para simular estas políticas, consulte las opciones de validación de políticas en Validació n de políticas de IAM para identificar y corregir estas políticas.

Uso del simulador de políticas de IAM (AWS CLI y API de AWS)

Normalmente, los comandos del simulador de políticas exigen realizar llamadas a las operaciones de API para hacer dos cosas:

1. Evaluar las políticas y devolver la lista de claves de contexto a las que hacen referencia. Debe conocer las claves de contexto a las que hacen referencia para que pueda proporcionarles valores en el siguiente paso.
2. Simular las políticas, proporcionando una lista de acciones, recursos y claves de contexto que se utilizan durante la simulación.

Por motivos de seguridad, las operaciones de API se han dividido en dos grupos:

- Operaciones de API que simulan únicamente políticas que se transmiten directamente a la API como cadenas. En este conjunto se incluyen GetContextKeysForCustomPolicy y SimulateCustomPolicy.
- Operaciones de API que simulan las políticas que se han asociado a un recurso, usuario, grupo de usuarios o rol de IAM. Dado que estas operaciones de API pueden revelar detalles de permisos asignados a otras entidades de IAM, debe plantearse restringir el acceso a ellas. En este conjunto se incluyen GetContextKeysForPrincipalPolicy y SimulatePrincipalPolicy. Para obtener más información sobre cómo restringir el acceso a operaciones de API, consulte Ejemplos de políticas: AWS Identity and Access Management (IAM).

En ambos casos, las operaciones de API simulan el efecto de una o varias políticas en una lista de acciones y recursos. Cada acción va emparejada con cada recurso y la simulación determina si las políticas permiten o deniegan esa acción para dicho recurso. También puede proporcionar valores para cualquier clave de contexto a la que sus políticas hagan referencia. Puede obtener la lista de claves de contexto a las que las políticas hacen referencia llamando primero a `GetContextKeysForCustomPolicy` o `GetContextKeysForPrincipalPolicy`. Si no proporciona un valor para una clave de contexto, la simulación sigue ejecutándose. Pero los resultados podrían no ser fiables, ya que el simulador de política no puede incluir la clave de contexto en la evaluación.

Para obtener la lista de claves de contexto (AWS CLI, API de AWS)

Utilice lo siguiente para evaluar una lista de las políticas y devolver una lista de claves de contexto que se utilizan en las políticas.

- AWS CLI: `aws iam get-context-keys-for-custom-policy` y `aws iam get-context-keys-for-principal-policy`
- API de AWS: `GetContextKeysForCustomPolicy` y `GetContextKeysForPrincipalPolicy`

Para simular políticas de IAM (AWS CLI, API de AWS)

Utilice lo siguiente para simular políticas de IAM con el fin de determinar los permisos en vigor del usuario.

- AWS CLI: `aws iam simulate-custom-policy` y `aws iam simulate-principal-policy`
- API de AWS: `SimulateCustomPolicy` y `SimulatePrincipalPolicy`

Adición y eliminación de permisos de identidad de IAM

Puede utilizar políticas para definir los permisos para una identidad (usuario, grupo de usuarios o rol). Puede agregar y eliminar permisos asociando y desasociando políticas de IAM para una identidad que utilice la AWS Management Console, AWS Command Line Interface (AWS CLI) o la API de AWS. También puede utilizar las políticas para establecer los límites de los permisos solo para las entidades (usuarios o roles) que están utilizando los mismos métodos. Los límites de permisos son una función avanzada de AWS que controla los permisos que puede tener una entidad como máximo.

Temas

- Terminología
- Ver actividad de la identidad
- Adición de permisos de identidad de IAM (consola)
- Eliminación de permisos de identidad de IAM (consola)
- Agregar políticas de IAM (AWS CLI)
- Eliminación de políticas de IAM (AWS CLI)
- Adición de políticas de IAM (API de AWS)
- Eliminación de políticas de IAM (API de AWS)

Terminología

Al asociar políticas de permisos a identidades (usuarios, grupos de usuarios y roles), la terminología y los procedimientos varían en función de si está trabajando con una política administrada o insertada:

- Asociar – Se utiliza con políticas administradas. Asocie una política administrada a una identidad (usuario, grupo de usuarios o rol). La conexión de una política aplica los permisos en la política a la identidad.
- Desasociar – Se utiliza con políticas administradas. Desvincula una política administrada de una identidad de IAM (usuario, grupo de usuarios o rol). Al desvincular una política se quitan sus permisos de la identidad.
- Integrar – Se utiliza con políticas insertadas. Integre una política insertada en una identidad (usuario, grupo de usuarios o rol). La integración de una política aplica los permisos en la política a la identidad. Dado que una política insertada se almacena en la identidad, se incrusta en lugar de conectarse, aunque el resultado es similar.

> **Note**
>
> Puede integrar una política insertada de un rol vinculado a un servicio solo en el servicio que depende del rol. Consulte la documentación de AWS de su servicio para saber si es compatible con esta característica.

- Eliminar – Se utiliza con políticas insertadas. Elimina una política insertada de una identidad de IAM (usuario, grupo de usuarios o rol). Al eliminar una política se quitan sus permisos de la identidad.

> **Note**
>
> Puede eliminar una política insertada de un rol vinculado a un servicio solo en el servicio que depende del rol. Consulte la documentación de AWS de su servicio para saber si es compatible con esta característica.

Puede utilizar la consola, la AWS CLI o la API de AWS para realizar cualquiera de estas acciones.

Más información

- Para obtener más información acerca de la diferencia entre las políticas administradas e insertadas, consulte Políticas administradas y políticas insertadas.
- Para obtener más información sobre los límites de permisos, consulte Límites de permisos para las entidades de IAM.
- Para obtener información general sobre las políticas de IAM, consulte Políticas y permisos en IAM.
- Para obtener información sobre validar las políticas de IAM, consulte Validación de políticas de IAM.
- El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Ver actividad de la identidad

Antes de cambiar los permisos de una identidad (usuario, grupo de usuarios o rol), debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Adición de permisos de identidad de IAM (consola)

Puede utilizar la AWS Management Console para agregar permisos a una identidad (usuario, grupo de usuarios o rol). Para ello, asocie las políticas administradas que controlan los permisos o especifique una política que sirva como límite de permisos. También puede integrar una política insertada.

Para utilizar una política administrada como una política de permisos para una entidad (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el botón de radio situado junto al nombre de la política que desee asociar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija Acciones y, a continuación, elija Adjuntar.

5. Seleccione una o más identidades a las que asociar la política. Puede utilizar el cuadro de búsqueda para filtrar la lista entidades principales. Después de seleccionar las identidades, elija Attach policy (Asociar política).

Para utilizar una política administrada para configurar un límite de permisos (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, elija el nombre de la política que desea configurar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. En la página de detalles de la política, seleccione la pestaña Entidades asociadas y, a continuación, si es necesario, abra la sección Asociadas como límites de permisos y seleccione Configurar esta política como límite de permisos.
5. Seleccione uno o varios usuarios o roles en los que va a utilizar la política para un límite de permisos. Puede utilizar el cuadro de búsqueda para filtrar la lista entidades principales. Después de seleccionar las entidades principales, seleccione Configurar límite de permisos.

Para integrar una política insertada de un usuario o un rol (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Users (Usuarios) o Roles.
3. En la lista, seleccione el nombre del usuario o rol en el que integrará una política.
4. Elija la pestaña Permissions (Permisos).
5. Seleccione Agregar permisos y, a continuación, Crear política insertada.

> **Note**
>
> No puede incrustar una política insertada en un rol vinculado a un servicio en IAM. Dado que el servicio vinculado define si puede modificar los permisos del rol, podría añadir las políticas adicionales del servicio desde la consola, la API o la AWS CLI. Para consultar la documentación relacionada con los roles vinculados a dicho servicio, visite Servicios

de AWS que funcionan con IAM y elija Yes (Sí) en la columna Service-Linked Role (Rol vinculado al servicio) del servicio.

6. Seleccione entre los siguientes métodos para ver los pasos necesarios para crear su política:

   - Importación de políticas administradas existentes - Puede importar una política administrada en la cuenta y, a continuación, editar la política para personalizarla a sus requisitos específicos. Una política administrada puede ser una política administrada por AWS o una política administrada por el cliente que haya creado anteriormente.
   - Creación de políticas con el editor visual - Puede construir una nueva política desde cero en el editor visual. Si utiliza el editor visual, no tiene que conocer la sintaxis JSON.
   - Creación de políticas mediante el editor JSON: en la opción JSON del editor, puede crear una política utilizando sintaxis JSON. Puede escribir un nuevo documento de política de JSON o pegar un ejemplo de política.

7. Después de crear una política insertada, se integra automáticamente en su usuario o rol.

Para integrar una política insertada para un grupo de usuarios (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija User groups (Grupos de usuarios).
3. En la lista, seleccione el nombre del grupo de usuarios en el que integrará una política.
4. Elija la pestaña de Permisos, elija Agregar permisos y luego Crear política insertada.
5. Haga una de las siguientes acciones:

   - Seleccione la opción Visual para crear la política. Para obtener más información, consulte Creación de políticas con el editor visual.
   - Seleccione la opción JSON para crear la política. Para obtener más información, consulte Creación de políticas mediante el editor JSON.

6. Cuando esté satisfecho con la política, elija Create policy (Crear política).

Para cambiar el límite de permisos para una o varias entidades (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, elija el nombre de la política que desea configurar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. En la página de detalles de la política, seleccione la pestaña Entidades asociadas y, a continuación, si es necesario, abra la sección Asociadas como límite de permisos. Seleccione la casilla de verificación situada junto a los usuarios o roles cuyos límites desee cambiar y, a continuación, seleccione Cambiar.
5. Seleccione la política nueva que desea utilizar para un límite de permisos. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas. Después de seleccionar la política, seleccione Configurar límite de permisos.

Eliminación de permisos de identidad de IAM (consola)

Puede utilizar la AWS Management Console para eliminar permisos de una identidad (usuario, grupo de usuarios o rol). Para ello, desasocie las políticas administradas que controlan los permisos o elimine la política aplicada como límite de permisos. También puede eliminar una política insertada.

Para desasociar una política administrada utilizada como política de permisos (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el botón de radio situado junto al nombre de la política que desee desasociar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija Acciones y a continuación seleccione Desconectar.
5. Seleccione las entidades de las que desasociar la política. Puede utilizar el cuadro de búsqueda para filtrar la lista de identidades. Después de seleccionar las identidades, elija Detach policy (Desasociar política).

Para eliminar un límite de permisos (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).

3. En la lista de políticas, elija el nombre de la política que desea configurar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. En la página de resumen de la política, seleccione la pestaña Entidades asociadas y, a continuación, si es necesario, abra la sección Asociadas como límite de permisos y seleccione las entidades de las que desee eliminar el límite de permisos. Después, seleccione Eliminar límite.
5. Confirme que desea eliminar el límite y seleccione Eliminar límite.

Para eliminar una política insertada (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios, Usuarios o Roles.
3. En la lista, elija el nombre del grupo de usuarios, usuario o rol que tiene la política que desea quitar.
4. Elija la pestaña Permissions (Permisos).
5. Seleccione la casilla de verificación situada junto a la política, y luego Eliminar.
6. Seleccione Eliminar en el cuadro de confirmación.

Agregar políticas de IAM (AWS CLI)

Puede utilizar la AWS CLI para agregar permisos a una identidad (usuario, grupo de usuarios o rol). Para ello, asocie las políticas administradas que controlan los permisos o especifique una política que sirva como límite de permisos. También puede integrar una política insertada.

Para utilizar una política administrada como una política de permisos para una entidad (AWS CLI)

1. (Opcional) Para ver información sobre una política administrada, ejecute los siguientes comandos:
   - Para ver una lista de las políticas administradas: aws iam list-policies
   - Para recuperar información detallada sobre una política administrada: get-policy
2. Para asociar una política administrada a una identidad (un usuario, grupo de usuarios o rol), utilice uno de los siguientes comandos:

- aws iam attach-user-policy
- aws iam attach-group-policy
- aws iam attach-role-policy

Para utilizar una política administrada para configurar un límite de permisos (AWS CLI)

1. (Opcional) Para ver información sobre una política administrada, ejecute los siguientes comandos:
   - Para ver una lista de las políticas administradas: aws iam list-policies
   - Para recuperar información detallada sobre una política administrada: aws iam get-policy
2. Para utilizar una política administrada para establecer el límite de permisos para una entidad (un usuario o un rol), utilice uno de los comandos siguientes:
   - aws iam put-user-permissions-boundary
   - aws iam put-role-permissions-boundary

Para integrar una política insertada (AWS CLI)

Para integrar una política insertada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), utilice uno de los siguientes comandos:

- aws iam put-user-policy
- aws iam put-group-policy
- aws iam put-role-policy

Eliminación de políticas de IAM (AWS CLI)

Puede utilizar la AWS CLI para desasociar las políticas administradas que controlan los permisos, o eliminar la política aplicada como límite de permisos. También puede eliminar una política insertada.

Para desasociar una política administrada utilizada como política de permisos (AWS CLI)

1. (Opcional) Para ver información de topología sobre una política, ejecute los siguientes comandos:
   - Para ver una lista de las políticas administradas: aws iam list-policies

- Para recuperar información detallada sobre una política administrada: aws iam get-policy

2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, ejecute los siguientes comandos:

   - Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada:
     - aws iam list-entities-for-policy
   - Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol), utilice uno de los siguientes comandos:
     - aws iam list-attached-user-policies
     - aws iam list-attached-group-policies
     - aws iam list-attached-role-policies

3. Para desasociar una política administrada de una identidad (un usuario, grupo de usuarios o rol), utilice uno de los siguientes comandos:

   - aws iam detach-user-policy
   - aws iam detach-group-policy
   - aws iam detach-role-policy

Para eliminar un límite de permisos (AWS CLI)

1. (Opcional) Para ver qué política administrada se está utilizando actualmente para establecer el límite de permisos para un usuario o un rol, ejecute los comandos siguientes:

   - aws iam get-user
   - aws iam get-role

2. (Opcional) Para ver con qué usuarios o roles se está utilizando una política administrada para un límite de permisos, ejecute el comando siguiente:

   - aws iam list-entities-for-policy

3. (Opcional) Para ver información sobre una política administrada, ejecute los siguientes comandos:

   - Para ver una lista de las políticas administradas: aws iam list-policies
   - Para recuperar información detallada sobre una política administrada: aws iam get-policy

4. Para eliminar un límite de permisos de un usuario o un rol, utilice uno de los comandos siguientes:

   - aws iam delete-user-permissions-boundary
   - aws iam delete-role-permissions-boundary

Para eliminar una política insertada (AWS CLI)

1. (Opcional) Para enumerar todas las políticas insertadas asociadas a una identidad (un usuario, grupo de usuarios o rol), utilice uno de los siguientes comandos:

   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies

2. (Opcional) Para recuperar un documento de política insertada integrada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), utilice uno de los siguientes comandos:

   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy

3. Para eliminar una política insertada de una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), utilice uno de los siguientes comandos:

   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Adición de políticas de IAM (API de AWS)

Puede utilizar la API de AWS para asociar las políticas administradas que controlan los permisos o especificar una política que sirva como límite de permisos. También puede integrar una política insertada.

Para utilizar una política administrada como una política de permisos para una entidad (API de AWS)

1. (Opcional) Para ver información de topología sobre una política, llame a las siguientes operaciones:
   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy
2. Para asociar una política administrada a una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:
   - AttachUserPolicy
   - AttachGroupPolicy
   - AttachRolePolicy

Para utilizar una política administrada para configurar un límite de permisos (API de AWS)

1. (Opcional) Para ver información sobre una política administrada, llame a las siguientes operaciones:
   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy
2. Para utilizar una política administrada para establecer el límite de permisos para una entidad (usuario o rol), llame a una de las operaciones siguientes:
   - PutUserPermissionsBoundary
   - PutRolePermissionsBoundary

Para integrar una política insertada (API de AWS)

Para integrar una política insertada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), llame a una de las siguientes operaciones:

- PutUserPolicy
- PutGroupPolicy
- PutRolePolicy

Eliminación de políticas de IAM (API de AWS)

Puede utilizar la API de AWS para desasociar las políticas administradas que controlan los permisos, o eliminar la política aplicada como límite de permisos. También puede eliminar una política insertada.

Para desasociar una política administrada utilizada como política de permisos (API de AWS)

1. (Opcional) Para ver información de topología sobre una política, llame a las siguientes operaciones:
   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy
2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, llame a las siguientes operaciones:
   - Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada:
     - ListEntitiesForPolicy
   - Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies
3. Para desasociar una política administrada de una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:
   - DetachUserPolicy
   - DetachGroupPolicy
   - DetachRolePolicy

Para eliminar un límite de permisos (API de AWS)

1. (Opcional) Para ver qué política administrada se está utilizando actualmente para establecer el límite de permisos para un usuario o un rol, llame a las operaciones siguientes:
   - GetUser

- GetRole

2. (Opcional) Para ver con qué usuarios o roles se está utilizando una política administrada para un límite de permisos, llame a la operación siguiente:

   - ListEntitiesForPolicy

3. (Opcional) Para ver información sobre una política administrada, llame a las siguientes operaciones:

   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy

4. Para eliminar un límite de permisos de un usuario o un rol, llame a una de las operaciones siguientes:

   - DeleteUserPermissionsBoundary
   - DeleteRolePermissionsBoundary

Para eliminar una política insertada (API de AWS)

1. (Opcional) Para enumerar todas las políticas insertadas asociadas a una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:

   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies

2. (Opcional) Para recuperar un documento de política insertada integrada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), llame a una de las siguientes operaciones:

   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy

3. Para eliminar una política insertada de una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), llame a una de las siguientes operaciones:

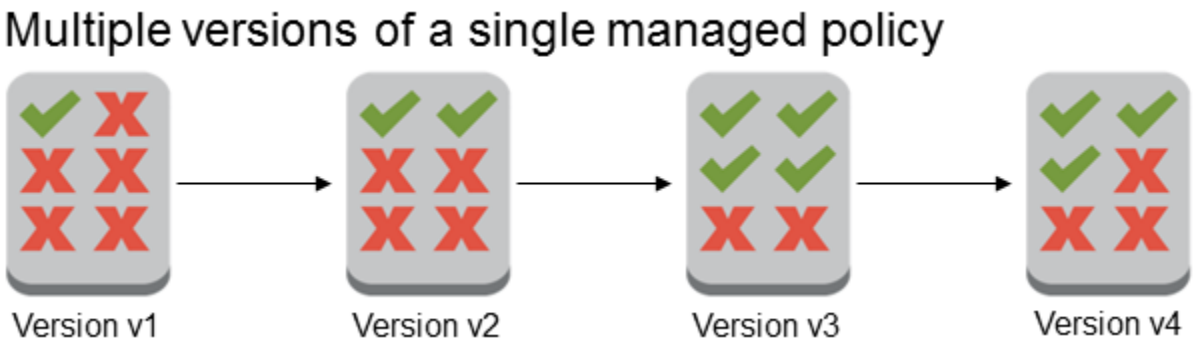
   - DeleteUserPolicy
   - DeleteGroupPolicy

- DeleteRolePolicy

Control de versiones de políticas de IAM

Al realizar cambios en una política administrada por el cliente de IAM y cuando AWS realiza cambios en una política administrada por AWS, la política modificada no sobrescribirá la política existente. En cambio, IAM crea una nueva versión de la política administrada. IAM almacena hasta cinco versiones de las políticas administradas por el cliente. IAM no es compatible con el control de versiones para políticas insertadas.

El siguiente diagrama ilustra el control de versiones para una política administrada por el cliente. En este ejemplo, las versiones de 1 a 4 se guardan. Puede guardar en IAM hasta cinco versiones de políticas administradas. Cuando edite una política que crearía una sexta versión para guardar, podrá elegir qué versión anterior ya no se deberá guardar. Puede volver a cualquiera de las otras cuatro versiones guardadas en cualquier momento.



Una versión de política es diferente de un elemento de política `Version`. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Para obtener más información sobre el elemento de política `Version`, consulte Elementos de política JSON de IAM: Version.

Puede utilizar las versiones para realizar un seguimiento de los cambios realizados en una política administrada. Por ejemplo, puede realizar un cambio en una política administrada y, a continuación, descubrir que el cambio tenía efectos no deseados. En este caso, puede volver a una versión anterior de la política administrada configurando la versión anterior como la versión predeterminada.

En las siguientes secciones, se explica cómo puede utilizar el control de versiones para las políticas administradas.

Temas

- Permisos para configurar la versión predeterminada de una política
- Configuración de la versión predeterminada de políticas administradas por el cliente
- Uso de versiones para revertir los cambios
- Límites de versión

Permisos para configurar la versión predeterminada de una política

Los permisos que se requieren para establecer la versión predeterminada de una política corresponden a las operaciones de API de AWS para la tarea. Puede utilizar las operaciones de API `CreatePolicyVersion` o `SetDefaultPolicyVersion` para establecer la versión predeterminada de una política. Para permitir que alguien establezca la versión predeterminada de una política existente, puede permitir el acceso a la acción `iam:CreatePolicyVersion` o a la acción `iam:SetDefaultPolicyVersion`. La acción `iam:CreatePolicyVersion` permite crear una nueva versión de la política y establecerla como predeterminada. La acción `iam:SetDefaultPolicyVersion` permite definir cualquier versión existente de la política como predeterminada.

> **Important**
>
> Denegar la acción `iam:SetDefaultPolicyVersion` en una política de usuario no impide a este último crear una nueva versión de la política y configurarla como predeterminada.

Puede utilizar la siguiente política para denegar a un usuario el acceso para cambiar una política existente administrada por el cliente:

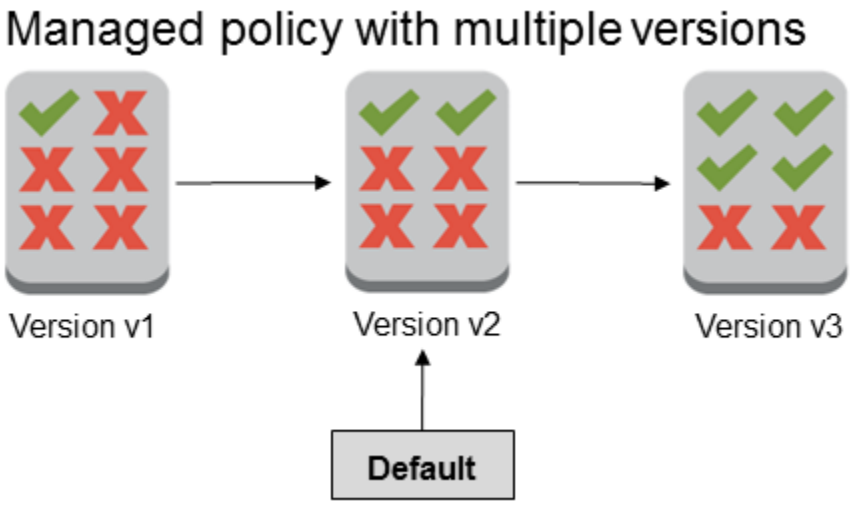
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
        }
    ]
```

```
}
```

Configuración de la versión predeterminada de políticas administradas por el cliente

Una de las versiones de una política administrada se establece como la versión predeterminada. La versión predeterminada de la política es la versión operativa, es decir, se trata de la versión que está en vigor para todas las entidades principales (usuarios, grupos de usuarios y roles) a las que la política administrada está asociada.

Al crear una política administrada por el cliente, la política comienza con una única versión identificada como v1. En el caso de las políticas administradas con una versión única, dicha versión se establece automáticamente como la opción predeterminada. En el caso de las políticas administradas por el cliente con más de una versión, puede seleccionar la versión que quiere establecer como la opción predeterminada. En el caso de las políticas administradas por AWS, AWS establece la versión predeterminada. Los siguientes diagramas ilustran este concepto.



Puede establecer la versión predeterminada de una política administrada por el cliente para aplicar dicha versión a cada identidad de IAM (usuario, grupo de usuarios y rol) a la que la política está asociada. No puede establecer la versión predeterminada de una política administrada por AWS o una política insertada.

Para establecer la versión predeterminada de una política administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, elija el nombre de la política de la que desea establecer la versión predeterminada. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija la pestaña Policy versions (Versiones de la política). Active la casilla de verificación situada junto a la versión que desea establecer como la versión predeterminada y, a continuación, elija Set as default (Establecer como predeterminada).

Para descubrir cómo establecer la versión predeterminada de una política administrada por el cliente desde AWS Command Line Interface o la API de AWS, consulte Edición de políticas administradas por el cliente (AWS CLI).

Uso de versiones para revertir los cambios

Puede establecer la versión predeterminada de una política administrada por el cliente para revertir los cambios. Por ejemplo, fíjese en el siguiente escenario:

Cree una política administrada por el cliente que permita a los usuarios administrar un determinado bucket de Amazon S3 con la AWS Management Console. Tras la creación, la política administrada por el cliente tiene una única versión, identificada como v1, por lo que dicha versión se establece automáticamente como la opción predeterminada. La política funciona según lo previsto.

Posteriormente, puede actualizar la política para agregar permisos para administrar un segundo bucket de Amazon S3. IAM crea una nueva versión de la política, identificada como v2, que incluye los cambios. Establezca la versión v2 como la opción predeterminada y poco tiempo después los usuarios le informan que no tienen permiso para utilizar la consola de Amazon S3. En este caso, puede volver a la versión v1 de la política, que sabe que funciona según lo previsto. Para ello, establezca la versión v1 como la versión predeterminada. Los usuarios pueden ahora utilizar la consola de Amazon S3 para administrar el bucket original.

Posteriormente, después de determinar el error en la versión v2 de la política, debe volver a actualizar la política para agregar permisos para administrar el segundo bucket de Amazon S3. IAM crea una nueva versión de la política, identificada como v3. Establezca la versión v3 como la opción predeterminada y esta versión funciona según lo previsto. En este momento, elimine la versión v2 de la política.

Límites de versión

Una política administrada puede tener hasta cinco versiones. Si necesita realizar cambios a partir de la quinta versión de la política administrada desde AWS Command Line Interface o la API de AWS, primero debe eliminar una o varias de las versiones anteriores. Si utiliza la AWS Management Console, no tiene que eliminar una versión antes de editar su política. Al guardar una sexta versión, aparecerá un cuadro de diálogo que le pedirá que elimine una o varias versiones no predeterminadas de la política. Puede ver el documento de política JSON de cada versión que le ayudará a decidir. Para obtener más información sobre este cuadro de diálogo, consulte the section called “Edición de políticas de IAM”.

Puede eliminar la versión de la política administrada que quiera, excepto la versión predeterminada. Al eliminar una versión, los identificadores de versión de las demás versiones no cambian. Por este motivo, es posible que los identificadores de versión no sean secuenciales. Por ejemplo, si elimina las versiones v2 y v4 de una política administrada y añade dos nuevas versiones, es posible que los demás identificadores de versión sean v1, v3, v5, v6 y v7.

Edición de políticas de IAM

Una política es una entidad que, cuando se asocia a una identidad o recurso, define sus permisos. Las políticas se almacenan en AWS como documentos JSON y se asocian a entidades principales como políticas basadas en identidad en IAM. Puede asociar una política basada en la identidad a una entidad principal (o identidad), como un grupo de usuarios, usuario o rol de IAM. Las políticas basadas en identidad incluyen políticas administradas por AWS, políticas administradas por el cliente y políticas insertadas. Solo puede editar las políticas administradas por el cliente y las políticas insertadas en IAM. Las políticas administradas por AWS no se pueden editar. El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Temas

Ver acceso a políticas

Antes de cambiar los permisos para una política, debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Edición de políticas administradas por el cliente (Consola)

Puede editar políticas administradas por el cliente para cambiar los permisos que están definidos en la política. Una política administrada por el cliente puede tener hasta cinco versiones. Esto es importante ya que si realiza cambios a partir de la quinta versión de la política administrada, la AWS Management Console le pedirá que decida cuál de las versiones anteriores quiere eliminar. También puede cambiar la versión predeterminada o eliminar una versión de una política antes de editarla para evitar indicaciones. Para obtener más información sobre versiones, consulte Control de versiones de políticas de IAM.

Para editar una política administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, elija el nombre de la política que desea editar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Seleccione la pestaña Permisos y, a continuación, Editar.
5. Haga una de las siguientes acciones:

   - Seleccione la opción Visual para cambiar la política sin conocer la sintaxis JSON. Puede realizar cambios en el servicio, acciones, recursos o condiciones opcionales para cada bloque de permisos en su política. También puede importar una política para añadir permisos adicionales en la parte inferior de la política. Cuando haya terminado con los cambios, seleccione Siguiente para continuar.
   - Seleccione la opción JSON para modificar la política escribiendo o pegando texto en el cuadro de texto JSON. También puede importar una política para añadir permisos adicionales en la parte inferior de la política. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next.

   > **Note**
   >
   > Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

6. En la página Revisar y guardar, revise los Permisos definidos en esta política y, a continuación, seleccione Guardar cambios para guardar su trabajo.
7. Si la política administrada ya tiene el máximo de cinco versiones, al seleccionar Guardar cambios aparecerá un cuadro de diálogo. Para guardar la nueva versión, se elimina la versión no predeterminada más antigua de la política y se sustituye con esta nueva versión. También puede configurar la nueva versión como versión predeterminada de la política.

   Seleccione Guardar cambios para guardar la nueva versión de la política.

Para establecer la versión predeterminada de una política administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, elija el nombre de la política de la que desea establecer la versión predeterminada. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija la pestaña Policy versions (Versiones de la política). Active la casilla de verificación situada junto a la versión que desea establecer como la versión predeterminada y, a continuación, elija Set as default (Establecer como predeterminada).

Para eliminar una versión de una política administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. Elija el nombre de la política administrada por el cliente que tenga una versión que desea eliminar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija la pestaña Policy versions (Versiones de la política). Seleccione la casilla de verificación situada junto a la versión que desea eliminar. A continuación, elija Delete (Eliminar).
5. Confirme que desea eliminar la versión y, a continuación, seleccione Delete (Eliminar).

Edición de políticas insertadas (consola)

Puede editar una política insertada desde la AWS Management Console.

Para editar una política insertada de un grupo de usuarios, un usuario o un rol (consola)

1. En el panel de navegación, elija Grupos de usuarios, Usuarios o Roles.
2. Seleccione el nombre del grupo de usuarios, usuario o rol que tenga la política que quiere modificar. A continuación, seleccione la pestaña Permissions (Permisos) y expanda la política.
3. Para editar una política insertada, elija Edit policy (Editar política).
4. Haga una de las siguientes acciones:

   - Seleccione la opción Visual para cambiar la política sin conocer la sintaxis JSON. Puede realizar cambios en el servicio, acciones, recursos o condiciones opcionales para cada

bloque de permisos en su política. También puede importar una política para añadir permisos adicionales en la parte inferior de la política. Cuando haya terminado con los cambios, seleccione Siguiente para continuar.

- Seleccione la opción JSON para modificar la política escribiendo o pegando texto en el cuadro de texto JSON. También puede importar una política para añadir permisos adicionales en la parte inferior de la política. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Next. Para guardar los cambios sin afectar a las entidades asociadas actualmente, desactive la casilla de verificación Save as default version (Guardar como versión predeterminada).

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

5. En la página Revisar, revise el resumen de la política y después seleccione Guardar cambios para guardar su trabajo.

Edición de políticas administradas por el cliente (AWS CLI)

Puede editar una política administrada por el cliente desde AWS Command Line Interface (AWS CLI).

> **Note**
>
> Una política administrada puede tener hasta cinco versiones. Si necesita realizar cambios a partir de la quinta versión de la política administrada del cliente, primero debe eliminar una o varias de las versiones anteriores.

Para editar una política administrada por el cliente (AWS CLI)

1. (Opcional) Para ver información de topología sobre una política, ejecute los siguientes comandos:

- Para enumerar las políticas administradas: list-policies
- Para recuperar información detallada sobre una política administrada: get-policy

2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, ejecute los siguientes comandos:

   - Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada:
     - list-entities-for-policy
   - Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol):
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies

3. Para editar una política administrada por el cliente, ejecute el siguiente comando:

   - create-policy-version

4. (Opcional) Para validar una política administrada por el cliente, ejecute el siguiente comando de IAM Access Analyzer:

   - validate-policy

Para establecer la versión predeterminada de una política administrada por el cliente (AWS CLI)

1. (Opcional) Para obtener una lista de políticas administradas, ejecute el siguiente comando:

   - list-policies

2. Para establecer la versión predeterminada de una política administrada por el cliente, ejecute el siguiente comando:

   - set-default-policy-version

Para eliminar una versión de una política administrada por el cliente (AWS CLI)

1. (Opcional) Para obtener una lista de políticas administradas, ejecute el siguiente comando:

   - list-policies

2. Para eliminar una política administrada por el cliente, ejecute el siguiente comando:

   - delete-policy-version

Edición de políticas administradas por el cliente (API de AWS)

Puede editar una política administrada por el cliente con la API de AWS.

> **Note**
>
> Una política administrada puede tener hasta cinco versiones. Si necesita realizar cambios a partir de la quinta versión de la política administrada del cliente, primero debe eliminar una o varias de las versiones anteriores.

Para editar una política administrada por el cliente (API de AWS)

1. (Opcional) Para ver información de topología sobre una política, llame a las siguientes operaciones:

   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy

2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, llame a las siguientes operaciones:

   - Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada:
     - ListEntitiesForPolicy
   - Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol):
     - ListAttachedUserPolicies
     - ListAttachedGroupPolicies
     - ListAttachedRolePolicies

3. Para editar una política administrada por el cliente, llame a la siguiente operación:

   - CreatePolicyVersion

4. (Opcional) Para validar una política administrada por el cliente, llame a la siguiente operación de IAM Access Analyzer:

   - ValidatePolicy

Para establecer la versión predeterminada de una política administrada por el cliente (API de AWS)

1. (Opcional) Para obtener una lista de políticas administradas, llame a la siguiente operación:

   - ListPolicies
2. Para establecer la versión predeterminada de una política administrada por el cliente, llame a la siguiente operación:

   - SetDefaultPolicyVersion

Para eliminar una versión de una política administrada por el cliente (API de AWS)

1. (Opcional) Para obtener una lista de políticas administradas, llame a la siguiente operación:

   - ListPolicies
2. Para eliminar una política administrada por el cliente, llame a la siguiente operación:

   - DeletePolicyVersion

Eliminación de políticas de IAM

Puede eliminar políticas de IAM mediante AWS Management Console, AWS Command Line Interface (AWS CLI) o la API de IAM.

> **Note**
>
> La eliminación de las políticas de IAM es permanente. Una vez que se elimina la política, no se puede recuperar.

Para obtener más información acerca de la diferencia entre las políticas administradas e insertadas, consulte Políticas administradas y políticas insertadas.

Para obtener información general sobre las políticas de IAM, consulte Políticas y permisos en IAM.

El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

Temas

- Ver acceso a políticas
- Eliminar políticas de IAM (Consola)
- Eliminación de políticas de IAM (AWS CLI)
- Eliminación de políticas de IAM (API de AWS)

Ver acceso a políticas

Antes de eliminar una política debe revisar su actividad de nivel de servicio reciente. Esto es importante porque no desea eliminar el acceso de un principal (persona o aplicación) que está utilizándolo. Para obtener más información acerca de cómo ver la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Eliminar políticas de IAM (Consola)

Puede eliminar una política administrada por el cliente para quitarla de su cuenta de Cuenta de AWS. No se puede eliminar políticas administradas por AWS.

Para eliminar una política administrada por el cliente (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. Seleccione el botón de radio situado junto a la política administrada por el cliente que desee eliminar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Elija Actions (Acciones) y, a continuación, elija Delete (Eliminar).
5. Siga las instrucciones para confirmar que desea eliminar la política y, a continuación, seleccione Eliminar.

Para eliminar una política insertada de un grupo de usuarios, un usuario o un rol (consola)

1. En el panel de navegación, elija Grupos de usuarios, Usuarios o Roles.

2. Seleccione el nombre del grupo de usuarios, usuario o rol que tenga la política que desea eliminar. Después seleccione la pestaña Permissions (Permisos).
3. Seleccione las casillas de verificación situadas junto a las políticas que desee eliminar, y luego Eliminar. Para eliminar una política insertada en Usuarios o Roles, seleccione Eliminar con el fin de confirmar su eliminación. Si va a eliminar una única política insertada en Grupos de usuarios, escriba el nombre de la política y elija Eliminar. Si va a eliminar varias políticas insertadas en Grupos de usuarios, escriba el número de políticas que está eliminando seguido de **`inline policies`** y elija Eliminar. Por ejemplo, si va a eliminar tres políticas insertadas, escriba **`3 inline policies`**.

Eliminación de políticas de IAM (AWS CLI)

Puede eliminar una política administrada por el cliente desde AWS Command Line Interface.

Para eliminar una política administrada por el cliente (AWS CLI)

1. (Opcional) Para ver información de topología sobre una política, ejecute los siguientes comandos:
   - Para enumerar las políticas administradas: list-policies
   - Para recuperar información detallada sobre una política administrada: get-policy
2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, ejecute los siguientes comandos:
   - Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada, ejecute el siguiente comando:
     - list-entities-for-policy
   - Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol), ejecute uno de los siguientes comandos:
     - list-attached-user-policies
     - list-attached-group-policies
     - list-attached-role-policies
3. Para eliminar una política administrada por el cliente, ejecute el siguiente comando:
   - delete-policy

Para eliminar una política insertada (AWS CLI)

1. (Opcional) Para enumerar todas las políticas insertadas asociadas a una identidad (un usuario, grupo de usuarios o rol), utilice uno de los siguientes comandos:
   - aws iam list-user-policies
   - aws iam list-group-policies
   - aws iam list-role-policies
2. (Opcional) Para recuperar un documento de política insertada integrada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), utilice uno de los siguientes comandos:
   - aws iam get-user-policy
   - aws iam get-group-policy
   - aws iam get-role-policy
3. Para eliminar una política insertada de una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), utilice uno de los siguientes comandos:
   - aws iam delete-user-policy
   - aws iam delete-group-policy
   - aws iam delete-role-policy

Eliminación de políticas de IAM (API de AWS)

Puede eliminar una política administrada por el cliente con la API de AWS.

Para eliminar una política administrada por el cliente (API de AWS)

1. (Opcional) Para ver información de topología sobre una política, llame a las siguientes operaciones:
   - Para enumerar las políticas administradas: ListPolicies
   - Para recuperar información detallada sobre una política administrada: GetPolicy
2. (Opcional) Para obtener información acerca de las relaciones entre las políticas e identidades, llame a las siguientes operaciones:

- Para enumerar las identidades (usuarios, grupos de usuarios y roles) a los que está asociada una política administrada, llame a la siguiente operación:
  - ListEntitiesForPolicy
- Para enumerar las políticas administradas asociadas a una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:
  - ListAttachedUserPolicies
  - ListAttachedGroupPolicies
  - ListAttachedRolePolicies

3. Para eliminar una política administrada por el cliente, llame a la siguiente operación:
   - DeletePolicy

Para eliminar una política insertada (API de AWS)

1. (Opcional) Para enumerar todas las políticas insertadas asociadas a una identidad (un usuario, grupo de usuarios o rol), llame a una de las siguientes operaciones:
   - ListUserPolicies
   - ListGroupPolicies
   - ListRolePolicies
2. (Opcional) Para recuperar un documento de política insertada integrada en una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), llame a una de las siguientes operaciones:
   - GetUserPolicy
   - GetGroupPolicy
   - GetRolePolicy
3. Para eliminar una política insertada de una identidad (usuario, grupo de usuarios o rol que no sea un rol vinculado a un servicio), llame a una de las siguientes operaciones:
   - DeleteUserPolicy
   - DeleteGroupPolicy
   - DeleteRolePolicy

Perfeccionar los permisos con la información sobre los últimos accesos en AWS

Como administrador, puede conceder permisos a recursos de IAM (roles, usuarios, grupos de usuarios o políticas) más allá de lo que necesitan. IAM proporciona información sobre los últimos accesos, lo que puede ayudarle a identificar los permisos que no se han utilizado para eliminarlos. Puede utilizar la información sobre los últimos accesos para perfeccionar las políticas y permitir el acceso exclusivo a los servicios y acciones que las identidades y políticas de IAM utilizan. Esto le ayudará a cumplir mejor las prácticas recomendadas del principio de privilegios mínimos. Puede consultar la información sobre los últimos accesos de identidades o políticas existentes en IAM o AWS Organizations.

Puede supervisar continuamente la información a la que se accedió por última vez con analizadores de acceso no utilizados. Para obtener más información, consulte Resultados relacionados con el acceso externo y no utilizado.

Temas

Tipos de información para IAM sobre los últimos accesos

Puede ver dos tipos de información sobre los últimos accesos para las identidades de IAM: información sobre los servicios de AWS permitidos e información sobre las acciones permitidas. Esta información incluye la fecha y la hora en las que se intentó acceder a la API de AWS. En el caso de las acciones, la información sobre los últimos accesos brinda información sobre las acciones de administración del servicio. Las acciones de la gerencia incluyen la creación, eliminación y

modificación de acciones. Para conocer más acerca de cómo ver la información de IAM sobre los últimos accesos, consulte Ver la información de acceso reciente de IAM.

Si desea ver escenarios de ejemplo en los que la información sobre los últimos accesos se utiliza para tomar decisiones sobre los permisos concedidos a las identidades de IAM, consulte Ejemplos de escenarios sobre el uso de información de acceso reciente.

Para obtener más información acerca de cómo se suministra la información sobre las acciones de administración, consulte Cosas que debe saber sobre la información de acceso reciente.

Información de acceso reciente de AWS Organizations

Si inicia sesión con las credenciales de la cuenta de administración, podrá ver información sobre los últimos accesos a servicios de una política o entidad de AWS Organizations de la organización. Las entidades de AWS Organizations pueden ser cuentas, unidades organizativas o la raíz de la organización. En la información de acceso reciente de AWS Organizations, se incluyen los servicios permitidos por una política de control de servicios (SCP). Esta información indica qué entidades principales (usuario raíz, usuario de IAM o rol) en una organización o cuenta intentaron acceder por última vez al servicio, además de cuándo lo hicieron. Para obtener más información sobre el informe y cómo consultar la información de AWS Organizations sobre acceso reciente, consulte Ver la información de último acceso de Organizations.

Si desea ver escenarios de ejemplo en los que la información de acceso reciente se utiliza para tomar decisiones sobre los permisos concedidos a las entidades de Organizations, consulte Ejemplos de escenarios sobre el uso de información de acceso reciente.

Cosas que debe saber sobre la información de acceso reciente

Antes de utilizar la información de un informe sobre los últimos accesos para modificar los permisos de una identidad de IAM o entidad de Organizations, revise los siguientes detalles sobre la información.

- Periodo de seguimiento: la actividad reciente aparece en la consola de IAM dentro de las cuatro horas. El periodo de seguimiento de la información sobre los servicios es de al menos 400 días, dependiendo de cuándo el servicio comenzó el seguimiento de la información sobre las acciones. El periodo de seguimiento de la información de las acciones de Amazon S3 comenzó el 12 de abril de 2020. El período de seguimiento de las acciones de Amazon EC2, IAM y Lambda comenzó el 7 de abril de 2021. El periodo de seguimiento de todos los demás servicios comenzó el 23 de mayo de 2023. Para ver una lista de los servicios sobre los que hay disponible información sobre los últimos accesos a la acción, consulte Servicios y acciones de la información sobre los últimos

accesos a la acción de IAM. Para obtener más información sobre las regiones en las que hay disponible información sobre los últimos accesos a la acción, consulte Dónde AWS se hace un seguimiento de la información de acceso reciente.

- Intentos informados: los datos de los últimos servicios a los que se ha accedido incluyen todos los intentos de acceso a una API de AWS, no solo los intentos que hayan funcionado. Esto incluye todos los intentos que se realizaron mediante la AWS Management Console, la API de AWS a través de cualquiera de los SDK, o cualquiera de las herramientas de la línea de comandos. Una entrada inesperada en los datos de los últimos servicios a los que se ha accedido no significa que su cuenta se haya visto comprometida, ya que puede haberse denegado la solicitud. Consulte los logs de CloudTrail, como la fuente autorizada de información sobre todas las llamadas a la API, y si funcionaron o se les denegó el acceso.
- PassRole: no se realiza ningún seguimiento de la acción `iam:PassRole` y esta acción no se incluye en la información de IAM sobre los últimos accesos.
- Información sobre los últimos accesos a la acción: la información sobre los últimos accesos a la acción está disponible para las acciones de administración de servicios a las que acceden las identidades de IAM. Consulte la lista de servicios y sus acciones sobre los que se informa acerca de la acción a la que se accedió por última vez.

> **Note**
>
> La información sobre los últimos accesos a la acción no está disponible para los eventos de datos de Amazon S3.

- Eventos de administración: IAM proporciona información sobre acciones para los eventos de administración de servicios que CloudTrail registra. A veces, los eventos de administración de CloudTrail también se denominan «operaciones de plano de control» o «eventos de plano de control». Los eventos de administración proporcionan visibilidad sobre las operaciones administrativas que se realizan en los recursos de su Cuenta de AWS. Para obtener más información sobre los eventos de administración en CloudTrail, consulte Registro de eventos de administración en la Guía del usuario AWS CloudTrail.
- Propietario del informe: solo la entidad principal que genera un informe puede ver los detalles del informe. Esto significa que, cuando consulte los datos en AWS Management Console, es posible que tenga que esperar a que se generen y se carguen. Si utiliza la AWS CLI o la API de AWS para obtener los detalles del informe, sus credenciales deben coincidir con las credenciales de la entidad principal que generó el informe. Si utiliza credenciales temporales para un rol o usuario federado, debe generar y recuperar el informe durante la misma sesión. Para obtener más

información acerca de las entidades principales de sesión de rol asumible, consulte Elemento de la política de JSON de AWS: Principal.

- Recursos de IAM: la información sobre los últimos accesos para IAM incluye los recursos de IAM (roles, usuarios, grupos de usuarios y políticas) en su cuenta. La información sobre los últimos accesos para Organizations incluye entidades principales (usuarios de IAM, roles de IAM o Usuario raíz de la cuenta de AWS) en la entidad de Organizations especificada. La información sobre los últimos accesos no incluye los intentos no autenticados.
- Tipos de política de IAM: la información sobre los últimos accesos para IAM incluye los servicios permitidos por las políticas de identidades de IAM. Estas son las políticas asociadas a un rol o asociadas a un usuario directamente o a través de un grupo. El acceso permitido por otros tipos de políticas no se incluye en su informe. Los tipos de políticas excluidos incluyen las políticas basadas en recursos, las listas de control de acceso, las SCP de AWS Organizations, los límites de permisos de IAM y las políticas de sesión. Los permisos que proporcionan los roles vinculados a servicios los define el servicio al que están vinculados y no se pueden modificar en IAM. Para obtener más información sobre los roles vinculados a servicios, consulte Uso de roles vinculados a servicios. Si necesita información acerca de cómo se evalúan los diferentes tipos de políticas para permitir o denegar el acceso, consulte Lógica de evaluación de políticas.
- Tipos de políticas de Organizations: la información de AWS Organizations solo incluye los servicios permitidos por las políticas de control de servicios (SCP) heredadas de la entidad de Organizations. Las SCP son políticas asociadas a una raíz, unidad organizativa o cuenta. El acceso permitido por otros tipos de políticas no se incluye en su informe. Los tipos de políticas excluidos incluyen las políticas basadas en identidades, las políticas basadas en recursos, las listas de control de acceso, los límites de permisos de IAM y las políticas de sesión. Para obtener información sobre cómo los diferentes tipos de políticas se evalúan para permitir o denegar el acceso, consulte Lógica de evaluación de políticas.
- Especificación de un ID de política: cuando usa AWS CLI o la API de AWS para generar un informe con la información de acceso reciente de Organizations, si lo desea, puede especificar el ID de una política. El informe resultante contendrá información sobre los servicios que están permitidos solo en esa política. Los datos contienen la actividad más reciente registrada en la cuenta de la entidad de Organizations especificada o los elementos secundarios de la entidad. Para obtener más información, consulte aws iam generate-organizations-access-report o GenerateOrganizationsAccessReport.
- Cuenta de gestión de Organizations - Debe iniciar sesión en la cuenta de administración de su organización para ver la última información de servicio a la que se ha accedido. Puede ver los datos de la cuenta de administración utilizando la consola de IAM, AWS CLI o la API deAWS .

El informe resultante muestra una lista de todos los servicios de AWS, ya que la cuenta de administración no se está limitada por SCP. Si especifica un ID de política en la CLI o la API, la política no se tiene en cuenta. En cada servicio, el informe incluye únicamente la información de la cuenta maestra. Sin embargo, los informes de otras entidades de Organizations no devuelven información sobre la actividad de la cuenta de administración.

- Configuración de Organizations: un administrador debe habilitar SCP en su raíz de la organización antes de que pueda generar datos para Organizations.

Permisos necesarios

Para poder ver la información de acceso reciente en AWS Management Console, debe tener una política que conceda los permisos necesarios.

Permisos para información de IAM

Si desea utilizar la consola de IAM para ver la información de acceso reciente de un usuario, rol o política de IAM, debe contar con una política que incluya las siguientes acciones:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:Get*`
- `iam:List*`

Estos permisos permiten a un usuario ver lo siguiente:

- Qué usuarios, grupos o roles están asociados a una política administrada
- A qué servicios puede acceder un usuario o rol
- La última vez que se accedió al servicio
- La última vez que intentaron utilizar una acción específica de Amazon EC2, IAM, Lambda, o Amazon S3

Para poder ver la información de acceso reciente de IAM con AWS CLI o la API de AWS, debe contar con los permisos adecuados sobre la operación que desee utilizar:

- `iam:GenerateServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetails`
- `iam:GetServiceLastAccessedDetailsWithEntities`

- iam:ListPoliciesGrantingServiceAccess

Este ejemplo muestra cómo podría crear una política basada en identidad que permite ver la información del último acceso de IAM. Además, permite acceso de solo lectura a todas las partes de IAM. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateServiceLastAccessedDetails",
            "iam:Get*",
            "iam:List*"
        ],
        "Resource": "*"
    }
```

Permisos para información de AWS Organizations

Para utilizar la consola de IAM para ver un informe de la raíz, unidad organizativa o entidades de la cuenta de Organizations, debe contar con una política que incluya las siguientes acciones:

- iam:GenerateOrganizationsAccessReport
- iam:GetOrganizationsAccessReport
- organizations:DescribeAccount
- organizations:DescribeOrganization
- organizations:DescribeOrganizationalUnit
- organizations:DescribePolicy
- organizations:ListChildren
- organizations:ListParents
- organizations:ListPoliciesForTarget
- organizations:ListRoots
- organizations:ListTargetsForPolicy

Para utilizar la información de Organizations sobre los últimos servicios a los que se ha accedido con la AWS CLI o la API de AWS, debe tener una política que incluya las siguientes acciones:

- `iam:GenerateOrganizationsAccessReport`
- `iam:GetOrganizationsAccessReport`
- `organizations:DescribePolicy`
- `organizations:ListChildren`
- `organizations:ListParents`
- `organizations:ListPoliciesForTarget`
- `organizations:ListRoots`
- `organizations:ListTargetsForPolicy`

Este ejemplo muestra cómo podría crear una política basada en identidad que permita ver la información del último acceso al servicio para las organizaciones. Además, permite acceso de solo lectura a todas las partes de Organizations. Esta política define los permisos para el acceso programático y a la consola.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": [
            "iam:GenerateOrganizationsAccessReport",
            "iam:GetOrganizationsAccessReport",
            "organizations:Describe*",
            "organizations:List*"
        ],
        "Resource": "*"
    }
}
```

También puede utilizar la clave de condición iam:OrganizationsPolicyId para poder generar un informe solo para una política de Organizations concreta. Para ver una política de ejemplo, consulte IAM: ver la información del último acceso al servicio para una política de Organizaciones.

Resolución de problemas de la actividad para entidades de IAM y Organizations

En algunos casos, la tabla de información de acceso reciente de AWS Management Console podría estar vacía. O es posible que AWS CLI o la API de AWS devuelvan un conjunto de información vacío o un campo nulo. En estos casos, revise los siguientes problemas:

- En el caso de la información sobre las últimas acciones a las que se ha accedido, es posible que la acción que espera ver no aparezca en la lista. Esto puede ocurrir porque la identidad de IAM no tiene permisos para esta acción o porque AWS aún no hace un seguimiento de la información sobre los últimos accesos a la acción.
- Para un usuario de IAM, asegúrese de que el usuario tenga al menos una política asociada administrada o insertada, ya sea directamente o a través de suscripciones a grupos.
- Para un grupo de IAM, verifique que el grupo tenga al menos una política asociada administrada o insertada.
- En el caso de los grupos de IAM, el informe solo devuelve información de acceso reciente de los servicios a los que accedieron los miembros que utilizaron las políticas del grupo para acceder al servicio. Para saber si un miembro ha utilizado otras políticas, consulte la información de acceso reciente de dicho usuario.
- Para un rol de IAM, verifique que el rol tenga al menos una política asociada administrada o insertada.
- Para una entidad de IAM (usuario o rol), revise otros tipos de políticas que puedan afectar a los permisos de dicha entidad. Estos incluyen las políticas basadas en recursos, las listas de control de acceso, las políticas de AWS Organizations, los límites de permisos de IAM o las políticas de sesión. Para obtener más información, consulte Tipos de políticas o Evaluación de políticas dentro de una misma cuenta.
- Para una política de IAM, asegúrese de que la política administrada especificada esté asociada al menos con un usuario, grupo con miembros o rol.
- Para una entidad de Organizations (raíz, unidad organizativa o cuenta), asegúrese de que ha iniciado sesión con las credenciales de la cuenta de administración de Organizations.
- Compruebe que las SCP están habilitadas en la raíz de su organización.
- La información sobre los últimos accesos de acciones solo está disponible para las acciones detalladas en Servicios y acciones de la información sobre los últimos accesos a la acción de IAM.

Cuando realice los cambios, espere al menos 4 horas para que aparezca la actividad en su informe de la consola de IAM. Si utiliza AWS CLI o la API de AWS, debe generar un nuevo informe para ver la información actualizada.

Dónde AWS se hace un seguimiento de la información de acceso reciente

AWS recopila la información de acceso reciente en las regiones estándar de AWS. Si se incorporan nuevas regiones en AWS, se agregarán a la tabla siguiente y se incluirá la fecha en que AWS comenzó a hacer un seguimiento de la información en cada región:

- Información sobre el servicio: el periodo de seguimiento para los servicios es de al menos 400 días, aunque puede ser inferior si la región comenzó a hacer seguimiento de esta característica dentro de los últimos 400 días.
- Información sobre las acciones: el período de seguimiento de las acciones de administración de Amazon S3 comenzó el 12 de abril de 2020. El período de seguimiento de las acciones de administración de Amazon EC2, IAM y Lambda comenzó el 7 de abril de 2021. El periodo de seguimiento para las acciones de administración de todos los demás servicios comenzó el 23 de mayo de 2023. Si la fecha de seguimiento de una región es posterior al 23 de mayo de 2023, la información sobre los últimos accesos a la acción de esa región se iniciará en la fecha posterior.

Nombre de la región	Región	Fecha de inicio del seguimien to
Este de EE. UU. (Ohio)	us-east-2	27 de octubre de 2017
Este de EE. UU. (Norte de Virginia)	us-east-1	1 de octubre de 2015
Oeste de EE. UU. (Norte de California)	us-west-1	1 de octubre de 2015
Oeste de EE. UU. (Oregón)	us-west-2	1 de octubre de 2015
África (Ciudad del Cabo)	af-south-1	22 de abril de 2020
Asia-Pacífico (Hong Kong)	ap-east-1	24 de abril de 2019
Asia-Pacífico (Hyderabad)	ap-south-2	22 de noviembre de 2022

Nombre de la región	Región	Fecha de inicio del seguimiento
Asia-Pacífico (Yakarta)	ap-southeast-3	13 de diciembre de 2021
Asia-Pacífico (Melbourne)	ap-southeast-4	23 de enero de 2023
Asia-Pacífico (Bombay)	ap-south-1	27 de junio de 2016
Asia-Pacífico (Osaka)	ap-northeast-3	11 de febrero de 2018
Asia-Pacífico (Seúl)	ap-northeast-2	6 de enero de 2016
Asia-Pacífico (Singapur)	ap-southeast-1	1 de octubre de 2015
Asia-Pacífico (Sídney)	ap-southeast-2	1 de octubre de 2015
Asia-Pacífico (Tokio)	ap-northeast-1	1 de octubre de 2015
Canadá (Central)	ca-central-1	28 de octubre de 2017
Europa (Frankfurt)	eu-central-1	1 de octubre de 2015
Europa (Irlanda)	eu-west-1	1 de octubre de 2015
Europa (Londres)	eu-west-2	28 de octubre de 2017
Europa (Milán)	eu-south-1	28 de abril de 2020
Europa (París)	eu-west-3	18 de diciembre de 2017
Europa (España)	eu-south-2	15 de noviembre de 2022
Europa (Estocolmo)	eu-north-1	12 de diciembre de 2018
Europa (Zúrich)	eu-central-2	8 de noviembre de 2022
Israel (Tel Aviv)	il-central-1	1 de agosto de 2023
Medio Oriente (Baréin)	me-south-1	29 de julio de 2019
Medio Oriente (EAU)	me-central-1	30 de agosto de 2022

Nombre de la región	Región	Fecha de inicio del seguimiento
América del Sur (São Paulo)	sa-east-1	11 de diciembre de 2015
AWS GovCloud (Este de EE. UU.)	us-gov-este-1	1 de julio de 2023
AWS GovCloud (Oeste de EE. UU.)	us-gov-oeste-1	1 de julio de 2023

Si una Región no aparece en la tabla anterior, significa que esa Región aún no proporciona información sobre el último acceso.

Una región de AWS es una colección de recursos de AWS que se encuentran en un área geográfica. Las regiones se agrupan en particiones. Las regiones estándar son las regiones que pertenecen a la partición `aws`. Para obtener más información de las distintas particiones, consulte este artículo sobre el formato de los nombres de recurso de Amazon (ARN) en la Referencia general de AWS. Para obtener más información sobre las regiones, consulte Acerca de las regiones de AWS, también en la Referencia general de AWS.

Ver la información de acceso reciente de IAM

Puede ver la información de acceso reciente de IAM con AWS Management Console, AWS CLI o la API de AWS. Consulte la lista de servicios y sus acciones para los que se muestra la información sobre los últimos accesos. Para obtener más información sobre la información de acceso reciente, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Puede consultar información sobre los siguientes tipos de recursos en IAM. En cada caso, la información incluirá los servicios permitidos durante el período concreto del informe:

- Usuario: ver la última vez que el usuario intentó acceder a cada servicio permitido.
- Grupo de usuarios: muestra información acerca de la última vez que un miembro del grupo de usuarios intentó acceder a cada servicio permitido. Este informe también incluye el número total de miembros que intentaron el acceso.
- Rol: muestra la última vez que alguien utilizó el rol en un intento de acceder a cada servicio permitido.

- Política: muestra información acerca de la última vez que un usuario o rol intentó acceder a cada servicio permitido. Este informe también incluye el número total de entidades que intentaron el acceso.

> **Note**
>
> Antes de ver la información de acceso de un recurso de IAM, asegúrese de que entiende el período del informe, las entidades consultadas y los tipos de política evaluados. Para obtener más información, consulte the section called “Cosas que debe saber sobre la información de acceso reciente”.

Ver información de IAM (consola)

Puede ver la información de acceso reciente de IAM en la pestaña Asesor de acceso de la consola de IAM.

Para ver información de IAM (consola)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Grupos de usuarios, Usuarios, Roles o Políticas.
3. Elija un nombre de usuario, grupo de usuarios, rol o política para abrir la página Resumen y elija la pestaña Asesor de acceso. Verá la siguiente información, en función del recurso que elija:

   - Grupo de usuarios: consulte la lista de servicios a los que pueden acceder los miembros de los grupos de usuarios. También puede ver la última vez que un miembro accedió al servicio, qué políticas de grupo de usuarios utilizó y qué miembro del grupo de usuarios realizó la solicitud. Elija el nombre de la política para obtener información acerca de si se trata de una política administrada o una política de grupo insertada de usuarios. Elija el nombre del miembro del grupo de usuarios para ver todos los miembros del grupo de usuarios y la última vez que accedió al servicio.
   - Usuario: permite ver la lista de servicios a los que puede obtener acceso el usuario. También puede ver cuándo accedió por última vez al servicio y qué políticas están actualmente asociadas con el usuario. Elija el nombre de la política para saber si se trata de una política administrada, una política de usuario en línea o una política insertada del grupo de usuarios.

- Rol: vea la lista de servicios a los que puede acceder el rol, cuándo el rol accedió por última vez al servicio y qué políticas se utilizaron. Elija el nombre de la política para obtener información acerca de si se trata de una política administrada o una política de rol insertada.
- Política: vea la lista de servicios con acciones permitidas en la política. También puede ver cuándo se usó la política por última vez para acceder al servicio y qué entidad (usuario o rol) la utilizó. La fecha del último acceso también incluye cuándo se concede el acceso a esta política a través de otra política. Seleccione el nombre de la entidad que quiere saber qué entidades tienen esta política asociada y cuando han accedido por última vez al servicio.

4. En la columna Servicio de la tabla, elija el nombre de uno de los servicios que incluye información sobre los últimos accesos a la acción para ver una lista de las acciones de administración a las que las entidades de IAM han intentado acceder. Podrá ver la Región de AWS y una marca de tiempo que indica la última vez que alguien intentó realizar la acción.
5. En la columna Últimos accesos, se muestran los servicios y las acciones de administración de los servicios que incluyen información sobre los últimos accesos a la acción. Consulte a continuación los resultados que pueden devolverse en esta columna. Estos resultados variarán en función de si un servicio o una acción están permitidos, si se ha accedido a ellos y si su información de acceso reciente se está supervisando en AWS.

   hace <número de> días

   Número de días desde que se utilizó el servicio o la acción en el período de seguimiento. El período de seguimiento de los servicios abarca los últimos 400 días. El periodo de seguimiento de las acciones de Amazon S3 comenzó el 12 de abril de 2020. El periodo de seguimiento de las acciones de Amazon EC2, IAM, y Lambda comenzó el 7 de abril de 2021. El periodo de seguimiento de todos los demás servicios comenzó el 23 de mayo de 2023. Para obtener más información sobre las fechas de inicio de seguimiento de cada Región de AWS, consulte Dónde AWS se hace un seguimiento de la información de acceso reciente.

   No se ha accedido en el período de seguimiento

   Ninguna entidad ha utilizado el servicio o la acción supervisados durante el período de seguimiento.

   Es posible que tenga permisos sobre una acción que no aparece en la lista. Esto puede suceder si AWS actualmente no incluye la información de seguimiento de la acción. No debe tomar decisiones de permisos basándose exclusivamente en la ausencia de cierta información de seguimiento. En su lugar, le recomendamos que utilice esta información para enriquecer y

sustentar la estrategia general de conceder los mínimos privilegios posibles. Consulte sus políticas para confirmar que el nivel de acceso es el adecuado.

Ver información de IAM (AWS CLI)

Puede utilizar la API de AWS CLI para recuperar información sobre la última vez que se utilizó un recurso de IAM para intentar acceder a servicios de AWS y acciones de Amazon S3, Amazon EC2, IAM, y Lambda. Un recurso de IAM puede ser un usuario, un grupo de usuarios, un rol o una política.

Para ver información de IAM (AWS CLI)

1. Genere un informe. La solicitud debe incluir el ARN del recurso de IAM (usuario, grupo de usuarios, rol o política) para el que desea un informe. Puede especificar el nivel de detalle con el que desea generar el informe para ver los datos de acceso solo de los servicios o tanto de los servicios como de las acciones. La solicitud devuelve un `job-id` que puede utilizar en las operaciones `get-service-last-accessed-details-with-entities` y `get-service-last-accessed-details` para monitorear el `job-status` hasta que se complete el trabajo.

   - aws iam generate-service-last-accessed-details

2. Recupere detalles sobre el informe utilizando el parámetro `job-id` del paso anterior.

   - aws iam get-service-last-accessed-details

   Esta operación devuelve la siguiente información, en función del tipo de recurso y el nivel de detalle solicitado en la operación `generate-service-last-accessed-details`:

   - Usuario: devuelve una lista de los servicios a los que puede acceder el usuario especificado. Para cada servicio, la operación devuelve la fecha y la hora del último intento del usuario y el ARN del usuario.
   - Grupo de usuarios: devuelve una lista de servicios a los que pueden acceder los miembros del grupo de usuarios especificado mediante las políticas adjuntas al grupo de usuarios. Para cada servicio, la operación devuelve la fecha y la hora del último intento realizado por cualquier miembro del grupo de usuarios. También devuelve el ARN de dicho usuario y el número total de los miembros del grupo de usuarios que han intentado para acceder al servicio. Utilice la operación GetServiceLastAccessedDetailsWithEntities para recuperar una lista de todos los miembros.

- Rol: devuelve una lista de los servicios a los que puede acceder el rol especificado. Para cada servicio, la operación devuelve la fecha y la hora del último intento del ro y el ARN del rol.
- Política— Devuelve una lista de servicios a los que la política especificada permite el acceso. Para cada servicio, la operación devuelve la fecha y la hora a la que una entidad (usuario o rol) intento acceder por última vez al servicio utilizando la política. También devuelve el ARN de dicha entidad y el número total de entidades que intentaron acceder.

3. Obtenga más información sobre las entidades que utilizaron permisos de política de grupo de usuarios o política en un intento por acceder a un servicio específico. Esta operación devuelve una lista de entidades con cada ARN, ID, nombre, ruta, tipo de entidad (usuario o rol) y cuando intentaron acceder al servicio por última vez. También puede utilizar esta operación para usuarios y roles, pero solo devuelve información acerca de dicha entidad.

   - aws iam get-service-last-accessed-details-with-entities

4. Más información sobre las políticas basadas en identidad que una identidad (usuario, grupo de usuarios o rol) utiliza en un intento de acceder a un servicio específico. Cuando se especifica una identidad y un servicio, esta operación devuelve una lista de políticas de permisos que la identidad puede utilizar para acceder al servicio especificado. Esta operación proporciona el estado actual de las políticas y no depende del informe generado. Tampoco devuelve otros tipos de políticas, como las políticas basadas en recursos, las listas de control de acceso, las políticas de AWS Organizations, los límites de permisos de IAM o las políticas de sesión. Para obtener más información, consulte Tipos de políticas o Evaluación de políticas dentro de una misma cuenta.

   - aws iam list-policies-granting-service-access

Ver información de IAM (API de AWS)

Puede utilizar la API de AWS para recuperar información sobre la última vez que se utilizó un recurso de IAM para intentar acceder a servicios de AWS y acciones de Amazon S3, Amazon EC2, IAM, y Lambda. Un recurso de IAM puede ser un usuario, un grupo de usuarios, un rol o una política. Puede especificar el nivel de detalle con el que se va a generar el informe para ver los datos de acceso solo de los servicios o tanto de los servicios como de las acciones.

Para ver información de IAM (API de AWS)

1. Genere un informe. La solicitud debe incluir el ARN del recurso de IAM (usuario, grupo de usuarios, rol o política) para el que desea un informe. Devuelve un `JobId` que puede

utilizar en las operaciones `GetServiceLastAccessedDetailsWithEntities` y `GetServiceLastAccessedDetails` para monitorizar el `JobStatus` hasta que se complete el trabajo.

- GenerateServiceLastAccessedDetails

2. Recupere detalles sobre el informe utilizando el parámetro `JobId` del paso anterior.

   - GetServiceLastAccessedDetails

   Esta operación devuelve la siguiente información, en función del tipo de recurso y el nivel de detalle solicitado en la operación `GenerateServiceLastAccessedDetails`:

   - Usuario: devuelve una lista de los servicios a los que puede acceder el usuario especificado. Para cada servicio, la operación devuelve la fecha y la hora del último intento del usuario y el ARN del usuario.
   - Grupo de usuarios: devuelve una lista de servicios a los que pueden acceder los miembros del grupo de usuarios especificado mediante las políticas adjuntas al grupo de usuarios. Para cada servicio, la operación devuelve la fecha y la hora del último intento realizado por cualquier miembro del grupo de usuarios. También devuelve el ARN de dicho usuario y el número total de los miembros del grupo de usuarios que han intentado para acceder al servicio. Utilice la operación GetServiceLastAccessedDetailsWithEntities para recuperar una lista de todos los miembros.
   - Rol: devuelve una lista de los servicios a los que puede acceder el rol especificado. Para cada servicio, la operación devuelve la fecha y la hora del último intento del ro y el ARN del rol.
   - Política— Devuelve una lista de servicios a los que la política especificada permite el acceso. Para cada servicio, la operación devuelve la fecha y la hora a la que una entidad (usuario o rol) intento acceder por última vez al servicio utilizando la política. También devuelve el ARN de dicha entidad y el número total de entidades que intentaron acceder.

3. Obtenga más información sobre las entidades que utilizaron permisos de política de grupo de usuarios o política en un intento por acceder a un servicio específico. Esta operación devuelve una lista de entidades con cada ARN, ID, nombre, ruta, tipo de entidad (usuario o rol) y cuando intentaron acceder al servicio por última vez. También puede utilizar esta operación para usuarios y roles, pero solo devuelve información acerca de dicha entidad.

   - GetServiceLastAccessedDetailsWithEntities

4. Más información sobre las políticas basadas en identidad que una identidad (usuario, grupo de usuarios o rol) utiliza en un intento de acceder a un servicio específico. Cuando se especifica una identidad y un servicio, esta operación devuelve una lista de políticas de permisos que la identidad puede utilizar para acceder al servicio especificado. Esta operación proporciona el estado actual de las políticas y no depende del informe generado. Tampoco devuelve otros tipos de políticas, como las políticas basadas en recursos, las listas de control de acceso, las políticas de AWS Organizations, los límites de permisos de IAM o las políticas de sesión. Para obtener más información, consulte Tipos de políticas o Evaluación de políticas dentro de una misma cuenta.

   - ListPoliciesGrantingServiceAccess

Ver la información de último acceso de Organizations

Puede consultar la información de acceso reciente de AWS Organizations con la consola de IAM, AWS CLI o la API de AWS. Para obtener información importante sobre los datos, los permisos necesarios, la solución de problemas y las regiones compatibles, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Cuando inicie sesión en la consola de IAM mediante las credenciales de administración de la cuenta AWS Organizations, puede ver la información de cualquier entidad de su organización. Las entidades de Organizations incluyen la raíz de la organización, las unidades organizativas (OU) y las cuentas. También puede utilizar la consola de IAM para ver información sobre las políticas de control de servicios (SCP) de su organización. IAM muestra una lista de servicios permitidos por las SCP que se aplican a la entidad. En cada servicio, puede ver la información más reciente sobre la actividad de la entidad de Organizations elegida o de sus elementos secundarios.

Si utiliza AWS CLI o la API de AWS con las credenciales de la cuenta de administración, podrá generar un informe de cualquier entidad o política de la organización. Un informe mediante programación de una entidad incluye una lista de los servicios que permiten las SCP que se aplican a la entidad. Para cada servicio, el informe incluye la actividad más reciente de las cuentas de la entidad de Organizations especificada o el subárbol de la entidad.

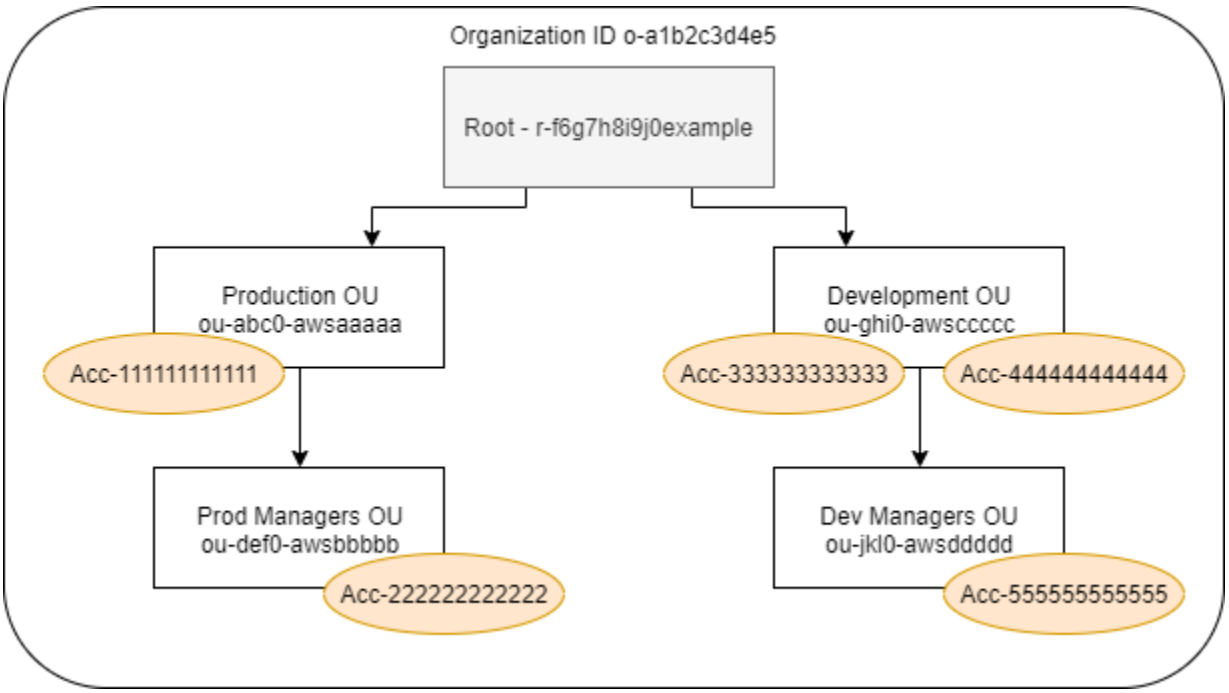
Si genera un informe de una política mediante programación, debe especificar una entidad de Organizations. Este informe incluye una lista de servicios permitidos por la SCP especificada. Para cada servicio, incluye la actividad de la cuenta más reciente en la entidad o los secundarios de la entidad a los que esa política concede permiso. Para obtener más información, consulte aws iam generate-organizations-access-report o GenerateOrganizationsAccessReport.

Antes de ver el informe, asegúrese de que entiende los requisitos y la información de la cuenta de administración, el período del informe, las entidades consultadas y los tipos de política evaluados. Para obtener más información, consulte the section called “Cosas que debe saber sobre la información de acceso reciente”.

Comprender la ruta de la entidad AWS Organizations

Cuando utilice las API de AWS CLI o AWS para generar un informe de acceso AWS Organizations, debe especificar una ruta de acceso de entidad. Una ruta es una representación de texto de la estructura de una entidad de Organizations.

Puede crear una ruta de entidad utilizando la estructura conocida de su organización. Por ejemplo, suponga que tiene la siguiente estructura organizativa en AWS Organizations.



La ruta de acceso para la unidad organizativa (OU) de Dev Managers se crea utilizando los ID de la organización, la raíz y todas las OU de la ruta hasta la OU incluida.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-ghi0-awsccccc/ou-jkl0-awsddddd/
```

La ruta de acceso de la cuenta en la unidad organizativa de Producción se genera utilizando los identificadores de la organización, la raíz, la unidad organizativa y el número de cuenta.

```
o-a1b2c3d4e5/r-f6g7h8i9j0example/ou-abc0-awsaaaaa/111111111111/
```

> **Note**
>
> Los ID de organización son únicos globalmente, pero los ID de unidad organizativa y los ID de raíz solo son únicos dentro de una organización. Esto significa que no hay dos organizaciones que compartan el mismo ID de organización. Sin embargo, otra organización puede tener una unidad organizativa o raíz con el mismo ID que la suya. Le recomendamos que incluya siempre el ID de organización cuando especifique una unidad organizativa o raíz.

Visualización de información para Organizations (consola)

Puede utilizar la consola de IAM para consultar información sobre los últimos servicios a los que ha accedido la raíz, la unidad organizativa, la cuenta o la política.

Para ver información de la raíz (consola)

1. Inicie la sesión en AWS Management Console con las credenciales de la cuenta de administración de Organizations y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación situado debajo de la sección Access reports (Informes de acceso), elija Organization activity (Actividad de la organización).
3. En la página de Organization activity (Actividad de la organización), elija Root (Raíz).
4. En la pestaña Details and activity (Detalles y actividad), examine la sección Service access report (Informe de acceso a servicios). La información contiene una lista de los servicios permitidos por las políticas que están asociadas directamente a la raíz. La información indica desde qué cuenta se accedió por última vez al servicio y cuándo se hizo. Para obtener más información sobre las entidades principales que han accedido al servicio, inicie sesión como administrador en esa cuenta y consulte la información sobre los últimos accesos al servicio de IAM.
5. Elija la pestaña SCP asociadas para ver la lista de políticas de control de servicio (SCP) que están asociadas a la raíz. IAM muestra el número de entidades de destino a las que está asociada cada política. Puede utilizar esta información para decidir qué SCP revisar.
6. Elija el nombre de una SCP para ver todos los servicios que permite la política. Para cada servicio, examine desde qué cuenta se accedió al servicio por última vez, y cuándo.
7. Seleccionar Editar en AWS Organizations para ver detalles adicionales y editar el SCP en la consola Organizations. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS Organizations.

Para ver información de una unidad organizativa o una cuenta (consola)

1. Inicie la sesión en AWS Management Console con las credenciales de la cuenta de administración de Organizations y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación situado debajo de la sección Access reports (Informes de acceso), elija Organization activity (Actividad de la organización).
3. En la página Organization activity (Actividad de la organización), amplíe la estructura de la organización. A continuación, elija el nombre de la unidad organizativa o cuenta que desea ver, excepto la cuenta de administración.
4. En la pestaña Details and activity (Detalles y actividad), examine la sección Service access report (Informe de acceso a servicios). La información contiene una lista de los servicios permitidos por las SCP asociadas a la unidad organizativa o la cuenta y todos sus elementos principales. La información indica desde qué cuenta se accedió por última vez al servicio y cuándo se hizo. Para obtener más información sobre las entidades principales que han accedido al servicio, inicie sesión como administrador en esa cuenta y consulte la información sobre los últimos accesos al servicio de IAM.
5. Elija la pestaña SCP asociadas para ver la lista de políticas de control de servicio (SCP) que están asociadas directamente a la unidad organizativa o la cuenta. IAM muestra el número de entidades de destino a las que está asociada cada política. Puede utilizar esta información para decidir qué SCP revisar.
6. Elija el nombre de una SCP para ver todos los servicios que permite la política. Para cada servicio, examine desde qué cuenta se accedió al servicio por última vez, y cuándo.
7. Seleccionar Editar en AWS Organizations para ver detalles adicionales y editar el SCP en la consola Organizations. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS Organizations.

Para ver información de la cuenta de administración (consola)

1. Inicie la sesión en AWS Management Console con las credenciales de la cuenta de administración de Organizations y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación situado debajo de la sección Access reports (Informes de acceso), elija Organization activity (Actividad de la organización).

3. En la página Actividad de la organización, amplíe la estructura de la organización y elija el nombre de la cuenta de administración.
4. En la pestaña Details and activity (Detalles y actividad), examine la sección Service access report (Informe de acceso a servicios). La información contiene una lista de todos los servicios de AWS. La cuenta de administración no está limitada por las SCP. La información indica si la cuenta accedió al servicio la última vez y cuándo lo hizo. Para obtener más información sobre las entidades principales que han accedido al servicio, inicie sesión como administrador en esa cuenta y consulte la información sobre los últimos accesos al servicio de IAM.
5. Elija la pestaña SCP asociadas para confirmar que no hay SCP asociadas porque la cuenta es la cuenta de administración.

Para ver información de una política (consola)

1. Inicie la sesión en AWS Management Console con las credenciales de la cuenta de administración de Organizations y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación situado debajo de la sección Access reports (Informes de acceso), elija Service control policies (SCPs) (Políticas de control de servicios [SCP]).
3. En la página Service control policies (SCPs) (Políticas de control de servicios [SCP]), consulte la lista de las políticas de su organización. Puede ver el número de entidades de destino a las que está asociada cada política.
4. Elija el nombre de una SCP para ver todos los servicios que permite la política. Para cada servicio, examine desde qué cuenta se accedió al servicio por última vez, y cuándo.
5. Seleccionar Editar en AWS Organizations para ver detalles adicionales y editar el SCP en la consola Organizations. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS Organizations.

Visualización de información para Organizations (AWS CLI)

Puede utilizar AWS CLI para recuperar información de la raíz, una unidad organizativa, una cuenta o una política de Organizations sobre los últimos accesos al servicio.

Para ver la información sobre los últimos accesos al servicio de Organizations (AWS CLI)

1. Utilice las credenciales de la cuenta de administración de Organizations con los permisos necesarios de IAM y Organizations y confirme que las SCP están activadas para su raíz. Para

obtener más información, consulte Cosas que debe saber sobre la información de acceso reciente.

2. Genere un informe. La solicitud debe incluir la ruta de la entidad de Organizations (raíz, unidad organizativa o cuenta) para la que desea un informe. Si lo desea, puede incluir un parámetro `organization-policy-id` para ver un informe para una política específica. El comando devuelve un `job-id` que puede utilizar en el comando `get-organizations-access-report` para monitorizarr el `job-status` hasta que se complete el trabajo.

   - aws iam generate-organizations-access-report

3. Recupere detalles sobre el informe utilizando el parámetro `job-id` del paso anterior.

   - aws iam get-organizations-access-report

   Este comando devuelve una lista de los servicios a los que pueden acceder los miembros de la entidad. Para cada servicio, el comando devuelve la fecha y la hora del último intento del miembro de la cuenta y la ruta de la entidad de la cuenta. También devuelve el número total de servicios a los que es posible acceder y el número de servicios a los que no se ha accedido. Si ha especificado el parámetro `organizations-policy-id` opcional, entonces, los servicios a los que es posible acceder son aquellos que están permitidos por la política especificada.

Visualización de información para Organizations (AWSAPI)

Puede utilizar la API de AWS para recuperar información de la raíz, una unidad organizativa, una cuenta o una política de Organizations sobre los últimos accesos al servicio.

Para ver la información sobre los últimos accesos al servicio de Organizations (API de AWS)

1. Utilice las credenciales de la cuenta de administración de Organizations con los permisos necesarios de IAM y Organizations y confirme que las SCP están activadas para su raíz. Para obtener más información, consulte Cosas que debe saber sobre la información de acceso reciente.
2. Genere un informe. La solicitud debe incluir la ruta de la entidad de Organizations (raíz, unidad organizativa o cuenta) para la que desea un informe. Si lo desea, puede incluir un parámetro `OrganizationsPolicyId` para ver un informe para una política específica. La operación devuelve un `JobId` que puede utilizar en la operación `GetOrganizationsAccessReport` para monitorizar el `JobStatus` hasta que se complete el trabajo.

- GenerateOrganizationsAccessReport

3. Recupere detalles sobre el informe utilizando el parámetro `JobId` del paso anterior.

   - GetOrganizationsAccessReport

   Este operación devuelve una lista de los servicios a los que pueden acceder los miembros de la entidad. Para cada servicio, la operación devuelve la fecha y la hora del último intento del miembro de la cuenta y la ruta de la entidad de la cuenta. También devuelve el número total de servicios a los que es posible acceder y el número de servicios a los que no se ha accedido. Si ha especificado el parámetro `OrganizationsPolicyId` opcional, entonces, los servicios a los que es posible acceder son aquellos que están permitidos por la política especificada.

Ejemplos de escenarios sobre el uso de información de acceso reciente

Puede utilizar la información de acceso reciente para tomar decisiones sobre los permisos que se conceden a las entidades de IAM o a las entidades de AWS Organizations. Para obtener más información, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

> **Note**
>
> Antes de consultar la información de acceso de una entidad o política de IAM o AWS Organizations, asegúrese de que entiende el período del informe, las entidades consultadas y los tipos de política evaluados. Para obtener más información, consulte the section called “Cosas que debe saber sobre la información de acceso reciente”.

Es responsabilidad del administrador elegir la accesibilidad y el principio de privilegios mínimos apropiado para su empresa.

Uso de información para reducir los permisos de un grupo de IAM

Puede utilizar la información de acceso reciente para reducir los permisos de un grupo de IAM e incluir exclusivamente aquellos servicios que necesitan los usuarios. Este método es un importante paso en la concesión de privilegios mínimos en un nivel de servicio.

Por ejemplo, Paulo Santos es el administrador encargado de definir los permisos de usuarios de AWS para Example Corp. Esta empresa acaba de comenzar a utilizar AWS, y el equipo de desarrollo

de software aún no ha definido qué servicios de AWS que utilizarán. Paulo quiere dar el equipo permiso para obtener acceso solo a los servicios que necesitan, pero como aún no se ha definido, les proporciona temporalmente permisos de usuario avanzado. Decide utilizar la información de acceso reciente para reducir los permisos del grupo.

Paulo crea una política administrada denominada `ExampleDevelopment` utilizando el siguiente texto JSON. A continuación, lo asocia a un grupo denominado `Development` y añade todos los desarrolladores al grupo.

> **Note**
>
> Es posible que los usuarios avanzados de Paulo necesiten permisos de `iam:CreateServiceLinkedRole` para utilizar algunos servicios y características. Él entiende que agregar este permiso los habilitará para crear cualquier rol vinculado al servicio. Acepta este riesgo para sus usuarios avanzados.

```
{

    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToAllServicesExceptPeopleManagement",
            "Effect": "Allow",
            "NotAction": [
                "iam:*",
                "organizations:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Paulo decide esperar 90 días antes de ver la información de acceso reciente del grupo `Development` a través de AWS Management Console. Ve la lista de servicios a los que han accedido los miembros del grupo. Se entera de que los usuarios accedieron a cinco servicios en la última semana: AWS CloudTrail, Amazon CloudWatch Logs, Amazon EC2, AWS KMS y Amazon S3. Accedieron a otros servicios cuando estaban evaluando por primera vez AWS y desde entonces no lo hicieron.

Paulo decide reducir los permisos de la política para incluir solo los cinco servicios y las acciones de IAM y Organizations necesarias. Edita la política de `ExampleDevelopment` con el siguiente texto JSON.

> **Note**
>
> Es posible que los usuarios avanzados de Paulo necesiten permisos de `iam:CreateServiceLinkedRole` para utilizar algunos servicios y características. Él entiende que agregar este permiso los habilitará para crear cualquier rol vinculado al servicio. Acepta este riesgo para sus usuarios avanzados.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessToListedServices",
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "kms:*",
                "cloudtrail:*",
                "logs:*",
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "RequiredIamAndOrgsActions",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
}
```

Para reducir aún más los permisos, Paulo puede ver los eventos de la cuenta en el historial de eventos de AWS CloudTrail. Allí puede ver información de eventos detallada que puede utilizar para reducir los permisos de la política para incluir solo las acciones y los recursos que los desarrolladores necesitan. Para obtener más información, consulte Ver eventos de CloudTrail en la consola de CloudTrail en la Guía del usuario de AWS CloudTrail.

Uso de información para reducir los permisos de un usuario de IAM

Puede utilizar la información de acceso reciente para reducir los permisos de un usuario específico de IAM.

Por ejemplo, Martha Rivera es una administradora de TI responsable de garantizar que las personas de su empresa no tengan demasiados permisos de AWS. En el marco de una comprobación de seguridad periódica, revisa los permisos de todos los usuarios de IAM. Uno de ellos es un desarrollador de aplicaciones llamado Nikhil Jayashankar, que previamente trabajaba como ingeniero de seguridad. Debido al cambio en los requisitos de trabajo, Nikhil es miembro del grupo `app-dev` y del grupo `security-team`. El grupo de `app-dev` por su nuevo trabajo concede permisos a varios servicios, como Amazon EC2, Amazon EBS, Auto Scaling, Amazon S3, Route 53 y Elastic Transcoder. El grupo `security-team` para su trabajo anterior concede permisos para IAM y CloudTrail.

Como administradora, Martha inicia sesión en la consola de IAM y elige Usuarios, selecciona el nombre `nikhilj` y elige la pestaña Asesor de acceso.

Martha revisa la columna Último acceso y se da cuenta de que Nikhil no ha accedido recientemente a IAM, CloudTrail, Route 53, Amazon Elastic Transcoder ni a una serie de otros servicios de AWS. Nikhil ha accedido a Amazon S3. Martha elige S3 en la lista de servicios y se entera de que Nikhil ha realizado algunas acciones `List` de S3 en las últimas dos semanas. Dentro de la empresa, Martha confirma que Nikhil ya no necesita acceder a IAM y CloudTrail porque ya no es miembro del equipo de seguridad interna.

Martha ahora está lista para actuar con arreglo a la información sobre los últimos accesos al servicio o la acción. Sin embargo, a diferencia del grupo del ejemplo anterior, un usuario de IAM como `nikhilj` podría estar sujeto a varias políticas y ser miembro de varios grupos. Martha debe continuar con precaución para evitar interrumpir de forma inadvertida el acceso de `nikhilj` u otros los miembros del grupo. Además de descubrir el acceso que Nikhil debe tener, debe determinar cómo está recibiendo estos permisos.

Martha elige la pestaña Permissions (Permisos), donde ve qué políticas están asociadas directamente a `nikhilj` y las asociadas desde un grupo. Amplía cada política y ve el resumen de la política para saber qué política permite el acceso a los servicios que Nikhil no está utilizando:

- IAM: la política administrada `IAMFullAccess` AWS se asigna directamente a `nikhilj` y se asigna al grupo `security-team`.
- CloudTrail: la política administrada de `AWSCloudTrailReadOnlyAccess` AWS se asigna al grupo `security-team`.
- Route 53: la política administrada por el cliente `App-Dev-Route53` se asocia al grupo `app-dev`.
- Elastic Transcoder: la política administrada por el cliente `App-Dev-ElasticTranscoder` se asocia al grupo `app-dev`.

Martha decide eliminar la política administrada `IAMFullAccess` AWS asociada directamente a `nikhilj`. También elimina la pertenencia de Nikhil al grupo `security-team`. Estas dos acciones eliminan el acceso innecesarios a IAM y CloudTrail.

Los permisos de Nikhil para acceder a Route 53 y Elastic Transcoder los concede el grupo `app-dev`. Aunque Nikhil no está utilizando dichos servicios, otros miembros del grupo podrían. Martha consulta la información de acceso reciente del grupo `app-dev` y ve que varios miembros accedieron recientemente a Route 53 y Amazon S3. Sin embargo, ningún miembro del grupo ha accedido a Elastic Transcoder en el último año. Elimina política administrada por el cliente `App-Dev-ElasticTranscoder` del grupo.

A continuación, Martha revisa la información de acceso reciente de la política `App-Dev-ElasticTranscoder` administrada por el cliente. Descubre que la política no está asociada a ninguna otra entidad de IAM. Investiga dentro de su empresa para asegurarse de que la política no se necesitará en el futuro y, a continuación, la elimina.

Uso de información antes de eliminar recursos de IAM

Puede utilizar la información de acceso reciente antes de eliminar un recurso de IAM para asegurarse de que ha transcurrido una determinada cantidad de tiempo desde que alguien utilizó el recurso por última vez. Esto se aplica a usuarios, grupos, roles y políticas. Para obtener más información acerca de estas acciones, consulte los siguientes temas:

- Usuarios: Eliminación de un usuario
- Grupos: Eliminación de un grupo
- Roles: Eliminación de un rol
- Políticas: Eliminación de una política administrada (esta también desasocia la política de las identidades)

Uso de información antes de editar políticas de IAM

Puede revisar la información de acceso reciente de una identidad de IAM (usuario, grupo o rol) o de una política de IAM antes de editar una política que afecte a dicho recurso. Esto es importante porque no desea eliminar el acceso de alguien que lo está utilizando.

Por ejemplo, Arnav Desai es desarrollador y administrador AWS de Example Corp. Cuando su equipo comenzó a utilizar AWS, le dieron acceso a todos los desarrolladores de usuarios avanzados que les permitían acceso completo a todos los servicios, excepto IAM y Organizations. Como primer paso hacia la concesión de privilegios mínimos, Arnav quiere utilizar la AWS CLI para revisar las políticas administradas de su cuenta.

Para ello, Arnav primero lista las políticas de permisos administrados por el cliente de su cuenta asociadas a una identidad, utilizando el comando siguiente:

```
aws iam list-policies --scope Local --only-attached --policy-usage-filter
 PermissionsPolicy
```

De la respuesta, captura el ARN de cada política. Arnav, a continuación, genera un informe con la información de acceso reciente de cada política utilizando el siguiente comando.

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

De esa respuesta, captura el ID del informe generado desde el campo `JobId`. Arnav a continuación, sondea el siguiente comando hasta que el campo `JobStatus` devuelva un valor de `COMPLETED` o `FAILED`. Si se produjo un error en el trabajo, captura el error.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Cuando el trabajo tiene un estado de `COMPLETED`, Arnav analiza el contenido de la matriz `ServicesLastAccessed` con formato JSON.

```
"ServicesLastAccessed": [
       {
           "TotalAuthenticatedEntities": 1,
           "LastAuthenticated": 2018-11-01T21:24:33.222Z,
           "ServiceNamespace": "dynamodb",
           "LastAuthenticatedEntity": "arn:aws:iam::123456789012:user/IAMExampleUser",
           "ServiceName": "Amazon DynamoDB"
       },

       {
           "TotalAuthenticatedEntities": 0,
           "ServiceNamespace": "ec2",
           "ServiceName": "Amazon EC2"
       },

       {
           "TotalAuthenticatedEntities": 3,
           "LastAuthenticated": 2018-08-25T15:29:51.156Z,
           "ServiceNamespace": "s3",
           "LastAuthenticatedEntity": "arn:aws:iam::123456789012:role/IAMExampleRole",
           "ServiceName": "Amazon S3"
       }
   ]
```

A partir de esta información, Arnav descubre que la política `ExamplePolicy1` permite el acceso a tres servicios, Amazon DynamoDB, Amazon S3, y Amazon EC2. El usuario de IAM llamado `IAMExampleUser` intentó accede por última vez a DynamoDB el 1 de noviembre y alguien utilizó el rol `IAMExampleRole` para intentar acceder a Amazon S3 el 25 de agosto. También hay dos entidades más que han intentado acceder a Amazon S3 en el último año. Sin embargo, nadie ha intentado acceder a Amazon EC2 en el último año.

Esto significa que Arnav puede eliminar de forma segura las acciones de Amazon EC2 de la política. Arnav desea revisar el documento JSON actual de la política. En primer lugar, debe determinar el número de versión de la política utilizando el siguiente comando.

```
aws iam list-policy-versions --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

De la respuesta, Arnav recopila los número de versión predeterminada actual de la matriz `Versions`. A continuación, utiliza ese número de versión (`v2`) para solicitar el documento de política JSON utilizando el siguiente comando.

```
aws iam get-policy-version --policy-arn arn:aws:iam::123456789012:policy/ExamplePolicy1
 --version-id v2
```

Arnav almacena el documento de política de JSON devuelto en el campo `Document` de la matriz `PolicyVersion`. Dentro del documento de política, Arnav busca acciones con el espacio de nombres `ec2`. Si no hay acciones de otros espacios de nombres restantes en la política, desasociará la política de las identidades afectadas (usuarios, grupos y roles). A continuación, elimina la política. En este caso, la política no incluye los servicios Amazon DynamoDB y Amazon S3. Por lo tanto, Arnav elimina las acciones de Amazon EC2 del documento y guarda los cambios. A continuación, utiliza el siguiente comando para actualizar la política utilizando la nueva versión del documento y establecer que dicha versión es la versión de política predeterminada.

```
aws iam create-policy-version --policy-arn arn:aws:iam::123456789012:policy/
ExamplePolicy1 --policy-document file://UpdatedPolicy.json --set-as-default
```

La política `ExamplePolicy1` ahora está actualizada para eliminar el acceso al servicio de Amazon EC2 innecesario.

Otros escenarios de IAM

La información sobre cuando un recurso de IAM (usuario, grupo, rol o política) ha intentado por última vez acceder a un servicio puede ayudarle a la hora de completar cualquiera de las siguientes tareas:

- Políticas: Edición de una política administrada por el cliente o insertada existente para eliminar permisos
- Políticas: Conversión de una política insertada en una política administrada y su eliminación a continuación

- Políticas: adición de una denegación explícita a una política existente
- Políticas: desconexión de una política administrada de una identidad (usuario, grupo o rol)
- Entidades: Establecer un límite de permisos para controlar los permisos máximos que una entidad (usuario o rol) puede tener
- Grupos: Eliminación de usuarios de un grupo

Uso de información para perfeccionar los permisos de una unidad organizativa

Puede utilizar la información de acceso reciente para perfeccionar los permisos de una unidad organizativa de AWS Organizations.

Por ejemplo, John Stiles es un administrador de AWS Organizations. Es responsable de garantizar que las personas de las cuentas de Cuentas de AWS de la empresa no tengan demasiados permisos. En el marco de una auditoría de seguridad periódica, revisa los permisos de la organización. Su unidad organizativa `Development` incluye cuentas que se suelen utilizar para probar nuevos servicios de AWS. John decide revisar periódicamente el informe de los servicios a los que no se ha accedido en más de 180 días. Luego elimina los permisos de los miembros de la unidad organizativa para acceder a dichos servicios.

John inicia sesión en la consola de IAM con sus credenciales de cuenta de administración. En la consola de IAM, localiza los datos de Organizations de la unidad organizativa `Development`. Revisa la tabla Service access report (Informe de acceso a servicios) y ve dos servicios de AWS a los que no se ha accedido en más de su período preferido de 180 días. Recuerda que añadió permisos para que los equipos de desarrollo accedieran a Amazon Lex y AWS Database Migration Service. John se pone en contacto con los equipos de desarrollo y confirma que ya no tienen una necesidad comercial de probar estos servicios.

Ahora John está listo para actuar con arreglo a la información de acceso reciente. Elije Editar en AWS Organizations y se le recuerda que la SCP está asociada a varias entidades. Elije Continue (Continuar). En AWS Organizations, revisa los destinos para averiguar a qué entidades de Organizations está asociada la SCP. Todas las entidades se encuentran dentro de la unidad organizativa `Development`.

John decide denegar el acceso a las acciones de Amazon Lex y AWS Database Migration Service en la SCP `NewServiceTest`. Esta acción elimina el acceso innecesario a los servicios.

Servicios y acciones de la información sobre los últimos accesos a la acción de IAM

En la siguiente tabla, se enumeran los servicios de AWS para los que se muestra la información sobre los últimos accesos a la acción de IAM. Para obtener una lista de acciones de cada servicio, consulte Acciones, recursos y claves de condiciones de Servicios de AWS en la Referencia de autorizaciones de servicio.

Servicio	Prefijo de servicio
Analizador de acceso de AWS Identity and Access Management	access-analyzer
AWS Account Management	account
AWS Certificate Manager	acm
Amazon Managed Workflows para Apache Airflow	airflow
AWS Amplify	amplify
Creador de UI de AWS Amplify	amplifyuibuilder
Integraciones de aplicaciones de Amazon	app-integrations
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Application Cost Profiler	application-cost-profiler
Información de aplicaciones de Amazon CloudWatch	applicationinsights
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service para Prometheus	aps
Amazon Athena	athena

Servicio	Prefijo de servicio
AWS Audit Manager	auditmanager
AWS Auto Scaling	autoscaling
AWS Marketplace	aws-marketplace
AWS Backup	backup
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	budgets
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Generador de perfiles de Amazon CodeGuru	codeguru-profiler
Revisor de Amazon CodeGuru	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar

Servicio	Prefijo de servicio
Notificaciones de AWS CodeStar	codestar-notifications
Identidad de Amazon Cognito	cognito-identity
Grupos de usuarios de Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clústeres elásticos de Amazon DocumentDB	docdb-elastic

Servicio	Prefijo de servicio
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store (EBS)	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR en EKS (Contenedores de EMR)	emr-containers
Amazon EMR sin servidor	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	events
Amazon CloudWatch Evidently	evidently
Amazon FinSpace	finspace

Servicio	Prefijo de servicio
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
Almacén de identidad de AWS	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot

Servicio	Prefijo de servicio
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming para Kafka Connect	kafkaconnect
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex	lex
Administrador de suscripciones de Linux de AWS License Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Registros de Amazon CloudWatch	logs

Servicio	Prefijo de servicio
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaPackage	mediapackage
AWS Elemental MediaPackage VOD	mediapackage-vod
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB para Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
Recomendaciones de estrategias de AWS Migration Hub	migrationhub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager

Servicio	Prefijo de servicio
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organizations
AWS Panorama	panorama
Información de rendimiento de AWS	pi
Canalizaciones de Amazon EventBridge	pipes
Amazon Polly	polly
Perfiles de clientes de Amazon Connect	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram
Papelera de reciclaje de AWS	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de datos de Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub

Servicio	Prefijo de servicio
Explorador de recursos de AWS	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
Funciones de AWS Identity and Access Management en cualquier lugar	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Preparación para recuperación de Amazon Route 53	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3
Amazon S3 en Outposts	s3-outposts
Capacidades geoespaciales de Amazon SageMaker	sagemaker-geospatial
Savings Plans	savingsplans
Esquemas de Amazon EventBridge	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	securityhub
Amazon Security Lake	securitylake

Servicio	Prefijo de servicio
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Servicio de SMS y voz de Amazon Pinpoint	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager para SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	tag

Servicio	Prefijo de servicio
Amazon Textract	textract
Amazon Timestream	timestream
Creador de redes de telecomunicaciones de AWS	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transfer
Amazon Translate	translate
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	workspaces
AWS X-Ray	xray

Acciones para la información sobre los últimos accesos a la acción

En la siguiente tabla, se enumeran las acciones para las que está disponible la información sobre los últimos accesos a la acción.

Prefijo de servicio	Acciones
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration

Prefijo de servicio	Acciones
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule
	access-analyzer:GetFinding
	access-analyzer:GetGeneratedPolicy
	access-analyzer:ListAccessPreviewFindings
	access-analyzer:ListAccessPreviews
	access-analyzer:ListAnalyzedResources
	access-analyzer:ListAnalyzers
	access-analyzer:ListArchiveRules
	access-analyzer:ListFindings
	access-analyzer:ListPolicyGenerations
	access-analyzer:StartPolicyGeneration
	access-analyzer:StartResourceScan
	access-analyzer:UpdateArchiveRule

Prefijo de servicio	Acciones
	access-analyzer:UpdateFindings access-analyzer:ValidatePolicy
cuenta	account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAlternateContact account:GetContactInformation account:GetRegionOptStatus account:ListRegions account:PutAlternateContact account:PutContactInformation

Prefijo de servicio	Acciones
acm	acm:DeleteCertificate acm:DescribeCertificate acm:ExportCertificate acm:GetAccountConfiguration acm:GetCertificate acm:ImportCertificate acm:ListCertificates acm:PutAccountConfiguration acm:RenewCertificate acm:RequestCertificate acm:ResendValidationEmail acm:UpdateCertificateOptions
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:UpdateEnvironment

Prefijo de servicio	Acciones
amplify	amplify:CreateApp
	amplify:CreateBackendEnvironment
	amplify:CreateBranch
	amplify:CreateDeployment
	amplify:CreateDomainAssociation
	amplify:CreateWebHook
	amplify:DeleteApp
	amplify:DeleteBackendEnvironment
	amplify:DeleteBranch
	amplify:DeleteDomainAssociation
	amplify:DeleteJob
	amplify:DeleteWebHook
	amplify:GenerateAccessLogs
	amplify:GetApp
	amplify:GetArtifactUrl
	amplify:GetBackendEnvironment
	amplify:GetBranch
	amplify:GetDomainAssociation
	amplify:GetJob
	amplify:GetWebHook
	amplify:ListApps

Prefijo de servicio	Acciones
	amplify:ListArtifacts amplify:ListBackendEnvironments amplify:ListBranches amplify:ListDomainAssociations amplify:ListJobs amplify:ListWebHooks amplify:StartDeployment amplify:StartJob amplify:StopJob amplify:UpdateApp amplify:UpdateBranch amplify:UpdateDomainAssociation amplify:UpdateWebHook

Prefijo de servicio	Acciones
amplifyuibuilder	amplifyuibuilder:CreateComponent amplifyuibuilder:CreateForm amplifyuibuilder:CreateTheme amplifyuibuilder:DeleteComponent amplifyuibuilder:DeleteForm amplifyuibuilder:DeleteTheme amplifyuibuilder:ExportComponents amplifyuibuilder:ExportThemes amplifyuibuilder:GetCodegenJob amplifyuibuilder:GetComponent amplifyuibuilder:GetForm amplifyuibuilder:GetTheme amplifyuibuilder:ListCodegenJobs amplifyuibuilder:ListComponents amplifyuibuilder:ListForms amplifyuibuilder:ListThemes amplifyuibuilder:ResetMetadataFlag amplifyuibuilder:StartCodegenJob amplifyuibuilder:UpdateComponent amplifyuibuilder:UpdateForm amplifyuibuilder:UpdateTheme

Prefijo de servicio	Acciones
app-integrations	app-integrations:CreateApplication app-integrations:CreateDataIntegration app-integrations:CreateEventIntegration app-integrations:DeleteDataIntegration app-integrations:DeleteEventIntegration app-integrations:GetApplication app-integrations:GetDataIntegration app-integrations:GetEventIntegration app-integrations:ListApplications app-integrations:ListDataIntegrationAssociations app-integrations:ListDataIntegrations app-integrations:ListEventIntegrationAssociations app-integrations:ListEventIntegrations app-integrations:UpdateApplication app-integrations:UpdateDataIntegration app-integrations:UpdateEventIntegration

Prefijo de servicio	Acciones
appconfig	appconfig:CreateApplication appconfig:CreateConfigurationProfile appconfig:CreateDeploymentStrategy appconfig:CreateEnvironment appconfig:CreateExtension appconfig:CreateExtensionAssociation appconfig:CreateHostedConfigurationVersion appconfig:DeleteApplication appconfig:DeleteConfigurationProfile appconfig:DeleteDeploymentStrategy appconfig:DeleteEnvironment appconfig:DeleteExtension appconfig:DeleteExtensionAssociation appconfig:DeleteHostedConfigurationVersion appconfig:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment appconfig:GetExtension

Prefijo de servicio	Acciones
	appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeployments appconfig:ListDeploymentStrategies appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

Prefijo de servicio	Acciones
appflow	appflow:CancelFlowExecutions appflow:CreateConnectorProfile appflow:CreateFlow appflow:DeleteConnectorProfile appflow:DeleteFlow appflow:DescribeConnector appflow:DescribeConnectorEntity appflow:DescribeConnectorProfiles appflow:DescribeConnectors appflow:DescribeFlow appflow:DescribeFlowExecutionRecords appflow:ListConnectorEntities appflow:ListConnectors appflow:ListFlows appflow:RegisterConnector appflow:ResetConnectorMetadataCache appflow:StartFlow appflow:StopFlow appflow:UnRegisterConnector appflow:UpdateConnectorProfile appflow:UpdateConnectorRegistration

Prefijo de servicio	Acciones
	appflow:UpdateFlow
application-cost-profiler	application-cost-profiler:DeleteReportDefinition application-cost-profiler:GetReportDefinition application-cost-profiler:ImportApplicationUsage application-cost-profiler:ListReportDefinitions application-cost-profiler:PutReportDefinition application-cost-profiler:UpdateReportDefinition

Prefijo de servicio	Acciones
applicationinsights	applicationinsights:AddWorkload
	applicationinsights:CreateApplication
	applicationinsights:CreateComponent
	applicationinsights:CreateLogPattern
	applicationinsights:DeleteApplication
	applicationinsights:DeleteComponent
	applicationinsights:DeleteLogPattern
	applicationinsights:DescribeApplication
	applicationinsights:DescribeComponent
	applicationinsights:DescribeComponentConfiguration
	applicationinsights:DescribeComponentConfigurationRecommendation
	applicationinsights:DescribeLogPattern
	applicationinsights:DescribeObservation
	applicationinsights:DescribeProblem
	applicationinsights:DescribeProblemObservations
	applicationinsights:DescribeWorkload
	applicationinsights:ListApplications
	applicationinsights:ListComponents
	applicationinsights:ListConfigurationHistory
	applicationinsights:ListLogPatterns

Prefijo de servicio	Acciones
	applicationinsights:ListLogPatternSets applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

Prefijo de servicio	Acciones
appmesh	appmesh:CreateGatewayRoute appmesh:CreateMesh appmesh:CreateRoute appmesh:CreateVirtualGateway appmesh:CreateVirtualNode appmesh:CreateVirtualRouter appmesh:CreateVirtualService appmesh:DeleteGatewayRoute appmesh:DeleteMesh appmesh:DeleteRoute appmesh:DeleteVirtualGateway appmesh:DeleteVirtualNode appmesh:DeleteVirtualRouter appmesh:DeleteVirtualService appmesh:DescribeGatewayRoute appmesh:DescribeMesh appmesh:DescribeRoute appmesh:DescribeVirtualGateway appmesh:DescribeVirtualNode appmesh:DescribeVirtualRouter appmesh:DescribeVirtualService

Prefijo de servicio	Acciones
	appmesh:ListGatewayRoutes appmesh:ListMeshes appmesh:ListRoutes appmesh:ListVirtualGateways appmesh:ListVirtualNodes appmesh:ListVirtualRouters appmesh:ListVirtualServices appmesh:StreamAggregatedResources appmesh:UpdateGatewayRoute appmesh:UpdateMesh appmesh:UpdateRoute appmesh:UpdateVirtualGateway appmesh:UpdateVirtualNode appmesh:UpdateVirtualRouter appmesh:UpdateVirtualService

Prefijo de servicio	Acciones
appstream	appstream:AssociateAppBlockBuilderAppBlock
	appstream:AssociateApplicationFleet
	appstream:AssociateApplicationToEntitlement
	appstream:AssociateFleet
	appstream:BatchAssociateUserStack
	appstream:BatchDisassociateUserStack
	appstream:CopyImage
	appstream:CreateAppBlock
	appstream:CreateAppBlockBuilder
	appstream:CreateAppBlockBuilderStreamingURL
	appstream:CreateApplication
	appstream:CreateDirectoryConfig
	appstream:CreateEntitlement
	appstream:CreateFleet
	appstream:CreateImageBuilder
	appstream:CreateImageBuilderStreamingURL
	appstream:CreateStack
	appstream:CreateStreamingURL
	appstream:CreateUpdatedImage
	appstream:CreateUsageReportSubscription
	appstream:CreateUser

Prefijo de servicio	Acciones
	appstream:DeleteAppBlock
	appstream:DeleteAppBlockBuilder
	appstream:DeleteApplication
	appstream:DeleteDirectoryConfig
	appstream:DeleteEntitlement
	appstream:DeleteFleet
	appstream:DeleteImage
	appstream:DeleteImageBuilder
	appstream:DeleteImagePermissions
	appstream:DeleteStack
	appstream:DeleteUsageReportSubscription
	appstream:DeleteUser
	appstream:DescribeAppBlockBuilderAppBlockAssociations
	appstream:DescribeAppBlockBuilders
	appstream:DescribeAppBlocks
	appstream:DescribeApplicationFleetAssociations
	appstream:DescribeApplications
	appstream:DescribeDirectoryConfigs
	appstream:DescribeEntitlements
	appstream:DescribeFleets
	appstream:DescribeImageBuilders

Prefijo de servicio	Acciones
	appstream:DescribeImagePermissions
	appstream:DescribeImages
	appstream:DescribeSessions
	appstream:DescribeStacks
	appstream:DescribeUsageReportSubscriptions
	appstream:DescribeUsers
	appstream:DescribeUserStackAssociations
	appstream:DisableUser
	appstream:DisassociateAppBlockBuilderAppBlock
	appstream:DisassociateApplicationFleet
	appstream:DisassociateApplicationFromEntitlement
	appstream:DisassociateFleet
	appstream:EnableUser
	appstream:ExpireSession
	appstream:ListAssociatedFleets
	appstream:ListAssociatedStacks
	appstream:ListEntitledApplications
	appstream:StartAppBlockBuilder
	appstream:StartFleet
	appstream:StartImageBuilder
	appstream:StopAppBlockBuilder

Prefijo de servicio	Acciones
	appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack

Prefijo de servicio	Acciones
appsync	appsync:AssociateApi appsync:AssociateMergedGraphqlApi appsync:AssociateSourceGraphqlApi appsync:CreateApiCache appsync:CreateApiKey appsync:CreateDataSource appsync:CreateDomainName appsync:CreateFunction appsync:CreateGraphqlApi appsync:CreateResolver appsync:CreateType appsync:DeleteApiCache appsync:DeleteApiKey appsync:DeleteDataSource appsync:DeleteDomainName appsync:DeleteFunction appsync:DeleteGraphqlApi appsync:DeleteResolver appsync:DeleteType appsync:DisassociateApi appsync:DisassociateMergedGraphqlApi

Prefijo de servicio	Acciones
	appsync:DisassociateSourceGraphqlApi appsync:EvaluateCode appsync:EvaluateMappingTemplate appsync:FlushApiCache appsync:GetApiAssociation appsync:GetApiCache appsync:GetDataSource appsync:GetDomainName appsync:GetFunction appsync:GetGraphqlApi appsync:GetIntrospectionSchema appsync:GetResolver appsync:GetSchemaCreationStatus appsync:GetSourceApiAssociation appsync:GetType appsync:ListApiKeys appsync:ListDataSources appsync:ListDomainNames appsync:ListFunctions appsync:ListGraphqlApis appsync:ListResolvers

Prefijo de servicio	Acciones
	appsync:ListResolversByFunction appsync:ListSourceApiAssociations appsync:ListTypes appsync:ListTypesByAssociation appsync:StartSchemaCreation appsync:StartSchemaMerge appsync:UpdateApiCache appsync:UpdateApiKey appsync:UpdateDataSource appsync:UpdateDomainName appsync:UpdateFunction appsync:UpdateGraphqlApi appsync:UpdateResolver appsync:UpdateSourceApiAssociation appsync:UpdateType

Prefijo de servicio	Acciones
aps	aps:CreateAlertManagerDefinition aps:CreateLoggingConfiguration aps:CreateRuleGroupsNamespace aps:CreateWorkspace aps:DeleteAlertManagerDefinition aps:DeleteLoggingConfiguration aps:DeleteRuleGroupsNamespace aps:DeleteWorkspace aps:DescribeAlertManagerDefinition aps:DescribeLoggingConfiguration aps:DescribeRuleGroupsNamespace aps:DescribeWorkspace aps:ListRuleGroupsNamespaces aps:ListWorkspaces aps:PutAlertManagerDefinition aps:PutRuleGroupsNamespace aps:UpdateLoggingConfiguration aps:UpdateWorkspaceAlias

Prefijo de servicio	Acciones
athena	athena:BatchGetNamedQuery athena:BatchGetPreparedStatement athena:BatchGetQueryExecution athena:CancelCapacityReservation athena:CreateCapacityReservation athena:CreateDataCatalog athena:CreateNamedQuery athena:CreateNotebook athena:CreatePreparedStatement athena:CreatePresignedNotebookUrl athena:CreateWorkGroup athena:DeleteCapacityReservation athena:DeleteDataCatalog athena:DeleteNamedQuery athena:DeleteNotebook athena:DeletePreparedStatement athena:DeleteWorkGroup athena:ExportNotebook athena:GetCalculationExecution athena:GetCalculationExecutionCode athena:GetCalculationExecutionStatus

Prefijo de servicio	Acciones
	athena:GetCapacityAssignmentConfiguration athena:GetCapacityReservation athena:GetDatabase athena:GetDataCatalog athena:GetNamedQuery athena:GetNotebookMetadata athena:GetPreparedStatement athena:GetQueryExecution athena:GetQueryResults athena:GetQueryResultsStream athena:GetQueryRuntimeStatistics athena:GetSession athena:GetSessionStatus athena:GetTableMetadata athena:GetWorkGroup athena:ImportNotebook athena:ListApplicationDPUSizes athena:ListCalculationExecutions athena:ListCapacityReservations athena:ListDatabases athena:ListDataCatalogs

Prefijo de servicio	Acciones
	athena:ListEngineVersions athena:ListExecutors athena:ListNamedQueries athena:ListNotebookMetadata athena:ListNotebookSessions athena:ListPreparedStatements athena:ListQueryExecutions athena:ListSessions athena:ListTableMetadata athena:ListWorkGroups athena:PutCapacityAssignmentConfiguration athena:StartCalculationExecution athena:StartQueryExecution athena:StartSession athena:StopCalculationExecution athena:StopQueryExecution athena:TerminateSession athena:UpdateCapacityReservation athena:UpdateDataCatalog athena:UpdateNamedQuery athena:UpdateNotebook

Prefijo de servicio	Acciones
	athena:UpdateNotebookMetadata athena:UpdatePreparedStatement athena:UpdateWorkGroup

Prefijo de servicio	Acciones
auditmanager	auditmanager:AssociateAssessmentReportEvidenceFolder
	auditmanager:BatchAssociateAssessmentReportEvidence
	auditmanager:BatchCreateDelegationByAssessment
	auditmanager:BatchDeleteDelegationByAssessment
	auditmanager:BatchDisassociateAssessmentReportEvidence
	auditmanager:BatchImportEvidenceToAssessmentControl
	auditmanager:CreateAssessment
	auditmanager:CreateAssessmentFramework
	auditmanager:CreateAssessmentReport
	auditmanager:CreateControl
	auditmanager:DeleteAssessment
	auditmanager:DeleteAssessmentFramework
	auditmanager:DeleteAssessmentFrameworkShare
	auditmanager:DeleteAssessmentReport
	auditmanager:DeleteControl
	auditmanager:DeregisterAccount
	auditmanager:DeregisterOrganizationAdminAccount
	auditmanager:DisassociateAssessmentReportEvidenceFolder
	auditmanager:GetAccountStatus
	auditmanager:GetAssessment
	auditmanager:GetAssessmentFramework

Prefijo de servicio	Acciones
	auditmanager:GetAssessmentReportUrl
	auditmanager:GetChangeLogs
	auditmanager:GetControl
	auditmanager:GetDelegations
	auditmanager:GetEvidence
	auditmanager:GetEvidenceByEvidenceFolder
	auditmanager:GetEvidenceFileUploadUrl
	auditmanager:GetEvidenceFolder
	auditmanager:GetEvidenceFoldersByAssessment
	auditmanager:GetEvidenceFoldersByAssessmentControl
	auditmanager:GetInsights
	auditmanager:GetInsightsByAssessment
	auditmanager:GetOrganizationAdminAccount
	auditmanager:GetServicesInScope
	auditmanager:GetSettings
	auditmanager:ListAssessmentControlInsightsByControlDomain
	auditmanager:ListAssessmentFrameworks
	auditmanager:ListAssessmentFrameworkShareRequests
	auditmanager:ListAssessmentReports
	auditmanager:ListAssessments
	auditmanager:ListControlDomainInsights

Prefijo de servicio	Acciones
	auditmanager:ListControlDomainInsightsByAssessment
	auditmanager:ListControlInsightsByControlDomain
	auditmanager:ListControls
	auditmanager:ListKeywordsForDataSource
	auditmanager:ListNotifications
	auditmanager:RegisterAccount
	auditmanager:RegisterOrganizationAdminAccount
	auditmanager:StartAssessmentFrameworkShare
	auditmanager:UpdateAssessment
	auditmanager:UpdateAssessmentControl
	auditmanager:UpdateAssessmentControlSetStatus
	auditmanager:UpdateAssessmentFramework
	auditmanager:UpdateAssessmentFrameworkShare
	auditmanager:UpdateAssessmentStatus
	auditmanager:UpdateControl
	auditmanager:UpdateSettings
	auditmanager:ValidateAssessmentReportIntegrity

Prefijo de servicio	Acciones
autoscaling	autoscaling:AttachInstances autoscaling:AttachLoadBalancers autoscaling:AttachLoadBalancerTargetGroups autoscaling:AttachTrafficSources autoscaling:BatchDeleteScheduledAction autoscaling:BatchPutScheduledUpdateGroupAction autoscaling:CancelInstanceRefresh autoscaling:CompleteLifecycleAction autoscaling:CreateAutoScalingGroup autoscaling:CreateLaunchConfiguration autoscaling:DeleteAutoScalingGroup autoscaling:DeleteLaunchConfiguration autoscaling:DeleteLifecycleHook autoscaling:DeleteNotificationConfiguration autoscaling:DeletePolicy autoscaling:DeleteScheduledAction autoscaling:DeleteWarmPool autoscaling:DescribeAccountLimits autoscaling:DescribeAdjustmentTypes autoscaling:DescribeAutoScalingGroups autoscaling:DescribeAutoScalingInstances

Prefijo de servicio	Acciones
	autoscaling:DescribeAutoScalingNotificationTypes autoscaling:DescribeInstanceRefreshes autoscaling:DescribeLaunchConfigurations autoscaling:DescribeLifecycleHooks autoscaling:DescribeLifecycleHookTypes autoscaling:DescribeLoadBalancers autoscaling:DescribeLoadBalancerTargetGroups autoscaling:DescribeMetricCollectionTypes autoscaling:DescribeNotificationConfigurations autoscaling:DescribePolicies autoscaling:DescribeScalingActivities autoscaling:DescribeScalingProcessTypes autoscaling:DescribeScheduledActions autoscaling:DescribeTerminationPolicyTypes autoscaling:DescribeTrafficSources autoscaling:DescribeWarmPool autoscaling:DetachInstances autoscaling:DetachLoadBalancers autoscaling:DetachLoadBalancerTargetGroups autoscaling:DetachTrafficSources autoscaling:DisableMetricsCollection

Prefijo de servicio	Acciones
	autoscaling:EnableMetricsCollection autoscaling:EnterStandby autoscaling:ExecutePolicy autoscaling:ExitStandby autoscaling:GetPredictiveScalingForecast autoscaling:PutLifecycleHook autoscaling:PutNotificationConfiguration autoscaling:PutScalingPolicy autoscaling:PutScheduledUpdateGroupAction autoscaling:PutWarmPool autoscaling:RecordLifecycleActionHeartbeat autoscaling:ResumeProcesses autoscaling:RollbackInstanceRefresh autoscaling:SetDesiredCapacity autoscaling:SetInstanceHealth autoscaling:SetInstanceProtection autoscaling:StartInstanceRefresh autoscaling:SuspendProcesses autoscaling:TerminateInstanceInAutoScalingGroup autoscaling:UpdateAutoScalingGroup
aws-marketplace	aws-marketplace:GetEntitlements

Prefijo de servicio	Acciones
backup	backup:CancelLegalHold backup:CreateBackupPlan backup:CreateBackupSelection backup:CreateBackupVault backup:CreateFramework backup:CreateLegalHold backup:CreateLogicallyAirGappedBackupVault backup:CreateReportPlan backup:DeleteBackupPlan backup:DeleteBackupSelection backup:DeleteBackupVault backup:DeleteBackupVaultAccessPolicy backup:DeleteBackupVaultLockConfiguration backup:DeleteBackupVaultNotifications backup:DeleteFramework backup:DeleteRecoveryPoint backup:DeleteReportPlan backup:DescribeBackupJob backup:DescribeBackupVault backup:DescribeCopyJob backup:DescribeFramework

Prefijo de servicio	Acciones
	backup:DescribeGlobalSettings backup:DescribeProtectedResource backup:DescribeRecoveryPoint backup:DescribeRegionSettings backup:DescribeReportJob backup:DescribeReportPlan backup:DescribeRestoreJob backup:DisassociateRecoveryPoint backup:DisassociateRecoveryPointFromParent backup:ExportBackupPlanTemplate backup:GetBackupPlan backup:GetBackupPlanFromJSON backup:GetBackupPlanFromTemplate backup:GetBackupSelection backup:GetBackupVaultAccessPolicy backup:GetBackupVaultNotifications backup:GetLegalHold backup:GetRecoveryPointRestoreMetadata backup:GetSupportedResourceTypes backup:ListBackupJobs backup:ListBackupPlans

Prefijo de servicio	Acciones
	backup:ListBackupPlanTemplates backup:ListBackupPlanVersions backup:ListBackupSelections backup:ListBackupVaults backup:ListCopyJobs backup:ListFrameworks backup:ListLegalHolds backup:ListProtectedResources backup:ListRecoveryPointsByBackupVault backup:ListRecoveryPointsByLegalHold backup:ListRecoveryPointsByResource backup:ListReportJobs backup:ListReportPlans backup:ListRestoreJobs backup:PutBackupVaultAccessPolicy backup:PutBackupVaultLockConfiguration backup:PutBackupVaultNotifications backup:StartBackupJob backup:StartCopyJob backup:StartReportJob backup:StartRestoreJob

Prefijo de servicio	Acciones
	backup:StopBackupJob backup:UpdateBackupPlan backup:UpdateFramework backup:UpdateGlobalSettings backup:UpdateRecoveryPointLifecycle backup:UpdateRegionSettings backup:UpdateReportPlan

Prefijo de servicio	Acciones
lote	batch:CancelJob batch:CreateComputeEnvironment batch:CreateJobQueue batch:CreateSchedulingPolicy batch:DeleteComputeEnvironment batch:DeleteJobQueue batch:DeleteSchedulingPolicy batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch:DescribeSchedulingPolicies batch:ListJobs batch:ListSchedulingPolicies batch:RegisterJobDefinition batch:SubmitJob batch:TerminateJob batch:UpdateComputeEnvironment batch:UpdateJobQueue batch:UpdateSchedulingPolicy

Prefijo de servicio	Acciones
braket	braket:CancelJob braket:CancelQuantumTask braket:CreateJob braket:CreateQuantumTask braket:GetDevice braket:GetJob braket:GetQuantumTask braket:SearchDevices braket:SearchJobs braket:SearchQuantumTasks

Prefijo de servicio	Acciones
budgets	budgets:ModifyBudget
	budgets:CreateBudgetAction
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:DeleteBudgetAction
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:ViewBudget
	budgets:DescribeBudgetAction
	budgets:DescribeBudgetActionHistories
	budgets:DescribeBudgetActionsForAccount
	budgets:DescribeBudgetActionsForBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ExecuteBudgetAction
	budgets:ModifyBudget
	budgets:UpdateBudgetAction

Prefijo de servicio	Acciones
	budgets:ModifyBudget budgets:ModifyBudget
cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironments cloud9:DescribeEnvironmentStatus cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

Prefijo de servicio	Acciones
cloudformation	cloudformation:BatchDescribeTypeConfigurations
	cloudformation:CancelUpdateStack
	cloudformation:ContinueUpdateRollback
	cloudformation:CreateChangeSet
	cloudformation:CreateStack
	cloudformation:CreateStackInstances
	cloudformation:CreateStackSet
	cloudformation:DeactivateType
	cloudformation:DeleteChangeSet
	cloudformation:DeleteStack
	cloudformation:DeleteStackInstances
	cloudformation:DeleteStackSet
	cloudformation:DeregisterType
	cloudformation:DescribeAccountLimits
	cloudformation:DescribeChangeSet
	cloudformation:DescribeChangeSetHooks
	cloudformation:DescribeOrganizationsAccess
	cloudformation:DescribePublisher
	cloudformation:DescribeStackDriftDetectionStatus
	cloudformation:DescribeStackEvents
	cloudformation:DescribeStackInstance

Prefijo de servicio	Acciones
	cloudformation:DescribeStackResource
	cloudformation:DescribeStackResourceDrifts
	cloudformation:DescribeStackResources
	cloudformation:DescribeStacks
	cloudformation:DescribeStackSet
	cloudformation:DescribeStackSetOperation
	cloudformation:DescribeType
	cloudformation:DescribeTypeRegistration
	cloudformation:DetectStackDrift
	cloudformation:DetectStackResourceDrift
	cloudformation:DetectStackSetDrift
	cloudformation:EstimateTemplateCost
	cloudformation:ExecuteChangeSet
	cloudformation:GetStackPolicy
	cloudformation:GetTemplate
	cloudformation:GetTemplateSummary
	cloudformation:ImportStacksToStackSet
	cloudformation:ListChangeSets
	cloudformation:ListExports
	cloudformation:ListImports
	cloudformation:ListStackInstanceResourceDrifts

Prefijo de servicio	Acciones
	cloudformation:ListStackInstances
	cloudformation:ListStackResources
	cloudformation:ListStackSetOperationResults
	cloudformation:ListStackSetOperations
	cloudformation:ListStackSets
	cloudformation:ListTypeRegistrations
	cloudformation:ListTypes
	cloudformation:ListTypeVersions
	cloudformation:PublishType
	cloudformation:RecordHandlerProgress
	cloudformation:RegisterPublisher
	cloudformation:RegisterType
	cloudformation:RollbackStack
	cloudformation:SetStackPolicy
	cloudformation:SetTypeConfiguration
	cloudformation:SetTypeDefaultVersion
	cloudformation:SignalResource
	cloudformation:StopStackSetOperation
	cloudformation:TestType
	cloudformation:UpdateStack
	cloudformation:UpdateStackInstances

Prefijo de servicio	Acciones
	cloudformation:UpdateStackSet cloudformation:UpdateTerminationProtection cloudformation:ValidateTemplate

Prefijo de servicio	Acciones
cloudfront	cloudfront:AssociateAlias cloudfront:CreateCachePolicy cloudfront:CreateCloudFrontOriginAccessIdentity cloudfront:CreateContinuousDeploymentPolicy cloudfront:CreateFieldLevelEncryptionConfig cloudfront:CreateFieldLevelEncryptionProfile cloudfront:CreateFunction cloudfront:CreateInvalidation cloudfront:CreateKeyGroup cloudfront:CreateMonitoringSubscription cloudfront:CreateOriginAccessControl cloudfront:CreateOriginRequestPolicy cloudfront:CreatePublicKey cloudfront:CreateRealtimeLogConfig cloudfront:CreateResponseHeadersPolicy cloudfront:DeleteCachePolicy cloudfront:DeleteCloudFrontOriginAccessIdentity cloudfront:DeleteContinuousDeploymentPolicy cloudfront:DeleteDistribution cloudfront:DeleteFieldLevelEncryptionConfig cloudfront:DeleteFieldLevelEncryptionProfile

Prefijo de servicio	Acciones
	cloudfront:DeleteFunction cloudfront:DeleteKeyGroup cloudfront:DeleteMonitoringSubscription cloudfront:DeleteOriginAccessControl cloudfront:DeleteOriginRequestPolicy cloudfront:DeletePublicKey cloudfront:DeleteRealtimeLogConfig cloudfront:DeleteResponseHeadersPolicy cloudfront:DeleteStreamingDistribution cloudfront:DescribeFunction cloudfront:GetCachePolicy cloudfront:GetCachePolicyConfig cloudfront:GetCloudFrontOriginAccessIdentity cloudfront:GetCloudFrontOriginAccessIdentityConfig cloudfront:GetContinuousDeploymentPolicy cloudfront:GetContinuousDeploymentPolicyConfig cloudfront:GetDistributionConfig cloudfront:GetFieldLevelEncryption cloudfront:GetFieldLevelEncryptionConfig cloudfront:GetFieldLevelEncryptionProfile cloudfront:GetFieldLevelEncryptionProfileConfig

Prefijo de servicio	Acciones
	cloudfront:GetFunction cloudfront:GetInvalidation cloudfront:GetKeyGroup cloudfront:GetKeyGroupConfig cloudfront:GetMonitoringSubscription cloudfront:GetOriginAccessControl cloudfront:GetOriginAccessControlConfig cloudfront:GetOriginRequestPolicy cloudfront:GetOriginRequestPolicyConfig cloudfront:GetPublicKey cloudfront:GetPublicKeyConfig cloudfront:GetRealtimeLogConfig cloudfront:GetResponseHeadersPolicy cloudfront:GetResponseHeadersPolicyConfig cloudfront:GetStreamingDistribution cloudfront:GetStreamingDistributionConfig cloudfront:ListCachePolicies cloudfront:ListCloudFrontOriginAccessIdentities cloudfront:ListConflictingAliases cloudfront:ListContinuousDeploymentPolicies cloudfront:ListDistributions

Prefijo de servicio	Acciones
	cloudfront:ListDistributionsByCachePolicyId cloudfront:ListDistributionsByKeyGroup cloudfront:ListDistributionsByOriginRequestPolicyId cloudfront:ListDistributionsByRealtimeLogConfig cloudfront:ListDistributionsByResponseHeadersPolicyId cloudfront:ListDistributionsByWebACLId cloudfront:ListFieldLevelEncryptionConfigs cloudfront:ListFieldLevelEncryptionProfiles cloudfront:ListFunctions cloudfront:ListInvalidations cloudfront:ListKeyGroups cloudfront:ListOriginAccessControls cloudfront:ListOriginRequestPolicies cloudfront:ListPublicKeys cloudfront:ListRealtimeLogConfigs cloudfront:ListResponseHeadersPolicies cloudfront:ListStreamingDistributions cloudfront:PublishFunction cloudfront:TestFunction cloudfront:UpdateCachePolicy cloudfront:UpdateCloudFrontOriginAccessIdentity

Prefijo de servicio	Acciones
	cloudfront:UpdateContinuousDeploymentPolicy cloudfront:UpdateDistribution cloudfront:UpdateFieldLevelEncryptionConfig cloudfront:UpdateFieldLevelEncryptionProfile cloudfront:UpdateFunction cloudfront:UpdateKeyGroup cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy

Prefijo de servicio	Acciones
cloudhsm	cloudhsm:CreateHapg
	cloudhsm:CreateLunaClient
	cloudhsm:DeleteBackup
	cloudhsm:DeleteHapg
	cloudhsm:DeleteHsm
	cloudhsm:DeleteLunaClient
	cloudhsm:DescribeBackups
	cloudhsm:DescribeClusters
	cloudhsm:DescribeHapg
	cloudhsm:DescribeHsm
	cloudhsm:DescribeLunaClient
	cloudhsm:GetConfig
	cloudhsm:InitializeCluster
	cloudhsm:ListAvailableZones
	cloudhsm:ListHapgs
	cloudhsm:ListHsms
	cloudhsm:ListLunaClients
	cloudhsm:ModifyBackupAttributes
	cloudhsm:ModifyCluster
	cloudhsm:ModifyHapg
	cloudhsm:ModifyLunaClient

Prefijo de servicio	Acciones
	cloudhsm:RestoreBackup

Prefijo de servicio	Acciones
cloudsearch	cloudsearch:BuildSuggesters cloudsearch:CreateDomain cloudsearch:DefineAnalysisScheme cloudsearch:DefineExpression cloudsearch:DefineIndexField cloudsearch:DefineSuggester cloudsearch:DeleteAnalysisScheme cloudsearch:DeleteDomain cloudsearch:DeleteExpression cloudsearch:DeleteIndexField cloudsearch:DeleteSuggester cloudsearch:DescribeAnalysisSchemes cloudsearch:DescribeAvailabilityOptions cloudsearch:DescribeDomainEndpointOptions cloudsearch:DescribeDomains cloudsearch:DescribeExpressions cloudsearch:DescribeIndexFields cloudsearch:DescribeScalingParameters cloudsearch:DescribeServiceAccessPolicies cloudsearch:DescribeSuggesters cloudsearch:IndexDocuments

Prefijo de servicio	Acciones
	cloudsearch:ListDomainNames cloudsearch:UpdateAvailabilityOptions cloudsearch:UpdateDomainEndpointOptions cloudsearch:UpdateScalingParameters cloudsearch:UpdateServiceAccessPolicies

Prefijo de servicio	Acciones
cloudtrail	cloudtrail:CancelQuery
	cloudtrail:CreateChannel
	cloudtrail:CreateEventDataStore
	cloudtrail:CreateTrail
	cloudtrail:DeleteChannel
	cloudtrail:DeleteEventDataStore
	cloudtrail:DeleteResourcePolicy
	cloudtrail:DeleteTrail
	cloudtrail:DeregisterOrganizationDelegatedAdmin
	cloudtrail:DescribeQuery
	cloudtrail:DescribeTrails
	cloudtrail:GetChannel
	cloudtrail:GetEventDataStore
	cloudtrail:GetEventSelectors
	cloudtrail:GetImport
	cloudtrail:GetInsightSelectors
	cloudtrail:GetQueryResults
	cloudtrail:GetResourcePolicy
	cloudtrail:GetTrail
	cloudtrail:GetTrailStatus
	cloudtrail:ListChannels

Prefijo de servicio	Acciones
	cloudtrail:ListEventDataStores
	cloudtrail:ListImportFailures
	cloudtrail:ListImports
	cloudtrail:ListPublicKeys
	cloudtrail:ListQueries
	cloudtrail:ListTrails
	cloudtrail:LookupEvents
	cloudtrail:PutEventSelectors
	cloudtrail:PutInsightSelectors
	cloudtrail:PutResourcePolicy
	cloudtrail:RegisterOrganizationDelegatedAdmin
	cloudtrail:RestoreEventDataStore
	cloudtrail:StartEventDataStoreIngestion
	cloudtrail:StartImport
	cloudtrail:StartLogging
	cloudtrail:StartQuery
	cloudtrail:StopEventDataStoreIngestion
	cloudtrail:StopImport
	cloudtrail:StopLogging
	cloudtrail:UpdateChannel
	cloudtrail:UpdateEventDataStore

Prefijo de servicio	Acciones
	cloudtrail:UpdateTrail

Prefijo de servicio	Acciones
cloudwatch	cloudwatch:DeleteAlarms cloudwatch:DeleteAnomalyDetector cloudwatch:DeleteDashboards cloudwatch:DeleteInsightRules cloudwatch:DeleteMetricStream cloudwatch:DescribeAlarmHistory cloudwatch:DescribeAlarms cloudwatch:DescribeAlarmsForMetric cloudwatch:DescribeAnomalyDetectors cloudwatch:DescribeInsightRules cloudwatch:DisableAlarmActions cloudwatch:DisableInsightRules cloudwatch:EnableAlarmActions cloudwatch:EnableInsightRules cloudwatch:GetDashboard cloudwatch:GetInsightRuleReport cloudwatch:GetMetricStream cloudwatch:ListDashboards cloudwatch:ListManagedInsightRules cloudwatch:ListMetricStreams cloudwatch:PutAnomalyDetector

Prefijo de servicio	Acciones
	cloudwatch:PutCompositeAlarm cloudwatch:PutDashboard cloudwatch:PutInsightRule cloudwatch:PutManagedInsightRules cloudwatch:PutMetricAlarm cloudwatch:PutMetricStream cloudwatch:SetAlarmState cloudwatch:StartMetricStreams cloudwatch:StopMetricStreams

Prefijo de servicio	Acciones
codeartifact	codeartifact:AssociateExternalConnection
	codeartifact:CopyPackageVersions
	codeartifact:CreateDomain
	codeartifact:CreateRepository
	codeartifact:DeleteDomain
	codeartifact:DeleteDomainPermissionsPolicy
	codeartifact:DeletePackage
	codeartifact:DeletePackageVersions
	codeartifact:DeleteRepository
	codeartifact:DeleteRepositoryPermissionsPolicy
	codeartifact:DescribeDomain
	codeartifact:DescribePackage
	codeartifact:DescribePackageVersion
	codeartifact:DescribeRepository
	codeartifact:DisassociateExternalConnection
	codeartifact:DisposePackageVersions
	codeartifact:GetAuthorizationToken
	codeartifact:GetDomainPermissionsPolicy
	codeartifact:GetPackageVersionAsset
	codeartifact:GetPackageVersionReadme
	codeartifact:GetRepositoryEndpoint

Prefijo de servicio	Acciones
	codeartifact:GetRepositoryPermissionsPolicy
	codeartifact:ListDomains
	codeartifact:ListPackages
	codeartifact:ListPackageVersionAssets
	codeartifact:ListPackageVersionDependencies
	codeartifact:ListPackageVersions
	codeartifact:ListRepositories
	codeartifact:ListRepositoriesInDomain
	codeartifact:PublishPackageVersion
	codeartifact:PutDomainPermissionsPolicy
	codeartifact:PutPackageMetadata
	codeartifact:PutPackageOriginConfiguration
	codeartifact:PutRepositoryPermissionsPolicy
	codeartifact:ReadFromRepository
	codeartifact:UpdatePackageVersionsStatus
	codeartifact:UpdateRepository

Prefijo de servicio	Acciones
codedeploy	codedeploy:BatchGetApplicationRevisions codedeploy:BatchGetApplications codedeploy:BatchGetDeploymentGroups codedeploy:BatchGetDeploymentInstances codedeploy:BatchGetDeployments codedeploy:BatchGetDeploymentTargets codedeploy:BatchGetOnPremisesInstances codedeploy:ContinueDeployment codedeploy:CreateApplication codedeploy:CreateDeployment codedeploy:CreateDeploymentConfig codedeploy:CreateDeploymentGroup codedeploy:DeleteApplication codedeploy:DeleteDeploymentConfig codedeploy:DeleteDeploymentGroup codedeploy:DeleteGitHubAccountToken codedeploy:DeleteResourcesByExternalId codedeploy:DeregisterOnPremisesInstance codedeploy:GetApplication codedeploy:GetApplicationRevision codedeploy:GetDeployment

Prefijo de servicio	Acciones
	codedeploy:GetDeploymentConfig codedeploy:GetDeploymentGroup codedeploy:GetDeploymentInstance codedeploy:GetDeploymentTarget codedeploy:GetOnPremisesInstance codedeploy:ListApplicationRevisions codedeploy:ListApplications codedeploy:ListDeploymentConfigs codedeploy:ListDeploymentGroups codedeploy:ListDeploymentInstances codedeploy:ListDeployments codedeploy:ListDeploymentTargets codedeploy:ListGitHubAccountTokenNames codedeploy:ListOnPremisesInstances codedeploy:PutLifecycleEventHookExecutionStatus codedeploy:RegisterApplicationRevision codedeploy:RegisterOnPremisesInstance codedeploy:SkipWaitTimeForInstanceTermination codedeploy:StopDeployment codedeploy:UpdateApplication codedeploy:UpdateDeploymentGroup

Prefijo de servicio	Acciones
codeguru-profiler	codeguru-profiler:AddNotificationChannels
	codeguru-profiler:BatchGetFrameMetricData
	codeguru-profiler:ConfigureAgent
	codeguru-profiler:CreateProfilingGroup
	codeguru-profiler:DeleteProfilingGroup
	codeguru-profiler:DescribeProfilingGroup
	codeguru-profiler:GetFindingsReportAccountSummary
	codeguru-profiler:GetNotificationConfiguration
	codeguru-profiler:GetPolicy
	codeguru-profiler:GetProfile
	codeguru-profiler:GetRecommendations
	codeguru-profiler:ListFindingsReports
	codeguru-profiler:ListProfileTimes
	codeguru-profiler:ListProfilingGroups
	codeguru-profiler:PutPermission
	codeguru-profiler:RemoveNotificationChannel
	codeguru-profiler:RemovePermission
	codeguru-profiler:SubmitFeedback
	codeguru-profiler:UpdateProfilingGroup

Prefijo de servicio	Acciones
codeguru-reviewer	codeguru-reviewer:AssociateRepository codeguru-reviewer:CreateCodeReview codeguru-reviewer:DescribeCodeReview codeguru-reviewer:DescribeRecommendationFeedback codeguru-reviewer:DescribeRepositoryAssociation codeguru-reviewer:DisassociateRepository codeguru-reviewer:ListCodeReviews codeguru-reviewer:ListRecommendationFeedback codeguru-reviewer:ListRecommendations codeguru-reviewer:ListRepositoryAssociations codeguru-reviewer:PutRecommendationFeedback

Prefijo de servicio	Acciones
codepipeline	codepipeline:AcknowledgeJob
	codepipeline:AcknowledgeThirdPartyJob
	codepipeline:CreateCustomActionType
	codepipeline:CreatePipeline
	codepipeline:DeleteCustomActionType
	codepipeline:DeletePipeline
	codepipeline:DeleteWebhook
	codepipeline:DeregisterWebhookWithThirdParty
	codepipeline:GetActionType
	codepipeline:GetJobDetails
	codepipeline:GetPipeline
	codepipeline:GetPipelineExecution
	codepipeline:GetPipelineState
	codepipeline:GetThirdPartyJobDetails
	codepipeline:ListActionExecutions
	codepipeline:ListActionTypes
	codepipeline:ListPipelineExecutions
	codepipeline:ListPipelines
	codepipeline:ListWebhooks
	codepipeline:PollForJobs
	codepipeline:PollForThirdPartyJobs

Prefijo de servicio	Acciones
	codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

Prefijo de servicio	Acciones
codestar	codestar:AssociateTeamMember codestar:CreateProject codestar:CreateUserProfile codestar:DeleteProject codestar:DeleteUserProfile codestar:DescribeProject codestar:DescribeUserProfile codestar:DisassociateTeamMember codestar:ListProjects codestar:ListResources codestar:ListTeamMembers codestar:ListUserProfiles codestar:UpdateProject codestar:UpdateTeamMember codestar:UpdateUserProfile

Prefijo de servicio	Acciones
codestar-notifications	codestar-notifications:CreateNotificationRule
	codestar-notifications:DeleteNotificationRule
	codestar-notifications:DeleteTarget
	codestar-notifications:DescribeNotificationRule
	codestar-notifications:ListEventTypes
	codestar-notifications:ListNotificationRules
	codestar-notifications:ListTargets
	codestar-notifications:Subscribe
	codestar-notifications:Unsubscribe
	codestar-notifications:UpdateNotificationRule

Prefijo de servicio	Acciones
cognito-identity	cognito-identity:CreateIdentityPool cognito-identity:DeleteIdentities cognito-identity:DeleteIdentityPool cognito-identity:DescribeIdentity cognito-identity:DescribeIdentityPool cognito-identity:GetIdentityPoolRoles cognito-identity:ListIdentities cognito-identity:ListIdentityPools cognito-identity:LookupDeveloperIdentity cognito-identity:MergeDeveloperIdentities cognito-identity:SetIdentityPoolRoles cognito-identity:UnlinkDeveloperIdentity cognito-identity:UpdateIdentityPool

Prefijo de servicio	Acciones
cognito-idp	cognito-idp:AddCustomAttributes
	cognito-idp:AdminAddUserToGroup
	cognito-idp:AdminConfirmSignUp
	cognito-idp:AdminCreateUser
	cognito-idp:AdminDeleteUser
	cognito-idp:AdminDeleteUserAttributes
	cognito-idp:AdminDisableProviderForUser
	cognito-idp:AdminDisableUser
	cognito-idp:AdminEnableUser
	cognito-idp:AdminForgetDevice
	cognito-idp:AdminGetDevice
	cognito-idp:AdminGetUser
	cognito-idp:AdminInitiateAuth
	cognito-idp:AdminLinkProviderForUser
	cognito-idp:AdminListDevices
	cognito-idp:AdminListGroupsForUser
	cognito-idp:AdminListUserAuthEvents
	cognito-idp:AdminRemoveUserFromGroup
	cognito-idp:AdminResetUserPassword
	cognito-idp:AdminRespondToAuthChallenge
	cognito-idp:AdminSetUserMFAPreference

Prefijo de servicio	Acciones
	cognito-idp:AdminSetUserPassword
	cognito-idp:AdminSetUserSettings
	cognito-idp:AdminUpdateAuthEventFeedback
	cognito-idp:AdminUpdateDeviceStatus
	cognito-idp:AdminUpdateUserAttributes
	cognito-idp:AdminUserGlobalSignOut
	cognito-idp:AssociateSoftwareToken
	cognito-idp:ChangePassword
	cognito-idp:ConfirmDevice
	cognito-idp:ConfirmForgotPassword
	cognito-idp:ConfirmSignUp
	cognito-idp:CreateGroup
	cognito-idp:CreateIdentityProvider
	cognito-idp:CreateResourceServer
	cognito-idp:CreateUserImportJob
	cognito-idp:CreateUserPool
	cognito-idp:CreateUserPoolClient
	cognito-idp:CreateUserPoolDomain
	cognito-idp:DeleteGroup
	cognito-idp:DeleteIdentityProvider
	cognito-idp:DeleteResourceServer

Prefijo de servicio	Acciones
	cognito-idp:DeleteUser
	cognito-idp:DeleteUserAttributes
	cognito-idp:DeleteUserPool
	cognito-idp:DeleteUserPoolClient
	cognito-idp:DeleteUserPoolDomain
	cognito-idp:DescribeIdentityProvider
	cognito-idp:DescribeResourceServer
	cognito-idp:DescribeRiskConfiguration
	cognito-idp:DescribeUserImportJob
	cognito-idp:DescribeUserPool
	cognito-idp:DescribeUserPoolClient
	cognito-idp:DescribeUserPoolDomain
	cognito-idp:ForgetDevice
	cognito-idp:ForgotPassword
	cognito-idp:GetCSVHeader
	cognito-idp:GetDevice
	cognito-idp:GetGroup
	cognito-idp:GetIdentityProviderByIdentifier
	cognito-idp:GetLogDeliveryConfiguration
	cognito-idp:GetSigningCertificate
	cognito-idp:GetUICustomization

Prefijo de servicio	Acciones
	cognito-idp:GetUser
	cognito-idp:GetUserAttributeVerificationCode
	cognito-idp:GetUserPoolMfaConfig
	cognito-idp:GlobalSignOut
	cognito-idp:InitiateAuth
	cognito-idp:ListDevices
	cognito-idp:ListGroups
	cognito-idp:ListIdentityProviders
	cognito-idp:ListResourceServers
	cognito-idp:ListUserImportJobs
	cognito-idp:ListUserPoolClients
	cognito-idp:ListUserPools
	cognito-idp:ListUsers
	cognito-idp:ListUsersInGroup
	cognito-idp:ResendConfirmationCode
	cognito-idp:RespondToAuthChallenge
	cognito-idp:RevokeToken
	cognito-idp:SetLogDeliveryConfiguration
	cognito-idp:SetRiskConfiguration
	cognito-idp:SetUICustomization
	cognito-idp:SetUserMFAPreference

Prefijo de servicio	Acciones
	cognito-idp:SetUserPoolMfaConfig
	cognito-idp:SetUserSettings
	cognito-idp:SignUp
	cognito-idp:StartUserImportJob
	cognito-idp:StopUserImportJob
	cognito-idp:UpdateAuthEventFeedback
	cognito-idp:UpdateDeviceStatus
	cognito-idp:UpdateGroup
	cognito-idp:UpdateIdentityProvider
	cognito-idp:UpdateResourceServer
	cognito-idp:UpdateUserAttributes
	cognito-idp:UpdateUserPool
	cognito-idp:UpdateUserPoolClient
	cognito-idp:UpdateUserPoolDomain
	cognito-idp:VerifySoftwareToken
	cognito-idp:VerifyUserAttribute

Prefijo de servicio	Acciones
cognito-sync	cognito-sync:BulkPublish cognito-sync:DeleteDataset cognito-sync:DescribeDataset cognito-sync:DescribeIdentityPoolUsage cognito-sync:DescribeIdentityUsage cognito-sync:GetBulkPublishDetails cognito-sync:GetCognitoEvents cognito-sync:GetIdentityPoolConfiguration cognito-sync:ListDatasets cognito-sync:ListIdentityPoolUsage cognito-sync:ListRecords cognito-sync:RegisterDevice cognito-sync:SetCognitoEvents cognito-sync:SetIdentityPoolConfiguration cognito-sync:SubscribeToDataset cognito-sync:UnsubscribeFromDataset cognito-sync:UpdateRecords

Prefijo de servicio	Acciones
comprehendmedical	comprehendmedical:DescribeEntitiesDetectionV2Job
	comprehendmedical:DescribeICD10CMInferenceJob
	comprehendmedical:DescribePHIDetectionJob
	comprehendmedical:DescribeRxNormInferenceJob
	comprehendmedical:DescribeSNOMEDCTInferenceJob
	comprehendmedical:DetectEntitiesV2
	comprehendmedical:DetectPHI
	comprehendmedical:InferICD10CM
	comprehendmedical:InferRxNorm
	comprehendmedical:InferSNOMEDCT
	comprehendmedical:ListEntitiesDetectionV2Jobs
	comprehendmedical:ListICD10CMInferenceJobs
	comprehendmedical:ListPHIDetectionJobs
	comprehendmedical:ListRxNormInferenceJobs
	comprehendmedical:ListSNOMEDCTInferenceJobs
	comprehendmedical:StartEntitiesDetectionV2Job
	comprehendmedical:StartICD10CMInferenceJob
	comprehendmedical:StartPHIDetectionJob
	comprehendmedical:StartRxNormInferenceJob
	comprehendmedical:StartSNOMEDCTInferenceJob
	comprehendmedical:StopEntitiesDetectionV2Job

Prefijo de servicio	Acciones
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

Prefijo de servicio	Acciones
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences compute-optimizer:DescribeRecommendationExportJobs compute-optimizer:ExportAutoScalingGroupRecommendations compute-optimizer:ExportEBSVolumeRecommendations compute-optimizer:ExportEC2InstanceRecommendations compute-optimizer:ExportECSServiceRecommendations compute-optimizer:ExportLambdaFunctionRecommendations compute-optimizer:ExportLicenseRecommendations compute-optimizer:GetEC2RecommendationProjectedMetrics compute-optimizer:GetECSServiceRecommendationProjectedMetrics compute-optimizer:GetEffectiveRecommendationPreferences compute-optimizer:GetEnrollmentStatus compute-optimizer:GetEnrollmentStatusesForOrganization compute-optimizer:GetRecommendationPreferences compute-optimizer:GetRecommendationSummaries compute-optimizer:PutRecommendationPreferences compute-optimizer:UpdateEnrollmentStatus

Prefijo de servicio	Acciones
config	config:BatchGetResourceConfig
	config:DeleteAggregationAuthorization
	config:DeleteConfigRule
	config:DeleteConfigurationAggregator
	config:DeleteConfigurationRecorder
	config:DeleteConformancePack
	config:DeleteDeliveryChannel
	config:DeleteEvaluationResults
	config:DeleteOrganizationConfigRule
	config:DeleteOrganizationConformancePack
	config:DeletePendingAggregationRequest
	config:DeleteRemediationConfiguration
	config:DeleteRemediationExceptions
	config:DeleteResourceConfig
	config:DeleteRetentionConfiguration
	config:DeleteStoredQuery
	config:DeliverConfigSnapshot
	config:DescribeAggregateComplianceByConfigRules
	config:DescribeAggregateComplianceByConformancePacks
	config:DescribeAggregationAuthorizations
	config:DescribeComplianceByConfigRule

Prefijo de servicio	Acciones
	config:DescribeComplianceByResource
	config:DescribeConfigRuleEvaluationStatus
	config:DescribeConfigRules
	config:DescribeConfigurationAggregators
	config:DescribeConfigurationAggregatorSourcesStatus
	config:DescribeConfigurationRecorders
	config:DescribeConfigurationRecorderStatus
	config:DescribeConformancePackCompliance
	config:DescribeConformancePacks
	config:DescribeConformancePackStatus
	config:DescribeDeliveryChannels
	config:DescribeDeliveryChannelStatus
	config:DescribeOrganizationConfigRules
	config:DescribeOrganizationConfigRuleStatuses
	config:DescribeOrganizationConformancePacks
	config:DescribeOrganizationConformancePackStatuses
	config:DescribePendingAggregationRequests
	config:DescribeRemediationConfigurations
	config:DescribeRemediationExceptions
	config:DescribeRemediationExecutionStatus
	config:DescribeRetentionConfigurations

Prefijo de servicio	Acciones
	config:GetComplianceDetailsByConfigRule
	config:GetComplianceDetailsByResource
	config:GetComplianceSummaryByConfigRule
	config:GetComplianceSummaryByResourceType
	config:GetConformancePackComplianceDetails
	config:GetConformancePackComplianceSummary
	config:GetCustomRulePolicy
	config:GetDiscoveredResourceCounts
	config:GetOrganizationConfigRuleDetailedStatus
	config:GetOrganizationConformancePackDetailedStatus
	config:GetOrganizationCustomRulePolicy
	config:GetResourceConfigHistory
	config:GetResourceEvaluationSummary
	config:GetStoredQuery
	config:ListConformancePackComplianceScores
	config:ListDiscoveredResources
	config:ListResourceEvaluations
	config:ListStoredQueries
	config:PutConfigRule
	config:PutConfigurationAggregator
	config:PutConfigurationRecorder

Prefijo de servicio	Acciones
	config:PutConformancePack
	config:PutDeliveryChannel
	config:PutEvaluations
	config:PutExternalEvaluation
	config:PutOrganizationConfigRule
	config:PutOrganizationConformancePack
	config:PutRemediationConfigurations
	config:PutRemediationExceptions
	config:PutResourceConfig
	config:PutRetentionConfiguration
	config:PutStoredQuery
	config:SelectResourceConfig
	config:StartConfigRulesEvaluation
	config:StartConfigurationRecorder
	config:StartRemediationExecution
	config:StartResourceEvaluation
	config:StopConfigurationRecorder

Prefijo de servicio	Acciones
connect	connect:ActivateEvaluationForm connect:AssociateApprovedOrigin connect:AssociateBot connect:AssociateDefaultVocabulary connect:AssociateInstanceStorageConfig connect:AssociateLambdaFunction connect:AssociateLexBot connect:AssociatePhoneNumberContactFlow connect:AssociateQueueQuickConnects connect:AssociateRoutingProfileQueues connect:AssociateSecurityKey connect:ClaimPhoneNumber connect:CreateAgentStatus connect:CreateContactFlow connect:CreateContactFlowModule connect:CreateEvaluationForm connect:CreateHoursOfOperation connect:CreateInstance connect:CreateIntegrationAssociation connect:CreateParticipant connect:CreatePrompt

Prefijo de servicio	Acciones
	connect:CreateQueue connect:CreateQuickConnect connect:CreateRoutingProfile connect:CreateRule connect:CreateSecurityProfile connect:CreateTaskTemplate connect:CreateTrafficDistributionGroup connect:CreateUseCase connect:CreateUser connect:CreateUserHierarchyGroup connect:CreateView connect:CreateViewVersion connect:CreateVocabulary connect:DeactivateEvaluationForm connect:DeleteContactEvaluation connect:DeleteContactFlow connect:DeleteContactFlowModule connect:DeleteEvaluationForm connect:DeleteHoursOfOperation connect:DeleteInstance connect:DeleteIntegrationAssociation

Prefijo de servicio	Acciones
	connect:DeletePrompt connect:DeleteQueue connect:DeleteQuickConnect connect:DeleteRoutingProfile connect:DeleteRule connect:DeleteSecurityProfile connect:DeleteTaskTemplate connect:DeleteTrafficDistributionGroup connect:DeleteUseCase connect:DeleteUser connect:DeleteUserHierarchyGroup connect:DeleteView connect:DeleteVocabulary connect:DescribeAgentStatus connect:DescribeContact connect:DescribeContactEvaluation connect:DescribeContactFlow connect:DescribeContactFlowModule connect:DescribeEvaluationForm connect:DescribeInstanceAttribute connect:DescribeInstanceStorageConfig

Prefijo de servicio	Acciones
	connect:DescribePhoneNumber connect:DescribeRule connect:DescribeTrafficDistributionGroup connect:DescribeUserHierarchyGroup connect:DescribeUserHierarchyStructure connect:DescribeView connect:DescribeVocabulary connect:DisassociateApprovedOrigin connect:DisassociateBot connect:DisassociateInstanceStorageConfig connect:DisassociateLambdaFunction connect:DisassociateLexBot connect:DisassociatePhoneNumberContactFlow connect:DisassociateQueueQuickConnects connect:DisassociateRoutingProfileQueues connect:DisassociateSecurityKey connect:DismissUserContact connect:GetContactAttributes connect:GetCurrentMetricData connect:GetCurrentUserData connect:GetFederationToken

Prefijo de servicio	Acciones
	connect:GetMetricData connect:GetMetricDataV2 connect:GetPromptFile connect:GetTaskTemplate connect:GetTrafficDistribution connect:ListApprovedOrigins connect:ListBots connect:ListContactEvaluations connect:ListContactFlowModules connect:ListContactFlows connect:ListContactReferences connect:ListDefaultVocabularies connect:ListEvaluationForms connect:ListEvaluationFormVersions connect:ListHoursOfOperations connect:ListInstanceAttributes connect:ListInstanceStorageConfigs connect:ListIntegrationAssociations connect:ListLambdaFunctions connect:ListLexBots connect:ListPhoneNumbers

Prefijo de servicio	Acciones
	connect:ListPhoneNumbersV2 connect:ListPrompts connect:ListQueueQuickConnects connect:ListQueues connect:ListQuickConnects connect:ListRoutingProfileQueues connect:ListRoutingProfiles connect:ListRules connect:ListSecurityKeys connect:ListSecurityProfileApplications connect:ListSecurityProfilePermissions connect:ListSecurityProfiles connect:ListTaskTemplates connect:ListTrafficDistributionGroups connect:ListUseCases connect:ListUserHierarchyGroups connect:ListUsers connect:ListViews connect:ListViewVersions connect:MonitorContact connect:PutUserStatus

Prefijo de servicio	Acciones
	connect:ReleasePhoneNumber connect:ReplicateInstance connect:ResumeContactRecording connect:SearchAvailablePhoneNumbers connect:SearchHoursOfOperations connect:SearchPrompts connect:SearchQueues connect:SearchQuickConnects connect:SearchRoutingProfiles connect:SearchSecurityProfiles connect:SearchVocabularies connect:StartChatContact connect:StartContactEvaluation connect:StartContactRecording connect:StartContactStreaming connect:StartOutboundVoiceContact connect:StartTaskContact connect:StopContact connect:StopContactRecording connect:StopContactStreaming connect:SubmitContactEvaluation

Prefijo de servicio	Acciones
	connect:SuspendContactRecording connect:TransferContact connect:UpdateAgentStatus connect:UpdateContact connect:UpdateContactAttributes connect:UpdateContactEvaluation connect:UpdateContactFlowContent connect:UpdateContactFlowMetadata connect:UpdateContactFlowModuleContent connect:UpdateContactFlowModuleMetadata connect:UpdateContactFlowName connect:UpdateContactSchedule connect:UpdateEvaluationForm connect:UpdateHoursOfOperation connect:UpdateInstanceAttribute connect:UpdateInstanceStorageConfig connect:UpdateParticipantRoleConfig connect:UpdatePhoneNumber connect:UpdatePhoneNumberMetadata connect:UpdatePrompt connect:UpdateQueueHoursOfOperation

Prefijo de servicio	Acciones
	connect:UpdateQueueMaxContacts connect:UpdateQueueName connect:UpdateQueueOutboundCallerConfig connect:UpdateQueueStatus connect:UpdateQuickConnectConfig connect:UpdateQuickConnectName connect:UpdateRoutingProfileAgentAvailabilityTimer connect:UpdateRoutingProfileConcurrency connect:UpdateRoutingProfileDefaultOutboundQueue connect:UpdateRoutingProfileName connect:UpdateRoutingProfileQueues connect:UpdateRule connect:UpdateSecurityProfile connect:UpdateTaskTemplate connect:UpdateTrafficDistribution connect:UpdateUserHierarchy connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect:UpdateUserPhoneConfig connect:UpdateUserRoutingProfile

Prefijo de servicio	Acciones
	connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

Prefijo de servicio	Acciones
databrew	databrew:BatchDeleteRecipeVersion databrew:CreateDataset databrew:CreateProfileJob databrew:CreateProject databrew:CreateRecipe databrew:CreateRecipeJob databrew:CreateRuleset databrew:CreateSchedule databrew:DeleteDataset databrew:DeleteJob databrew:DeleteProject databrew:DeleteRecipeVersion databrew:DeleteRuleset databrew:DeleteSchedule databrew:DescribeDataset databrew:DescribeJob databrew:DescribeJobRun databrew:DescribeProject databrew:DescribeRecipe databrew:DescribeRuleset databrew:DescribeSchedule

Prefijo de servicio	Acciones
	databrew:ListDatasets databrew:ListJobRuns databrew:ListJobs databrew:ListProjects databrew:ListRecipes databrew:ListRecipeVersions databrew:ListRulesets databrew:ListSchedules databrew:PublishRecipe databrew:SendProjectSessionAction databrew:StartJobRun databrew:StartProjectSession databrew:StopJobRun databrew:UpdateDataset databrew:UpdateProfileJob databrew:UpdateProject databrew:UpdateRecipe databrew:UpdateRecipeJob databrew:UpdateRuleset databrew:UpdateSchedule

Prefijo de servicio	Acciones
dataexchange	dataexchange:CancelJob
	dataexchange:CreateDataSet
	dataexchange:CreateEventAction
	dataexchange:CreateJob
	dataexchange:CreateRevision
	dataexchange:DeleteAsset
	dataexchange:DeleteEventAction
	dataexchange:DeleteRevision
	dataexchange:GetEventAction
	dataexchange:GetJob
	dataexchange:ListDataSetRevisions
	dataexchange:ListDataSets
	dataexchange:ListEventActions
	dataexchange:ListJobs
	dataexchange:ListRevisionAssets
	dataexchange:RevokeRevision
	dataexchange:StartJob
	dataexchange:UpdateAsset
	dataexchange:UpdateDataSet
	dataexchange:UpdateEventAction
	dataexchange:UpdateRevision

Prefijo de servicio	Acciones
datapipeline	datapipeline:ActivatePipeline datapipeline:CreatePipeline datapipeline:DeactivatePipeline datapipeline:DeletePipeline datapipeline:DescribeObjects datapipeline:DescribePipelines datapipeline:EvaluateExpression datapipeline:GetPipelineDefinition datapipeline:ListPipelines datapipeline:PollForTask datapipeline:PutPipelineDefinition datapipeline:QueryObjects datapipeline:ReportTaskProgress datapipeline:ReportTaskRunnerHeartbeat datapipeline:SetStatus datapipeline:SetTaskStatus datapipeline:ValidatePipelineDefinition

Prefijo de servicio	Acciones
dax	dax:CreateCluster
	dax:DecreaseReplicationFactor
	dax:DeleteCluster
	dax:DeleteParameterGroup
	dax:DeleteSubnetGroup
	dax:DescribeClusters
	dax:DescribeDefaultParameters
	dax:DescribeEvents
	dax:DescribeParameterGroups
	dax:DescribeParameters
	dax:DescribeSubnetGroups
	dax:IncreaseReplicationFactor
	dax:RebootNode
	dax:UpdateCluster
	dax:UpdateParameterGroup
	dax:UpdateSubnetGroup

Prefijo de servicio	Acciones
devicefarm	devicefarm:CreateDevicePool devicefarm:CreateInstanceProfile devicefarm:CreateNetworkProfile devicefarm:CreateProject devicefarm:CreateRemoteAccessSession devicefarm:CreateTestGridProject devicefarm:CreateTestGridUrl devicefarm:CreateUpload devicefarm:CreateVPCEConfiguration devicefarm:DeleteDevicePool devicefarm:DeleteInstanceProfile devicefarm:DeleteNetworkProfile devicefarm:DeleteProject devicefarm:DeleteRemoteAccessSession devicefarm:DeleteRun devicefarm:DeleteTestGridProject devicefarm:DeleteUpload devicefarm:DeleteVPCEConfiguration devicefarm:GetAccountSettings devicefarm:GetDevice devicefarm:GetDeviceInstance

Prefijo de servicio	Acciones
	devicefarm:GetDevicePool devicefarm:GetDevicePoolCompatibility devicefarm:GetInstanceProfile devicefarm:GetJob devicefarm:GetNetworkProfile devicefarm:GetOfferingStatus devicefarm:GetProject devicefarm:GetRemoteAccessSession devicefarm:GetRun devicefarm:GetSuite devicefarm:GetTest devicefarm:GetTestGridProject devicefarm:GetTestGridSession devicefarm:GetUpload devicefarm:GetVPCEConfiguration devicefarm:ListArtifacts devicefarm:ListDeviceInstances devicefarm:ListDevicePools devicefarm:ListDevices devicefarm:ListInstanceProfiles devicefarm:ListJobs

Prefijo de servicio	Acciones
	devicefarm:ListNetworkProfiles devicefarm:ListOfferingPromotions devicefarm:ListOfferings devicefarm:ListOfferingTransactions devicefarm:ListProjects devicefarm:ListRemoteAccessSessions devicefarm:ListRuns devicefarm:ListSamples devicefarm:ListSuites devicefarm:ListTestGridProjects devicefarm:ListTestGridSessionActions devicefarm:ListTestGridSessionArtifacts devicefarm:ListTestGridSessions devicefarm:ListTests devicefarm:ListUniqueProblems devicefarm:ListUploads devicefarm:ListVPCEConfigurations devicefarm:PurchaseOffering devicefarm:RenewOffering devicefarm:ScheduleRun devicefarm:StopJob

Prefijo de servicio	Acciones
	devicefarm:StopRemoteAccessSession devicefarm:StopRun devicefarm:UpdateDeviceInstance devicefarm:UpdateDevicePool devicefarm:UpdateInstanceProfile devicefarm:UpdateNetworkProfile devicefarm:UpdateProject devicefarm:UpdateTestGridProject devicefarm:UpdateUpload devicefarm:UpdateVPCEConfiguration

Prefijo de servicio	Acciones
devops-guru	devops-guru:AddNotificationChannel
	devops-guru:DeleteInsight
	devops-guru:DescribeAccountHealth
	devops-guru:DescribeAccountOverview
	devops-guru:DescribeAnomaly
	devops-guru:DescribeEventSourcesConfig
	devops-guru:DescribeFeedback
	devops-guru:DescribeInsight
	devops-guru:DescribeOrganizationHealth
	devops-guru:DescribeOrganizationOverview
	devops-guru:DescribeOrganizationResourceCollectionHealth
	devops-guru:DescribeResourceCollectionHealth
	devops-guru:DescribeServiceIntegration
	devops-guru:GetCostEstimation
	devops-guru:GetResourceCollection
	devops-guru:ListAnomaliesForInsight
	devops-guru:ListAnomalousLogGroups
	devops-guru:ListEvents
	devops-guru:ListInsights
	devops-guru:ListMonitoredResources
	devops-guru:ListNotificationChannels

Prefijo de servicio	Acciones
	devops-guru:ListOrganizationInsights devops-guru:ListRecommendations devops-guru:PutFeedback devops-guru:RemoveNotificationChannel devops-guru:SearchInsights devops-guru:SearchOrganizationInsights devops-guru:StartCostEstimation devops-guru:UpdateEventSourcesConfig devops-guru:UpdateResourceCollection devops-guru:UpdateServiceIntegration

Prefijo de servicio	Acciones
directconnect	directconnect:AcceptDirectConnectGatewayAssociationProposal directconnect:AllocateConnectionOnInterconnect directconnect:AllocateHostedConnection directconnect:AllocatePrivateVirtualInterface directconnect:AllocatePublicVirtualInterface directconnect:AllocateTransitVirtualInterface directconnect:AssociateConnectionWithLag directconnect:AssociateHostedConnection directconnect:AssociateMacSecKey directconnect:AssociateVirtualInterface directconnect:ConfirmConnection directconnect:ConfirmCustomerAgreement directconnect:ConfirmPrivateVirtualInterface directconnect:ConfirmPublicVirtualInterface directconnect:ConfirmTransitVirtualInterface directconnect:CreateBGPPeer directconnect:CreateConnection directconnect:CreateDirectConnectGateway directconnect:CreateDirectConnectGatewayAssociation directconnect:CreateDirectConnectGatewayAssociationProposal directconnect:CreateInterconnect

Prefijo de servicio	Acciones
	directconnect:CreateLag directconnect:CreatePrivateVirtualInterface directconnect:CreatePublicVirtualInterface directconnect:CreateTransitVirtualInterface directconnect:DeleteBGPPeer directconnect:DeleteConnection directconnect:DeleteDirectConnectGateway directconnect:DeleteDirectConnectGatewayAssociation directconnect:DeleteDirectConnectGatewayAssociationProposal directconnect:DeleteInterconnect directconnect:DeleteLag directconnect:DeleteVirtualInterface directconnect:DescribeConnectionLoa directconnect:DescribeConnections directconnect:DescribeConnectionsOnInterconnect directconnect:DescribeCustomerMetadata directconnect:DescribeDirectConnectGatewayAssociationProposals directconnect:DescribeDirectConnectGatewayAssociations directconnect:DescribeDirectConnectGatewayAttachments directconnect:DescribeDirectConnectGateways directconnect:DescribeHostedConnections

Prefijo de servicio	Acciones
	directconnect:DescribeInterconnectLoa directconnect:DescribeInterconnects directconnect:DescribeLags directconnect:DescribeLoa directconnect:DescribeLocations directconnect:DescribeRouterConfiguration directconnect:DescribeVirtualGateways directconnect:DescribeVirtualInterfaces directconnect:DisassociateConnectionFromLag directconnect:DisassociateMacSecKey directconnect:ListVirtualInterfaceTestHistory directconnect:StartBgpFailoverTest directconnect:StopBgpFailoverTest directconnect:UpdateConnection directconnect:UpdateDirectConnectGateway directconnect:UpdateDirectConnectGatewayAssociation directconnect:UpdateLag directconnect:UpdateVirtualInterfaceAttributes

Prefijo de servicio	Acciones
dlm	dlm:CreateLifecyclePolicy
	dlm:DeleteLifecyclePolicy
	dlm:GetLifecyclePolicies
	dlm:GetLifecyclePolicy
	dlm:UpdateLifecyclePolicy

Prefijo de servicio	Acciones
dms	dms:ApplyPendingMaintenanceAction dms:BatchStartRecommendations dms:CancelReplicationTaskAssessmentRun dms:CreateDataProvider dms:CreateEndpoint dms:CreateEventSubscription dms:CreateInstanceProfile dms:CreateMigrationProject dms:CreateReplicationConfig dms:CreateReplicationInstance dms:CreateReplicationSubnetGroup dms:CreateReplicationTask dms:DeleteCertificate dms:DeleteConnection dms:DeleteDataProvider dms:DeleteEndpoint dms:DeleteEventSubscription dms:DeleteFleetAdvisorCollector dms:DeleteFleetAdvisorDatabases dms:DeleteInstanceProfile dms:DeleteMigrationProject

Prefijo de servicio	Acciones
	dms:DeleteReplicationConfig dms:DeleteReplicationInstance dms:DeleteReplicationSubnetGroup dms:DeleteReplicationTask dms:DeleteReplicationTaskAssessmentRun dms:DescribeAccountAttributes dms:DescribeApplicableIndividualAssessments dms:DescribeCertificates dms:DescribeConnections dms:DescribeEndpoints dms:DescribeEndpointSettings dms:DescribeEndpointTypes dms:DescribeEngineVersions dms:DescribeEventCategories dms:DescribeEvents dms:DescribeEventSubscriptions dms:DescribeFleetAdvisorCollectors dms:DescribeFleetAdvisorDatabases dms:DescribeFleetAdvisorLsaAnalysis dms:DescribeFleetAdvisorSchemaObjectSummary dms:DescribeFleetAdvisorSchemas

Prefijo de servicio	Acciones
	dms:DescribeMetadataModelImports dms:DescribeOrderableReplicationInstances dms:DescribePendingMaintenanceActions dms:DescribeRecommendationLimitations dms:DescribeRecommendations dms:DescribeRefreshSchemasStatus dms:DescribeReplicationConfigs dms:DescribeReplicationInstances dms:DescribeReplicationInstanceTaskLogs dms:DescribeReplications dms:DescribeReplicationSubnetGroups dms:DescribeReplicationTableStatistics dms:DescribeReplicationTaskAssessmentResults dms:DescribeReplicationTaskAssessmentRuns dms:DescribeReplicationTaskIndividualAssessments dms:DescribeReplicationTasks dms:DescribeSchemas dms:DescribeTableStatistics dms:ExportMetadataModelAssessment dms:ImportCertificate dms:ModifyEndpoint

Prefijo de servicio	Acciones
	dms:ModifyEventSubscription dms:ModifyReplicationConfig dms:ModifyReplicationInstance dms:ModifyReplicationSubnetGroup dms:ModifyReplicationTask dms:MoveReplicationTask dms:RebootReplicationInstance dms:RefreshSchemas dms:ReloadReplicationTables dms:ReloadTables dms:RunFleetAdvisorLsaAnalysis dms:StartMetadataModelAssessment dms:StartMetadataModelConversion dms:StartMetadataModelExportToTarget dms:StartRecommendations dms:StartReplication dms:StartReplicationTask dms:StartReplicationTaskAssessment dms:StopReplicationTask dms:TestConnection dms:UpdateSubscriptionsToEventBridge

Prefijo de servicio	Acciones
docdb-elastic	docdb-elastic:CreateCluster
	docdb-elastic:CreateClusterSnapshot
	docdb-elastic:DeleteCluster
	docdb-elastic:DeleteClusterSnapshot
	docdb-elastic:GetCluster
	docdb-elastic:GetClusterSnapshot
	docdb-elastic:ListClusters
	docdb-elastic:ListClusterSnapshots
	docdb-elastic:RestoreClusterFromSnapshot
	docdb-elastic:UpdateCluster

Prefijo de servicio	Acciones
ds	ds:AcceptSharedDirectory ds:AddIpRoutes ds:AddRegion ds:CancelSchemaExtension ds:ConnectDirectory ds:CreateAlias ds:CreateComputer ds:CreateConditionalForwarder ds:CreateDirectory ds:CreateLogSubscription ds:CreateMicrosoftAD ds:CreateSnapshot ds:CreateTrust ds:DeleteConditionalForwarder ds:DeleteDirectory ds:DeleteLogSubscription ds:DeleteSnapshot ds:DeleteTrust ds:DeregisterCertificate ds:DeregisterEventTopic ds:DescribeCertificate

Prefijo de servicio	Acciones
	ds:DescribeClientAuthenticationSettings ds:DescribeConditionalForwarders ds:DescribeDirectories ds:DescribeDomainControllers ds:DescribeEventTopics ds:DescribeLDAPSSettings ds:DescribeRegions ds:DescribeSettings ds:DescribeSharedDirectories ds:DescribeSnapshots ds:DescribeTrusts ds:DescribeUpdateDirectory ds:DisableClientAuthentication ds:DisableLDAPS ds:DisableRadius ds:DisableSso ds:EnableClientAuthentication ds:EnableLDAPS ds:EnableRadius ds:EnableSso ds:GetDirectoryLimits

Prefijo de servicio	Acciones
	ds:GetSnapshotLimits ds:ListCertificates ds:ListIpRoutes ds:ListLogSubscriptions ds:ListSchemaExtensions ds:RegisterCertificate ds:RegisterEventTopic ds:RejectSharedDirectory ds:RemoveIpRoutes ds:RemoveRegion ds:ResetUserPassword ds:RestoreFromSnapshot ds:ShareDirectory ds:StartSchemaExtension ds:UnshareDirectory ds:UpdateConditionalForwarder ds:UpdateDirectorySetup ds:UpdateNumberOfDomainControllers ds:UpdateRadius ds:UpdateSettings ds:UpdateTrust

Prefijo de servicio	Acciones
	ds:VerifyTrust

Prefijo de servicio	Acciones
dynamodb	dynamodb:CreateBackup
	dynamodb:CreateGlobalTable
	dynamodb:CreateTable
	dynamodb:DeleteBackup
	dynamodb:DeleteTable
	dynamodb:DescribeBackup
	dynamodb:DescribeContinuousBackups
	dynamodb:DescribeContributorInsights
	dynamodb:DescribeEndpoints
	dynamodb:DescribeExport
	dynamodb:DescribeGlobalTable
	dynamodb:DescribeGlobalTableSettings
	dynamodb:DescribeImport
	dynamodb:DescribeKinesisStreamingDestination
	dynamodb:DescribeLimits
	dynamodb:DescribeStream
	dynamodb:DescribeTable
	dynamodb:DescribeTableReplicaAutoScaling
	dynamodb:DescribeTimeToLive
	dynamodb:DisableKinesisStreamingDestination
	dynamodb:EnableKinesisStreamingDestination

Prefijo de servicio	Acciones
	dynamodb:ExportTableToPointInTime dynamodb:ImportTable dynamodb:ListBackups dynamodb:ListContributorInsights dynamodb:ListExports dynamodb:ListGlobalTables dynamodb:ListImports dynamodb:ListStreams dynamodb:ListTables dynamodb:RestoreTableFromBackup dynamodb:RestoreTableToPointInTime dynamodb:UpdateContinuousBackups dynamodb:UpdateContributorInsights dynamodb:UpdateGlobalTable dynamodb:UpdateGlobalTableSettings dynamodb:UpdateTable dynamodb:UpdateTableReplicaAutoScaling dynamodb:UpdateTimeToLive
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

Prefijo de servicio	Acciones
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptReservedInstancesExchangeQuote
	ec2:AcceptTransitGatewayMulticastDomainAssociations
	ec2:AcceptTransitGatewayPeeringAttachment
	ec2:AcceptTransitGatewayVpcAttachment
	ec2:AcceptVpcEndpointConnections
	ec2:AcceptVpcPeeringConnection
	ec2:AdvertiseByoipCidr
	ec2:AllocateAddress
	ec2:AllocateHosts
	ec2:AllocateIpamPoolCidr
	ec2:ApplySecurityGroupsToClientVpnTargetNetwork
	ec2:AssignIpv6Addresses
	ec2:AssignPrivateIpAddresses
	ec2:AssignPrivateNatGatewayAddress
	ec2:AssociateAddress
	ec2:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow

Prefijo de servicio	Acciones
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteTable
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask
	ec2:CancelCapacityReservation

Prefijo de servicio	Acciones
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway
	ec2:CreateDefaultSubnet

Prefijo de servicio	Acciones
	ec2:CreateDefaultVpc
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2:CreateInternetGateway
	ec2:CreateIpam
	ec2:CreateIpamPool
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplate
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable

Prefijo de servicio	Acciones
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	ec2:CreateManagedPrefixList
	ec2:CreateNatGateway
	ec2:CreateNetworkAcl
	ec2:CreateNetworkAclEntry
	ec2:CreateNetworkInsightsAccessScope
	ec2:CreateNetworkInsightsPath
	ec2:CreateNetworkInterface
	ec2:CreateNetworkInterfacePermission
	ec2:CreatePlacementGroup
	ec2:CreatePublicIpv4Pool
	ec2:CreateReplaceRootVolumeTask
	ec2:CreateReservedInstancesListing
	ec2:CreateRestoreImageTask
	ec2:CreateRoute
	ec2:CreateRouteTable
	ec2:CreateSecurityGroup
	ec2:CreateSnapshot
	ec2:CreateSnapshots

Prefijo de servicio	Acciones
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2:CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup

Prefijo de servicio	Acciones
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2:CreateVpc
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2:DeleteCarrierGateway
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs

Prefijo de servicio	Acciones
	ec2:DeleteFpgaImage
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamPool
	ec2:DeleteIpamResourceDiscovery
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope

Prefijo de servicio	Acciones
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteTable
	ec2:DeleteSecurityGroup
	ec2:DeleteSnapshot
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2:DeleteTransitGatewayConnect

Prefijo de servicio	Acciones
	ec2:DeleteTransitGatewayConnectPeer
	ec2:DeleteTransitGatewayMulticastDomain
	ec2:DeleteTransitGatewayPeeringAttachment
	ec2:DeleteTransitGatewayPolicyTable
	ec2:DeleteTransitGatewayPrefixListReference
	ec2:DeleteTransitGatewayRoute
	ec2:DeleteTransitGatewayRouteTable
	ec2:DeleteTransitGatewayRouteTableAnnouncement
	ec2:DeleteTransitGatewayVpcAttachment
	ec2:DeleteVerifiedAccessEndpoint
	ec2:DeleteVerifiedAccessGroup
	ec2:DeleteVerifiedAccessInstance
	ec2:DeleteVerifiedAccessTrustProvider
	ec2:DeleteVolume
	ec2:DeleteVpc
	ec2:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcEndpointServiceConfigurations
	ec2:DeleteVpcPeeringConnection
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute

Prefijo de servicio	Acciones
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAddressTransfers
	ec2:DescribeAggregateIdFormat
	ec2:DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityReservationFleets
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways
	ec2:DescribeClassicLinkInstances

Prefijo de servicio	Acciones
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks
	ec2:DescribeCustomerGateways
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute
	ec2:DescribeFpgaImages

Prefijo de servicio	Acciones
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdentityIdFormat
	ec2:DescribeIdFormat
	ec2:DescribeImageAttribute
	ec2:DescribeImages
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceAttribute
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstances
	ec2:DescribeInstanceStatus
	ec2:DescribeInstanceTypeOfferings
	ec2:DescribeInstanceTypes
	ec2:DescribeInternetGateways
	ec2:DescribeIpamPools

Prefijo de servicio	Acciones
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpams
	ec2:DescribeIpamScopes
	ec2:DescribeIpv6Pools
	ec2:DescribeKeyPairs
	ec2:DescribeLaunchTemplates
	ec2:DescribeLaunchTemplateVersions
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGateways
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeManagedPrefixLists
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes

Prefijo de servicio	Acciones
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteTables
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroups

Prefijo de servicio	Acciones
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshots
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups
	ec2:DescribeStoreImageTasks
	ec2:DescribeSubnets
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables

Prefijo de servicio	Acciones
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGateways
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVolumeStatus
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcClassicLink
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcEndpointServiceConfigurations

Prefijo de servicio	Acciones
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcPeeringConnections
	ec2:DescribeVpcs
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableIpamOrganizationAdminAccount

Prefijo de servicio	Acciones
	ec2:DisableSerialConsoleAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteTable
	ec2:DisassociateSubnetCidrBlock
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAwsNetworkPerformanceMetricSubscription

Prefijo de servicio	Acciones
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableIpamOrganizationAdminAccount
	ec2:EnableReachabilityAnalyzerOrganizationSharing
	ec2:EnableSerialConsoleAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData

Prefijo de servicio	Acciones
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceTypesFromInstanceRequirements
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	ec2:GetIpamDiscoveredResourceCidrs
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations

Prefijo de servicio	Acciones
	ec2:GetManagedPrefixListEntries
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessGroupPolicy
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList

Prefijo de servicio	Acciones
	ec2:ImportImage
	ec2:ImportInstance
	ec2:ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2:ModifyAddressAttribute
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdentityIdFormat
	ec2:ModifyIdFormat
	ec2:ModifyImageAttribute
	ec2:ModifyInstanceAttribute

Prefijo de servicio	Acciones
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstancePlacement
	ec2:ModifyIpam
	ec2:ModifyIpamPool
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier

Prefijo de servicio	Acciones
	ec2:ModifySpotFleetRequest
	ec2:ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2:ModifyTransitGatewayPrefixListReference
	ec2:ModifyTransitGatewayVpcAttachment
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2:ModifyVpcAttribute
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration

Prefijo de servicio	Acciones
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers

Prefijo de servicio	Acciones
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute
	ec2:ResetEbsDefaultKmsKeyId

Prefijo de servicio	Acciones
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis

Prefijo de servicio	Acciones
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:StopInstances
	ec2:TerminateClientVpnConnections
	ec2:TerminateInstances
	ec2:UnassignIpv6Addresses
	ec2:UnassignPrivateIpAddresses
	ec2:UnassignPrivateNatGatewayAddress
	ec2:UnmonitorInstances
	ec2:UpdateSecurityGroupRuleDescriptionsEgress
	ec2:UpdateSecurityGroupRuleDescriptionsIngress
	ec2:WithdrawByoipCidr

Prefijo de servicio	Acciones
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepository ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryPolicy ecr:DescribeImageReplicationStatus ecr:DescribeImages ecr:DescribeImageScanFindings ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy

Prefijo de servicio	Acciones
	ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr:InitiateLayerUpload ecr:ListImages ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UploadLayerPart

Prefijo de servicio	Acciones
ecr-public	ecr-public:BatchCheckLayerAvailability ecr-public:BatchDeleteImage ecr-public:CompleteLayerUpload ecr-public:CreateRepository ecr-public:DeleteRepository ecr-public:DeleteRepositoryPolicy ecr-public:DescribeImages ecr-public:DescribeRegistries ecr-public:DescribeRepositories ecr-public:GetAuthorizationToken ecr-public:GetRegistryCatalogData ecr-public:GetRepositoryCatalogData ecr-public:GetRepositoryPolicy ecr-public:InitiateLayerUpload ecr-public:PutImage ecr-public:PutRegistryCatalogData ecr-public:PutRepositoryCatalogData ecr-public:SetRepositoryPolicy ecr-public:UploadLayerPart

Prefijo de servicio	Acciones
ecs	ecs:CreateCapacityProvider ecs:CreateCluster ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTasks ecs:DescribeTaskSets ecs:DiscoverPollEndpoint

Prefijo de servicio	Acciones
	ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs:RegisterTaskDefinition ecs:RunTask ecs:StartTask ecs:StopTask ecs:SubmitAttachmentStateChanges

Prefijo de servicio	Acciones
	ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

Prefijo de servicio	Acciones
eks	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAddon
	eks:DeleteCluster
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeregisterCluster
	eks:DescribeAddon
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions
	eks:DescribeCluster
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeNodegroup
	eks:DescribeUpdate
	eks:DisassociateIdentityProviderConfig
	eks:ListAddons

Prefijo de servicio	Acciones
	eks:ListClusters eks:ListFargateProfiles eks:ListIdentityProviderConfigs eks:ListNodegroups eks:ListUpdates eks:RegisterCluster eks:UpdateAddon eks:UpdateClusterConfig eks:UpdateClusterVersion eks:UpdateNodegroupConfig eks:UpdateNodegroupVersion
elastic-inference	elastic-inference:DescribeAcceleratorOfferings elastic-inference:DescribeAccelerators elastic-inference:DescribeAcceleratorTypes

Prefijo de servicio	Acciones
elasticache	elasticache:AuthorizeCacheSecurityGroupIngress
	elasticache:BatchApplyUpdateAction
	elasticache:BatchStopUpdateAction
	elasticache:CompleteMigration
	elasticache:CopySnapshot
	elasticache:CreateCacheCluster
	elasticache:CreateCacheParameterGroup
	elasticache:CreateCacheSecurityGroup
	elasticache:CreateCacheSubnetGroup
	elasticache:CreateGlobalReplicationGroup
	elasticache:CreateReplicationGroup
	elasticache:CreateSnapshot
	elasticache:CreateUser
	elasticache:CreateUserGroup
	elasticache:DecreaseNodeGroupsInGlobalReplicationGroup
	elasticache:DecreaseReplicaCount
	elasticache:DeleteCacheCluster
	elasticache:DeleteCacheParameterGroup
	elasticache:DeleteCacheSecurityGroup
	elasticache:DeleteCacheSubnetGroup
	elasticache:DeleteGlobalReplicationGroup

Prefijo de servicio	Acciones
	elasticache:DeleteReplicationGroup
	elasticache:DeleteSnapshot
	elasticache:DeleteUser
	elasticache:DeleteUserGroup
	elasticache:DescribeCacheClusters
	elasticache:DescribeCacheEngineVersions
	elasticache:DescribeCacheParameterGroups
	elasticache:DescribeCacheParameters
	elasticache:DescribeCacheSecurityGroups
	elasticache:DescribeCacheSubnetGroups
	elasticache:DescribeEngineDefaultParameters
	elasticache:DescribeEvents
	elasticache:DescribeGlobalReplicationGroups
	elasticache:DescribeReplicationGroups
	elasticache:DescribeReservedCacheNodes
	elasticache:DescribeReservedCacheNodesOfferings
	elasticache:DescribeServiceUpdates
	elasticache:DescribeSnapshots
	elasticache:DescribeUpdateActions
	elasticache:DescribeUserGroups
	elasticache:DescribeUsers

Prefijo de servicio	Acciones
	elasticache:DisassociateGlobalReplicationGroup
	elasticache:FailoverGlobalReplicationGroup
	elasticache:IncreaseNodeGroupsInGlobalReplicationGroup
	elasticache:IncreaseReplicaCount
	elasticache:ListAllowedNodeTypeModifications
	elasticache:ModifyCacheCluster
	elasticache:ModifyCacheParameterGroup
	elasticache:ModifyCacheSubnetGroup
	elasticache:ModifyGlobalReplicationGroup
	elasticache:ModifyReplicationGroup
	elasticache:ModifyReplicationGroupShardConfiguration
	elasticache:ModifyUser
	elasticache:ModifyUserGroup
	elasticache:PurchaseReservedCacheNodesOffering
	elasticache:RebalanceSlotsInGlobalReplicationGroup
	elasticache:RebootCacheCluster
	elasticache:ResetCacheParameterGroup
	elasticache:RevokeCacheSecurityGroupIngress
	elasticache:StartMigration
	elasticache:TestFailover
	elasticache:TestMigration

Prefijo de servicio	Acciones
elasticbeanstalk	elasticbeanstalk:AbortEnvironmentUpdate elasticbeanstalk:ApplyEnvironmentManagedAction elasticbeanstalk:AssociateEnvironmentOperationsRole elasticbeanstalk:CheckDNSAvailability elasticbeanstalk:ComposeEnvironments elasticbeanstalk:CreateApplication elasticbeanstalk:CreateApplicationVersion elasticbeanstalk:CreateConfigurationTemplate elasticbeanstalk:CreateEnvironment elasticbeanstalk:CreatePlatformVersion elasticbeanstalk:CreateStorageLocation elasticbeanstalk:DeleteApplication elasticbeanstalk:DeleteApplicationVersion elasticbeanstalk:DeleteConfigurationTemplate elasticbeanstalk:DeleteEnvironmentConfiguration elasticbeanstalk:DeletePlatformVersion elasticbeanstalk:DescribeAccountAttributes elasticbeanstalk:DescribeApplications elasticbeanstalk:DescribeApplicationVersions elasticbeanstalk:DescribeConfigurationOptions elasticbeanstalk:DescribeConfigurationSettings

Prefijo de servicio	Acciones
	elasticbeanstalk:DescribeEnvironmentHealth elasticbeanstalk:DescribeEnvironmentManagedActionHistory elasticbeanstalk:DescribeEnvironmentManagedActions elasticbeanstalk:DescribeEnvironmentResources elasticbeanstalk:DescribeEnvironments elasticbeanstalk:DescribeEvents elasticbeanstalk:DescribeInstancesHealth elasticbeanstalk:DescribePlatformVersion elasticbeanstalk:DisassociateEnvironmentOperationsRole elasticbeanstalk:ListAvailableSolutionStacks elasticbeanstalk:ListPlatformBranches elasticbeanstalk:ListPlatformVersions elasticbeanstalk:RebuildEnvironment elasticbeanstalk:RequestEnvironmentInfo elasticbeanstalk:RestartAppServer elasticbeanstalk:RetrieveEnvironmentInfo elasticbeanstalk:SwapEnvironmentCNAMEs elasticbeanstalk:TerminateEnvironment elasticbeanstalk:UpdateApplication elasticbeanstalk:UpdateApplicationResourceLifecycle elasticbeanstalk:UpdateApplicationVersion

Prefijo de servicio	Acciones
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

Prefijo de servicio	Acciones
elasticfilesystem	elasticfilesystem:CreateAccessPoint
	elasticfilesystem:CreateFileSystem
	elasticfilesystem:CreateMountTarget
	elasticfilesystem:CreateReplicationConfiguration
	elasticfilesystem:DeleteAccessPoint
	elasticfilesystem:DeleteFileSystem
	elasticfilesystem:DeleteFileSystemPolicy
	elasticfilesystem:DeleteMountTarget
	elasticfilesystem:DeleteReplicationConfiguration
	elasticfilesystem:DescribeAccessPoints
	elasticfilesystem:DescribeAccountPreferences
	elasticfilesystem:DescribeBackupPolicy
	elasticfilesystem:DescribeFileSystemPolicy
	elasticfilesystem:DescribeFileSystems
	elasticfilesystem:DescribeLifecycleConfiguration
	elasticfilesystem:DescribeMountTargets
	elasticfilesystem:DescribeMountTargetSecurityGroups
	elasticfilesystem:DescribeReplicationConfigurations
	elasticfilesystem:ModifyMountTargetSecurityGroups
	elasticfilesystem:PutAccountPreferences
	elasticfilesystem:PutBackupPolicy

Prefijo de servicio	Acciones
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem

Prefijo de servicio	Acciones
elasticloadbalancing	elasticloadbalancing:AddListenerCertificates elasticloadbalancing:ApplySecurityGroupsToLoadBalancer elasticloadbalancing:AttachLoadBalancerToSubnets elasticloadbalancing:ConfigureHealthCheck elasticloadbalancing:CreateAppCookieStickinessPolicy elasticloadbalancing:CreateLBCookieStickinessPolicy elasticloadbalancing:CreateListener elasticloadbalancing:CreateLoadBalancer elasticloadbalancing:CreateLoadBalancerListeners elasticloadbalancing:CreateLoadBalancerPolicy elasticloadbalancing:CreateRule elasticloadbalancing:CreateTargetGroup elasticloadbalancing:DeleteListener elasticloadbalancing:DeleteLoadBalancer elasticloadbalancing:DeleteLoadBalancerListeners elasticloadbalancing:DeleteLoadBalancerPolicy elasticloadbalancing:DeleteRule elasticloadbalancing:DeleteTargetGroup elasticloadbalancing:DeregisterInstancesFromLoadBalancer elasticloadbalancing:DeregisterTargets elasticloadbalancing:DescribeAccountLimits

Prefijo de servicio	Acciones
	elasticloadbalancing:DescribeInstanceHealth elasticloadbalancing:DescribeListenerCertificates elasticloadbalancing:DescribeListeners elasticloadbalancing:DescribeLoadBalancerAttributes elasticloadbalancing:DescribeLoadBalancerPolicies elasticloadbalancing:DescribeLoadBalancerPolicyTypes elasticloadbalancing:DescribeLoadBalancers elasticloadbalancing:DescribeRules elasticloadbalancing:DescribeSSLPolicies elasticloadbalancing:DescribeTargetGroupAttributes elasticloadbalancing:DescribeTargetGroups elasticloadbalancing:DescribeTargetHealth elasticloadbalancing:DetachLoadBalancerFromSubnets elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer elasticloadbalancing:ModifyListener elasticloadbalancing:ModifyLoadBalancerAttributes elasticloadbalancing:ModifyRule elasticloadbalancing:ModifyTargetGroup elasticloadbalancing:ModifyTargetGroupAttributes elasticloadbalancing:RegisterInstancesWithLoadBalancer

Prefijo de servicio	Acciones
	elasticloadbalancing:RegisterTargets elasticloadbalancing:RemoveListenerCertificates elasticloadbalancing:SetIpAddressType elasticloadbalancing:SetLoadBalancerListenerSSLCertificate elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer elasticloadbalancing:SetLoadBalancerPoliciesOfListener elasticloadbalancing:SetRulePriorities elasticloadbalancing:SetSecurityGroups elasticloadbalancing:SetSubnets

Prefijo de servicio	Acciones
elastictranscoder	elastictranscoder:CancelJob
	elastictranscoder:CreateJob
	elastictranscoder:CreatePipeline
	elastictranscoder:CreatePreset
	elastictranscoder:DeletePipeline
	elastictranscoder:DeletePreset
	elastictranscoder:ListJobsByPipeline
	elastictranscoder:ListJobsByStatus
	elastictranscoder:ListPipelines
	elastictranscoder:ListPresets
	elastictranscoder:ReadJob
	elastictranscoder:ReadPipeline
	elastictranscoder:ReadPreset
	elastictranscoder:TestRole
	elastictranscoder:UpdatePipeline
	elastictranscoder:UpdatePipelineNotifications
	elastictranscoder:UpdatePipelineStatus

Prefijo de servicio	Acciones
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListVirtualClusters emr-containers:StartJobRun

Prefijo de servicio	Acciones
emr-serverless	emr-serverless:CancelJobRun
	emr-serverless:CreateApplication
	emr-serverless:DeleteApplication
	emr-serverless:GetApplication
	emr-serverless:GetDashboardForJobRun
	emr-serverless:GetJobRun
	emr-serverless:ListApplications
	emr-serverless:ListJobRuns
	emr-serverless:StartApplication
	emr-serverless:StartJobRun
	emr-serverless:StopApplication
	emr-serverless:UpdateApplication

Prefijo de servicio	Acciones
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection es:DeletePackage es:DeleteVpcEndpoint

Prefijo de servicio	Acciones
	es:DescribeDomain es:DescribeDomainAutoTunes es:DescribeDomainChangeProgress es:DescribeDomainConfig es:DescribeDomainHealth es:DescribeDomainNodes es:DescribeDomains es:DescribeDryRunProgress es:DescribeElasticsearchDomain es:DescribeElasticsearchDomainConfig es:DescribeElasticsearchDomains es:DescribeElasticsearchInstanceTypeLimits es:DescribeInboundConnections es:DescribeInboundCrossClusterSearchConnections es:DescribeInstanceTypeLimits es:DescribeOutboundConnections es:DescribeOutboundCrossClusterSearchConnections es:DescribePackages es:DescribeReservedElasticsearchInstanceOfferings es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings

Prefijo de servicio	Acciones
	es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage es:GetCompatibleElasticsearchVersions es:GetCompatibleVersions es:GetDomainMaintenanceStatus es:GetPackageVersionHistory es:GetUpgradeHistory es:GetUpgradeStatus es:ListDomainNames es:ListDomainsForPackage es:ListElasticsearchInstanceTypes es:ListElasticsearchVersions es:ListInstanceTypeDetails es:ListPackagesForDomain es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering

Prefijo de servicio	Acciones
	es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es:UpdatePackage es:UpdateScheduledAction es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

Prefijo de servicio	Acciones
eventos	events:ActivateEventSource
	events:CancelReplay
	events:CreateApiDestination
	events:CreateArchive
	events:CreateConnection
	events:CreateEndpoint
	events:CreateEventBus
	events:CreatePartnerEventSource
	events:DeactivateEventSource
	events:DeauthorizeConnection
	events:DeleteApiDestination
	events:DeleteArchive
	events:DeleteConnection
	events:DeleteEndpoint
	events:DeleteEventBus
	events:DeletePartnerEventSource
	events:DeleteRule
	events:DescribeApiDestination
	events:DescribeArchive
	events:DescribeConnection
	events:DescribeEndpoint

Prefijo de servicio	Acciones
	events:DescribeEventBus
	events:DescribeEventSource
	events:DescribePartnerEventSource
	events:DescribeReplay
	events:DescribeRule
	events:DisableRule
	events:EnableRule
	events:ListApiDestinations
	events:ListArchives
	events:ListConnections
	events:ListEndpoints
	events:ListEventBuses
	events:ListEventSources
	events:ListPartnerEventSourceAccounts
	events:ListPartnerEventSources
	events:ListReplays
	events:ListRuleNamesByTarget
	events:ListRules
	events:ListTargetsByRule
	events:PutPermission
	events:PutRule

Prefijo de servicio	Acciones
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint

Prefijo de servicio	Acciones
evidently	evidently:CreateExperiment evidently:CreateFeature evidently:CreateLaunch evidently:CreateProject evidently:CreateSegment evidently:DeleteExperiment evidently:DeleteFeature evidently:DeleteLaunch evidently:DeleteProject evidently:DeleteSegment evidently:GetExperiment evidently:GetExperimentResults evidently:GetFeature evidently:GetLaunch evidently:GetProject evidently:GetSegment evidently:ListExperiments evidently:ListFeatures evidently:ListLaunches evidently:ListProjects evidently:ListSegmentReferences

Prefijo de servicio	Acciones
	evidently:ListSegments evidently:StartExperiment evidently:StartLaunch evidently:StopExperiment evidently:StopLaunch evidently:TestSegmentPattern evidently:UpdateExperiment evidently:UpdateFeature evidently:UpdateLaunch evidently:UpdateProject evidently:UpdateProjectDataDelivery

Prefijo de servicio	Acciones
finspace	finspace:CreateEnvironment
	finspace:CreateKxChangeset
	finspace:CreateKxCluster
	finspace:CreateKxDatabase
	finspace:CreateKxEnvironment
	finspace:CreateKxUser
	finspace:CreateUser
	finspace:DeleteEnvironment
	finspace:DeleteKxCluster
	finspace:DeleteKxDatabase
	finspace:DeleteKxEnvironment
	finspace:DeleteKxUser
	finspace:GetEnvironment
	finspace:GetKxChangeset
	finspace:GetKxCluster
	finspace:GetKxConnectionString
	finspace:GetKxDatabase
	finspace:GetKxEnvironment
	finspace:GetKxUser
	finspace:GetLoadSampleDataSetGroupIntoEnvironmentStatus
	finspace:GetUser

Prefijo de servicio	Acciones
	finspace:ListEnvironments finspace:ListKxChangesets finspace:ListKxClusterNodes finspace:ListKxClusters finspace:ListKxDatabases finspace:ListKxEnvironments finspace:ListKxUsers finspace:ListUsers finspace:LoadSampleDataSetGroupIntoEnvironment finspace:ResetUserPassword finspace:UpdateEnvironment finspace:UpdateKxClusterDatabases finspace:UpdateKxDatabase finspace:UpdateKxEnvironment finspace:UpdateKxEnvironmentNetwork finspace:UpdateKxUser finspace:UpdateUser

Prefijo de servicio	Acciones
firehose	firehose:CreateDeliveryStream firehose:DeleteDeliveryStream firehose:DescribeDeliveryStream firehose:ListDeliveryStreams firehose:StartDeliveryStreamEncryption firehose:StopDeliveryStreamEncryption firehose:UpdateDestination
fis	fis:CreateExperimentTemplate fis:DeleteExperimentTemplate fis:GetAction fis:GetExperiment fis:GetExperimentTemplate fis:GetTargetResourceType fis:ListActions fis:ListExperiments fis:ListExperimentTemplates fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate

Prefijo de servicio	Acciones
fms	fms:AssociateAdminAccount fms:AssociateThirdPartyFirewall fms:BatchAssociateResource fms:BatchDisassociateResource fms:DeleteAppsList fms:DeleteNotificationChannel fms:DeletePolicy fms:DeleteProtocolsList fms:DeleteResourceSet fms:DisassociateAdminAccount fms:DisassociateThirdPartyFirewall fms:GetAdminAccount fms:GetAdminScope fms:GetAppsList fms:GetComplianceDetail fms:GetNotificationChannel fms:GetPolicy fms:GetProtectionStatus fms:GetProtocolsList fms:GetResourceSet fms:GetThirdPartyFirewallAssociationStatus

Prefijo de servicio	Acciones
	fms:GetViolationDetails fms:ListAdminAccountsForOrganization fms:ListAdminsManagingAccount fms:ListAppsLists fms:ListComplianceStatus fms:ListDiscoveredResources fms:ListMemberAccounts fms:ListPolicies fms:ListProtocolsLists fms:ListResourceSetResources fms:ListResourceSets fms:ListThirdPartyFirewallFirewallPolicies fms:PutAdminAccount fms:PutAppsList fms:PutNotificationChannel fms:PutPolicy fms:PutProtocolsList fms:PutResourceSet

Prefijo de servicio	Acciones
frauddetector	frauddetector:BatchCreateVariable
	frauddetector:BatchGetVariable
	frauddetector:CancelBatchImportJob
	frauddetector:CancelBatchPredictionJob
	frauddetector:CreateBatchImportJob
	frauddetector:CreateBatchPredictionJob
	frauddetector:CreateDetectorVersion
	frauddetector:CreateList
	frauddetector:CreateModel
	frauddetector:CreateModelVersion
	frauddetector:CreateRule
	frauddetector:CreateVariable
	frauddetector:DeleteBatchImportJob
	frauddetector:DeleteBatchPredictionJob
	frauddetector:DeleteDetector
	frauddetector:DeleteDetectorVersion
	frauddetector:DeleteEntityType
	frauddetector:DeleteEvent
	frauddetector:DeleteEventsByEventType
	frauddetector:DeleteEventType
	frauddetector:DeleteExternalModel

Prefijo de servicio	Acciones
	frauddetector:DeleteLabel
	frauddetector:DeleteList
	frauddetector:DeleteModel
	frauddetector:DeleteModelVersion
	frauddetector:DeleteOutcome
	frauddetector:DeleteRule
	frauddetector:DeleteVariable
	frauddetector:DescribeDetector
	frauddetector:DescribeModelVersions
	frauddetector:GetBatchImportJobs
	frauddetector:GetBatchPredictionJobs
	frauddetector:GetDeleteEventsByEventTypeStatus
	frauddetector:GetDetectors
	frauddetector:GetDetectorVersion
	frauddetector:GetEntityTypes
	frauddetector:GetEvent
	frauddetector:GetEventPrediction
	frauddetector:GetEventPredictionMetadata
	frauddetector:GetEventTypes
	frauddetector:GetExternalModels
	frauddetector:GetKMSEncryptionKey

Prefijo de servicio	Acciones
	frauddetector:GetLabels
	frauddetector:GetListElements
	frauddetector:GetListsMetadata
	frauddetector:GetModels
	frauddetector:GetModelVersion
	frauddetector:GetOutcomes
	frauddetector:GetRules
	frauddetector:GetVariables
	frauddetector:ListEventPredictions
	frauddetector:PutDetector
	frauddetector:PutEntityType
	frauddetector:PutEventType
	frauddetector:PutExternalModel
	frauddetector:PutKMSEncryptionKey
	frauddetector:PutLabel
	frauddetector:PutOutcome
	frauddetector:SendEvent
	frauddetector:UpdateDetectorVersion
	frauddetector:UpdateDetectorVersionMetadata
	frauddetector:UpdateDetectorVersionStatus
	frauddetector:UpdateEventLabel

Prefijo de servicio	Acciones
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

Prefijo de servicio	Acciones
fsx	fsx:AssociateFileSystemAliases fsx:CancelDataRepositoryTask fsx:CopyBackup fsx:CreateDataRepositoryTask fsx:CreateFileCache fsx:CreateFileSystem fsx:CreateFileSystemFromBackup fsx:CreateSnapshot fsx:CreateStorageVirtualMachine fsx:CreateVolume fsx:CreateVolumeFromBackup fsx:DeleteBackup fsx:DeleteFileCache fsx:DeleteFileSystem fsx:DeleteSnapshot fsx:DeleteStorageVirtualMachine fsx:DeleteVolume fsx:DescribeBackups fsx:DescribeDataRepositoryAssociations fsx:DescribeDataRepositoryTasks fsx:DescribeFileCaches

Prefijo de servicio	Acciones
	fsx:DescribeFileSystemAliases fsx:DescribeFileSystems fsx:DescribeSnapshots fsx:DescribeStorageVirtualMachines fsx:DescribeVolumes fsx:DisassociateFileSystemAliases fsx:ReleaseFileSystemNfsV3Locks fsx:RestoreVolumeFromSnapshot fsx:StartMisconfiguredStateRecovery fsx:UpdateDataRepositoryAssociation fsx:UpdateFileCache fsx:UpdateFileSystem fsx:UpdateSnapshot fsx:UpdateStorageVirtualMachine fsx:UpdateVolume

Prefijo de servicio	Acciones
gamelift	gamelift:AcceptMatch
	gamelift:ClaimGameServer
	gamelift:CreateAlias
	gamelift:CreateBuild
	gamelift:CreateFleet
	gamelift:CreateFleetLocations
	gamelift:CreateGameServerGroup
	gamelift:CreateGameSession
	gamelift:CreateGameSessionQueue
	gamelift:CreateLocation
	gamelift:CreateMatchmakingConfiguration
	gamelift:CreateMatchmakingRuleSet
	gamelift:CreatePlayerSession
	gamelift:CreatePlayerSessions
	gamelift:CreateScript
	gamelift:CreateVpcPeeringAuthorization
	gamelift:CreateVpcPeeringConnection
	gamelift:DeleteAlias
	gamelift:DeleteBuild
	gamelift:DeleteFleet
	gamelift:DeleteFleetLocations

Prefijo de servicio	Acciones
	gamelift:DeleteGameServerGroup
	gamelift:DeleteGameSessionQueue
	gamelift:DeleteLocation
	gamelift:DeleteMatchmakingConfiguration
	gamelift:DeleteMatchmakingRuleSet
	gamelift:DeleteScalingPolicy
	gamelift:DeleteScript
	gamelift:DeleteVpcPeeringAuthorization
	gamelift:DeleteVpcPeeringConnection
	gamelift:DeregisterCompute
	gamelift:DeregisterGameServer
	gamelift:DescribeAlias
	gamelift:DescribeBuild
	gamelift:DescribeCompute
	gamelift:DescribeEC2InstanceLimits
	gamelift:DescribeFleetAttributes
	gamelift:DescribeFleetCapacity
	gamelift:DescribeFleetEvents
	gamelift:DescribeFleetLocationAttributes
	gamelift:DescribeFleetLocationCapacity
	gamelift:DescribeFleetLocationUtilization

Prefijo de servicio	Acciones
	gamelift:DescribeFleetPortSettings
	gamelift:DescribeFleetUtilization
	gamelift:DescribeGameServer
	gamelift:DescribeGameServerGroup
	gamelift:DescribeGameServerInstances
	gamelift:DescribeGameSessionDetails
	gamelift:DescribeGameSessionPlacement
	gamelift:DescribeGameSessionQueues
	gamelift:DescribeGameSessions
	gamelift:DescribeInstances
	gamelift:DescribeMatchmaking
	gamelift:DescribeMatchmakingConfigurations
	gamelift:DescribeMatchmakingRuleSets
	gamelift:DescribePlayerSessions
	gamelift:DescribeRuntimeConfiguration
	gamelift:DescribeScalingPolicies
	gamelift:DescribeScript
	gamelift:DescribeVpcPeeringAuthorizations
	gamelift:DescribeVpcPeeringConnections
	gamelift:GetComputeAccess
	gamelift:GetComputeAuthToken

Prefijo de servicio	Acciones
	gamelift:GetGameSessionLogUrl
	gamelift:GetInstanceAccess
	gamelift:ListAliases
	gamelift:ListBuilds
	gamelift:ListCompute
	gamelift:ListFleets
	gamelift:ListGameServerGroups
	gamelift:ListGameServers
	gamelift:ListLocations
	gamelift:ListScripts
	gamelift:PutScalingPolicy
	gamelift:RegisterCompute
	gamelift:RegisterGameServer
	gamelift:RequestUploadCredentials
	gamelift:ResolveAlias
	gamelift:ResumeGameServerGroup
	gamelift:SearchGameSessions
	gamelift:StartFleetActions
	gamelift:StartGameSessionPlacement
	gamelift:StartMatchBackfill
	gamelift:StartMatchmaking

Prefijo de servicio	Acciones
	gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

Prefijo de servicio	Acciones
geo	geo:AssociateTrackerConsumer geo:BatchDeleteDevicePositionHistory geo:BatchDeleteGeofence geo:BatchEvaluateGeofences geo:BatchGetDevicePosition geo:BatchPutGeofence geo:BatchUpdateDevicePosition geo:CalculateRoute geo:CalculateRouteMatrix geo:CreateGeofenceCollection geo:CreateMap geo:CreatePlaceIndex geo:CreateRouteCalculator geo:CreateTracker geo:DeleteGeofenceCollection geo:DeleteKey geo:DeleteMap geo:DeletePlaceIndex geo:DeleteRouteCalculator geo:DeleteTracker geo:DescribeGeofenceCollection

Prefijo de servicio	Acciones
	geo:DescribeKey geo:DescribeMap geo:DescribePlaceIndex geo:DescribeRouteCalculator geo:DescribeTracker geo:DisassociateTrackerConsumer geo:GetDevicePosition geo:GetDevicePositionHistory geo:GetGeofence geo:GetMapGlyphs geo:GetMapSprites geo:GetMapStyleDescriptor geo:GetMapTile geo:GetPlace geo:ListDevicePositions geo:ListGeofenceCollections geo:ListGeofences geo:ListKeys geo:ListMaps geo:ListPlaceIndexes geo:ListRouteCalculators

Prefijo de servicio	Acciones
	geo:ListTrackerConsumers geo:ListTrackers geo:PutGeofence geo:SearchPlaceIndexForPosition geo:SearchPlaceIndexForSuggestions geo:SearchPlaceIndexForText geo:UpdateGeofenceCollection geo:UpdateKey geo:UpdateMap geo:UpdatePlaceIndex geo:UpdateRouteCalculator geo:UpdateTracker

Prefijo de servicio	Acciones
glacier	glacier:AbortMultipartUpload
	glacier:AbortVaultLock
	glacier:CompleteMultipartUpload
	glacier:CompleteVaultLock
	glacier:CreateVault
	glacier:DeleteArchive
	glacier:DeleteVault
	glacier:DeleteVaultAccessPolicy
	glacier:DeleteVaultNotifications
	glacier:DescribeJob
	glacier:DescribeVault
	glacier:GetDataRetrievalPolicy
	glacier:GetJobOutput
	glacier:GetVaultAccessPolicy
	glacier:GetVaultLock
	glacier:GetVaultNotifications
	glacier:InitiateJob
	glacier:InitiateMultipartUpload
	glacier:InitiateVaultLock
	glacier:ListJobs
	glacier:ListMultipartUploads

Prefijo de servicio	Acciones
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

Prefijo de servicio	Acciones
grafana	grafana:AssociateLicense grafana:CreateWorkspace grafana:CreateWorkspaceApiKey grafana:DeleteWorkspace grafana:DeleteWorkspaceApiKey grafana:DescribeWorkspace grafana:DescribeWorkspaceAuthentication grafana:DescribeWorkspaceConfiguration grafana:DisassociateLicense grafana:ListPermissions grafana:ListVersions grafana:ListWorkspaces grafana:UpdatePermissions grafana:UpdateWorkspace grafana:UpdateWorkspaceAuthentication grafana:UpdateWorkspaceConfiguration

Prefijo de servicio	Acciones
greengrass	greengrass:AssociateRoleToGroup
	greengrass:AssociateServiceRoleToAccount
	greengrass:BatchAssociateClientDeviceWithCoreDevice
	greengrass:BatchDisassociateClientDeviceFromCoreDevice
	greengrass:CancelDeployment
	greengrass:CreateComponentVersion
	greengrass:CreateConnectorDefinition
	greengrass:CreateConnectorDefinitionVersion
	greengrass:CreateCoreDefinition
	greengrass:CreateCoreDefinitionVersion
	greengrass:CreateDeployment
	greengrass:CreateDeviceDefinition
	greengrass:CreateDeviceDefinitionVersion
	greengrass:CreateFunctionDefinition
	greengrass:CreateFunctionDefinitionVersion
	greengrass:CreateGroup
	greengrass:CreateGroupCertificateAuthority
	greengrass:CreateGroupVersion
	greengrass:CreateLoggerDefinition
	greengrass:CreateLoggerDefinitionVersion
	greengrass:CreateResourceDefinition

Prefijo de servicio	Acciones
	greengrass:CreateResourceDefinitionVersion
	greengrass:CreateSoftwareUpdateJob
	greengrass:CreateSubscriptionDefinition
	greengrass:CreateSubscriptionDefinitionVersion
	greengrass:DeleteComponent
	greengrass:DeleteConnectorDefinition
	greengrass:DeleteCoreDefinition
	greengrass:DeleteCoreDevice
	greengrass:DeleteDeployment
	greengrass:DeleteDeviceDefinition
	greengrass:DeleteFunctionDefinition
	greengrass:DeleteGroup
	greengrass:DeleteLoggerDefinition
	greengrass:DeleteResourceDefinition
	greengrass:DeleteSubscriptionDefinition
	greengrass:DescribeComponent
	greengrass:DisassociateRoleFromGroup
	greengrass:DisassociateServiceRoleFromAccount
	greengrass:GetAssociatedRole
	greengrass:GetBulkDeploymentStatus
	greengrass:GetComponent

Prefijo de servicio	Acciones
	greengrass:GetComponentVersionArtifact
	greengrass:GetConnectivityInfo
	greengrass:GetConnectorDefinition
	greengrass:GetConnectorDefinitionVersion
	greengrass:GetCoreDefinition
	greengrass:GetCoreDefinitionVersion
	greengrass:GetCoreDevice
	greengrass:GetDeployment
	greengrass:GetDeploymentStatus
	greengrass:GetDeviceDefinition
	greengrass:GetDeviceDefinitionVersion
	greengrass:GetFunctionDefinition
	greengrass:GetFunctionDefinitionVersion
	greengrass:GetGroup
	greengrass:GetGroupCertificateAuthority
	greengrass:GetGroupCertificateConfiguration
	greengrass:GetGroupVersion
	greengrass:GetLoggerDefinition
	greengrass:GetLoggerDefinitionVersion
	greengrass:GetResourceDefinition
	greengrass:GetResourceDefinitionVersion

Prefijo de servicio	Acciones
	greengrass:GetServiceRoleForAccount
	greengrass:GetSubscriptionDefinition
	greengrass:GetSubscriptionDefinitionVersion
	greengrass:GetThingRuntimeConfiguration
	greengrass:ListBulkDeploymentDetailedReports
	greengrass:ListBulkDeployments
	greengrass:ListClientDevicesAssociatedWithCoreDevice
	greengrass:ListComponents
	greengrass:ListComponentVersions
	greengrass:ListConnectorDefinitions
	greengrass:ListConnectorDefinitionVersions
	greengrass:ListCoreDefinitions
	greengrass:ListCoreDefinitionVersions
	greengrass:ListCoreDevices
	greengrass:ListDeployments
	greengrass:ListDeviceDefinitions
	greengrass:ListDeviceDefinitionVersions
	greengrass:ListEffectiveDeployments
	greengrass:ListFunctionDefinitions
	greengrass:ListFunctionDefinitionVersions
	greengrass:ListGroupCertificateAuthorities

Prefijo de servicio	Acciones
	greengrass:ListGroups
	greengrass:ListGroupVersions
	greengrass:ListInstalledComponents
	greengrass:ListLoggerDefinitions
	greengrass:ListLoggerDefinitionVersions
	greengrass:ListResourceDefinitions
	greengrass:ListResourceDefinitionVersions
	greengrass:ListSubscriptionDefinitions
	greengrass:ListSubscriptionDefinitionVersions
	greengrass:ResetDeployments
	greengrass:StartBulkDeployment
	greengrass:StopBulkDeployment
	greengrass:UpdateConnectivityInfo
	greengrass:UpdateConnectorDefinition
	greengrass:UpdateCoreDefinition
	greengrass:UpdateDeviceDefinition
	greengrass:UpdateFunctionDefinition
	greengrass:UpdateGroup
	greengrass:UpdateGroupCertificateConfiguration
	greengrass:UpdateLoggerDefinition
	greengrass:UpdateResourceDefinition

Prefijo de servicio	Acciones
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

Prefijo de servicio	Acciones
groundstation	groundstation:CancelContact groundstation:CreateConfig groundstation:CreateDataflowEndpointGroup groundstation:CreateEphemeris groundstation:CreateMissionProfile groundstation:DeleteConfig groundstation:DeleteDataflowEndpointGroup groundstation:DeleteEphemeris groundstation:DeleteMissionProfile groundstation:DescribeContact groundstation:DescribeEphemeris groundstation:GetConfig groundstation:GetDataflowEndpointGroup groundstation:GetMinuteUsage groundstation:GetMissionProfile groundstation:GetSatellite groundstation:ListConfigs groundstation:ListContacts groundstation:ListDataflowEndpointGroups groundstation:ListEphemerides groundstation:ListGroundStations

Prefijo de servicio	Acciones
	groundstation:ListMissionProfiles groundstation:ListSatellites groundstation:RegisterAgent groundstation:ReserveContact groundstation:UpdateAgentStatus groundstation:UpdateConfig groundstation:UpdateEphemeris groundstation:UpdateMissionProfile

Prefijo de servicio	Acciones
guardduty	guardduty:AcceptAdministratorInvitation
	guardduty:AcceptInvitation
	guardduty:ArchiveFindings
	guardduty:CreateDetector
	guardduty:CreateFilter
	guardduty:CreateIPSet
	guardduty:CreateMembers
	guardduty:CreatePublishingDestination
	guardduty:CreateSampleFindings
	guardduty:CreateThreatIntelSet
	guardduty:DeclineInvitations
	guardduty:DeleteDetector
	guardduty:DeleteFilter
	guardduty:DeleteInvitations
	guardduty:DeleteIPSet
	guardduty:DeleteMembers
	guardduty:DeletePublishingDestination
	guardduty:DeleteThreatIntelSet
	guardduty:DescribeMalwareScans
	guardduty:DescribeOrganizationConfiguration
	guardduty:DescribePublishingDestination

Prefijo de servicio	Acciones
	guardduty:DisableOrganizationAdminAccount
	guardduty:DisassociateFromAdministratorAccount
	guardduty:DisassociateFromMasterAccount
	guardduty:DisassociateMembers
	guardduty:EnableOrganizationAdminAccount
	guardduty:GetAdministratorAccount
	guardduty:GetCoverageStatistics
	guardduty:GetDetector
	guardduty:GetFilter
	guardduty:GetFindings
	guardduty:GetFindingsStatistics
	guardduty:GetInvitationsCount
	guardduty:GetIPSet
	guardduty:GetMalwareScanSettings
	guardduty:GetMasterAccount
	guardduty:GetMemberDetectors
	guardduty:GetMembers
	guardduty:GetRemainingFreeTrialDays
	guardduty:GetThreatIntelSet
	guardduty:GetUsageStatistics
	guardduty:InviteMembers

Prefijo de servicio	Acciones
	guardduty:ListCoverage
	guardduty:ListDetectors
	guardduty:ListFilters
	guardduty:ListFindings
	guardduty:ListInvitations
	guardduty:ListIPSets
	guardduty:ListMembers
	guardduty:ListOrganizationAdminAccounts
	guardduty:ListPublishingDestinations
	guardduty:ListThreatIntelSets
	guardduty:SendSecurityTelemetry
	guardduty:StartMalwareScan
	guardduty:StartMonitoringMembers
	guardduty:StopMonitoringMembers
	guardduty:UnarchiveFindings
	guardduty:UpdateDetector
	guardduty:UpdateFilter
	guardduty:UpdateFindingsFeedback
	guardduty:UpdateIPSet
	guardduty:UpdateMalwareScanSettings
	guardduty:UpdateMemberDetectors

Prefijo de servicio	Acciones
	guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatIntelSet
healthlake	healthlake:CreateFHIRDatastore healthlake:CreateResource healthlake:DeleteFHIRDatastore healthlake:DeleteResource healthlake:DescribeFHIRDatastore healthlake:DescribeFHIRExportJob healthlake:DescribeFHIRImportJob healthlake:GetCapabilities healthlake:ListFHIRDatastores healthlake:ListFHIRExportJobs healthlake:ListFHIRImportJobs healthlake:ReadResource healthlake:SearchWithGet healthlake:SearchWithPost healthlake:StartFHIRExportJob healthlake:StartFHIRImportJob healthlake:UpdateResource

Prefijo de servicio	Acciones
honeycode	honeycode:BatchCreateTableRows honeycode:BatchDeleteTableRows honeycode:BatchUpdateTableRows honeycode:BatchUpsertTableRows honeycode:DescribeTableDataImportJob honeycode:GetScreenData honeycode:InvokeScreenAutomation honeycode:ListTableColumns honeycode:ListTableRows honeycode:ListTables honeycode:QueryTableRows honeycode:StartTableDataImportJob

Prefijo de servicio	Acciones
iam	iam:AddClientIDToOpenIDConnectProvider iam:AddRoleToInstanceProfile iam:AddUserToGroup iam:AttachGroupPolicy iam:AttachRolePolicy iam:AttachUserPolicy iam:ChangePassword iam:CreateAccessKey iam:CreateAccountAlias iam:CreateGroup iam:CreateInstanceProfile iam:CreateLoginProfile iam:CreateOpenIDConnectProvider iam:CreatePolicy iam:CreatePolicyVersion iam:CreateRole iam:CreateSAMLProvider iam:CreateServiceLinkedRole iam:CreateServiceSpecificCredential iam:CreateUser iam:CreateVirtualMFADevice

Prefijo de servicio	Acciones
	iam:DeactivateMFADevice iam:DeleteAccessKey iam:DeleteAccountAlias iam:DeleteAccountPasswordPolicy iam:DeleteCloudFrontPublicKey iam:DeleteGroup iam:DeleteGroupPolicy iam:DeleteInstanceProfile iam:DeleteLoginProfile iam:DeleteOpenIDConnectProvider iam:DeletePolicy iam:DeletePolicyVersion iam:DeleteRole iam:DeleteRolePermissionsBoundary iam:DeleteRolePolicy iam:DeleteSAMLProvider iam:DeleteServerCertificate iam:DeleteServiceLinkedRole iam:DeleteServiceSpecificCredential iam:DeleteSigningCertificate iam:DeleteSSHPublicKey

Prefijo de servicio	Acciones
	iam:DeleteUser iam:DeleteUserPermissionsBoundary iam:DeleteUserPolicy iam:DeleteVirtualMFADevice iam:DetachGroupPolicy iam:DetachRolePolicy iam:DetachUserPolicy iam:EnableMFADevice iam:GenerateCredentialReport iam:GenerateOrganizationsAccessReport iam:GenerateServiceLastAccessedDetails iam:GetAccessKeyLastUsed iam:GetAccountAuthorizationDetails iam:GetAccountEmailAddress iam:GetAccountName iam:GetAccountPasswordPolicy iam:GetAccountSummary iam:GetCloudFrontPublicKey iam:GetContextKeysForCustomPolicy iam:GetContextKeysForPrincipalPolicy iam:GetCredentialReport

Prefijo de servicio	Acciones
	iam:GetGroup iam:GetGroupPolicy iam:GetInstanceProfile iam:GetLoginProfile iam:GetMFADevice iam:GetOpenIDConnectProvider iam:GetOrganizationsAccessReport iam:GetPolicy iam:GetPolicyVersion iam:GetRole iam:GetRolePolicy iam:GetSAMLProvider iam:GetServerCertificate iam:GetServiceLastAccessedDetails iam:GetServiceLastAccessedDetailsWithEntities iam:GetServiceLinkedRoleDeletionStatus iam:GetSSHPublicKey iam:GetUser iam:GetUserPolicy iam:ListAccessKeys iam:ListAccountAliases

Prefijo de servicio	Acciones
	iam:ListAttachedGroupPolicies iam:ListAttachedRolePolicies iam:ListAttachedUserPolicies iam:ListCloudFrontPublicKeys iam:ListEntitiesForPolicy iam:ListGroupPolicies iam:ListGroups iam:ListGroupsForUser iam:ListInstanceProfiles iam:ListInstanceProfilesForRole iam:ListMFADevices iam:ListOpenIDConnectProviders iam:ListPolicies iam:ListPoliciesGrantingServiceAccess iam:ListPolicyVersions iam:ListRolePolicies iam:ListRoles iam:ListSAMLProviders iam:ListServerCertificates iam:ListServiceSpecificCredentials iam:ListSigningCertificates

Prefijo de servicio	Acciones
	iam:ListSSHPublicKeys iam:ListSTSRegionalEndpointsStatus iam:ListUserPolicies iam:ListUsers iam:ListVirtualMFADevices iam:PutGroupPolicy iam:PutRolePermissionsBoundary iam:PutRolePolicy iam:PutUserPermissionsBoundary iam:PutUserPolicy iam:RemoveClientIDFromOpenIDConnectProvider iam:RemoveRoleFromInstanceProfile iam:RemoveUserFromGroup iam:ResetServiceSpecificCredential iam:ResyncMFADevice iam:SetDefaultPolicyVersion iam:SetSecurityTokenServicePreferences iam:SetSTSRegionalEndpointStatus iam:SimulateCustomPolicy iam:SimulatePrincipalPolicy iam:UpdateAccessKey

Prefijo de servicio	Acciones
	iam:UpdateAccountEmailAddress iam:UpdateAccountName iam:UpdateAccountPasswordPolicy iam:UpdateAssumeRolePolicy iam:UpdateCloudFrontPublicKey iam:UpdateGroup iam:UpdateLoginProfile iam:UpdateOpenIDConnectProviderThumbprint iam:UpdateRole iam:UpdateRoleDescription iam:UpdateSAMLProvider iam:UpdateServerCertificate iam:UpdateServiceSpecificCredential iam:UpdateSigningCertificate iam:UpdateSSHPublicKey iam:UpdateUser iam:UploadCloudFrontPublicKey iam:UploadServerCertificate iam:UploadSigningCertificate iam:UploadSSHPublicKey

Prefijo de servicio	Acciones
identitystore	identitystore:CreateGroup identitystore:CreateGroupMembership identitystore:CreateUser identitystore:DeleteGroup identitystore:DeleteGroupMembership identitystore:DeleteUser identitystore:DescribeGroup identitystore:DescribeGroupMembership identitystore:DescribeUser identitystore:GetGroupId identitystore:GetGroupMembershipId identitystore:GetUserId identitystore:IsMemberInGroups identitystore:ListGroupMemberships identitystore:ListGroupMembershipsForMember identitystore:ListGroups identitystore:ListUsers identitystore:UpdateGroup identitystore:UpdateUser

Prefijo de servicio	Acciones
imagebuilder	imagebuilder:CancelImageCreation imagebuilder:CreateComponent imagebuilder:CreateContainerRecipe imagebuilder:CreateDistributionConfiguration imagebuilder:CreateImage imagebuilder:CreateImagePipeline imagebuilder:CreateImageRecipe imagebuilder:CreateInfrastructureConfiguration imagebuilder:DeleteComponent imagebuilder:DeleteContainerRecipe imagebuilder:DeleteDistributionConfiguration imagebuilder:DeleteImage imagebuilder:DeleteImagePipeline imagebuilder:DeleteImageRecipe imagebuilder:DeleteInfrastructureConfiguration imagebuilder:GetComponentPolicy imagebuilder:GetContainerRecipePolicy imagebuilder:GetImagePolicy imagebuilder:GetImageRecipePolicy imagebuilder:GetWorkflowExecution imagebuilder:GetWorkflowStepExecution

Prefijo de servicio	Acciones
	imagebuilder:ImportComponent imagebuilder:ImportVmImage imagebuilder:ListComponentBuildVersions imagebuilder:ListComponents imagebuilder:ListContainerRecipes imagebuilder:ListDistributionConfigurations imagebuilder:ListImageBuildVersions imagebuilder:ListImagePackages imagebuilder:ListImagePipelineImages imagebuilder:ListImagePipelines imagebuilder:ListImageRecipes imagebuilder:ListImages imagebuilder:ListImageScanFindingAggregations imagebuilder:ListImageScanFindings imagebuilder:ListInfrastructureConfigurations imagebuilder:ListWorkflowExecutions imagebuilder:ListWorkflowStepExecutions imagebuilder:PutComponentPolicy imagebuilder:PutContainerRecipePolicy imagebuilder:PutImagePolicy imagebuilder:PutImageRecipePolicy

Prefijo de servicio	Acciones
	imagebuilder:StartImagePipelineExecution imagebuilder:UpdateDistributionConfiguration imagebuilder:UpdateImagePipeline imagebuilder:UpdateInfrastructureConfiguration

Prefijo de servicio	Acciones
inspector	inspector:AddAttributesToFindings
	inspector:CreateAssessmentTarget
	inspector:CreateAssessmentTemplate
	inspector:CreateExclusionsPreview
	inspector:CreateResourceGroup
	inspector:DeleteAssessmentRun
	inspector:DeleteAssessmentTarget
	inspector:DeleteAssessmentTemplate
	inspector:DescribeAssessmentRuns
	inspector:DescribeAssessmentTargets
	inspector:DescribeAssessmentTemplates
	inspector:DescribeCrossAccountAccessRole
	inspector:DescribeExclusions
	inspector:DescribeFindings
	inspector:DescribeResourceGroups
	inspector:DescribeRulesPackages
	inspector:GetAssessmentReport
	inspector:GetExclusionsPreview
	inspector:GetTelemetryMetadata
	inspector:ListAssessmentRunAgents
	inspector:ListAssessmentRuns

Prefijo de servicio	Acciones
	inspector:ListAssessmentTargets
	inspector:ListAssessmentTemplates
	inspector:ListEventSubscriptions
	inspector:ListExclusions
	inspector:ListFindings
	inspector:ListRulesPackages
	inspector:PreviewAgents
	inspector:RegisterCrossAccountAccessRole
	inspector:RemoveAttributesFromFindings
	inspector:StartAssessmentRun
	inspector:StopAssessmentRun
	inspector:SubscribeToEvent
	inspector:UnsubscribeFromEvent
	inspector:UpdateAssessmentTarget

Prefijo de servicio	Acciones
inspector2	inspector2:AssociateMember inspector2:BatchGetAccountStatus inspector2:BatchGetCodeSnippet inspector2:BatchGetFindingDetails inspector2:BatchGetFreeTrialInfo inspector2:BatchGetMemberEc2DeepInspectionStatus inspector2:BatchUpdateMemberEc2DeepInspectionStatus inspector2:CancelFindingsReport inspector2:CancelSbomExport inspector2:CreateFilter inspector2:CreateFindingsReport inspector2:CreateSbomExport inspector2:DeleteFilter inspector2:DescribeOrganizationConfiguration inspector2:Disable inspector2:DisableDelegatedAdminAccount inspector2:DisassociateMember inspector2:Enable inspector2:EnableDelegatedAdminAccount inspector2:GetConfiguration inspector2:GetDelegatedAdminAccount

Prefijo de servicio	Acciones
	inspector2:GetEc2DeepInspectionConfiguration inspector2:GetEncryptionKey inspector2:GetFindingsReportStatus inspector2:GetMember inspector2:GetSbomExport inspector2:ListAccountPermissions inspector2:ListCoverage inspector2:ListCoverageStatistics inspector2:ListDelegatedAdminAccounts inspector2:ListFilters inspector2:ListFindingAggregations inspector2:ListFindings inspector2:ListMembers inspector2:ListUsageTotals inspector2:ResetEncryptionKey inspector2:SearchVulnerabilities inspector2:UpdateConfiguration inspector2:UpdateEc2DeepInspectionConfiguration inspector2:UpdateEncryptionKey inspector2:UpdateFilter inspector2:UpdateOrganizationConfiguration

Prefijo de servicio	Acciones
	inspector2:UpdateOrgEc2DeepInspectionConfiguration

Prefijo de servicio	Acciones
iot	iot:AcceptCertificateTransfer
	iot:AddThingToBillingGroup
	iot:AddThingToThingGroup
	iot:AssociateTargetsWithJob
	iot:AttachPolicy
	iot:AttachPrincipalPolicy
	iot:AttachSecurityProfile
	iot:AttachThingPrincipal
	iot:CancelAuditMitigationActionsTask
	iot:CancelAuditTask
	iot:CancelCertificateTransfer
	iot:CancelDetectMitigationActionsTask
	iot:CancelJob
	iot:CancelJobExecution
	iot:ClearDefaultAuthorizer
	iot:ConfirmTopicRuleDestination
	iot:CreateAuditSuppression
	iot:CreateAuthorizer
	iot:CreateBillingGroup
	iot:CreateCertificateFromCsr
	iot:CreateCustomMetric

Prefijo de servicio	Acciones
	iot:CreateDimension
	iot:CreateDomainConfiguration
	iot:CreateDynamicThingGroup
	iot:CreateFleetMetric
	iot:CreateJob
	iot:CreateJobTemplate
	iot:CreateKeysAndCertificate
	iot:CreateMitigationAction
	iot:CreateOTAUpdate
	iot:CreatePackage
	iot:CreatePackageVersion
	iot:CreatePolicy
	iot:CreatePolicyVersion
	iot:CreateProvisioningClaim
	iot:CreateProvisioningTemplate
	iot:CreateProvisioningTemplateVersion
	iot:CreateRoleAlias
	iot:CreateScheduledAudit
	iot:CreateSecurityProfile
	iot:CreateStream
	iot:CreateThing

Prefijo de servicio	Acciones
	iot:CreateThingGroup
	iot:CreateThingType
	iot:CreateTopicRule
	iot:CreateTopicRuleDestination
	iot:DeleteAccountAuditConfiguration
	iot:DeleteAuditSuppression
	iot:DeleteAuthorizer
	iot:DeleteBillingGroup
	iot:DeleteCACertificate
	iot:DeleteCertificate
	iot:DeleteCustomMetric
	iot:DeleteDimension
	iot:DeleteDomainConfiguration
	iot:DeleteDynamicThingGroup
	iot:DeleteFleetMetric
	iot:DeleteJob
	iot:DeleteJobExecution
	iot:DeleteJobTemplate
	iot:DeleteMitigationAction
	iot:DeleteOTAUpdate
	iot:DeletePackage

Prefijo de servicio	Acciones
	iot:DeletePackageVersion
	iot:DeletePolicy
	iot:DeletePolicyVersion
	iot:DeleteProvisioningTemplate
	iot:DeleteProvisioningTemplateVersion
	iot:DeleteRegistrationCode
	iot:DeleteRoleAlias
	iot:DeleteScheduledAudit
	iot:DeleteSecurityProfile
	iot:DeleteStream
	iot:DeleteThing
	iot:DeleteThingGroup
	iot:DeleteThingType
	iot:DeleteTopicRule
	iot:DeleteTopicRuleDestination
	iot:DeleteV2LoggingLevel
	iot:DeprecateThingType
	iot:DescribeAccountAuditConfiguration
	iot:DescribeAuditFinding
	iot:DescribeAuditMitigationActionsTask
	iot:DescribeAuditSuppression

Prefijo de servicio	Acciones
	iot:DescribeAuditTask
	iot:DescribeAuthorizer
	iot:DescribeBillingGroup
	iot:DescribeCACertificate
	iot:DescribeCertificate
	iot:DescribeCustomMetric
	iot:DescribeDefaultAuthorizer
	iot:DescribeDetectMitigationActionsTask
	iot:DescribeDimension
	iot:DescribeDomainConfiguration
	iot:DescribeEndpoint
	iot:DescribeEventConfigurations
	iot:DescribeFleetMetric
	iot:DescribeIndex
	iot:DescribeJob
	iot:DescribeJobExecution
	iot:DescribeJobTemplate
	iot:DescribeManagedJobTemplate
	iot:DescribeMitigationAction
	iot:DescribeProvisioningTemplate
	iot:DescribeProvisioningTemplateVersion

Prefijo de servicio	Acciones
	iot:DescribeRoleAlias
	iot:DescribeScheduledAudit
	iot:DescribeSecurityProfile
	iot:DescribeStream
	iot:DescribeThing
	iot:DescribeThingGroup
	iot:DescribeThingRegistrationTask
	iot:DescribeThingType
	iot:DetachPolicy
	iot:DetachPrincipalPolicy
	iot:DetachSecurityProfile
	iot:DetachThingPrincipal
	iot:DisableTopicRule
	iot:EnableTopicRule
	iot:GetBehaviorModelTrainingSummaries
	iot:GetBucketsAggregation
	iot:GetCardinality
	iot:GetEffectivePolicies
	iot:GetJobDocument
	iot:GetLoggingOptions
	iot:GetOTAUpdate

Prefijo de servicio	Acciones
	iot:GetPackage
	iot:GetPackageConfiguration
	iot:GetPackageVersion
	iot:GetPercentiles
	iot:GetPolicy
	iot:GetPolicyVersion
	iot:GetRegistrationCode
	iot:GetStatistics
	iot:GetTopicRule
	iot:GetTopicRuleDestination
	iot:GetV2LoggingOptions
	iot:ListActiveViolations
	iot:ListAttachedPolicies
	iot:ListAuditFindings
	iot:ListAuditMitigationActionsExecutions
	iot:ListAuditMitigationActionsTasks
	iot:ListAuditSuppressions
	iot:ListAuditTasks
	iot:ListAuthorizers
	iot:ListBillingGroups
	iot:ListCACertificates

Prefijo de servicio	Acciones
	iot:ListCertificates
	iot:ListCertificatesByCA
	iot:ListCustomMetrics
	iot:ListDetectMitigationActionsExecutions
	iot:ListDetectMitigationActionsTasks
	iot:ListDimensions
	iot:ListDomainConfigurations
	iot:ListFleetMetrics
	iot:ListIndices
	iot:ListJobExecutionsForJob
	iot:ListJobExecutionsForThing
	iot:ListJobs
	iot:ListJobTemplates
	iot:ListManagedJobTemplates
	iot:ListMetricValues
	iot:ListMitigationActions
	iot:ListOTAUpdates
	iot:ListOutgoingCertificates
	iot:ListPackages
	iot:ListPackageVersions
	iot:ListPolicies

Prefijo de servicio	Acciones
	iot:ListPolicyPrincipals
	iot:ListPolicyVersions
	iot:ListPrincipalPolicies
	iot:ListPrincipalThings
	iot:ListProvisioningTemplates
	iot:ListProvisioningTemplateVersions
	iot:ListRelatedResourcesForAuditFinding
	iot:ListRoleAliases
	iot:ListScheduledAudits
	iot:ListSecurityProfiles
	iot:ListSecurityProfilesForTarget
	iot:ListStreams
	iot:ListTargetsForPolicy
	iot:ListTargetsForSecurityProfile
	iot:ListThingGroups
	iot:ListThingGroupsForThing
	iot:ListThingPrincipals
	iot:ListThingRegistrationTaskReports
	iot:ListThingRegistrationTasks
	iot:ListThings
	iot:ListThingsInBillingGroup

Prefijo de servicio	Acciones
	iot:ListThingsInThingGroup
	iot:ListThingTypes
	iot:ListTopicRuleDestinations
	iot:ListTopicRules
	iot:ListV2LoggingLevels
	iot:ListViolationEvents
	iot:PutVerificationStateOnViolation
	iot:RegisterCACertificate
	iot:RegisterCertificate
	iot:RegisterCertificateWithoutCA
	iot:RegisterThing
	iot:RejectCertificateTransfer
	iot:RemoveThingFromBillingGroup
	iot:RemoveThingFromThingGroup
	iot:ReplaceTopicRule
	iot:SearchIndex
	iot:SetDefaultAuthorizer
	iot:SetDefaultPolicyVersion
	iot:SetLoggingOptions
	iot:SetV2LoggingLevel
	iot:SetV2LoggingOptions

Prefijo de servicio	Acciones
	iot:StartAuditMitigationActionsTask
	iot:StartDetectMitigationActionsTask
	iot:StartOnDemandAuditTask
	iot:StartThingRegistrationTask
	iot:StopThingRegistrationTask
	iot:TestAuthorization
	iot:TestInvokeAuthorizer
	iot:TransferCertificate
	iot:UpdateAccountAuditConfiguration
	iot:UpdateAuditSuppression
	iot:UpdateAuthorizer
	iot:UpdateBillingGroup
	iot:UpdateCACertificate
	iot:UpdateCertificate
	iot:UpdateCustomMetric
	iot:UpdateDimension
	iot:UpdateDomainConfiguration
	iot:UpdateDynamicThingGroup
	iot:UpdateEventConfigurations
	iot:UpdateFleetMetric
	iot:UpdateIndexingConfiguration

Prefijo de servicio	Acciones
	iot:UpdateJob iot:UpdateMitigationAction iot:UpdatePackage iot:UpdatePackageConfiguration iot:UpdatePackageVersion iot:UpdateProvisioningTemplate iot:UpdateRoleAlias iot:UpdateScheduledAudit iot:UpdateSecurityProfile iot:UpdateStream iot:UpdateThing iot:UpdateThingGroup iot:UpdateThingGroupsForThing iot:UpdateTopicRuleDestination iot:ValidateSecurityProfileBehaviors

Prefijo de servicio	Acciones
iotanalytics	iotanalytics:CancelPipelineReprocessing iotanalytics:CreateChannel iotanalytics:CreateDataset iotanalytics:CreateDatasetContent iotanalytics:CreateDatastore iotanalytics:CreatePipeline iotanalytics:DeleteChannel iotanalytics:DeleteDataset iotanalytics:DeleteDatasetContent iotanalytics:DeleteDatastore iotanalytics:DeletePipeline iotanalytics:DescribeChannel iotanalytics:DescribeDataset iotanalytics:DescribeDatastore iotanalytics:DescribeLoggingOptions iotanalytics:DescribePipeline iotanalytics:GetDatasetContent iotanalytics:ListChannels iotanalytics:ListDatasetContents iotanalytics:ListDatasets iotanalytics:ListDatastores

Prefijo de servicio	Acciones
	iotanalytics:ListPipelines iotanalytics:PutLoggingOptions iotanalytics:RunPipelineActivity iotanalytics:SampleChannelData iotanalytics:StartPipelineReprocessing iotanalytics:UpdateChannel iotanalytics:UpdateDataset iotanalytics:UpdateDatastore iotanalytics:UpdatePipeline
iotdeviceadvisor	iotdeviceadvisor:CreateSuiteDefinition iotdeviceadvisor:DeleteSuiteDefinition iotdeviceadvisor:GetEndpoint iotdeviceadvisor:GetSuiteDefinition iotdeviceadvisor:GetSuiteRun iotdeviceadvisor:GetSuiteRunReport iotdeviceadvisor:ListSuiteDefinitions iotdeviceadvisor:ListSuiteRuns iotdeviceadvisor:StartSuiteRun iotdeviceadvisor:StopSuiteRun iotdeviceadvisor:UpdateSuiteDefinition

Prefijo de servicio	Acciones
iotevents	iotevents:BatchAcknowledgeAlarm iotevents:BatchDeleteDetector iotevents:BatchDisableAlarm iotevents:BatchEnableAlarm iotevents:BatchResetAlarm iotevents:BatchSnoozeAlarm iotevents:BatchUpdateDetector iotevents:CreateAlarmModel iotevents:CreateDetectorModel iotevents:CreateInput iotevents:DeleteAlarmModel iotevents:DeleteDetectorModel iotevents:DeleteInput iotevents:DescribeAlarm iotevents:DescribeAlarmModel iotevents:DescribeDetector iotevents:DescribeDetectorModel iotevents:DescribeDetectorModelAnalysis iotevents:DescribeInput iotevents:DescribeLoggingOptions iotevents:GetDetectorModelAnalysisResults

Prefijo de servicio	Acciones
	iotevents:ListAlarmModels iotevents:ListAlarmModelVersions iotevents:ListAlarms iotevents:ListDetectorModels iotevents:ListDetectorModelVersions iotevents:ListDetectors iotevents:ListInputRoutings iotevents:ListInputs iotevents:PutLoggingOptions iotevents:StartDetectorModelAnalysis iotevents:UpdateAlarmModel iotevents:UpdateDetectorModel iotevents:UpdateInput
iotfleethub	iotfleethub:CreateApplication iotfleethub:DeleteApplication iotfleethub:DescribeApplication iotfleethub:ListApplications iotfleethub:UpdateApplication

Prefijo de servicio	Acciones
iotsitewise	iotsitewise:AssociateAssets iotsitewise:AssociateTimeSeriesToAssetProperty iotsitewise:BatchAssociateProjectAssets iotsitewise:BatchDisassociateProjectAssets iotsitewise:CreateAccessPolicy iotsitewise:CreateAsset iotsitewise:CreateAssetModel iotsitewise:CreateBulkImportJob iotsitewise:CreateDashboard iotsitewise:CreateGateway iotsitewise:CreatePortal iotsitewise:CreateProject iotsitewise:DeleteAccessPolicy iotsitewise:DeleteAsset iotsitewise:DeleteAssetModel iotsitewise:DeleteDashboard iotsitewise:DeleteGateway iotsitewise:DeletePortal iotsitewise:DeleteProject iotsitewise:DeleteTimeSeries iotsitewise:DescribeAccessPolicy

Prefijo de servicio	Acciones
	iotsitewise:DescribeAsset iotsitewise:DescribeAssetModel iotsitewise:DescribeAssetProperty iotsitewise:DescribeBulkImportJob iotsitewise:DescribeDashboard iotsitewise:DescribeDefaultEncryptionConfiguration iotsitewise:DescribeGateway iotsitewise:DescribeGatewayCapabilityConfiguration iotsitewise:DescribeLoggingOptions iotsitewise:DescribePortal iotsitewise:DescribeProject iotsitewise:DescribeStorageConfiguration iotsitewise:DescribeTimeSeries iotsitewise:DisassociateAssets iotsitewise:DisassociateTimeSeriesFromAssetProperty iotsitewise:ListAccessPolicies iotsitewise:ListAssetModelProperties iotsitewise:ListAssetModels iotsitewise:ListAssetProperties iotsitewise:ListAssetRelationships iotsitewise:ListAssets

Prefijo de servicio	Acciones
	iotsitewise:ListAssociatedAssets iotsitewise:ListBulkImportJobs iotsitewise:ListDashboards iotsitewise:ListGateways iotsitewise:ListPortals iotsitewise:ListProjectAssets iotsitewise:ListProjects iotsitewise:ListTimeSeries iotsitewise:PutDefaultEncryptionConfiguration iotsitewise:PutLoggingOptions iotsitewise:PutStorageConfiguration iotsitewise:UpdateAccessPolicy iotsitewise:UpdateAsset iotsitewise:UpdateAssetModel iotsitewise:UpdateAssetProperty iotsitewise:UpdateDashboard iotsitewise:UpdateGateway iotsitewise:UpdateGatewayCapabilityConfiguration iotsitewise:UpdatePortal iotsitewise:UpdateProject

Prefijo de servicio	Acciones
iottwinmaker	iottwinmaker:CreateComponentType
	iottwinmaker:CreateEntity
	iottwinmaker:CreateScene
	iottwinmaker:CreateSyncJob
	iottwinmaker:CreateWorkspace
	iottwinmaker:DeleteComponentType
	iottwinmaker:DeleteEntity
	iottwinmaker:DeleteScene
	iottwinmaker:DeleteSyncJob
	iottwinmaker:DeleteWorkspace
	iottwinmaker:ExecuteQuery
	iottwinmaker:GetPricingPlan
	iottwinmaker:GetScene
	iottwinmaker:GetSyncJob
	iottwinmaker:ListComponentTypes
	iottwinmaker:ListEntities
	iottwinmaker:ListScenes
	iottwinmaker:ListSyncJobs
	iottwinmaker:ListSyncResources
	iottwinmaker:ListWorkspaces
	iottwinmaker:UpdateComponentType

Prefijo de servicio	Acciones
	iottwinmaker:UpdateEntity iottwinmaker:UpdatePricingPlan iottwinmaker:UpdateScene iottwinmaker:UpdateWorkspace

Prefijo de servicio	Acciones
iotwireless	iotwireless:AssociateAwsAccountWithPartnerAccount iotwireless:AssociateMulticastGroupWithFuotaTask iotwireless:AssociateWirelessDeviceWithFuotaTask iotwireless:AssociateWirelessDeviceWithMulticastGroup iotwireless:AssociateWirelessDeviceWithThing iotwireless:AssociateWirelessGatewayWithCertificate iotwireless:AssociateWirelessGatewayWithThing iotwireless:CancelMulticastGroupSession iotwireless:CreateDestination iotwireless:CreateDeviceProfile iotwireless:CreateFuotaTask iotwireless:CreateMulticastGroup iotwireless:CreateNetworkAnalyzerConfiguration iotwireless:CreateServiceProfile iotwireless:CreateWirelessDevice iotwireless:CreateWirelessGateway iotwireless:CreateWirelessGatewayTask iotwireless:CreateWirelessGatewayTaskDefinition iotwireless:DeleteDestination iotwireless:DeleteDeviceProfile iotwireless:DeleteFuotaTask

Prefijo de servicio	Acciones
	iotwireless:DeleteMulticastGroup iotwireless:DeleteNetworkAnalyzerConfiguration iotwireless:DeleteQueuedMessages iotwireless:DeleteServiceProfile iotwireless:DeleteWirelessDevice iotwireless:DeleteWirelessDeviceImportTask iotwireless:DeleteWirelessGateway iotwireless:DeleteWirelessGatewayTask iotwireless:DeleteWirelessGatewayTaskDefinition iotwireless:DeregisterWirelessDevice iotwireless:DisassociateAwsAccountFromPartnerAccount iotwireless:DisassociateMulticastGroupFromFuotaTask iotwireless:DisassociateWirelessDeviceFromFuotaTask iotwireless:DisassociateWirelessDeviceFromMulticastGroup iotwireless:DisassociateWirelessDeviceFromThing iotwireless:DisassociateWirelessGatewayFromCertificate iotwireless:DisassociateWirelessGatewayFromThing iotwireless:GetDestination iotwireless:GetDeviceProfile iotwireless:GetEventConfigurationByResourceTypes iotwireless:GetFuotaTask

Prefijo de servicio	Acciones
	iotwireless:GetLogLevelsByResourceTypes iotwireless:GetMulticastGroup iotwireless:GetMulticastGroupSession iotwireless:GetNetworkAnalyzerConfiguration iotwireless:GetPartnerAccount iotwireless:GetPosition iotwireless:GetPositionConfiguration iotwireless:GetPositionEstimate iotwireless:GetResourceEventConfiguration iotwireless:GetResourceLogLevel iotwireless:GetResourcePosition iotwireless:GetServiceEndpoint iotwireless:GetServiceProfile iotwireless:GetWirelessDevice iotwireless:GetWirelessDeviceImportTask iotwireless:GetWirelessDeviceStatistics iotwireless:GetWirelessGateway iotwireless:GetWirelessGatewayCertificate iotwireless:GetWirelessGatewayFirmwareInformation iotwireless:GetWirelessGatewayStatistics iotwireless:GetWirelessGatewayTask

Prefijo de servicio	Acciones
	iotwireless:GetWirelessGatewayTaskDefinition iotwireless:ListDestinations iotwireless:ListDeviceProfiles iotwireless:ListDevicesForWirelessDeviceImportTask iotwireless:ListEventConfigurations iotwireless:ListFuotaTasks iotwireless:ListMulticastGroups iotwireless:ListMulticastGroupsByFuotaTask iotwireless:ListNetworkAnalyzerConfigurations iotwireless:ListPartnerAccounts iotwireless:ListPositionConfigurations iotwireless:ListQueuedMessages iotwireless:ListServiceProfiles iotwireless:ListWirelessDeviceImportTasks iotwireless:ListWirelessDevices iotwireless:ListWirelessGateways iotwireless:ListWirelessGatewayTaskDefinitions iotwireless:PutPositionConfiguration iotwireless:PutResourceLogLevel iotwireless:ResetAllResourceLogLevels iotwireless:ResetResourceLogLevel

Prefijo de servicio	Acciones
	iotwireless:SendDataToMulticastGroup iotwireless:SendDataToWirelessDevice iotwireless:StartBulkAssociateWirelessDeviceWithMulticastGroup iotwireless:StartBulkDisassociateWirelessDeviceFromMulticastGroup iotwireless:StartFuotaTask iotwireless:StartMulticastGroupSession iotwireless:StartNetworkAnalyzerStream iotwireless:StartSingleWirelessDeviceImportTask iotwireless:StartWirelessDeviceImportTask iotwireless:TestWirelessDevice iotwireless:UpdateDestination iotwireless:UpdateEventConfigurationByResourceTypes iotwireless:UpdateFuotaTask iotwireless:UpdateLogLevelsByResourceTypes iotwireless:UpdateMulticastGroup iotwireless:UpdateNetworkAnalyzerConfiguration iotwireless:UpdatePartnerAccount iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition

Prefijo de servicio	Acciones
	iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

Prefijo de servicio	Acciones
ivs	ivs:BatchGetChannel
	ivs:BatchGetStreamKey
	ivs:BatchStartViewerSessionRevocation
	ivs:CreateChannel
	ivs:CreateParticipantToken
	ivs:CreateRecordingConfiguration
	ivs:CreateStreamKey
	ivs:DeleteChannel
	ivs:DeletePlaybackKeyPair
	ivs:DeleteRecordingConfiguration
	ivs:DeleteStreamKey
	ivs:DisconnectParticipant
	ivs:GetChannel
	ivs:GetParticipant
	ivs:GetPlaybackKeyPair
	ivs:GetRecordingConfiguration
	ivs:GetStream
	ivs:GetStreamKey
	ivs:GetStreamSession
	ivs:ImportPlaybackKeyPair
	ivs:ListChannels

Prefijo de servicio	Acciones
	ivs:ListParticipantEvents ivs:ListParticipants ivs:ListPlaybackKeyPairs ivs:ListRecordingConfigurations ivs:ListStreamKeys ivs:ListStreams ivs:ListStreamSessions ivs:PutMetadata ivs:StartViewerSessionRevocation ivs:StopStream ivs:UpdateChannel

Prefijo de servicio	Acciones
ivschat	ivschat:CreateChatToken ivschat:CreateLoggingConfiguration ivschat:CreateRoom ivschat:DeleteLoggingConfiguration ivschat:DeleteMessage ivschat:DeleteRoom ivschat:DisconnectUser ivschat:GetLoggingConfiguration ivschat:GetRoom ivschat:ListLoggingConfigurations ivschat:ListRooms ivschat:SendEvent ivschat:UpdateLoggingConfiguration ivschat:UpdateRoom

Prefijo de servicio	Acciones
kafka	kafka:BatchAssociateScramSecret
	kafka:BatchDisassociateScramSecret
	kafka:CreateCluster
	kafka:CreateClusterV2
	kafka:CreateConfiguration
	kafka:DeleteCluster
	kafka:DeleteClusterPolicy
	kafka:DeleteConfiguration
	kafka:DeleteReplicator
	kafka:DeleteVpcConnection
	kafka:DescribeCluster
	kafka:DescribeClusterOperation
	kafka:DescribeClusterOperationV2
	kafka:DescribeClusterV2
	kafka:DescribeConfiguration
	kafka:DescribeConfigurationRevision
	kafka:DescribeVpcConnection
	kafka:GetBootstrapBrokers
	kafka:GetClusterPolicy
	kafka:GetCompatibleKafkaVersions
	kafka:ListClientVpcConnections

Prefijo de servicio	Acciones
	kafka:ListClusterOperations
	kafka:ListClusterOperationsV2
	kafka:ListClusters
	kafka:ListClustersV2
	kafka:ListConfigurationRevisions
	kafka:ListConfigurations
	kafka:ListKafkaVersions
	kafka:ListNodes
	kafka:ListReplicators
	kafka:ListScramSecrets
	kafka:ListVpcConnections
	kafka:PutClusterPolicy
	kafka:RebootBroker
	kafka:RejectClientVpcConnection
	kafka:UpdateBrokerCount
	kafka:UpdateBrokerStorage
	kafka:UpdateBrokerType
	kafka:UpdateClusterConfiguration
	kafka:UpdateClusterKafkaVersion
	kafka:UpdateConfiguration
	kafka:UpdateConnectivity

Prefijo de servicio	Acciones
	kafka:UpdateMonitoring kafka:UpdateReplicationInfo kafka:UpdateSecurity kafka:UpdateStorage
kafkaconnect	kafkaconnect:CreateConnector kafkaconnect:CreateCustomPlugin kafkaconnect:CreateWorkerConfiguration kafkaconnect:DeleteConnector kafkaconnect:DeleteCustomPlugin kafkaconnect:DescribeConnector kafkaconnect:DescribeCustomPlugin kafkaconnect:DescribeWorkerConfiguration kafkaconnect:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

Prefijo de servicio	Acciones
kendra	kendra:AssociateEntitiesToExperience
	kendra:AssociatePersonasToEntities
	kendra:BatchDeleteDocument
	kendra:BatchDeleteFeaturedResultsSet
	kendra:BatchGetDocumentStatus
	kendra:BatchPutDocument
	kendra:ClearQuerySuggestions
	kendra:CreateAccessControlConfiguration
	kendra:CreateDataSource
	kendra:CreateExperience
	kendra:CreateFaq
	kendra:CreateFeaturedResultsSet
	kendra:CreateIndex
	kendra:CreateQuerySuggestionsBlockList
	kendra:CreateThesaurus
	kendra:DeleteDataSource
	kendra:DeleteExperience
	kendra:DeleteFaq
	kendra:DeleteIndex
	kendra:DeletePrincipalMapping
	kendra:DeleteQuerySuggestionsBlockList

Prefijo de servicio	Acciones
	kendra:DeleteThesaurus
	kendra:DescribeAccessControlConfiguration
	kendra:DescribeDataSource
	kendra:DescribeExperience
	kendra:DescribeFaq
	kendra:DescribeFeaturedResultsSet
	kendra:DescribeIndex
	kendra:DescribePrincipalMapping
	kendra:DescribeQuerySuggestionsBlockList
	kendra:DescribeQuerySuggestionsConfig
	kendra:DescribeThesaurus
	kendra:DisassociateEntitiesFromExperience
	kendra:DisassociatePersonasFromEntities
	kendra:GetQuerySuggestions
	kendra:GetSnapshots
	kendra:ListAccessControlConfigurations
	kendra:ListDataSources
	kendra:ListDataSourceSyncJobs
	kendra:ListEntityPersonas
	kendra:ListExperienceEntities
	kendra:ListExperiences

Prefijo de servicio	Acciones
	kendra:ListFaqs kendra:ListFeaturedResultsSets kendra:ListGroupsOlderThanOrderingId kendra:ListIndices kendra:ListQuerySuggestionsBlockLists kendra:ListThesauri kendra:PutPrincipalMapping kendra:Query kendra:Retrieve kendra:StartDataSourceSyncJob kendra:StopDataSourceSyncJob kendra:SubmitFeedback kendra:UpdateDataSource kendra:UpdateExperience kendra:UpdateFeaturedResultsSet kendra:UpdateIndex kendra:UpdateQuerySuggestionsBlockList kendra:UpdateQuerySuggestionsConfig kendra:UpdateThesaurus

Prefijo de servicio	Acciones
kinesis	kinesis:CreateStream kinesis:DecreaseStreamRetentionPeriod kinesis:DeleteStream kinesis:DeregisterStreamConsumer kinesis:DescribeLimits kinesis:DescribeStream kinesis:DescribeStreamConsumer kinesis:DescribeStreamSummary kinesis:DisableEnhancedMonitoring kinesis:EnableEnhancedMonitoring kinesis:IncreaseStreamRetentionPeriod kinesis:ListShards kinesis:ListStreamConsumers kinesis:ListStreams kinesis:MergeShards kinesis:RegisterStreamConsumer kinesis:SplitShard kinesis:StartStreamEncryption kinesis:StopStreamEncryption kinesis:UpdateShardCount kinesis:UpdateStreamMode

Prefijo de servicio	Acciones
kinesisanalytics	kinesisanalytics:AddApplicationCloudWatchLoggingOption kinesisanalytics:AddApplicationInput kinesisanalytics:AddApplicationInputProcessingConfiguration kinesisanalytics:AddApplicationOutput kinesisanalytics:AddApplicationReferenceDataSource kinesisanalytics:AddApplicationVpcConfiguration kinesisanalytics:CreateApplication kinesisanalytics:CreateApplicationPresignedUrl kinesisanalytics:CreateApplicationSnapshot kinesisanalytics:DeleteApplication kinesisanalytics:DeleteApplicationCloudWatchLoggingOption kinesisanalytics:DeleteApplicationInputProcessingConfiguration kinesisanalytics:DeleteApplicationOutput kinesisanalytics:DeleteApplicationReferenceDataSource kinesisanalytics:DeleteApplicationSnapshot kinesisanalytics:DeleteApplicationVpcConfiguration kinesisanalytics:DescribeApplication kinesisanalytics:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema kinesisanalytics:ListApplications

Prefijo de servicio	Acciones
	kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

Prefijo de servicio	Acciones
kms	kms:CancelKeyDeletion kms:ConnectCustomKeyStore KMS:createAlias kms:CreateCustomKeyStore kms:CreateGrant kms:CreateKey kms:Decrypt kms:DeleteAlias kms:DeleteCustomKeyStore kms:DeleteImportedKeyMaterial kms:DescribeCustomKeyStores kms:DescribeKey kms:DisableKey kms:DisableKeyRotation kms:DisconnectCustomKeyStore kms:EnableKey kms:EnableKeyRotation kms:Encrypt kms:GenerateDataKey kms:GenerateDataKeyPair kms:GenerateDataKeyPairWithoutPlaintext

Prefijo de servicio	Acciones
	kms:GenerateDataKeyWithoutPlaintext kms:GenerateMac kms:GenerateRandom kms:GetKeyPolicy kms:GetKeyRotationStatus kms:GetParametersForImport kms:GetPublicKey kms:ImportKeyMaterial kms:ListAliases kms:ListGrants kms:ListKeyPolicies kms:ListKeys kms:ListRetirableGrants kms:ReplicateKey kms:RetireGrant kms:RevokeGrant kms:ScheduleKeyDeletion kms:Sign kms:UpdateAlias kms:UpdateCustomKeyStore kms:UpdateKeyDescription

Prefijo de servicio	Acciones
	kms:UpdatePrimaryRegion kms:Verify kms:VerifyMac

Prefijo de servicio	Acciones
lambda	lambda:AddLayerVersionPermission
	lambda:AddLayerVersionPermission
	lambda:AddPermission
	lambda:AddPermission
	lambda:AddPermission
	lambda:CreateAlias
	lambda:CreateAlias
	lambda:CreateCodeSigningConfig
	lambda:CreateEventSourceMapping
	lambda:CreateEventSourceMapping
	lambda:CreateFunction
	lambda:CreateFunction
	lambda:CreateFunctionUrlConfig
	lambda:DeleteAlias
	lambda:DeleteAlias
	lambda:DeleteCodeSigningConfig
	lambda:DeleteEventSourceMapping
	lambda:DeleteEventSourceMapping
	lambda:DeleteFunction
	lambda:DeleteFunction
	lambda:DeleteFunctionCodeSigningConfig

Prefijo de servicio	Acciones
	lambda:DeleteFunctionConcurrency
	lambda:DeleteFunctionConcurrency
	lambda:DeleteFunctionEventInvokeConfig
	lambda:DeleteFunctionUrlConfig
	lambda:DeleteLayerVersion
	lambda:DeleteLayerVersion
	lambda:DeleteProvisionedConcurrencyConfig
	lambda:GetAccountSettings
	lambda:GetAccountSettings
	lambda:GetAlias
	lambda:GetAlias
	lambda:GetCodeSigningConfig
	lambda:GetEventSourceMapping
	lambda:GetEventSourceMapping
	lambda:GetFunction
	lambda:GetFunction
	lambda:GetFunction
	lambda:GetFunctionCodeSigningConfig
	lambda:GetFunctionConcurrency
	lambda:GetFunctionConfiguration
	lambda:GetFunctionConfiguration

Prefijo de servicio	Acciones
	lambda:GetFunctionConfiguration
	lambda:GetFunctionEventInvokeConfig
	lambda:GetFunctionUrlConfig
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersionPolicy
	lambda:GetLayerVersionPolicy
	lambda:GetPolicy
	lambda:GetPolicy
	lambda:GetPolicy
	lambda:GetProvisionedConcurrencyConfig
	lambda:GetRuntimeManagementConfig
	lambda:ListAliases
	lambda:ListAliases
	lambda:ListCodeSigningConfigs
	lambda:ListEventSourceMappings
	lambda:ListEventSourceMappings
	lambda:ListFunctionEventInvokeConfigs
	lambda:ListFunctions

Prefijo de servicio	Acciones
	lambda:ListFunctions
	lambda:ListFunctionsByCodeSigningConfig
	lambda:ListFunctionUrlConfigs
	lambda:ListLayers
	lambda:ListLayers
	lambda:ListLayerVersions
	lambda:ListLayerVersions
	lambda:ListProvisionedConcurrencyConfigs
	lambda:ListVersionsByFunction
	lambda:ListVersionsByFunction
	lambda:PublishLayerVersion
	lambda:PublishLayerVersion
	lambda:PublishVersion
	lambda:PublishVersion
	lambda:PutFunctionCodeSigningConfig
	lambda:PutFunctionConcurrency
	lambda:PutFunctionConcurrency
	lambda:PutFunctionEventInvokeConfig
	lambda:PutProvisionedConcurrencyConfig
	lambda:PutRuntimeManagementConfig
	lambda:RemoveLayerVersionPermission

Prefijo de servicio	Acciones
	lambda:RemoveLayerVersionPermission
	lambda:RemovePermission
	lambda:RemovePermission
	lambda:RemovePermission
	lambda:UpdateAlias
	lambda:UpdateAlias
	lambda:UpdateCodeSigningConfig
	lambda:UpdateEventSourceMapping
	lambda:UpdateEventSourceMapping
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionEventInvokeConfig
	lambda:UpdateFunctionUrlConfig

Prefijo de servicio	Acciones
lex	lex:BatchCreateCustomVocabularyItem lex:BatchDeleteCustomVocabularyItem lex:BatchUpdateCustomVocabularyItem lex:BuildBotLocale lex:CreateBotAlias lex:CreateBotVersion lex:CreateExport lex:CreateIntentVersion lex:CreateResourcePolicy lex:CreateSlotTypeVersion lex:CreateTestSetDiscrepancyReport lex:CreateUploadUrl lex:DeleteBot lex:DeleteBotChannelAssociation lex:DeleteExport lex:DeleteImport lex:DeleteIntentVersion lex:DeleteResourcePolicy lex:DeleteSlotTypeVersion lex:DeleteTestSet lex:DeleteUtterances

Prefijo de servicio	Acciones
	lex:DescribeBotAlias lex:DescribeBotRecommendation lex:DescribeBotVersion lex:DescribeCustomVocabularyMetadata lex:DescribeExport lex:DescribeImport lex:DescribeResourcePolicy lex:DescribeTestExecution lex:DescribeTestSet lex:DescribeTestSetDiscrepancyReport lex:DescribeTestSetGeneration lex:GetBot lex:GetBotAlias lex:GetBotAliases lex:GetBotChannelAssociation lex:GetBotChannelAssociations lex:GetBots lex:GetBotVersions lex:GetBuiltinIntent lex:GetBuiltinIntents lex:GetBuiltinSlotTypes

Prefijo de servicio	Acciones
	lex:GetExport lex:GetImport lex:GetIntent lex:GetIntents lex:GetIntentVersions lex:GetMigration lex:GetMigrations lex:GetSlotType lex:GetSlotTypes lex:GetSlotTypeVersions lex:GetTestExecutionArtifactsUrl lex:GetUtterancesView lex:ListBotAliases lex:ListBotRecommendations lex:ListBots lex:ListBotVersions lex:ListBuiltInIntents lex:ListBuiltInSlotTypes lex:ListCustomVocabularyItems lex:ListExports lex:ListImports

Prefijo de servicio	Acciones
	lex:ListIntentMetrics lex:ListIntentPaths lex:ListRecommendedIntents lex:ListSessionAnalyticsData lex:ListSessionMetrics lex:ListTestExecutionResultItems lex:ListTestExecutions lex:ListTestSets lex:PutBot lex:PutBotAlias lex:PutIntent lex:PutSlotType lex:SearchAssociatedTranscripts lex:StartBotRecommendation lex:StartImport lex:StartMigration lex:StartTestExecution lex:StartTestSetGeneration lex:StopBotRecommendation lex:UpdateBotAlias lex:UpdateBotRecommendation

Prefijo de servicio	Acciones
	lex:UpdateExport lex:UpdateResourcePolicy
license-manager-linux-subscriptions	license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:UpdateServiceSettings

Prefijo de servicio	Acciones
lightsail	lightsail: AllocateStaticIp lightsail: AttachCertificateToDistribution lightsail: aAttachDisk lightsail:AttachInstancesToLoadBalancer lightsail: aAttachLoadBalancerTlsCertificate lightsail: AttachStaticIp lightsail: CloseInstancePublicPorts lightsail: CopySnapshot lightsail: CreateBucket lightsail: CreateBucketAccessKey lightsail: CreateCertificate lightsail: CreateCloudFormationStack lightsail: CreateContactMethod lightsail: CreateContainerService lightsail: CreateContainerServiceDeployment lightsail: CreateContainerServiceRegistryLogin lightsail: CreateDisk lightsail: CreateDiskFromSnapshot lightsail: CreateDiskSnapshot lightsail: CreateDistribution lightsail: CreateDomain

Prefijo de servicio	Acciones
	lightsail: CreateGUISessionAccessDetails lightsail: CreateInstances lightsail: CreateInstancesFromSnapshot lightsail: CreateInstanceSnapshot lightsail: CreateKeyPair lightsail: CreateLoadBalancer lightsail: CreateLoadBalancerTlsCertificate lightsail: CreateRelationalDatabase lightsail: CreateRelationalDatabaseFromSnapshot lightsail: CreateRelationalDatabaseSnapshot lightsail: DeleteAlarm lightsail: DeleteAutoSnapshot lightsail: DeleteBucket lightsail: DeleteBucketAccessKey lightsail: DeleteCertificate lightsail: DeleteContactMethod lightsail: DeleteContainerImage lightsail: DeleteContainerService lightsail: DeleteDisk lightsail: DeleteDiskSnapshot lightsail: DeleteDistribution

Prefijo de servicio	Acciones
	lightsail: DeleteDomain lightsail: DeleteDomainEntry lightsail: DeleteInstance lightsail: DeleteInstanceSnapshot lightsail: DeleteKeyPair lightsail: DeleteKnownHostKeys lightsail: DeleteLoadBalancer lightsail: DeleteLoadBalancerTlsCertificate lightsail: DeleteRelationalDatabase lightsail: DeleteRelationalDatabaseSnapshot lightsail: DetachCertificateFromDistribution lightsail: DetachDisk lightsail: DetachInstancesFromLoadBalancer lightsail: DetachStaticIp lightsail: DisableAddOn lightsail: DownloadDefaultKeyPair lightsail: EnableAddOn lightsail: ExportSnapshot lightsail: GetActiveNames lightsail: GetAlarms lightsail: GetAutoSnapshots

Prefijo de servicio	Acciones
	lightsail: GetBlueprints lightsail: GetBucketAccessKeys lightsail: GetBucketBundles lightsail: GetBucketMetricData lightsail: GetBuckets lightsail: GetBundles lightsail: GetCertificates lightsail: GetCloudFormationStackRecords lightsail: GetContactMethods lightsail: GetContainerAPIMetadata lightsail: GetContainerImages lightsail: GetContainerLog lightsail: GetContainerServiceDeployments lightsail: GetContainerServiceMetricData lightsail: GetContainerServicePowers lightsail: GetContainerServices lightsail: GetCostEstimate lightsail: GetDisk lightsail: GetDisks lightsail: GetDiskSnapshot lightsail: GetDiskSnapshots

Prefijo de servicio	Acciones
	lightsail: GetDistributionBundles lightsail: GetDistributionLatestCacheReset lightsail: GetDistributionMetricData lightsail: GetDistributions lightsail: GetDomain lightsail: GetExportSnapshotRecords lightsail: GetInstance lightsail: GetInstanceAccessDetails lightsail: GetInstanceMetricData lightsail: GetInstancePortStates lightsail: GetInstances lightsail: GetInstanceSnapshot lightsail: GetInstanceSnapshots lightsail: GetInstanceState lightsail: GetKeyPair lightsail: GetKeyPairs lightsail: GetLoadBalancer lightsail: GetLoadBalancerMetricData lightsail: GetLoadBalancers lightsail: GetLoadBalancerTlsCertificates lightsail: GetLoadBalancerTlsPolicies

Prefijo de servicio	Acciones
	lightsail: GetOperation lightsail: GetOperations lightsail: GetOperationsForResource lightsail: GetRegions lightsail: GetRelationalDatabase lightsail: GetRelationalDatabaseBlueprints lightsail: GetRelationalDatabaseBundles lightsail: GetRelationalDatabaseEvents lightsail: GetRelationalDatabaseLogEvents lightsail: GetRelationalDatabaseLogStreams lightsail: GetRelationalDatabaseMasterUserPassword lightsail: GetRelationalDatabaseMetricData lightsail: GetRelationalDatabaseParameters lightsail: GetRelationalDatabases lightsail: GetRelationalDatabaseSnapshot lightsail: GetRelationalDatabaseSnapshots lightsail: GetStaticIp lightsail: GetStaticIps lightsail: ImportKeyPair lightsail: IsVpcPeered lightsail: OpenInstancePublicPorts

Prefijo de servicio	Acciones
	lightsail: PeerVPC lightsail: PutAlarm lightsail: PutInstancePublicPorts lightsail: RebootInstance lightsail: RebootRelationalDatabase lightsail: RegisterContainerImage lightsail: ReleaseStaticIp lightsail: ResetDistributionCache lightsail: SendContactMethodVerification lightsail: SetIpAddressType lightsail: SetResourceAccessForBucket lightsail: StartGUISession lightsail: StartInstance lightsail: StartRelationalDatabase lightsail: StopGUISession lightsail: StopInstance lightsail: StopRelationalDatabase lightsail: TestAlarm lightsail: UnpeerVpc lightsail: UpdateBucket lightsail: UpdateBucketBundle

Prefijo de servicio	Acciones
	lightsail: UpdateContainerService lightsail: UpdateDistribution lightsail: UpdateDistributionBundle lightsail: UpdateDomainEntry lightsail: UpdateInstanceMetadataOptions lightsail: UpdateLoadBalancerAttribute lightsail: UpdateRelationalDatabase lightsail: UpdateRelationalDatabaseParameters

Prefijo de servicio	Acciones
registros	logs:AssociateKmsKey
	logs:CancelExportTask
	logs:CreateExportTask
	logs:CreateLogGroup
	logs:CreateLogStream
	logs:DeleteDataProtectionPolicy
	logs:DeleteDestination
	logs:DeleteLogGroup
	logs:DeleteLogStream
	logs:DeleteMetricFilter
	logs:DeleteQueryDefinition
	logs:DeleteResourcePolicy
	logs:DeleteRetentionPolicy
	logs:DeleteSubscriptionFilter
	logs:DescribeAccountPolicies
	logs:DescribeDestinations
	logs:DescribeExportTasks
	logs:DescribeLogGroups
	logs:DescribeLogStreams
	logs:DescribeMetricFilters
	logs:DescribeQueries

Prefijo de servicio	Acciones
	logs:DescribeQueryDefinitions
	logs:DescribeResourcePolicies
	logs:DescribeSubscriptionFilters
	logs:DisassociateKmsKey
	logs:GetDataProtectionPolicy
	logs:GetLogGroupFields
	logs:GetLogRecord
	logs:GetQueryResults
	logs:PutDataProtectionPolicy
	logs:PutDestination
	logs:PutDestinationPolicy
	logs:PutMetricFilter
	logs:PutQueryDefinition
	logs:PutResourcePolicy
	logs:PutRetentionPolicy
	logs:PutSubscriptionFilter
	logs:StartLiveTail
	logs:StartQuery
	logs:StopQuery
	logs:TestMetricFilter

Prefijo de servicio	Acciones
lookoutequipment	lookoutequipment:CreateDataset lookoutequipment:CreateInferenceScheduler lookoutequipment:CreateLabel lookoutequipment:CreateLabelGroup lookoutequipment:CreateModel lookoutequipment:DeleteDataset lookoutequipment:DeleteInferenceScheduler lookoutequipment:DeleteLabel lookoutequipment:DeleteLabelGroup lookoutequipment:DeleteModel lookoutequipment:DeleteResourcePolicy lookoutequipment:DeleteRetrainingScheduler lookoutequipment:DescribeDataIngestionJob lookoutequipment:DescribeDataset lookoutequipment:DescribeInferenceScheduler lookoutequipment:Describelabel lookoutequipment:DescribeLabelGroup lookoutequipment:DescribeModel lookoutequipment:DescribeModelVersion lookoutequipment:DescribeResourcePolicy lookoutequipment:DescribeRetrainingScheduler

Prefijo de servicio	Acciones
	lookoutequipment:ImportDataset lookoutequipment:ImportModelVersion lookoutequipment:ListDataIngestionJobs lookoutequipment:ListDatasets lookoutequipment:ListInferenceEvents lookoutequipment:ListInferenceExecutions lookoutequipment:ListInferenceSchedulers lookoutequipment:ListLabelGroups lookoutequipment:ListLabels lookoutequipment:ListModels lookoutequipment:ListModelVersions lookoutequipment:ListRetrainingSchedulers lookoutequipment:ListSensorStatistics lookoutequipment:PutResourcePolicy lookoutequipment:StartDataIngestionJob lookoutequipment:StartInferenceScheduler lookoutequipment:StartRetrainingScheduler lookoutequipment:StopInferenceScheduler lookoutequipment:StopRetrainingScheduler lookoutequipment:UpdateActiveModelVersion lookoutequipment:UpdateInferenceScheduler

Prefijo de servicio	Acciones
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

Prefijo de servicio	Acciones
lookoutmetrics	lookoutmetrics:ActivateAnomalyDetector
	lookoutmetrics:BackTestAnomalyDetector
	lookoutmetrics:CreateAlert
	lookoutmetrics:CreateAnomalyDetector
	lookoutmetrics:CreateMetricSet
	lookoutmetrics:DeactivateAnomalyDetector
	lookoutmetrics:DeleteAlert
	lookoutmetrics:DeleteAnomalyDetector
	lookoutmetrics:DescribeAlert
	lookoutmetrics:DescribeAnomalyDetectionExecutions
	lookoutmetrics:DescribeAnomalyDetector
	lookoutmetrics:DescribeMetricSet
	lookoutmetrics:DetectMetricSetConfig
	lookoutmetrics:GetAnomalyGroup
	lookoutmetrics:GetDataQualityMetrics
	lookoutmetrics:GetFeedback
	lookoutmetrics:GetSampleData
	lookoutmetrics:ListAlerts
	lookoutmetrics:ListAnomalyDetectors
	lookoutmetrics:ListAnomalyGroupRelatedMetrics
	lookoutmetrics:ListAnomalyGroupSummaries

Prefijo de servicio	Acciones
	lookoutmetrics:ListAnomalyGroupTimeSeries
	lookoutmetrics:ListMetricSets
	lookoutmetrics:PutFeedback
	lookoutmetrics:UpdateAlert
	lookoutmetrics:UpdateAnomalyDetector
	lookoutmetrics:UpdateMetricSet

Prefijo de servicio	Acciones
lookoutvision	lookoutvision:CreateDataset lookoutvision:CreateModel lookoutvision:CreateProject lookoutvision:DeleteDataset lookoutvision:DeleteModel lookoutvision:DeleteProject lookoutvision:DescribeDataset lookoutvision:DescribeModel lookoutvision:DescribeModelPackagingJob lookoutvision:DescribeProject lookoutvision:DetectAnomalies lookoutvision:ListDatasetEntries lookoutvision:ListModelPackagingJobs lookoutvision:ListModels lookoutvision:ListProjects lookoutvision:StartModel lookoutvision:StartModelPackagingJob lookoutvision:StopModel lookoutvision:UpdateDatasetEntries

Prefijo de servicio	Acciones
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplications
	m2:ListApplicationVersions
	m2:ListBatchJobDefinitions
	m2:ListBatchJobExecutions
	m2:ListDataSetImportHistory

Prefijo de servicio	Acciones
	m2:ListDataSets m2:ListDeployments m2:ListEngineVersions m2:ListEnvironments m2:StartApplication m2:StartBatchJob m2:StopApplication m2:UpdateApplication m2:UpdateEnvironment

Prefijo de servicio	Acciones
managedblockchain	managedblockchain:CreateAccessor managedblockchain:CreateMember managedblockchain:CreateNetwork managedblockchain:CreateNode managedblockchain:CreateProposal managedblockchain:DeleteAccessor managedblockchain:DeleteMember managedblockchain:DeleteNode managedblockchain:GetAccessor managedblockchain:GetMember managedblockchain:GetNetwork managedblockchain:GetNode managedblockchain:GetProposal managedblockchain:ListAccessors managedblockchain:ListInvitations managedblockchain:ListMembers managedblockchain:ListNetworks managedblockchain:ListNodes managedblockchain:ListProposals managedblockchain:ListProposalVotes managedblockchain:RejectInvitation

Prefijo de servicio	Acciones
	managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

Prefijo de servicio	Acciones
mediaconnect	mediaconnect:AddBridgeOutputs
	mediaconnect:AddBridgeSources
	mediaconnect:AddFlowMediaStreams
	mediaconnect:AddFlowOutputs
	mediaconnect:AddFlowSources
	mediaconnect:AddFlowVpcInterfaces
	mediaconnect:CreateBridge
	mediaconnect:CreateFlow
	mediaconnect:CreateGateway
	mediaconnect:DeleteBridge
	mediaconnect:DeleteFlow
	mediaconnect:DeleteGateway
	mediaconnect:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeGateway
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect:GrantFlowEntitlements
	mediaconnect:ListBridges

Prefijo de servicio	Acciones
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect:StopFlow
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState
	mediaconnect:UpdateFlow

Prefijo de servicio	Acciones
	mediaconnect:UpdateFlowEntitlement mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateFlowOutput mediaconnect:UpdateFlowSource mediaconnect:UpdateGatewayInstance

Prefijo de servicio	Acciones
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobs mediaconvert:ListJobTemplates mediaconvert:ListPresets mediaconvert:ListQueues

Prefijo de servicio	Acciones
	mediaconvert:PutPolicy mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

Prefijo de servicio	Acciones
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreatePartnerInput
	medialive:DeleteChannel
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DescribeAccountConfiguration

Prefijo de servicio	Acciones
	medialive:DescribeChannel
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeThumbnails
	medialive:ListChannels
	medialive:ListInputDevices
	medialive:ListInputDeviceTransfers
	medialive:ListInputs
	medialive:ListInputSecurityGroups
	medialive:ListMultiplexes
	medialive:ListMultiplexPrograms
	medialive:ListOfferings
	medialive:ListReservations
	medialive:PurchaseOffering

Prefijo de servicio	Acciones
	medialive:RebootInputDevice
	medialive:RejectInputDeviceTransfer
	medialive:StartChannel
	medialive:StartInputDevice
	medialive:StartInputDeviceMaintenanceWindow
	medialive:StartMultiplex
	medialive:StopChannel
	medialive:StopInputDevice
	medialive:StopMultiplex
	medialive:TransferInputDevice
	medialive:UpdateAccountConfiguration
	medialive:UpdateChannel
	medialive:UpdateChannelClass
	medialive:UpdateInput
	medialive:UpdateInputDevice
	medialive:UpdateInputSecurityGroup
	medialive:UpdateMultiplex
	medialive:UpdateMultiplexProgram
	medialive:UpdateReservation

Prefijo de servicio	Acciones
mediapackage	mediapackage:ConfigureLogs mediapackage:CreateChannel mediapackage:CreateHarvestJob mediapackage:CreateOriginEndpoint mediapackage:DeleteChannel mediapackage:DeleteOriginEndpoint mediapackage:DescribeChannel mediapackage:DescribeHarvestJob mediapackage:DescribeOriginEndpoint mediapackage:ListChannels mediapackage:ListHarvestJobs mediapackage:ListOriginEndpoints mediapackage:RotateChannelCredentials mediapackage:RotateIngestEndpointCredentials mediapackage:UpdateChannel mediapackage:UpdateOriginEndpoint

Prefijo de servicio	Acciones
mediapackage-vod	mediapackage-vod:ConfigureLogs mediapackage-vod:CreateAsset mediapackage-vod:CreatePackagingConfiguration mediapackage-vod:CreatePackagingGroup mediapackage-vod:DeleteAsset mediapackage-vod:DeletePackagingConfiguration mediapackage-vod:DeletePackagingGroup mediapackage-vod:DescribeAsset mediapackage-vod:DescribePackagingConfiguration mediapackage-vod:DescribePackagingGroup mediapackage-vod:ListAssets mediapackage-vod:ListPackagingConfigurations mediapackage-vod:ListPackagingGroups mediapackage-vod:UpdatePackagingGroup

Prefijo de servicio	Acciones
mediastore	mediastore:CreateContainer mediastore:DeleteContainer mediastore:DeleteContainerPolicy mediastore:DeleteCorsPolicy mediastore:DeleteLifecyclePolicy mediastore:DeleteMetricPolicy mediastore:DescribeContainer mediastore:GetContainerPolicy mediastore:GetCorsPolicy mediastore:GetLifecyclePolicy mediastore:GetMetricPolicy mediastore:ListContainers mediastore:PutContainerPolicy mediastore:PutCorsPolicy mediastore:PutLifecyclePolicy mediastore:PutMetricPolicy mediastore:StartAccessLogging mediastore:StopAccessLogging

Prefijo de servicio	Acciones
mediatailor	mediatailor:ConfigureLogsForPlaybackConfiguration
	mediatailor:CreateChannel
	mediatailor:CreateLiveSource
	mediatailor:CreatePrefetchSchedule
	mediatailor:CreateProgram
	mediatailor:CreateSourceLocation
	mediatailor:CreateVodSource
	mediatailor:DeleteChannel
	mediatailor:DeleteChannelPolicy
	mediatailor:DeleteLiveSource
	mediatailor:DeletePlaybackConfiguration
	mediatailor:DeletePrefetchSchedule
	mediatailor:DeleteProgram
	mediatailor:DeleteSourceLocation
	mediatailor:DeleteVodSource
	mediatailor:DescribeChannel
	mediatailor:DescribeLiveSource
	mediatailor:DescribeProgram
	mediatailor:DescribeSourceLocation
	mediatailor:DescribeVodSource
	mediatailor:GetChannelPolicy

Prefijo de servicio	Acciones
	mediatailor:GetChannelSchedule
	mediatailor:GetPlaybackConfiguration
	mediatailor:GetPrefetchSchedule
	mediatailor:ListAlerts
	mediatailor:ListChannels
	mediatailor:ListLiveSources
	mediatailor:ListPlaybackConfigurations
	mediatailor:ListPrefetchSchedules
	mediatailor:ListSourceLocations
	mediatailor:ListVodSources
	mediatailor:PutChannelPolicy
	mediatailor:PutPlaybackConfiguration
	mediatailor:StartChannel
	mediatailor:StopChannel
	mediatailor:UpdateChannel
	mediatailor:UpdateLiveSource
	mediatailor:UpdateProgram
	mediatailor:UpdateSourceLocation
	mediatailor:UpdateVodSource

Prefijo de servicio	Acciones
memorydb	memorydb:BatchUpdateCluster memorydb:CopySnapshot memorydb:CreateAcl memorydb:CreateCluster memorydb:CreateParameterGroup memorydb:CreateSnapshot memorydb:CreateSubnetGroup memorydb:CreateUser memorydb:DeleteAcl memorydb:DeleteCluster memorydb:DeleteParameterGroup memorydb:DeleteSnapshot memorydb:DeleteSubnetGroup memorydb:DeleteUser memorydb:DescribeAcls memorydb:DescribeClusters memorydb:DescribeEngineVersions memorydb:DescribeEvents memorydb:DescribeParameterGroups memorydb:DescribeParameters memorydb:DescribeReservedNodes

Prefijo de servicio	Acciones
	memorydb:DescribeReservedNodesOfferings memorydb:DescribeServiceUpdates memorydb:DescribeSnapshots memorydb:DescribeSubnetGroups memorydb:DescribeUsers memorydb:FailoverShard memorydb:ListAllowedNodeTypeUpdates memorydb:PurchaseReservedNodesOffering memorydb:ResetParameterGroup memorydb:UpdateAcl memorydb:UpdateCluster memorydb:UpdateParameterGroup memorydb:UpdateSubnetGroup memorydb:UpdateUser

Prefijo de servicio	Acciones
mgh	mgh:AssociateCreatedArtifact
	mgh:AssociateDiscoveredResource
	mgh:CreateHomeRegionControl
	mgh:CreateProgressUpdateStream
	mgh:DeleteHomeRegionControl
	mgh:DeleteProgressUpdateStream
	mgh:DescribeApplicationState
	mgh:DescribeHomeRegionControls
	mgh:DescribeMigrationTask
	mgh:DisassociateCreatedArtifact
	mgh:DisassociateDiscoveredResource
	mgh:GetHomeRegion
	mgh:ImportMigrationTask
	mgh:ListApplicationStates
	mgh:ListCreatedArtifacts
	mgh:ListDiscoveredResources
	mgh:ListMigrationTasks
	mgh:ListProgressUpdateStreams
	mgh:NotifyApplicationState
	mgh:NotifyMigrationTaskState
	mgh:PutResourceAttributes

Prefijo de servicio	Acciones
mgn	mgn:ArchiveApplication mgn:ArchiveWave mgn:AssociateApplications mgn:AssociateSourceServers mgn:ChangeServerLifeCycleState mgn:CreateApplication mgn:CreateConnector mgn:CreateLaunchConfigurationTemplate mgn:CreateReplicationConfigurationTemplate mgn:CreateWave mgn:DeleteApplication mgn:DeleteConnector mgn:DeleteJob mgn:DeleteLaunchConfigurationTemplate mgn:DeleteReplicationConfigurationTemplate mgn:DeleteSourceServer mgn:DeleteVcenterClient mgn:DeleteWave mgn:DescribeJobLogItems mgn:DescribeJobs mgn:DescribeLaunchConfigurationTemplates

Prefijo de servicio	Acciones
	mgn:DescribeReplicationConfigurationTemplates mgn:DescribeVcenterClients mgn:DisassociateApplications mgn:DisassociateSourceServers mgn:DisconnectFromService mgn:FinalizeCutover mgn:GetReplicationConfiguration mgn:InitializeService mgn:ListConnectors mgn:ListExportErrors mgn:ListExports mgn:ListImportErrors mgn:ListImports mgn:ListManagedAccounts mgn:ListSourceServerActions mgn:ListTemplateActions mgn:MarkAsArchived mgn:PauseReplication mgn:PutSourceServerAction mgn:PutTemplateAction mgn:RemoveSourceServerAction

Prefijo de servicio	Acciones
	mgn:RemoveTemplateAction mgn:ResumeReplication mgn:RetryDataReplication mgn:StartCutover mgn:StartExport mgn:StartImport mgn:StartReplication mgn:StartTest mgn:StopReplication mgn:TerminateTargetInstances mgn:UnarchiveApplication mgn:UnarchiveWave mgn:UpdateApplication mgn:UpdateConnector mgn:UpdateLaunchConfigurationTemplate mgn:UpdateReplicationConfiguration mgn:UpdateReplicationConfigurationTemplate mgn:UpdateSourceServer mgn:UpdateSourceServerReplicationType mgn:UpdateWave

Prefijo de servicio	Acciones
migrationhub-strategy	migrationhub-strategy:GetAntiPattern
	migrationhub-strategy:GetApplicationComponentDetails
	migrationhub-strategy:GetApplicationComponentStrategies
	migrationhub-strategy:GetAssessment
	migrationhub-strategy:GetImportFileTask
	migrationhub-strategy:GetLatestAssessmentId
	migrationhub-strategy:GetPortfolioPreferences
	migrationhub-strategy:GetPortfolioSummary
	migrationhub-strategy:GetRecommendationReportDetails
	migrationhub-strategy:GetServerDetails
	migrationhub-strategy:GetServerStrategies
	migrationhub-strategy:ListAntiPatterns
	migrationhub-strategy:ListApplicationComponents
	migrationhub-strategy:ListCollectors
	migrationhub-strategy:ListImportFileTask
	migrationhub-strategy:ListJarArtifacts
	migrationhub-strategy:ListServers
	migrationhub-strategy:PutPortfolioPreferences
	migrationhub-strategy:RegisterCollector
	migrationhub-strategy:StartAssessment
	migrationhub-strategy:StartImportFileTask

Prefijo de servicio	Acciones
	migrationhub-strategy:StartRecommendationReportGeneration migrationhub-strategy:StopAssessment migrationhub-strategy:UpdateApplicationComponentConfig migrationhub-strategy:UpdateCollectorConfiguration migrationhub-strategy:UpdateServerConfig

Prefijo de servicio	Acciones
mobiletargeting	mobiletargeting:CreateApp mobiletargeting:CreateCampaign mobiletargeting:CreateEmailTemplate mobiletargeting:CreateExportJob mobiletargeting:CreateImportJob mobiletargeting:CreateInAppTemplate mobiletargeting:CreateJourney mobiletargeting:CreatePushTemplate mobiletargeting:CreateRecommenderConfiguration mobiletargeting:CreateSegment mobiletargeting:CreateSmsTemplate mobiletargeting:CreateVoiceTemplate mobiletargeting:DeleteAdmChannel mobiletargeting:DeleteApnsChannel mobiletargeting:DeleteApnsSandboxChannel mobiletargeting:DeleteApnsVoipChannel mobiletargeting:DeleteApnsVoipSandboxChannel mobiletargeting:DeleteApp mobiletargeting:DeleteBaiduChannel mobiletargeting:DeleteCampaign mobiletargeting:DeleteEmailChannel

Prefijo de servicio	Acciones
	mobiletargeting:DeleteEmailTemplate mobiletargeting:DeleteEndpoint mobiletargeting:DeleteEventStream mobiletargeting:DeleteGcmChannel mobiletargeting:DeleteInAppTemplate mobiletargeting:DeleteJourney mobiletargeting:DeletePushTemplate mobiletargeting:DeleteRecommenderConfiguration mobiletargeting:DeleteSegment mobiletargeting:DeleteSmsChannel mobiletargeting:DeleteSmsTemplate mobiletargeting:DeleteUserEndpoints mobiletargeting:DeleteVoiceChannel mobiletargeting:DeleteVoiceTemplate mobiletargeting:GetAdmChannel mobiletargeting:GetApnsChannel mobiletargeting:GetApnsSandboxChannel mobiletargeting:GetApnsVoipChannel mobiletargeting:GetApnsVoipSandboxChannel mobiletargeting:GetApp mobiletargeting:GetApplicationDateRangeKpi

Prefijo de servicio	Acciones
	mobiletargeting:GetApplicationSettings mobiletargeting:GetApps mobiletargeting:GetBaiduChannel mobiletargeting:GetCampaign mobiletargeting:GetCampaignActivities mobiletargeting:GetCampaignDateRangeKpi mobiletargeting:GetCampaigns mobiletargeting:GetCampaignVersion mobiletargeting:GetCampaignVersions mobiletargeting:GetChannels mobiletargeting:GetEmailChannel mobiletargeting:GetEmailTemplate mobiletargeting:GetEndpoint mobiletargeting:GetEventStream mobiletargeting:GetExportJob mobiletargeting:GetExportJobs mobiletargeting:GetGcmChannel mobiletargeting:GetImportJob mobiletargeting:GetImportJobs mobiletargeting:GetInAppMessages mobiletargeting:GetInAppTemplate

Prefijo de servicio	Acciones
	mobiletargeting:GetJourney mobiletargeting:GetJourneyDateRangeKpi mobiletargeting:GetJourneyExecutionActivityMetrics mobiletargeting:GetJourneyExecutionMetrics mobiletargeting:GetJourneyRunExecutionActivityMetrics mobiletargeting:GetJourneyRunExecutionMetrics mobiletargeting:GetJourneyRuns mobiletargeting:GetPushTemplate mobiletargeting:GetRecommenderConfiguration mobiletargeting:GetRecommenderConfigurations mobiletargeting:GetSegment mobiletargeting:GetSegmentExportJobs mobiletargeting:GetSegmentImportJobs mobiletargeting:GetSegments mobiletargeting:GetSegmentVersion mobiletargeting:GetSegmentVersions mobiletargeting:GetSmsChannel mobiletargeting:GetSmsTemplate mobiletargeting:GetUserEndpoints mobiletargeting:GetVoiceChannel mobiletargeting:GetVoiceTemplate

Prefijo de servicio	Acciones
	mobiletargeting:ListJourneys mobiletargeting:ListTemplates mobiletargeting:ListTemplateVersions mobiletargeting:PhoneNumberValidate mobiletargeting:PutEventStream mobiletargeting:RemoveAttributes mobiletargeting:UpdateAdmChannel mobiletargeting:UpdateApnsChannel mobiletargeting:UpdateApnsSandboxChannel mobiletargeting:UpdateApnsVoipChannel mobiletargeting:UpdateApnsVoipSandboxChannel mobiletargeting:UpdateApplicationSettings mobiletargeting:UpdateBaiduChannel mobiletargeting:UpdateCampaign mobiletargeting:UpdateEmailChannel mobiletargeting:UpdateEmailTemplate mobiletargeting:UpdateEndpoint mobiletargeting:UpdateEndpointsBatch mobiletargeting:UpdateGcmChannel mobiletargeting:UpdateInAppTemplate mobiletargeting:UpdateJourney

Prefijo de servicio	Acciones
	mobiletargeting:UpdateJourneyState mobiletargeting:UpdatePushTemplate mobiletargeting:UpdateRecommenderConfiguration mobiletargeting:UpdateSegment mobiletargeting:UpdateSmsChannel mobiletargeting:UpdateSmsTemplate mobiletargeting:UpdateTemplateActiveVersion mobiletargeting:UpdateVoiceChannel mobiletargeting:UpdateVoiceTemplate mobiletargeting:VerifyOTPMessage

Prefijo de servicio	Acciones
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration mq:UpdateUser

Prefijo de servicio	Acciones
networkmanager	networkmanager:AcceptAttachment networkmanager:AssociateConnectPeer networkmanager:AssociateCustomerGateway networkmanager:AssociateLink networkmanager:AssociateTransitGatewayConnectPeer networkmanager:CreateConnectAttachment networkmanager:CreateConnection networkmanager:CreateConnectPeer networkmanager:CreateCoreNetwork networkmanager:CreateDevice networkmanager:CreateGlobalNetwork networkmanager:CreateLink networkmanager:CreateSite networkmanager:CreateSiteToSiteVpnAttachment networkmanager:CreateTransitGatewayPeering networkmanager:CreateTransitGatewayRouteTableAttachment networkmanager:CreateVpcAttachment networkmanager:DeleteAttachment networkmanager:DeleteConnection networkmanager:DeleteConnectPeer networkmanager:DeleteCoreNetwork

Prefijo de servicio	Acciones
	networkmanager:DeleteCoreNetworkPolicyVersion networkmanager:DeleteDevice networkmanager:DeleteGlobalNetwork networkmanager:DeleteLink networkmanager:DeletePeering networkmanager:DeleteResourcePolicy networkmanager:DeleteSite networkmanager:DeregisterTransitGateway networkmanager:DescribeGlobalNetworks networkmanager:DisassociateConnectPeer networkmanager:DisassociateCustomerGateway networkmanager:DisassociateLink networkmanager:DisassociateTransitGatewayConnectPeer networkmanager:ExecuteCoreNetworkChangeSet networkmanager:GetConnectAttachment networkmanager:GetConnections networkmanager:GetConnectPeer networkmanager:GetConnectPeerAssociations networkmanager:GetCoreNetwork networkmanager:GetCoreNetworkChangeEvents networkmanager:GetCoreNetworkChangeSet

Prefijo de servicio	Acciones
	networkmanager:GetCoreNetworkPolicy networkmanager:GetCustomerGatewayAssociations networkmanager:GetDevices networkmanager:GetLinkAssociations networkmanager:GetLinks networkmanager:GetNetworkResourceCounts networkmanager:GetNetworkResourceRelationships networkmanager:GetNetworkResources networkmanager:GetNetworkRoutes networkmanager:GetNetworkTelemetry networkmanager:GetResourcePolicy networkmanager:GetRouteAnalysis networkmanager:GetSites networkmanager:GetSiteToSiteVpnAttachment networkmanager:GetTransitGatewayConnectPeerAssociations networkmanager:GetTransitGatewayPeering networkmanager:GetTransitGatewayRegistrations networkmanager:GetTransitGatewayRouteTableAttachment networkmanager:GetVpcAttachment networkmanager:ListAttachments networkmanager:ListConnectPeers

Prefijo de servicio	Acciones
	networkmanager:ListCoreNetworkPolicyVersions networkmanager:ListCoreNetworks networkmanager:ListOrganizationServiceAccessStatus networkmanager:ListPeerings networkmanager:PutCoreNetworkPolicy networkmanager:PutResourcePolicy networkmanager:RegisterTransitGateway networkmanager:RejectAttachment networkmanager:RestoreCoreNetworkPolicyVersion networkmanager:StartOrganizationServiceAccessUpdate networkmanager:StartRouteAnalysis networkmanager:UpdateConnection networkmanager:UpdateCoreNetwork networkmanager:UpdateDevice networkmanager:UpdateGlobalNetwork networkmanager:UpdateLink networkmanager:UpdateNetworkResourceMetadata networkmanager:UpdateSite networkmanager:UpdateVpcAttachment

Prefijo de servicio	Acciones
nimble	nimble:AcceptEulas
	nimble:CreateLaunchProfile
	nimble:CreateStreamingImage
	nimble:CreateStreamingSession
	nimble:CreateStreamingSessionStream
	nimble:CreateStudio
	nimble:CreateStudioComponent
	nimble:DeleteLaunchProfile
	nimble:DeleteLaunchProfileMember
	nimble:DeleteStreamingImage
	nimble:DeleteStreamingSession
	nimble:DeleteStudio
	nimble:DeleteStudioComponent
	nimble:DeleteStudioMember
	nimble:GetEula
	nimble:GetLaunchProfileDetails
	nimble:GetStreamingImage
	nimble:GetStreamingSession
	nimble:GetStreamingSessionBackup
	nimble:GetStreamingSessionStream
	nimble:GetStudio

Prefijo de servicio	Acciones
	nimble:GetStudioComponent
	nimble:GetStudioMember
	nimble:ListEulas
	nimble:ListLaunchProfileMembers
	nimble:ListLaunchProfiles
	nimble:ListStreamingImages
	nimble:ListStreamingSessionBackups
	nimble:ListStreamingSessions
	nimble:ListStudioComponents
	nimble:ListStudioMembers
	nimble:ListStudios
	nimble:PutLaunchProfileMembers
	nimble:PutStudioMembers
	nimble:StartStreamingSession
	nimble:StartStudioSSOConfigurationRepair
	nimble:StopStreamingSession
	nimble:UpdateLaunchProfile
	nimble:UpdateLaunchProfileMember
	nimble:UpdateStreamingImage
	nimble:UpdateStudio
	nimble:UpdateStudioComponent

Prefijo de servicio	Acciones
omics	omics:AbortMultipartReadSetUpload omics:BatchDeleteReadSet omics:CancelAnnotationImportJob omics:CancelRun omics:CancelVariantImportJob omics:CompleteMultipartReadSetUpload omics:CreateAnnotationStore omics:CreateMultipartReadSetUpload omics:CreateReferenceStore omics:CreateRunGroup omics:CreateSequenceStore omics:CreateVariantStore omics:CreateWorkflow omics:DeleteAnnotationStore omics:DeleteReference omics:DeleteReferenceStore omics:DeleteRun omics:DeleteRunGroup omics:DeleteSequenceStore omics:DeleteVariantStore omics:DeleteWorkflow

Prefijo de servicio	Acciones
	omics:GetAnnotationImportJob omics:GetAnnotationStore omics:GetReadSet omics:GetReadSetActivationJob omics:GetReadSetExportJob omics:GetReadSetImportJob omics:GetReadSetMetadata omics:GetReference omics:GetReferenceImportJob omics:GetReferenceMetadata omics:GetReferenceStore omics:GetRun omics:GetRunGroup omics:GetRunTask omics:GetSequenceStore omics:GetVariantImportJob omics:GetVariantStore omics:GetWorkflow omics:ListAnnotationImportJobs omics:ListAnnotationStores omics:ListMultipartReadSetUploads

Prefijo de servicio	Acciones
	omics:ListReadSetActivationJobs omics:ListReadSetExportJobs omics:ListReadSetImportJobs omics:ListReadSets omics:ListReadSetUploadParts omics:ListReferenceImportJobs omics:ListReferences omics:ListReferenceStores omics:ListRunGroups omics:ListRuns omics:ListRunTasks omics:ListSequenceStores omics:ListVariantImportJobs omics:ListVariantStores omics:ListWorkflows omics:StartAnnotationImportJob omics:StartReadSetActivationJob omics:StartReadSetExportJob omics:StartReadSetImportJob omics:StartReferenceImportJob omics:StartRun

Prefijo de servicio	Acciones
	omics:StartVariantImportJob omics:UpdateAnnotationStore omics:UpdateRunGroup omics:UpdateVariantStore omics:UpdateWorkflow omics:UploadReadSetPart

Prefijo de servicio	Acciones
opsworks	opsworks:AssignInstance
	opsworks:AssignVolume
	opsworks:AssociateElasticIp
	opsworks:AttachElasticLoadBalancer
	opsworks:CloneStack
	opsworks:CreateApp
	opsworks:CreateDeployment
	opsworks:CreateInstance
	opsworks:CreateLayer
	opsworks:CreateStack
	opsworks:CreateUserProfile
	opsworks:DeleteApp
	opsworks:DeleteInstance
	opsworks:DeleteLayer
	opsworks:DeleteStack
	opsworks:DeleteUserProfile
	opsworks:DeregisterEcsCluster
	opsworks:DeregisterElasticIp
	opsworks:DeregisterInstance
	opsworks:DeregisterRdsDbInstance
	opsworks:DeregisterVolume

Prefijo de servicio	Acciones
	opsworks:DescribeAgentVersions
	opsworks:DescribeApps
	opsworks:DescribeCommands
	opsworks:DescribeDeployments
	opsworks:DescribeEcsClusters
	opsworks:DescribeElasticIps
	opsworks:DescribeElasticLoadBalancers
	opsworks:DescribeInstances
	opsworks:DescribeLayers
	opsworks:DescribeLoadBasedAutoScaling
	opsworks:DescribeMyUserProfile
	opsworks:DescribeOperatingSystems
	opsworks:DescribePermissions
	opsworks:DescribeRaidArrays
	opsworks:DescribeRdsDbInstances
	opsworks:DescribeServiceErrors
	opsworks:DescribeStackProvisioningParameters
	opsworks:DescribeStacks
	opsworks:DescribeStackSummary
	opsworks:DescribeTimeBasedAutoScaling
	opsworks:DescribeUserProfiles

Prefijo de servicio	Acciones
	opsworks:DescribeVolumes
	opsworks:DetachElasticLoadBalancer
	opsworks:DisassociateElasticIp
	opsworks:GetHostnameSuggestion
	opsworks:GrantAccess
	opsworks:RebootInstance
	opsworks:RegisterEcsCluster
	opsworks:RegisterElasticIp
	opsworks:RegisterInstance
	opsworks:RegisterRdsDbInstance
	opsworks:RegisterVolume
	opsworks:SetLoadBasedAutoScaling
	opsworks:SetPermission
	opsworks:SetTimeBasedAutoScaling
	opsworks:StartInstance
	opsworks:StartStack
	opsworks:StopInstance
	opsworks:StopStack
	opsworks:UnassignInstance
	opsworks:UnassignVolume
	opsworks:UpdateApp

Prefijo de servicio	Acciones
	opsworks:UpdateElasticIp
	opsworks:UpdateInstance
	opsworks:UpdateLayer
	opsworks:UpdateMyUserProfile
	opsworks:UpdateRdsDbInstance
	opsworks:UpdateStack
	opsworks:UpdateUserProfile
	opsworks:UpdateVolume

Prefijo de servicio	Acciones
opsworks-cm	opsworks-cm:AssociateNode opsworks-cm:CreateBackup opsworks-cm:CreateServer opsworks-cm:DeleteBackup opsworks-cm:DeleteServer opsworks-cm:DescribeAccountAttributes opsworks-cm:DescribeBackups opsworks-cm:DescribeEvents opsworks-cm:DescribeNodeAssociationStatus opsworks-cm:DescribeServers opsworks-cm:DisassociateNode opsworks-cm:ExportServerEngineAttribute opsworks-cm:RestoreServer opsworks-cm:StartMaintenance opsworks-cm:UpdateServer opsworks-cm:UpdateServerEngineAttributes

Prefijo de servicio	Acciones
organizations	organizations:AcceptHandshake
	organizaciones: AttachPolicy
	organizaciones: CancelHandshake
	organizations:CloseAccount
	organizations:CreateAccount
	organizations:CreateGovCloudAccount
	organizations:CreateOrganization
	organizations:CreateOrganizationalUnit
	organizations:CreatePolicy
	organizations:DeclineHandshake
	organizations:DeleteOrganization
	organizations:DeleteOrganizationalUnit
	organizations:DeletePolicy
	organizations:DeleteResourcePolicy
	organizations:DeregisterDelegatedAdministrator
	organizations:DescribeAccount
	organizations:DescribeCreateAccountStatus
	organizations:DescribeEffectivePolicy
	organizations:DescribeHandshake
	organizations:DescribeOrganization
	organizations:DescribeOrganizationalUnit

Prefijo de servicio	Acciones
	organizations:DescribePolicy
	organizations:DescribeResourcePolicy
	organizations:DetachPolicy
	organizations:DisableAWSServiceAccess
	organizations:DisablePolicyType
	organizations:EnableAllFeatures
	organizations:EnableAWSServiceAccess
	organizations:EnablePolicyType
	organizations:InviteAccountToOrganization
	organizations:LeaveOrganization
	organizations:ListAccounts
	organizations:ListAccountsForParent
	organizations:ListAWSServiceAccessForOrganization
	organizations:ListChildren
	organizations:ListCreateAccountStatus
	organizations:ListDelegatedAdministrators
	organizations:ListDelegatedServicesForAccount
	organizations:ListHandshakesForAccount
	organizations:ListHandshakesForOrganization
	organizations:ListOrganizationalUnitsForParent
	organizations:ListParents

Prefijo de servicio	Acciones
	organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations:UpdateOrganizationalUnit organizations:UpdatePolicy

Prefijo de servicio	Acciones
outposts	outposts:CancelOrder outposts:CreateOrder outposts:CreateOutpost outposts:CreatePrivateConnectivityConfig outposts:CreateSite outposts:DeleteOutpost outposts:DeleteSite outposts:GetCatalogItem outposts:GetConnection outposts:GetOrder outposts:GetOutpost outposts:GetOutpostInstanceTypes outposts:GetPrivateConnectivityConfig outposts:GetSite outposts:GetSiteAddress outposts:ListAssets outposts:ListCatalogItems outposts:ListOrders outposts:ListOutposts outposts:ListSites outposts:StartConnection

Prefijo de servicio	Acciones
	outposts:UpdateOutpost outposts:UpdateSite outposts:UpdateSiteAddress outposts:UpdateSiteRackPhysicalProperties

Prefijo de servicio	Acciones
panorama	panorama:CreateApplicationInstance
	panorama:CreateJobForDevices
	panorama:CreateNodeFromTemplateJob
	panorama:CreatePackage
	panorama:CreatePackageImportJob
	panorama:DeleteDevice
	panorama:DeletePackage
	panorama:DeregisterPackageVersion
	panorama:DescribeApplicationInstance
	panorama:DescribeApplicationInstanceDetails
	panorama:DescribeDevice
	panorama:DescribeDeviceJob
	panorama:DescribeNode
	panorama:DescribeNodeFromTemplateJob
	panorama:DescribePackage
	panorama:DescribePackageImportJob
	panorama:DescribePackageVersion
	panorama:ListApplicationInstanceDependencies
	panorama:ListApplicationInstanceNodeInstances
	panorama:ListApplicationInstances
	panorama:ListDevices

Prefijo de servicio	Acciones
	panorama:ListDevicesJobs panorama:ListNodeFromTemplateJobs panorama:ListNodes panorama:ListPackageImportJobs panorama:ListPackages panorama:ProvisionDevice panorama:RegisterPackageVersion panorama:RemoveApplicationInstance panorama:SignalApplicationInstanceNodeInstances panorama:UpdateDeviceMetadata
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports

Prefijo de servicio	Acciones
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

Prefijo de servicio	Acciones
profile	profile:AddProfileKey
	profile:CreateCalculatedAttributeDefinition
	profile:CreateDomain
	profile:CreateEventStream
	profile:CreateProfile
	profile:DeleteCalculatedAttributeDefinition
	profile:DeleteDomain
	profile:DeleteEventStream
	profile:DeleteIntegration
	profile:DeleteProfile
	profile:DeleteProfileKey
	profile:DeleteProfileObject
	profile:DeleteProfileObjectType
	profile:DeleteWorkflow
	profile:GetAutoMergingPreview
	profile:GetCalculatedAttributeDefinition
	profile:GetCalculatedAttributeForProfile
	profile:GetDomain
	profile:GetEventStream
	profile:GetIdentityResolutionJob
	profile:GetIntegration

Prefijo de servicio	Acciones
	profile:GetMatches
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile:GetSimilarProfiles
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations
	profile:ListProfileObjects
	profile:ListProfileObjectTypes
	profile:ListProfileObjectTypeTemplates
	profile:ListRuleBasedMatches
	profile:ListWorkflows
	profile:MergeProfiles
	profile:PutIntegration
	profile:PutProfileObject

Prefijo de servicio	Acciones
	profile:PutProfileObjectType profile:SearchProfiles profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile

Prefijo de servicio	Acciones
qldb	qldb:CancelJournalKinesisStream qldb:CreateLedger qldb:DeleteLedger qldb:DescribeJournalKinesisStream qldb:DescribeJournalS3Export qldb:DescribeLedger qldb:ExportJournalToS3 qldb:GetBlock qldb:GetDigest qldb:GetRevision qldb:ListJournalKinesisStreamsForLedger qldb:ListJournalS3Exports qldb:ListJournalS3ExportsForLedger qldb:ListLedgers qldb:StreamJournalToKinesis qldb:UpdateLedger qldb:UpdateLedgerPermissionsMode

Prefijo de servicio	Acciones
ram	ram:AcceptResourceShareInvitation
	ram:AssociateResourceShare
	ram:AssociateResourceSharePermission
	ram:CreatePermission
	ram:CreatePermissionVersion
	ram:CreateResourceShare
	ram:DeletePermission
	ram:DeletePermissionVersion
	ram:DeleteResourceShare
	ram:DisassociateResourceShare
	ram:DisassociateResourceSharePermission
	ram:EnableSharingWithAwsOrganization
	ram:GetPermission
	ram:GetResourcePolicies
	ram:GetResourceShareAssociations
	ram:GetResourceShareInvitations
	ram:GetResourceShares
	ram:ListPendingInvitationResources
	ram:ListPermissionAssociations
	ram:ListPermissions
	ram:ListPermissionVersions

Prefijo de servicio	Acciones
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResources ram:ListResourceSharePermissions ram:ListResourceTypes ram:PromotePermissionCreatedFromPolicy ram:PromoteResourceShareCreatedFromPolicy ram:RejectResourceShareInvitation ram:ReplacePermissionAssociations ram:SetDefaultPermissionVersion ram:UpdateResourceShare
rbin	rbin:CreateRule rbin:DeleteRule rbin:GetRule rbin:ListRules rbin:LockRule rbin:UnlockRule rbin:UpdateRule

Prefijo de servicio	Acciones
rds	rds:AddRoleToDBCluster
	rds:AddRoleToDBInstance
	rds:AddSourceIdentifierToSubscription
	rds:ApplyPendingMaintenanceAction
	rds:AuthorizeDBSecurityGroupIngress
	rds:BacktrackDBCluster
	rds:CancelExportTask
	rds:CopyDBClusterParameterGroup
	rds:CopyDBClusterSnapshot
	rds:CopyDBParameterGroup
	rds:CopyDBSnapshot
	rds:CopyOptionGroup
	rds:CreateCustomDBEngineVersion
	rds:CreateDBClusterParameterGroup
	rds:CreateDBClusterSnapshot
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSnapshot
	rds:CreateDBSubnetGroup

Prefijo de servicio	Acciones
	rds:CreateEventSubscription
	rds:CreateGlobalCluster
	rds:CreateOptionGroup
	rds:DeleteBlueGreenDeployment
	rds:DeleteDBClusterAutomatedBackup
	rds:DeleteDBClusterParameterGroup
	rds:DeleteDBClusterSnapshot
	rds:DeleteDBInstanceAutomatedBackup
	rds:DeleteDBParameterGroup
	rds:DeleteDBProxy
	rds:DeleteDBProxyEndpoint
	rds:DeleteDBSecurityGroup
	rds:DeleteDBSnapshot
	rds:DeleteDBSubnetGroup
	rds:DeleteEventSubscription
	rds:DeleteGlobalCluster
	rds:DeleteOptionGroup
	rds:DeregisterDBProxyTargets
	rds:DescribeAccountAttributes
	rds:DescribeBlueGreenDeployments
	rds:DescribeCertificates

Prefijo de servicio	Acciones
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes
	rds:DescribeDBSnapshots

Prefijo de servicio	Acciones
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEvents
	rds:DescribeEventSubscriptions
	rds:DescribeExportTasks
	rds:DescribeGlobalClusters
	rds:DescribeOptionGroupOptions
	rds:DescribeOptionGroups
	rds:DescribeOrderableDBInstanceOptions
	rds:DescribePendingMaintenanceActions
	rds:DescribeReservedDBInstances
	rds:DescribeReservedDBInstancesOfferings
	rds:DescribeSourceRegions
	rds:DescribeValidDBInstanceModifications
	rds:DownloadCompleteDBLogFile
	rds:DownloadDBLogFilePortion
	rds:FailoverDBCluster
	rds:FailoverGlobalCluster
	rds:ModifyActivityStream

Prefijo de servicio	Acciones
	rds:ModifyCertificates
	rds:ModifyCurrentDBClusterCapacity
	rds:ModifyDBClusterEndpoint
	rds:ModifyDBClusterParameterGroup
	rds:ModifyDBClusterSnapshotAttribute
	rds:ModifyDBParameterGroup
	rds:ModifyDBProxy
	rds:ModifyDBProxyEndpoint
	rds:ModifyDBProxyTargetGroup
	rds:ModifyDBSnapshot
	rds:ModifyDBSnapshotAttribute
	rds:ModifyDBSubnetGroup
	rds:ModifyEventSubscription
	rds:ModifyGlobalCluster
	rds:ModifyOptionGroup
	rds:PurchaseReservedDBInstancesOffering
	rds:RebootDBCluster
	rds:RegisterDBProxyTargets
	rds:RemoveFromGlobalCluster
	rds:RemoveRoleFromDBCluster
	rds:RemoveRoleFromDBInstance

Prefijo de servicio	Acciones
	rds:RemoveSourceIdentifierFromSubscription
	rds:ResetDBClusterParameterGroup
	rds:ResetDBParameterGroup
	rds:RestoreDBClusterFromS3
	rds:RestoreDBClusterFromSnapshot
	rds:RestoreDBClusterToPointInTime
	rds:RestoreDBInstanceFromDBSnapshot
	rds:RestoreDBInstanceFromS3
	rds:RestoreDBInstanceToPointInTime
	rds:RevokeDBSecurityGroupIngress
	rds:StartActivityStream
	rds:StartDBCluster
	rds:StartDBInstance
	rds:StartDBInstanceAutomatedBackupsReplication
	rds:StartExportTask
	rds:StopActivityStream
	rds:StopDBCluster
	rds:StopDBInstance
	rds:StopDBInstanceAutomatedBackupsReplication
	rds:SwitchoverBlueGreenDeployment
	rds:SwitchoverGlobalCluster

Prefijo de servicio	Acciones
	rds:SwitchoverReadReplica

Prefijo de servicio	Acciones
redshift	redshift:AcceptReservedNodeExchange redshift:AddPartner redshift:AssociateDataShareConsumer redshift:AuthorizeClusterSecurityGroupIngress redshift:AuthorizeDataShare redshift:AuthorizeEndpointAccess redshift:AuthorizeSnapshotAccess redshift:BatchDeleteClusterSnapshots redshift:BatchModifyClusterSnapshots redshift:CancelResize redshift:CopyClusterSnapshot redshift:CreateAuthenticationProfile redshift:CreateCluster redshift:CreateClusterParameterGroup redshift:CreateClusterSecurityGroup redshift:CreateClusterSnapshot redshift:CreateClusterSubnetGroup redshift:CreateCustomDomainAssociation redshift:CreateEndpointAccess redshift:CreateEventSubscription redshift:CreateHsmClientCertificate

Prefijo de servicio	Acciones
	redshift:CreateHsmConfiguration redshift:CreateScheduledAction redshift:CreateSnapshotCopyGrant redshift:CreateSnapshotSchedule redshift:CreateUsageLimit redshift:DeauthorizeDataShare redshift:DeleteAuthenticationProfile redshift:DeleteCluster redshift:DeleteClusterParameterGroup redshift:DeleteClusterSecurityGroup redshift:DeleteClusterSnapshot redshift:DeleteClusterSubnetGroup redshift:DeleteCustomDomainAssociation redshift:DeleteEndpointAccess redshift:DeleteEventSubscription redshift:DeleteHsmClientCertificate redshift:DeleteHsmConfiguration redshift:DeletePartner redshift:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant redshift:DeleteSnapshotSchedule

Prefijo de servicio	Acciones
	redshift:DeleteUsageLimit redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories redshift:DescribeEvents

Prefijo de servicio	Acciones
	redshift:DescribeEventSubscriptions redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging

Prefijo de servicio	Acciones
	redshift:EnableSnapshotCopy redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule

Prefijo de servicio	Acciones
	redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:RebootCluster redshift:RejectDataShare redshift:ResetClusterParameterGroup redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus

Prefijo de servicio	Acciones
redshift-data	redshift-data:BatchExecuteStatement
	redshift-data:CancelStatement
	redshift-data:DescribeStatement
	redshift-data:DescribeTable
	redshift-data:ExecuteStatement
	redshift-data:GetStatementResult
	redshift-data:ListDatabases
	redshift-data:ListSchemas
	redshift-data:ListStatements
	redshift-data:ListTables

Prefijo de servicio	Acciones
refactor-spaces	refactor-spaces:CreateApplication refactor-spaces:CreateEnvironment refactor-spaces:CreateRoute refactor-spaces:CreateService refactor-spaces:DeleteApplication refactor-spaces:DeleteEnvironment refactor-spaces:DeleteResourcePolicy refactor-spaces:DeleteRoute refactor-spaces:DeleteService refactor-spaces:GetApplication refactor-spaces:GetEnvironment refactor-spaces:GetResourcePolicy refactor-spaces:GetRoute refactor-spaces:GetService refactor-spaces:ListApplications refactor-spaces:ListEnvironments refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

Prefijo de servicio	Acciones
rekognition	rekognition:AssociateFaces
	rekognition:CompareFaces
	rekognition:CopyProjectVersion
	rekognition:CreateCollection
	rekognition:CreateDataset
	rekognition:CreateFaceLivenessSession
	rekognition:CreateProject
	rekognition:CreateProjectVersion
	rekognition:CreateStreamProcessor
	rekognition:CreateUser
	rekognition:DeleteCollection
	rekognition:DeleteDataset
	rekognition:DeleteFaces
	rekognition:DeleteProject
	rekognition:DeleteProjectPolicy
	rekognition:DeleteProjectVersion
	rekognition:DeleteStreamProcessor
	rekognition:DeleteUser
	rekognition:DescribeCollection
	rekognition:DescribeDataset
	rekognition:DescribeProjects

Prefijo de servicio	Acciones
	rekognition:DescribeProjectVersions
	rekognition:DescribeStreamProcessor
	rekognition:DetectCustomLabels
	rekognition:DetectFaces
	rekognition:DetectLabels
	rekognition:DetectModerationLabels
	rekognition:DetectProtectiveEquipment
	rekognition:DetectText
	rekognition:DisassociateFaces
	rekognition:DistributeDatasetEntries
	rekognition:GetCelebrityInfo
	rekognition:GetCelebrityRecognition
	rekognition:GetContentModeration
	rekognition:GetFaceDetection
	rekognition:GetFaceLivenessSessionResults
	rekognition:GetFaceSearch
	rekognition:GetLabelDetection
	rekognition:GetMediaAnalysisJob
	rekognition:GetPersonTracking
	rekognition:GetSegmentDetection
	rekognition:GetTextDetection

Prefijo de servicio	Acciones
	rekognition:IndexFaces
	rekognition:ListCollections
	rekognition:ListDatasetEntries
	rekognition:ListDatasetLabels
	rekognition:ListFaces
	rekognition:ListMediaAnalysisJobs
	rekognition:ListProjectPolicies
	rekognition:ListStreamProcessors
	rekognition:ListUsers
	rekognition:PutProjectPolicy
	rekognition:RecognizeCelebrities
	rekognition:SearchFaces
	rekognition:SearchFacesByImage
	rekognition:SearchUsers
	rekognition:SearchUsersByImage
	rekognition:StartCelebrityRecognition
	rekognition:StartContentModeration
	rekognition:StartFaceDetection
	rekognition:StartFaceLivenessSession
	rekognition:StartFaceSearch
	rekognition:StartLabelDetection

Prefijo de servicio	Acciones
	rekognition:StartMediaAnalysisJob
	rekognition:StartPersonTracking
	rekognition:StartProjectVersion
	rekognition:StartSegmentDetection
	rekognition:StartStreamProcessor
	rekognition:StartTextDetection
	rekognition:StopProjectVersion
	rekognition:StopStreamProcessor
	rekognition:UpdateDatasetEntries
	rekognition:UpdateStreamProcessor

Prefijo de servicio	Acciones
resiliencehub	resiliencehub:AddDraftAppVersionResourceMappings resiliencehub:CreateApp resiliencehub:CreateAppVersionAppComponent resiliencehub:CreateAppVersionResource resiliencehub:CreateRecommendationTemplate resiliencehub:CreateResiliencyPolicy resiliencehub:DeleteApp resiliencehub:DeleteAppAssessment resiliencehub:DeleteAppInputSource resiliencehub:DeleteAppVersionAppComponent resiliencehub:DeleteAppVersionResource resiliencehub:DeleteRecommendationTemplate resiliencehub:DeleteResiliencyPolicy resiliencehub:DescribeApp resiliencehub:DescribeAppAssessment resiliencehub:DescribeAppVersion resiliencehub:DescribeAppVersionAppComponent resiliencehub:DescribeAppVersionResource resiliencehub:DescribeAppVersionResourcesResolutionStatus resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus

Prefijo de servicio	Acciones
	resiliencehub:DescribeResiliencyPolicy resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListApps resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RemoveDraftAppVersionResourceMappings

Prefijo de servicio	Acciones
	resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

Prefijo de servicio	Acciones
resource-explorer-2	resource-explorer-2:AssociateDefaultView resource-explorer-2:BatchGetView resource-explorer-2:CreateIndex resource-explorer-2:CreateView resource-explorer-2:DeleteIndex resource-explorer-2:DeleteView resource-explorer-2:DisassociateDefaultView resource-explorer-2:GetDefaultView resource-explorer-2:GetIndex resource-explorer-2:ListIndexes resource-explorer-2:ListSupportedResourceTypes resource-explorer-2:ListViews resource-explorer-2:Search resource-explorer-2:UpdateIndexType resource-explorer-2:UpdateView

Prefijo de servicio	Acciones
resource-groups	resource-groups:CreateGroup resource-groups:DeleteGroup resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroups resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

Prefijo de servicio	Acciones
robomaker	robomaker:BatchDeleteWorlds
	robomaker:BatchDescribeSimulationJob
	robomaker:CancelDeploymentJob
	robomaker:CancelSimulationJob
	robomaker:CancelSimulationJobBatch
	robomaker:CancelWorldExportJob
	robomaker:CancelWorldGenerationJob
	robomaker:CreateDeploymentJob
	robomaker:CreateFleet
	robomaker:CreateRobot
	robomaker:CreateRobotApplication
	robomaker:CreateRobotApplicationVersion
	robomaker:CreateSimulationApplication
	robomaker:CreateSimulationApplicationVersion
	robomaker:CreateSimulationJob
	robomaker:CreateWorldExportJob
	robomaker:CreateWorldGenerationJob
	robomaker:CreateWorldTemplate
	robomaker:DeleteFleet
	robomaker:DeleteRobot
	robomaker:DeleteRobotApplication

Prefijo de servicio	Acciones
	robomaker:DeleteSimulationApplication
	robomaker:DeleteWorldTemplate
	robomaker:DeregisterRobot
	robomaker:DescribeDeploymentJob
	robomaker:DescribeFleet
	robomaker:DescribeRobot
	robomaker:DescribeRobotApplication
	robomaker:DescribeSimulationApplication
	robomaker:DescribeSimulationJob
	robomaker:DescribeSimulationJobBatch
	robomaker:DescribeWorld
	robomaker:DescribeWorldExportJob
	robomaker:DescribeWorldGenerationJob
	robomaker:DescribeWorldTemplate
	robomaker:GetWorldTemplateBody
	robomaker:ListDeploymentJobs
	robomaker:ListFleets
	robomaker:ListRobotApplications
	robomaker:ListRobots
	robomaker:ListSimulationApplications
	robomaker:ListSimulationJobBatches

Prefijo de servicio	Acciones
	robomaker:ListSimulationJobs robomaker:ListWorldExportJobs robomaker:ListWorldGenerationJobs robomaker:ListWorlds robomaker:ListWorldTemplates robomaker:RegisterRobot robomaker:RestartSimulationJob robomaker:StartSimulationJobBatch robomaker:SyncDeploymentJob robomaker:UpdateRobotApplication robomaker:UpdateSimulationApplication robomaker:UpdateWorldTemplate

Prefijo de servicio	Acciones
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor rolesanywhere:DeleteCrl rolesanywhere:DeleteProfile rolesanywhere:DeleteTrustAnchor rolesanywhere:DisableCrl rolesanywhere:DisableProfile rolesanywhere:DisableTrustAnchor rolesanywhere:EnableCrl rolesanywhere:EnableProfile rolesanywhere:EnableTrustAnchor rolesanywhere:GetCrl rolesanywhere:GetProfile rolesanywhere:GetSubject rolesanywhere:GetTrustAnchor rolesanywhere:ImportCrl rolesanywhere:ListCrls rolesanywhere:ListProfiles rolesanywhere:ListSubjects rolesanywhere:ListTrustAnchors rolesanywhere:PutNotificationSettings

Prefijo de servicio	Acciones
	rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

Prefijo de servicio	Acciones
route53	route53:ActivateKeySigningKey
	route53:AssociateVPCWithHostedZone
	route53:ChangeCidrCollection
	route53:ChangeResourceRecordSets
	route53:CreateCidrCollection
	route53:CreateHealthCheck
	route53:CreateHostedZone
	route53:CreateKeySigningKey
	route53:CreateQueryLoggingConfig
	route53:CreateReusableDelegationSet
	route53:CreateTrafficPolicy
	route53:CreateTrafficPolicyInstance
	route53:CreateTrafficPolicyVersion
	route53:CreateVPCAssociationAuthorization
	route53:DeactivateKeySigningKey
	route53:DeleteCidrCollection
	route53:DeleteHealthCheck
	route53:DeleteHostedZone
	route53:DeleteKeySigningKey
	route53:DeleteQueryLoggingConfig
	route53:DeleteReusableDelegationSet

Prefijo de servicio	Acciones
	route53:DeleteTrafficPolicy
	route53:DeleteTrafficPolicyInstance
	route53:DeleteVPCAssociationAuthorization
	route53:DisableHostedZoneDNSSEC
	route53:DisassociateVPCFromHostedZone
	route53:EnableHostedZoneDNSSEC
	route53:GetAccountLimit
	route53:GetChange
	route53:GetCheckerIpRanges
	route53:GetDNSSEC
	route53:GetGeoLocation
	route53:GetHealthCheck
	route53:GetHealthCheckCount
	route53:GetHealthCheckLastFailureReason
	route53:GetHealthCheckStatus
	route53:GetHostedZone
	route53:GetHostedZoneCount
	route53:GetHostedZoneLimit
	route53:GetQueryLoggingConfig
	route53:GetReusableDelegationSet
	route53:GetReusableDelegationSetLimit

Prefijo de servicio	Acciones
	route53:GetTrafficPolicy
	route53:GetTrafficPolicyInstance
	route53:GetTrafficPolicyInstanceCount
	route53:ListCidrBlocks
	route53:ListCidrCollections
	route53:ListCidrLocations
	route53:ListGeoLocations
	route53:ListHealthChecks
	route53:ListHostedZones
	route53:ListHostedZonesByName
	route53:ListHostedZonesByVPC
	route53:ListQueryLoggingConfigs
	route53:ListResourceRecordSets
	route53:ListReusableDelegationSets
	route53:ListTrafficPolicies
	route53:ListTrafficPolicyInstances
	route53:ListTrafficPolicyInstancesByHostedZone
	route53:ListTrafficPolicyInstancesByPolicy
	route53:ListTrafficPolicyVersions
	route53:ListVPCAssociationAuthorizations
	route53:TestDNSAnswer

Prefijo de servicio	Acciones
	route53:UpdateHealthCheck route53:UpdateHostedZoneComment route53:UpdateTrafficPolicyComment route53:UpdateTrafficPolicyInstance

Prefijo de servicio	Acciones
route53-recovery-control-config	route53-recovery-control-config:CreateCluster route53-recovery-control-config:CreateControlPanel route53-recovery-control-config:CreateRoutingControl route53-recovery-control-config:CreateSafetyRule route53-recovery-control-config:DeleteCluster route53-recovery-control-config:DeleteControlPanel route53-recovery-control-config:DeleteRoutingControl route53-recovery-control-config:DeleteSafetyRule route53-recovery-control-config:DescribeCluster route53-recovery-control-config:DescribeControlPanel route53-recovery-control-config:DescribeRoutingControl route53-recovery-control-config:DescribeSafetyRule route53-recovery-control-config:GetResourcePolicy route53-recovery-control-config:ListAssociatedRoute53HealthChecks route53-recovery-control-config:ListClusters route53-recovery-control-config:ListControlPanels route53-recovery-control-config:ListRoutingControls route53-recovery-control-config:ListSafetyRules route53-recovery-control-config:UpdateControlPanel route53-recovery-control-config:UpdateRoutingControl

Prefijo de servicio	Acciones
	route53-recovery-control-config:UpdateSafetyRule

Prefijo de servicio	Acciones
route53-recovery-readiness	route53-recovery-readiness:CreateCell route53-recovery-readiness:CreateCrossAccountAuthorization route53-recovery-readiness:CreateReadinessCheck route53-recovery-readiness:CreateRecoveryGroup route53-recovery-readiness:CreateResourceSet route53-recovery-readiness:DeleteCell route53-recovery-readiness:DeleteCrossAccountAuthorization route53-recovery-readiness:DeleteReadinessCheck route53-recovery-readiness:DeleteRecoveryGroup route53-recovery-readiness:DeleteResourceSet route53-recovery-readiness:GetArchitectureRecommendations route53-recovery-readiness:GetCell route53-recovery-readiness:GetCellReadinessSummary route53-recovery-readiness:GetReadinessCheck route53-recovery-readiness:GetReadinessCheckResourceStatus route53-recovery-readiness:GetReadinessCheckStatus route53-recovery-readiness:GetRecoveryGroup route53-recovery-readiness:GetRecoveryGroupReadinessSummary route53-recovery-readiness:GetResourceSet route53-recovery-readiness:ListCells route53-recovery-readiness:ListCrossAccountAuthorizations

Prefijo de servicio	Acciones
	route53-recovery-readiness:ListReadinessChecks
	route53-recovery-readiness:ListRecoveryGroups
	route53-recovery-readiness:ListResourceSets
	route53-recovery-readiness:ListRules
	route53-recovery-readiness:UpdateCell
	route53-recovery-readiness:UpdateReadinessCheck
	route53-recovery-readiness:UpdateRecoveryGroup
	route53-recovery-readiness:UpdateResourceSet

Prefijo de servicio	Acciones
route53resolver	route53resolver:AssociateFirewallRuleGroup route53resolver:AssociateResolverEndpointIpAddress route53resolver:AssociateResolverQueryLogConfig route53resolver:AssociateResolverRule route53resolver:CreateFirewallDomainList route53resolver:CreateFirewallRule route53resolver:CreateFirewallRuleGroup route53resolver:CreateResolverEndpoint route53resolver:CreateResolverQueryLogConfig route53resolver:CreateResolverRule route53resolver:DeleteFirewallDomainList route53resolver:DeleteFirewallRule route53resolver:DeleteFirewallRuleGroup route53resolver:DeleteOutpostResolver route53resolver:DeleteResolverEndpoint route53resolver:DeleteResolverQueryLogConfig route53resolver:DeleteResolverRule route53resolver:DisassociateFirewallRuleGroup route53resolver:DisassociateResolverEndpointIpAddress route53resolver:DisassociateResolverQueryLogConfig route53resolver:DisassociateResolverRule

Prefijo de servicio	Acciones
	route53resolver:GetFirewallConfig route53resolver:GetFirewallDomainList route53resolver:GetFirewallRuleGroup route53resolver:GetFirewallRuleGroupAssociation route53resolver:GetFirewallRuleGroupPolicy route53resolver:GetOutpostResolver route53resolver:GetResolverConfig route53resolver:GetResolverDnssecConfig route53resolver:GetResolverEndpoint route53resolver:GetResolverQueryLogConfig route53resolver:GetResolverQueryLogConfigAssociation route53resolver:GetResolverQueryLogConfigPolicy route53resolver:GetResolverRule route53resolver:GetResolverRuleAssociation route53resolver:GetResolverRulePolicy route53resolver:ImportFirewallDomains route53resolver:ListFirewallConfigs route53resolver:ListFirewallDomainLists route53resolver:ListFirewallDomains route53resolver:ListFirewallRuleGroupAssociations route53resolver:ListFirewallRuleGroups

Prefijo de servicio	Acciones
	route53resolver:ListFirewallRules route53resolver:ListOutpostResolvers route53resolver:ListResolverConfigs route53resolver:ListResolverDnssecConfigs route53resolver:ListResolverEndpointIpAddresses route53resolver:ListResolverEndpoints route53resolver:ListResolverQueryLogConfigAssociations route53resolver:ListResolverQueryLogConfigs route53resolver:ListResolverRuleAssociations route53resolver:ListResolverRules route53resolver:PutFirewallRuleGroupPolicy route53resolver:PutResolverQueryLogConfigPolicy route53resolver:UpdateFirewallConfig route53resolver:UpdateFirewallDomains route53resolver:UpdateFirewallRule route53resolver:UpdateFirewallRuleGroupAssociation route53resolver:UpdateOutpostResolver route53resolver:UpdateResolverConfig route53resolver:UpdateResolverDnssecConfig route53resolver:UpdateResolverEndpoint route53resolver:UpdateResolverRule

Prefijo de servicio	Acciones
rum	rum:BatchCreateRumMetricDefinitions rum:BatchDeleteRumMetricDefinitions rum:BatchGetRumMetricDefinitions rum:CreateAppMonitor rum:DeleteAppMonitor rum:DeleteRumMetricsDestination rum:GetAppMonitor rum:GetAppMonitorData rum:ListAppMonitors rum:ListRumMetricsDestinations rum:PutRumMetricsDestination rum:UpdateAppMonitor rum:UpdateRumMetricDefinition

Prefijo de servicio	Acciones
s3	s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:DeleteBucket s3:PutAnalyticsConfiguration s3:PutBucketCORS s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutBucketOwnershipControls s3:DeleteBucketPolicy s3:PutBucketPublicAccessBlock

Prefijo de servicio	Acciones
	s3:PutReplicationConfiguration s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:GetAccelerateConfiguration s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetBucketAcl s3:GetAnalyticsConfiguration s3:GetBucketCORS s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration

Prefijo de servicio	Acciones
	s3:GetLifecycleConfiguration s3:GetBucketLocation s3:GetBucketLogging s3:GetMetricsConfiguration s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetReplicationConfiguration s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetMultiRegionAccessPoint s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetObjectAttributes s3:GetStorageLensConfiguration s3:GetStorageLensDashboard

Prefijo de servicio	Acciones
	s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets s3:ListJobs s3:ListBucketMultipartUploads s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutBucketAcl s3:PutAnalyticsConfiguration s3:PutBucketCORS s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutBucketLogging s3:PutMetricsConfiguration

Prefijo de servicio	Acciones
	s3:PutBucketNotification s3:PutBucketObjectLockConfiguration s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutReplicationConfiguration s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutMultiRegionAccessPointPolicy s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateJobPriority s3:UpdateJobStatus
s3-outposts	s3-outposts:CreateEndpoint s3-outposts:DeleteEndpoint s3-outposts:ListEndpoints s3-outposts:ListOutpostsWithS3 s3-outposts:ListSharedEndpoints

Prefijo de servicio	Acciones
sagemaker-geospatial	sagemaker-geospatial:DeleteEarthObservationJob sagemaker-geospatial:DeleteVectorEnrichmentJob sagemaker-geospatial:ExportEarthObservationJob sagemaker-geospatial:ExportVectorEnrichmentJob sagemaker-geospatial:GetEarthObservationJob sagemaker-geospatial:GetRasterDataCollection sagemaker-geospatial:GetTile sagemaker-geospatial:GetVectorEnrichmentJob sagemaker-geospatial:ListEarthObservationJobs sagemaker-geospatial:ListRasterDataCollections sagemaker-geospatial:ListVectorEnrichmentJobs sagemaker-geospatial:SearchRasterDataCollection sagemaker-geospatial:StartEarthObservationJob sagemaker-geospatial:StartVectorEnrichmentJob sagemaker-geospatial:StopEarthObservationJob sagemaker-geospatial:StopVectorEnrichmentJob

Prefijo de servicio	Acciones
savingsplans	savingsplans:CreateSavingsPlan savingsplans:DeleteQueuedSavingsPlan savingsplans:DescribeSavingsPlanRates savingsplans:DescribeSavingsPlans savingsplans:DescribeSavingsPlansOfferingRates savingsplans:DescribeSavingsPlansOfferings

Prefijo de servicio	Acciones
schemas	schemas:CreateDiscoverer schemas:CreateRegistry schemas:CreateSchema schemas:DeleteDiscoverer schemas:DeleteRegistry schemas:DeleteResourcePolicy schemas:DeleteSchema schemas:DeleteSchemaVersion schemas:DescribeCodeBinding schemas:DescribeDiscoverer schemas:DescribeRegistry schemas:DescribeSchema schemas:ExportSchema schemas:GetCodeBindingSource schemas:GetDiscoveredSchema schemas:GetResourcePolicy schemas:ListDiscoverers schemas:ListRegistries schemas:ListSchemas schemas:ListSchemaVersions schemas:PutCodeBinding

Prefijo de servicio	Acciones
	schemas:PutResourcePolicy schemas:SearchSchemas schemas:StartDiscoverer schemas:StopDiscoverer schemas:UpdateDiscoverer schemas:UpdateRegistry schemas:UpdateSchema
sdb	sdb:CreateDomain sdb:DeleteDomain sdb:DomainMetadata sdb:ListDomains

Prefijo de servicio	Acciones
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecrets secretsmanager:ListSecretVersionIds secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

Prefijo de servicio	Acciones
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:CreateActionTarget
	securityhub:CreateAutomationRule
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight

Prefijo de servicio	Acciones
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub:GetAdministratorAccount
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings

Prefijo de servicio	Acciones
	securityhub:GetInsightResults securityhub:GetInsights securityhub:GetInvitationsCount securityhub:GetMasterAccount securityhub:GetMembers securityhub:InviteMembers securityhub:ListAutomationRules securityhub:ListEnabledProductsForImport securityhub:ListFindingAggregators securityhub:ListInvitations securityhub:ListMembers securityhub:ListOrganizationAdminAccounts securityhub:ListSecurityControlDefinitions securityhub:ListStandardsControlAssociations securityhub:UpdateActionTarget securityhub:UpdateFindingAggregator securityhub:UpdateFindings securityhub:UpdateInsight securityhub:UpdateOrganizationConfiguration securityhub:UpdateSecurityHubConfiguration

Prefijo de servicio	Acciones
securitylake	securitylake:CreateAwsLogSource securitylake:CreateCustomLogSource securitylake:CreateDataLakeExceptionSubscription securitylake:CreateDataLakeOrganizationConfiguration securitylake:CreateSubscriber securitylake:CreateSubscriberNotification securitylake:DeleteAwsLogSource securitylake:DeleteCustomLogSource securitylake:DeleteDataLakeExceptionSubscription securitylake:DeleteDataLakeOrganizationConfiguration securitylake:DeleteSubscriber securitylake:DeleteSubscriberNotification securitylake:DeregisterDataLakeDelegatedAdministrator securitylake:GetDataLakeExceptionSubscription securitylake:GetDataLakeOrganizationConfiguration securitylake:GetDataLakeSources securitylake:GetSubscriber securitylake:ListDataLakes securitylake:ListLogSources securitylake:ListSubscribers securitylake:RegisterDataLakeDelegatedAdministrator

Prefijo de servicio	Acciones
	securitylake:UpdateDataLakeExceptionSubscription securitylake:UpdateSubscriber securitylake:UpdateSubscriberNotification
serverlessrepo	serverlessrepo:CreateApplication serverlessrepo:CreateApplicationVersion serverlessrepo:CreateCloudFormationChangeSet serverlessrepo:CreateCloudFormationTemplate serverlessrepo:DeleteApplication serverlessrepo:GetApplication serverlessrepo:GetApplicationPolicy serverlessrepo:GetCloudFormationTemplate serverlessrepo:ListApplicationDependencies serverlessrepo:ListApplications serverlessrepo:ListApplicationVersions serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

Prefijo de servicio	Acciones
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan

Prefijo de servicio	Acciones
	servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShares servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact

Prefijo de servicio	Acciones
	servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact

Prefijo de servicio	Acciones
	servicecatalog:ListStackInstancesForProvisionedProduct servicecatalog:NotifyProvisionProductEngineWorkflowResult servicecatalog:NotifyTerminateProvisionedProductEngine WorkflowResult servicecatalog:NotifyUpdateProvisionedProductEngineWor kflowResult servicecatalog:ProvisionProduct servicecatalog:RejectPortfolioShare servicecatalog:ScanProvisionedProducts servicecatalog:SearchProducts servicecatalog:SearchProductsAsAdmin servicecatalog:SearchProvisionedProducts servicecatalog:TerminateProvisionedProduct servicecatalog:UpdateConstraint servicecatalog:UpdatePortfolio servicecatalog:UpdatePortfolioShare servicecatalog:UpdateProduct servicecatalog:UpdateProvisionedProduct servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

Prefijo de servicio	Acciones
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	servicediscovery:DeregisterInstance
	servicediscovery:GetInstance
	servicediscovery:GetInstancesHealthStatus
	servicediscovery:GetNamespace
	servicediscovery:GetOperation
	servicediscovery:GetService
	servicediscovery:ListInstances
	servicediscovery:ListNamespaces
	servicediscovery:ListOperations
	servicediscovery:ListServices
	servicediscovery:RegisterInstance
	servicediscovery:UpdateHttpNamespace
	servicediscovery:UpdateInstanceCustomHealthStatus
	servicediscovery:UpdatePrivateDnsNamespace
	servicediscovery:UpdatePublicDnsNamespace

Prefijo de servicio	Acciones
	servicediscovery:UpdateService
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAssociationForServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease

Prefijo de servicio	Acciones
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	ses:CreateConfigurationSet
	ses:CreateConfigurationSetEventDestination
	ses:CreateConfigurationSetTrackingOptions
	ses:CreateContact
	ses:CreateContactList
	ses:CreateCustomVerificationEmailTemplate
	ses:CreateDedicatedIpPool
	ses:CreateDeliverabilityTestReport
	ses:CreateEmailIdentity
	ses:CreateEmailIdentityPolicy
	ses:CreateEmailTemplate
	ses:CreateImportJob
	ses:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateTemplate
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions

Prefijo de servicio	Acciones
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteVerifiedEmailAddress
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled

Prefijo de servicio	Acciones
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes

Prefijo de servicio	Acciones
	ses:GetImportJob
	ses:GetMessageInsights
	ses:GetSendQuota
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets

Prefijo de servicio	Acciones
	ses:ListRecommendations
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes

Prefijo de servicio	Acciones
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy
	ses:PutSuppressedDestination
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled

Prefijo de servicio	Acciones
	ses:UpdateConfigurationSetTrackingOptions
	ses:UpdateContact
	ses:UpdateContactList
	ses:UpdateCustomVerificationEmailTemplate
	ses:UpdateEmailIdentityPolicy
	ses:UpdateEmailTemplate
	ses:UpdateReceiptRule
	ses:UpdateTemplate
	ses:VerifyDomainDkim
	ses:VerifyDomainIdentity
	ses:VerifyEmailAddress
	ses:VerifyEmailIdentity

Prefijo de servicio	Acciones
shield	shield:AssociateDRTLogBucket shield:AssociateHealthCheck shield:AssociateProactiveEngagementDetails shield:CreateProtection shield:CreateProtectionGroup shield:CreateSubscription shield:DeleteProtection shield:DeleteProtectionGroup shield:DeleteSubscription shield:DescribeAttack shield:DescribeAttackStatistics shield:DescribeDRTAccess shield:DescribeEmergencyContactSettings shield:DescribeProtection shield:DescribeProtectionGroup shield:DescribeSubscription shield:DisableApplicationLayerAutomaticResponse shield:DisableProactiveEngagement shield:DisassociateDRTLogBucket shield:DisassociateDRTRole shield:DisassociateHealthCheck

Prefijo de servicio	Acciones
	shield:EnableApplicationLayerAutomaticResponse shield:EnableProactiveEngagement shield:GetSubscriptionState shield:ListAttacks shield:ListProtectionGroups shield:ListProtections shield:ListResourcesInProtectionGroup shield:UpdateApplicationLayerAutomaticResponse shield:UpdateEmergencyContactSettings shield:UpdateProtectionGroup shield:UpdateSubscription

Prefijo de servicio	Acciones
signer	signer:AddProfilePermission
	signer:CancelSigningProfile
	signer:DescribeSigningJob
	signer:GetRevocationStatus
	signer:GetSigningPlatform
	signer:GetSigningProfile
	signer:ListProfilePermissions
	signer:ListSigningJobs
	signer:ListSigningPlatforms
	signer:ListSigningProfiles
	signer:PutSigningProfile
	signer:RemoveProfilePermission
	signer:RevokeSignature
	signer:RevokeSigningProfile
	signer:SignPayload
	signer:StartSigningJob

Prefijo de servicio	Acciones
simspaceweaver	simspaceweaver:CreateSnapshot simspaceweaver:DeleteApp simspaceweaver:DeleteSimulation simspaceweaver:DescribeApp simspaceweaver:DescribeSimulation simspaceweaver:ListApps simspaceweaver:ListSimulations simspaceweaver:StartApp simspaceweaver:StartClock simspaceweaver:StartSimulation simspaceweaver:StopApp simspaceweaver:StopClock simspaceweaver:StopSimulation

Prefijo de servicio	Acciones
sms	sms:CreateApp sms:CreateReplicationJob sms:DeleteApp sms:DeleteAppLaunchConfiguration sms:DeleteAppReplicationConfiguration sms:DeleteAppValidationConfiguration sms:DeleteReplicationJob sms:DeleteServerCatalog sms:DisassociateConnector sms:GenerateChangeSet sms:GenerateTemplate sms:GetApp sms:GetAppLaunchConfiguration sms:GetAppReplicationConfiguration sms:GetAppValidationConfiguration sms:GetAppValidationOutput sms:GetConnectors sms:GetReplicationJobs sms:GetReplicationRuns sms:GetServers sms:ImportAppCatalog

Prefijo de servicio	Acciones
	sms:ImportServerCatalog
	sms:LaunchApp
	sms:ListApps
	sms:NotifyAppValidationOutput
	sms:PutAppLaunchConfiguration
	sms:PutAppReplicationConfiguration
	sms:PutAppValidationConfiguration
	sms:StartAppReplication
	sms:StartOnDemandAppReplication
	sms:StartOnDemandReplicationRun
	sms:StopAppReplication
	sms:TerminateApp
	sms:UpdateApp
	sms:UpdateReplicationJob

Prefijo de servicio	Acciones
sms-voice	sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteOptedOutNumber sms-voice:DeleteOptOutList sms-voice:DeletePool sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptedOutNumbers

Prefijo de servicio	Acciones
	sms-voice:DescribeOptOutLists sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DisassociateOriginationIdentity sms-voice:GetConfigurationSetEventDestinations sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:ReleasePhoneNumber sms-voice:RequestPhoneNumber sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber sms-voice:UpdatePool

Prefijo de servicio	Acciones
snowball	snowball:CancelCluster
	snowball:CancelJob
	snowball:CreateAddress
	snowball:CreateCluster
	snowball:CreateJob
	snowball:CreateLongTermPricing
	snowball:CreateReturnShippingLabel
	snowball:DescribeAddress
	snowball:DescribeAddresses
	snowball:DescribeCluster
	snowball:DescribeJob
	snowball:DescribeReturnShippingLabel
	snowball:GetJobManifest
	snowball:GetJobUnlockCode
	snowball:GetSnowballUsage
	snowball:GetSoftwareUpdates
	snowball:ListClusterJobs
	snowball:ListClusters
	snowball:ListCompatibleImages
	snowball:ListJobs
	snowball:ListLongTermPricing

Prefijo de servicio	Acciones
	snowball:ListPickupLocations snowball:ListServiceVersions snowball:UpdateCluster snowball:UpdateJob snowball:UpdateJobShipmentState snowball:UpdateLongTermPricing
sqs	sqs:AddPermission sqs:CancelMessageMoveTask sqs:CreateQueue sqs:DeleteQueue sqs:PurgeQueue sqs:RemovePermission sqs:SetQueueAttributes

Prefijo de servicio	Acciones
ssm	ssm:AssociateOpsItemRelatedItem
	ssm:CancelCommand
	ssm:CancelMaintenanceWindowExecution
	ssm:CreateActivation
	ssm:CreateAssociation
	ssm:CreateAssociationBatch
	ssm:CreateDocument
	ssm:CreateMaintenanceWindow
	ssm:CreateOpsItem
	ssm:CreateOpsMetadata
	ssm:CreatePatchBaseline
	ssm:CreateResourceDataSync
	ssm:DeleteActivation
	ssm:DeleteAssociation
	ssm:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter
	ssm:DeleteParameters
	ssm:DeletePatchBaseline

Prefijo de servicio	Acciones
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutions
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus
	ssm:DescribeInstanceInformation
	ssm:DescribeInstancePatches

Prefijo de servicio	Acciones
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroups
	ssm:DescribePatchGroupState
	ssm:DescribePatchProperties
	ssm:DescribeSessions
	ssm:DisassociateOpsItemRelatedItem
	ssm:GetAutomationExecution

Prefijo de servicio	Acciones
	ssm:GetCalendarState
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath
	ssm:GetPatchBaseline

Prefijo de servicio	Acciones
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociations
	ssm:ListAssociationVersions
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocuments
	ssm:ListDocumentVersions
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync
	ssm:ModifyDocumentPermission

Prefijo de servicio	Acciones
	ssm:PutComplianceItems
	ssm:PutInventory
	ssm:PutParameter
	ssm:PutResourcePolicy
	ssm:RegisterDefaultPatchBaseline
	ssm:RegisterManagedInstance
	ssm:RegisterPatchBaselineForPatchGroup
	ssm:RegisterTargetWithMaintenanceWindow
	ssm:RegisterTaskWithMaintenanceWindow
	ssm:ResetServiceSetting
	ssm:ResumeSession
	ssm:SendAutomationSignal
	ssm:SendCommand
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion
	ssm:UpdateAssociation

Prefijo de servicio	Acciones
	ssm:UpdateAssociationStatus
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline
	ssm:UpdateResourceDataSync
	ssm:UpdateServiceSetting

Prefijo de servicio	Acciones
ssm-incidents	ssm-incidents:CreateReplicationSet ssm-incidents:CreateResponsePlan ssm-incidents:CreateTimelineEvent ssm-incidents:DeleteIncidentRecord ssm-incidents:DeleteReplicationSet ssm-incidents:DeleteResourcePolicy ssm-incidents:DeleteResponsePlan ssm-incidents:DeleteTimelineEvent ssm-incidents:GetIncidentRecord ssm-incidents:GetReplicationSet ssm-incidents:GetResourcePolicies ssm-incidents:GetResponsePlan ssm-incidents:GetTimelineEvent ssm-incidents:ListIncidentRecords ssm-incidents:ListRelatedItems ssm-incidents:ListReplicationSets ssm-incidents:ListResponsePlans ssm-incidents:ListTimelineEvents ssm-incidents:PutResourcePolicy ssm-incidents:StartIncident ssm-incidents:UpdateDeletionProtection

Prefijo de servicio	Acciones
	ssm-incidents:UpdateIncidentRecord ssm-incidents:UpdateRelatedItems ssm-incidents:UpdateReplicationSet ssm-incidents:UpdateResponsePlan ssm-incidents:UpdateTimelineEvent

Prefijo de servicio	Acciones
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap:ListDatabases
	ssm-sap:ListOperations
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase
	ssm-sap:StartApplicationRefresh
	ssm-sap:UpdateApplicationSettings
	ssm-sap:UpdateHANABackupSettings

Prefijo de servicio	Acciones
states	states:CreateActivity states:CreateStateMachine states:CreateStateMachineAlias states:DeleteActivity states:DeleteStateMachine states:DeleteStateMachineAlias states:DeleteStateMachineVersion states:DescribeActivity states:DescribeExecution states:DescribeMapRun states:DescribeStateMachine states:DescribeStateMachineAlias states:DescribeStateMachineForExecution states:GetExecutionHistory states:ListActivities states:ListExecutions states:ListMapRuns states:ListStateMachineAliases states:ListStateMachines states:ListStateMachineVersions states:SendTaskFailure

Prefijo de servicio	Acciones
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken

Prefijo de servicio	Acciones
swf	swf:DeprecateActivityType swf:DeprecateDomain swf:DeprecateWorkflowType swf:DescribeActivityType swf:DescribeDomain swf:DescribeWorkflowType swf:ListActivityTypes swf:ListDomains swf:ListWorkflowTypes swf:RegisterActivityType swf:RegisterDomain swf:RegisterWorkflowType swf:UndeprecateActivityType swf:UndeprecateDomain swf:UndeprecateWorkflowType

Prefijo de servicio	Acciones
synthetics	synthetics:AssociateResource
	synthetics:CreateCanary
	synthetics:CreateGroup
	synthetics:DeleteCanary
	synthetics:DeleteGroup
	synthetics:DescribeCanaries
	synthetics:DescribeCanariesLastRun
	synthetics:DescribeRuntimeVersions
	synthetics:DisassociateResource
	synthetics:GetCanary
	synthetics:GetCanaryRuns
	synthetics:GetGroup
	synthetics:ListAssociatedGroups
	synthetics:ListGroupResources
	synthetics:ListGroups
	synthetics:StartCanary
	synthetics:StopCanary
	synthetics:UpdateCanary

Prefijo de servicio	Acciones
etiqueta	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

Prefijo de servicio	Acciones
textract	textract:AnalyzeDocument
	textract:AnalyzeExpense
	textract:AnalyzeID
	textract:CreateAdapter
	textract:CreateAdapterVersion
	textract:DeleteAdapter
	textract:DeleteAdapterVersion
	textract:DetectDocumentText
	textract:GetAdapter
	textract:GetAdapterVersion
	textract:GetDocumentAnalysis
	textract:GetDocumentTextDetection
	textract:GetExpenseAnalysis
	textract:GetLendingAnalysis
	textract:GetLendingAnalysisSummary
	textract:ListAdapters
	textract:ListAdapterVersions
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

Prefijo de servicio	Acciones
	textract:UpdateAdapter
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

Prefijo de servicio	Acciones
tnb	tnb:CancelSolNetworkOperation
	tnb:CreateSolFunctionPackage
	tnb:CreateSolNetworkInstance
	tnb:CreateSolNetworkPackage
	tnb:DeleteSolFunctionPackage
	tnb:DeleteSolNetworkInstance
	tnb:DeleteSolNetworkPackage
	tnb:GetSolFunctionInstance
	tnb:GetSolFunctionPackage
	tnb:GetSolFunctionPackageContent
	tnb:GetSolFunctionPackageDescriptor
	tnb:GetSolNetworkInstance
	tnb:GetSolNetworkOperation
	tnb:GetSolNetworkPackage
	tnb:GetSolNetworkPackageContent
	tnb:GetSolNetworkPackageDescriptor
	tnb:InstantiateSolNetworkInstance
	tnb:ListSolFunctionInstances
	tnb:ListSolFunctionPackages
	tnb:ListSolNetworkInstances
	tnb:ListSolNetworkOperations

Prefijo de servicio	Acciones
	tnb:ListSolNetworkPackages tnb:PutSolFunctionPackageContent tnb:PutSolNetworkPackageContent tnb:TerminateSolNetworkInstance tnb:UpdateSolFunctionPackage tnb:UpdateSolNetworkInstance tnb:UpdateSolNetworkPackage tnb:ValidateSolFunctionPackageContent tnb:ValidateSolNetworkPackageContent

Prefijo de servicio	Acciones
transcribe	transcribe:CreateCallAnalyticsCategory transcribe:CreateLanguageModel transcribe:CreateMedicalVocabulary transcribe:CreateVocabulary transcribe:CreateVocabularyFilter transcribe:DeleteCallAnalyticsCategory transcribe:DeleteCallAnalyticsJob transcribe:DeleteLanguageModel transcribe:DeleteMedicalTranscriptionJob transcribe:DeleteMedicalVocabulary transcribe:DeleteTranscriptionJob transcribe:DeleteVocabulary transcribe:DeleteVocabularyFilter transcribe:DescribeLanguageModel transcribe:GetCallAnalyticsCategory transcribe:GetCallAnalyticsJob transcribe:GetMedicalTranscriptionJob transcribe:GetMedicalVocabulary transcribe:GetTranscriptionJob transcribe:GetVocabulary transcribe:GetVocabularyFilter

Prefijo de servicio	Acciones
	transcribe:ListCallAnalyticsCategories transcribe:ListCallAnalyticsJobs transcribe:ListLanguageModels transcribe:ListMedicalTranscriptionJobs transcribe:ListMedicalVocabularies transcribe:ListTranscriptionJobs transcribe:ListVocabularies transcribe:ListVocabularyFilters transcribe:StartCallAnalyticsJob transcribe:StartMedicalTranscriptionJob transcribe:StartTranscriptionJob transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

Prefijo de servicio	Acciones
transfer	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWorkflow
	transfer:DescribeAccess
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector

Prefijo de servicio	Acciones
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListHostKeys
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWorkflows
	transfer:SendWorkflowStepState

Prefijo de servicio	Acciones
	transfer:StartFileTransfer
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile
	transfer:UpdateServer
	transfer:UpdateUser

Prefijo de servicio	Acciones
translate	translate:CreateParallelData translate:DeleteParallelData translate:DeleteTerminology translate:DescribeTextTranslationJob translate:GetParallelData translate:GetTerminology translate:ImportTerminology translate:ListLanguages translate:ListParallelData translate:ListTerminologies translate:ListTextTranslationJobs translate:StartTextTranslationJob translate:StopTextTranslationJob translate:TranslateDocument translate:TranslateText translate:UpdateParallelData

Prefijo de servicio	Acciones
voiceid	voiceid:AssociateFraudster
	voiceid:CreateDomain
	voiceid:CreateWatchlist
	voiceid:DeleteDomain
	voiceid:DeleteFraudster
	voiceid:DeleteSpeaker
	voiceid:DeleteWatchlist
	voiceid:DescribeDomain
	voiceid:DescribeFraudster
	voiceid:DescribeFraudsterRegistrationJob
	voiceid:DescribeSpeaker
	voiceid:DescribeSpeakerEnrollmentJob
	voiceid:DescribeWatchlist
	voiceid:DisassociateFraudster
	voiceid:EvaluateSession
	voiceid:ListDomains
	voiceid:ListFraudsterRegistrationJobs
	voiceid:ListFraudsters
	voiceid:ListSpeakerEnrollmentJobs
	voiceid:ListSpeakers
	voiceid:ListWatchlists

Prefijo de servicio	Acciones
	voiceid:OptOutSpeaker voiceid:StartFraudsterRegistrationJob voiceid:StartSpeakerEnrollmentJob voiceid:UpdateDomain voiceid:UpdateWatchlist

Prefijo de servicio	Acciones
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice:DeleteListener vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy

Prefijo de servicio	Acciones
	vpc-lattice:GetListener vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions vpc-lattice:ListListeners vpc-lattice:ListRules vpc-lattice:ListServiceNetworks vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice:UpdateAccessLogSubscription

Prefijo de servicio	Acciones
	vpc-lattice:UpdateListener vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

Prefijo de servicio	Acciones
wafv2	wafv2:AssociateWebACL
	wafv2:CheckCapacity
	wafv2:CreateAPIKey
	wafv2:CreateIPSet
	wafv2:CreateRegexPatternSet
	wafv2:CreateRuleGroup
	wafv2:CreateWebACL
	wafv2:DeleteFirewallManagerRuleGroups
	wafv2:DeleteIPSet
	wafv2:DeleteLoggingConfiguration
	wafv2:DeletePermissionPolicy
	wafv2:DeleteRegexPatternSet
	wafv2:DeleteRuleGroup
	wafv2:DeleteWebACL
	wafv2:DescribeAllManagedProducts
	wafv2:DescribeManagedProductsByVendor
	wafv2:DescribeManagedRuleGroup
	wafv2:DisassociateWebACL
	wafv2:GenerateMobileSdkReleaseUrl
	wafv2:GetDecryptedAPIKey
	wafv2:GetIPSet

Prefijo de servicio	Acciones
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetPermissionPolicy
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACL
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL
	wafv2:ListRuleGroups
	wafv2:ListWebACLs

Prefijo de servicio	Acciones
	wafv2:PutLoggingConfiguration wafv2:PutManagedRuleSetVersions wafv2:PutPermissionPolicy wafv2:UpdateIPSet wafv2:UpdateManagedRuleSetVersionExpiryDate wafv2:UpdateRegexPatternSet wafv2:UpdateRuleGroup wafv2:UpdateWebACL

Prefijo de servicio	Acciones
wellarchitected	wellarchitected:AssociateLenses wellarchitected:AssociateProfiles wellarchitected:CreateLensShare wellarchitected:CreateLensVersion wellarchitected:CreateMilestone wellarchitected:CreateProfile wellarchitected:CreateProfileShare wellarchitected:CreateReviewTemplate wellarchitected:CreateWorkload wellarchitected:CreateWorkloadShare wellarchitected:DeleteLens wellarchitected:DeleteLensShare wellarchitected:DeleteProfile wellarchitected:DeleteProfileShare wellarchitected:DeleteReviewTemplate wellarchitected:DeleteTemplateShare wellarchitected:DeleteWorkload wellarchitected:DeleteWorkloadShare wellarchitected:DisassociateLenses wellarchitected:DisassociateProfiles wellarchitected:ExportLens

Prefijo de servicio	Acciones
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport wellarchitected:GetLens wellarchitected:GetLensReview wellarchitected:GetLensReviewReport wellarchitected:GetLensVersionDifference wellarchitected:GetMilestone wellarchitected:GetProfile wellarchitected:GetProfileTemplate wellarchitected:GetReviewTemplate wellarchitected:GetReviewTemplateAnswer wellarchitected:GetReviewTemplateLensReview wellarchitected:GetWorkload wellarchitected:ImportLens wellarchitected:ListAnswers wellarchitected:ListCheckDetails wellarchitected:ListCheckSummaries wellarchitected:ListLenses wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

Prefijo de servicio	Acciones
	wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfiles wellarchitected:ListProfileShares wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloads wellarchitected:ListWorkloadShares wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview

Prefijo de servicio	Acciones
	wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

Prefijo de servicio	Acciones
wisdom	wisdom:CreateAssistant
	wisdom:CreateAssistantAssociation
	wisdom:CreateContent
	wisdom:CreateKnowledgeBase
	wisdom:CreateSession
	wisdom:DeleteAssistant
	wisdom:DeleteAssistantAssociation
	wisdom:DeleteContent
	wisdom:DeleteKnowledgeBase
	wisdom:GetAssistant
	wisdom:GetAssistantAssociation
	wisdom:GetContent
	wisdom:GetContentSummary
	wisdom:GetKnowledgeBase
	wisdom:GetRecommendations
	wisdom:GetSession
	wisdom:ListAssistantAssociations
	wisdom:ListAssistants
	wisdom:ListContents
	wisdom:ListKnowledgeBases
	wisdom:NotifyRecommendationsReceived

Prefijo de servicio	Acciones
	wisdom:QueryAssistant wisdom:RemoveKnowledgeBaseTemplateUri wisdom:SearchContent wisdom:SearchSessions wisdom:StartContentUpload wisdom:UpdateContent wisdom:UpdateKnowledgeBaseTemplateUri

Prefijo de servicio	Acciones
worklink	worklink:AssociateDomain worklink:AssociateWebsiteAuthorizationProvider worklink:AssociateWebsiteCertificateAuthority worklink:CreateFleet worklink:DeleteFleet worklink:DescribeAuditStreamConfiguration worklink:DescribeCompanyNetworkConfiguration worklink:DescribeDevice worklink:DescribeDevicePolicyConfiguration worklink:DescribeDomain worklink:DescribeFleetMetadata worklink:DescribeIdentityProviderConfiguration worklink:DescribeWebsiteCertificateAuthority worklink:DisassociateDomain worklink:DisassociateWebsiteAuthorizationProvider worklink:DisassociateWebsiteCertificateAuthority worklink:ListDevices worklink:ListDomains worklink:ListFleets worklink:ListWebsiteAuthorizationProviders worklink:ListWebsiteCertificateAuthorities

Prefijo de servicio	Acciones
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

Prefijo de servicio	Acciones
workspaces	workspaces:AssociateConnectionAlias
	workspaces:AssociateIpGroups
	workspaces:AssociateWorkspaceApplication
	workspaces:CopyWorkspaceImage
	workspaces:CreateConnectClientAddIn
	workspaces:CreateConnectionAlias
	workspaces:CreateIpGroup
	workspaces:CreateStandbyWorkspaces
	workspaces:CreateUpdatedWorkspaceImage
	workspaces:CreateWorkspaceBundle
	workspaces:CreateWorkspaceImage
	workspaces:CreateWorkspaces
	workspaces:DeleteClientBranding
	workspaces:DeleteConnectClientAddIn
	workspaces:DeleteConnectionAlias
	workspaces:DeleteIpGroup
	workspaces:DeleteWorkspaceBundle
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount

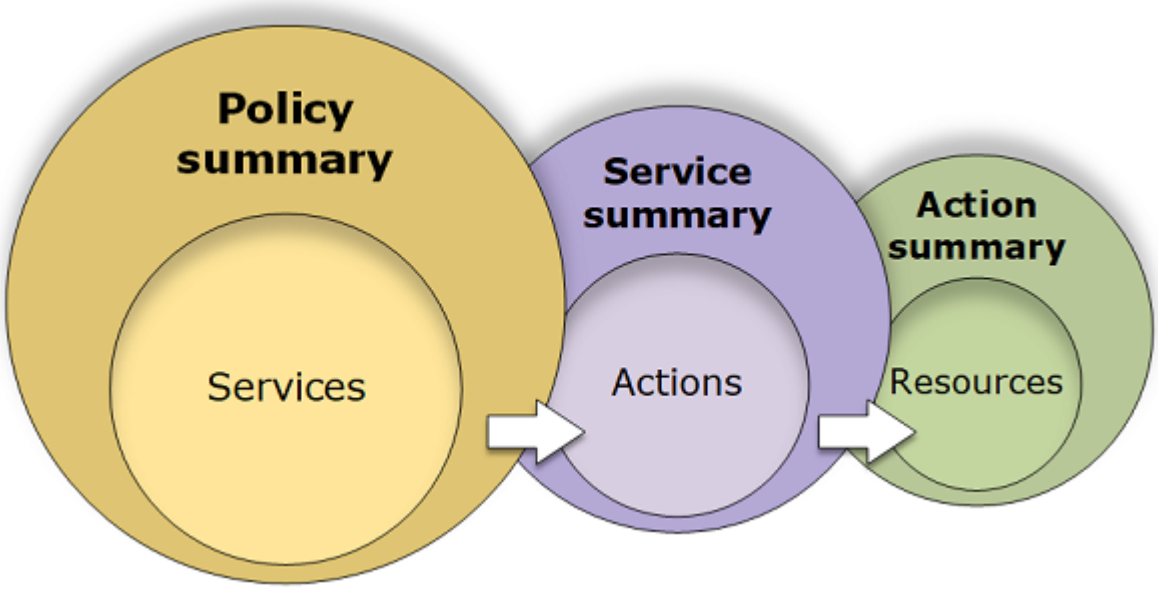
Prefijo de servicio	Acciones
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliases
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspaceSnapshots
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication

Prefijo de servicio	Acciones
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyWorkspaceAccessProperties
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StopWorkspaces
	workspaces:TerminateWorkspaces
	workspaces:UpdateConnectClientAddIn

Prefijo de servicio	Acciones
	workspaces:UpdateConnectionAliasPermission workspaces:UpdateWorkspaceBundle workspaces:UpdateWorkspaceImagePermission
xray	xray:CreateGroup xray:CreateSamplingRule xray:DeleteGroup xray:DeleteResourcePolicy xray:DeleteSamplingRule xray:GetEncryptionConfig xray:GetGroup xray:GetGroups xray:GetInsight xray:GetInsightEvents xray:GetInsightImpactGraph xray:GetInsightSummaries xray:GetSamplingRules xray:ListResourcePolicies xray:PutEncryptionConfig xray:PutResourcePolicy xray:UpdateGroup xray:UpdateSamplingRule

Permisos concedidos por una política

La consola de IAM incluye tablas de resumen de política que describen el nivel de acceso, los recursos y las condiciones permitidos o rechazados para cada servicio de una política. Las políticas se resumen en tres tablas: el resumen de política, el resumen de servicio y el resumen de acción. La tabla de resumen de política contiene una lista de servicios. Elija un servicio para ver el resumen de servicio. Esta tabla de resumen incluye una lista de las acciones y permisos asociados con el servicio elegido. Puede elegir una acción de dicha tabla para ver el resumen de acción. Esta tabla incluye una lista de recursos y condiciones para la acción que haya elegido.



Puede ver los resúmenes de las políticas en la página Users (Usuarios) o Roles de todas las políticas (administradas e insertadas) asociadas a dicho usuario. Puede ver los resúmenes de las políticas administradas en la página Policies (Políticas). Las políticas administradas incluyen políticas administradas por AWS, políticas de función administradas por AWS y las políticas administradas por el cliente. Puede ver resúmenes de estas políticas en la página Policies (Politicas), independientemente de si están asignadas a un usuario u otra identidad de IAM.

Puede utilizar la información de los resúmenes de política para comprender los permisos que autoriza o deniega la política. Los resúmenes de políticas pueden ayudarle a solucionar problemas y arreglar las políticas que no están proporcionando los permisos que espera.

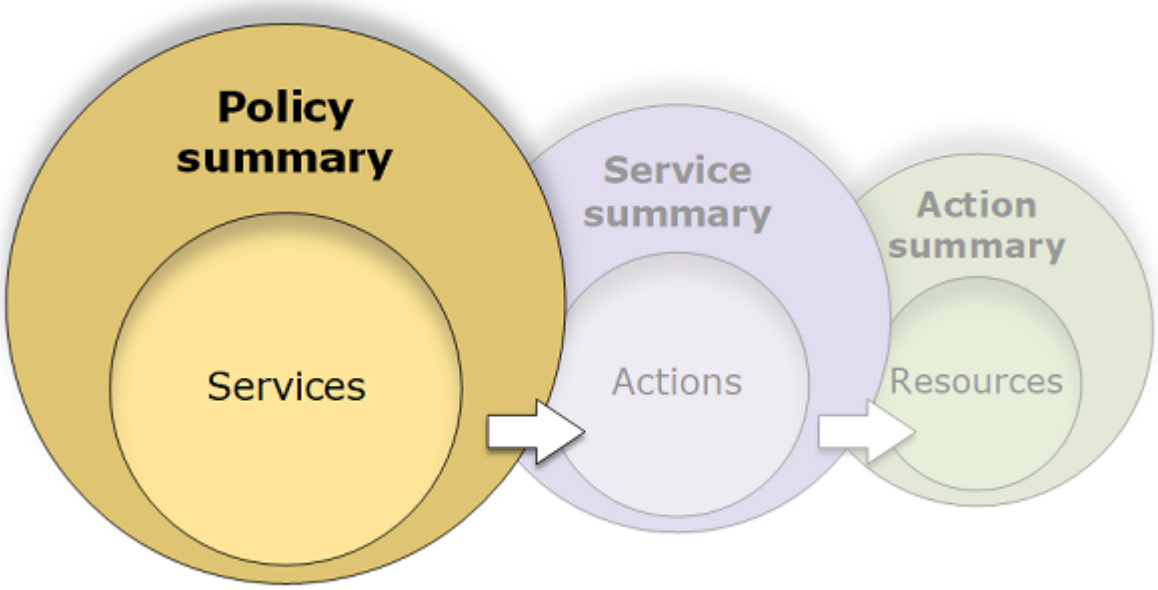
Temas

- Resumen de política (lista de servicios)
- Resumen de servicios (lista de acciones)

- Resumen de acción (lista de recursos)
- Ejemplos de resúmenes de políticas

Resumen de política (lista de servicios)

Las políticas se resumen en tres tablas: el resumen de política, el resumen de servicio y el resumen de acción. La tabla resumen de política incluye una lista de servicios y resúmenes de los permisos que la política elegida define.



La tabla de resumen de política se agrupa en una o varias secciones Uncategorized services (Servicios sin categorizar), Explicit deny (Denegar explícitamente) y Allow (Permitir). Si la política incluye un servicio que IAM no reconoce, el servicio se incluye en la sección Servicios sin categorizar de la tabla. Si IAM reconoce el servicio, este se incluye en las secciones Denegación explícita o Permitir de la tabla, en función del efecto de la política (`Deny` o `Allow`).

Visualización de los resúmenes de políticas

Puede ver los resúmenes de las políticas que se han asociado a un usuario seleccionando el nombre de cada política en la pestaña Permisos de la página de detalles del usuario. Puede ver los resúmenes de las políticas que se han asociado a un rol seleccionando el nombre de cada política en la pestaña Permisos de la página de detalles del rol. Puede ver el resumen de política de las políticas administradas en la página Policies (Políticas). Si la política no incluye ningún resumen de política, consulte Resumen de la política que falta para descubrir el motivo.

Para ver el resumen de política desde la página Policies (Políticas)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el nombre de la política que desea ver.
4. En la página Detalles de la política correspondiente a la política, consulte la pestaña Permisos para ver el resumen de la política.

Para ver el resumen de una política asociada a un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija la opción Users (Usuarios) en el panel de navegación.
3. En la lista de usuarios, seleccione el nombre del usuario cuya política desea ver.
4. En la página Summary (Resumen) del usuario, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas directamente al usuario o desde un grupo.
5. En la tabla de políticas del usuario, amplíe la fila de la política que desea ver.

Para ver el resumen de una política asociada a un rol

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Roles en el panel de navegación.
3. En la lista de roles, seleccione el nombre del rol cuya política desea ver.
4. En la página Summary (Resumen) del rol, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas al rol.
5. En la tabla de políticas del rol, amplíe la fila de la política que desea ver.

Edición de políticas para solucionar mensajes de advertencia

Mientras visualiza el resumen de una política, puede encontrar un error tipográfico o descubrir que la política no proporciona los permisos que esperaba. No se puede editar un resumen de política directamente. Sin embargo, puede editar una política administrada por el cliente con el editor visual de políticas, que detecta muchos de los mismos errores y advertencias que informa el resumen de

políticas. A continuación, puede ver los cambios en el resumen de política para confirmar que han resuelto todos los problemas. Para obtener información sobre cómo editar una política insertada, consulte the section called “Edición de políticas de IAM”. No se puede editar políticas administradas por AWS.

Para editar una política para el resumen de política mediante la opción Visual

1. Abra el resumen de política, tal y como se ha explicado en los últimos procedimientos.
2. Elija Editar.

   Si está en la página Users (Usuarios) y opta por editar una política administrada por el cliente asociada a dicho usuario, se le redirigirá a la página Policies (Políticas). Solo puede editar las políticas administradas por el cliente en la página Policies (Políticas).
3. Seleccione la opción Visual para ver la representación visual editable de la política. IAM podría reestructurar la política a fin de optimizarla para el editor visual y para facilitar la detección y corrección de problemas. Los mensajes de advertencia y error de la página le ayudarán a solucionar cualquier problema con la política. Para obtener más información sobre cómo IAM reestructura las políticas, consulte Reestructuración de políticas.
4. Edite la política y seleccione Siguiente para ver los cambios reflejados en el resumen de política. Si sigue viendo un problema, elija Previous (Anterior) para volver a la pantalla de edición.
5. Elija Guardar cambios para guardar los cambios.

Para editar una política para el resumen de política mediante la opción JSON

1. Abra el resumen de política, tal y como se ha explicado en los últimos procedimientos.
2. Puede utilizar los botones Resumen y JSON para comparar el resumen de política con el documento de política de JSON. Puede utilizar esta información para determinar qué líneas del documento de política desea cambiar.
3. Seleccione Editar y después la opción JSON para editar el documento de política de JSON.

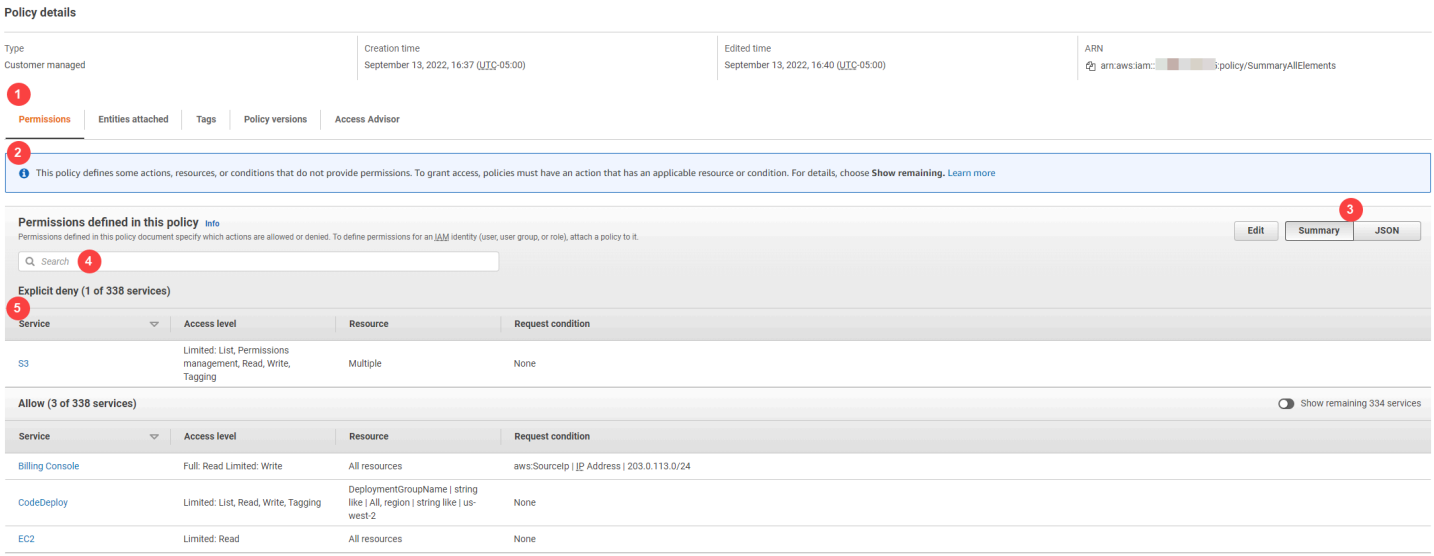
> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

Si está en la página Users (Usuarios) y opta por editar una política administrada por el cliente asociada a dicho usuario, se le redirigirá a la página Policies (Políticas). Solo puede editar las políticas administradas por el cliente en la página Policies (Políticas).

4. Edite su política. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Siguiente. Si sigue viendo un problema, elija Previous (Anterior) para volver a la pantalla de edición.
5. Elija Guardar cambios para guardar los cambios.

Descripción de los elementos de un resumen de política

En el siguiente ejemplo de página de detalles de una política, la política SummaryAllElements es una política administrada (política administrada por el cliente) que está asociada directamente al usuario. Esta política se ha ampliado para mostrar el resumen de política.

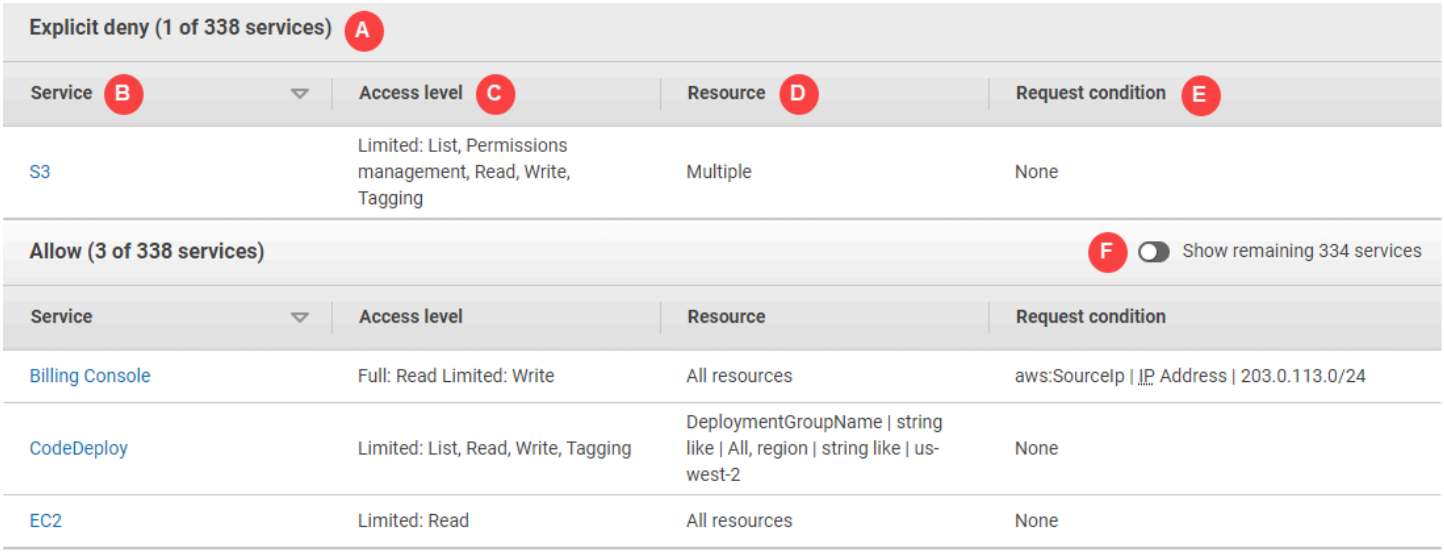


En la imagen anterior, el resumen de política es visible desde la página Políticas:

1. La pestaña Permisos incluye los permisos definidos en la política.
2. Si la política no concede permisos a todas las acciones, recursos y condiciones definidos por la política, aparece un banner de advertencia o error en la parte superior de la página. El resumen de política incluye información sobre el problema. Para obtener más información acerca de cómo los resúmenes de políticas pueden ayudarle a comprender y solucionar problemas de los permisos que concede su política, consulte the section called “Mi política no concede los permisos esperados”.

3. Utilice los botones Resumen y JSON para alternar entre el resumen de política y el documento de política de JSON.
4. Utilice el cuadro Buscar para reducir la lista de servicios y buscar un determinado servicio.
5. La vista ampliada muestra detalles adicionales de la política SummaryAllElements.

En la siguiente imagen de tabla de resumen de política se muestra la política SummaryAllElements ampliada en la página de detalles de la política.



En la imagen anterior, el resumen de política es visible desde la página Políticas:

A. En el caso de aquellos servicios que IAM reconoce, los servicios se organizan en función de si la política permite o deniega explícitamente el uso del servicio. En este ejemplo, la política incluye una instrucción `Deny` para el servicio Amazon S3, e instrucciones `Allow` para los servicios Facturación, CodeDeploy y Amazon EC2.

B. Servicio - Esta columna enumera los servicios definidos en la política y ofrece detalles de cada servicio. Cada nombre de servicio incluido en la tabla de resumen de política es un enlace a la tabla resumen de servicio, que se explica en Resumen de servicios (lista de acciones). En este ejemplo, se definen permisos para los servicios Amazon S3, Facturación, CodeDeploy y Amazon EC2.

C. Nivel de acceso: esta columna indica si las acciones de cada nivel de acceso (`List`, `Read`, `Write`, `Permission Management` y `Tagging`) tienen permisos `Full` o `Limited` definidos en la política. Para obtener información y ejemplos adicionales del resumen de nivel de acceso, consulte Descripción de los niveles de acceso en los resúmenes de políticas.

- Acceso total - Esta entrada indica que el servicio tiene acceso a todas las acciones de los cuatro niveles de acceso disponibles para el servicio.
- Si la entrada no incluye Full access (Acceso total), el servicio tiene acceso a algunas, pero no todas, acciones del servicio. El acceso viene definido por las siguientes descripciones de cada clasificación de nivel de acceso (`List`, `Read`, `Write`, `Permission Management` y `Tagging`):

  Full (Total): la política proporciona acceso a todas las acciones de cada clasificación de nivel de acceso indicada. En este ejemplo, la política proporciona acceso a todas las acciones `Read` del servicio Facturación.

  Limited (Limitado): la política proporciona acceso a una o varias, pero no todas, acciones de la clasificación de nivel de acceso indicada. En este ejemplo, la política proporciona acceso a algunas de las acciones `Write` del servicio Facturación.

D. Recurso - Esta columna muestra los recursos que la política especifica para cada servicio.

- Múltiples - La política incluye más de uno, pero no todos los recursos, del servicio. En este ejemplo, el acceso se deniega explícitamente a más de un recurso de Amazon S3.
- Todos los recursos: la política se define para todos los recursos del servicio. En este ejemplo, la política permite que las acciones indicadas se realicen en todos los recursos del servicio Facturación.
- Texto de recurso - La política incluye un recurso del servicio. En este ejemplo, las acciones indicadas solo están permitidas para el recurso `DeploymentGroupName` de CodeDeploy. En función de la información que el servicio proporcione a IAM, es posible que aparezca un ARN o bien el tipo de recurso definido.

> **Note**
>
> Esta columna puede incluir un recurso de otro servicio. Si la instrucción de la política que incluye el recurso no incluye ambas acciones y recursos del mismo servicio, el servicio incluirá recursos erróneos. IAM no le avisa de recursos no coincidentes al crear una política o al visualizar una política en el resumen de política. Si esta columna incluye un recurso no coincidente, debe comprobar que la política no tiene errores. Para comprender mejor las políticas, pruébelas siempre con el simulador de políticas.

E. Condición de solicitud - Esta columna indica si los servicios o acciones asociados al recurso están sujetos a las condiciones.

- Ninguna - La política no incluye ninguna condición para el servicio. En este ejemplo, no se aplica ninguna condición a las acciones denegadas en el servicio de Amazon S3.
- Texto de condición - La política incluye una condición para el servicio. En este ejemplo, las acciones del servicio Facturación indicadas solo están permitidas si la dirección IP del origen coincide con `203.0.113.0/24`.
- Múltiples - La política incluye más de una condición para el servicio. Para ver cada una de las diversas condiciones de la política, seleccione JSON con el fin de ver el documento de política.

F. Mostrar servicios restantes: utilice este botón para ampliar la tabla con el fin de incluir los servicios que no están definidos por la política. Estos servicios se deniegan implícitamente (o se deniegan de forma predeterminada) en esta política. Sin embargo, una instrucción de otra política podría seguir permitiendo o denegar explícitamente el uso del servicio. El resumen de política resume los permisos de una única política. Para obtener información sobre cómo el servicio de AWS decide si se permite o deniega una determinada solicitud, consulte Lógica de evaluación de políticas.

Cuando una política o un elemento dentro de la política no concede permisos, IAM, proporciona información y advertencias adicionales en la política de resumen. En la siguiente tabla de resumen de política se muestran los servicios de Mostrar servicios restantes ampliados en la página de detalles de la política SummaryAllElements junto con las posibles advertencias.

Explicit deny (1 of 338 services)

Service	Access level	Resource a	Request condition b
S3	Limited: List, Permissions management, Read, Write, Tagging	c Multiple| ⚠ One or more actions do not have an applicable resource.	None

Allow (3 of 338 services) Show remaining 334 services

Service	Access level	Resource	Request condition
Billing Console	Full: Read Limited: Write	All resources	aws:SourceIp | IP Address | 203.0.113.0/24
CodeCommit	None	d ⚠ No resources are defined.	None
CodeDeploy	Limited: List, Read, Write, Tagging	e DeploymentGroupName | string like | All, region | string like | us-west-2| ⚠ One or more actions do not have an applicable resource.	None
EC2	Limited: Read	All resources	None
S3	None	None| ⚠ One or more actions do not have an applicable resource.	f None | ⚠ One or more conditions do not have an applicable action.

En la imagen anterior, puede ver todos los servicios que incluyen acciones, recursos o condiciones definidos sin permisos:

a. Advertencia del recurso - En el caso de servicios que no conceden permisos para todas las acciones o recursos incluidos, verá una de las siguientes advertencias en la columna Recurso de la tabla:

   - 

     No resources are defined. (No hay recursos definidos.) - Esto significa que el servicio ha definido acciones, pero no se incluyen recursos admitidos en la política.

   - 

     One or more actions do not have an applicable resource. (Una o más acciones no tienen un recurso aplicable.) - Esto significa que el servicio ha definido acciones, pero que algunas de ellas no incluyen un recurso admitido.

   - 

     One or more resources do not have an applicable action. (Uno o más recursos no tienen una acción aplicable.) - Esto significa que el servicio ha definido recursos, pero que algunos de ellos no incluyen una acción admitida.

   Si un servicio incluye tanto acciones que no tienen un recurso aplicable como recursos que sí tienen un recurso aplicable, solo aparece la advertencia Uno o varios recursos no tienen una acción aplicable. Esto se debe a que el resumen de servicio del servicio en cuestión no muestra los recursos que no son aplicables a ninguna acción. En el caso de la acción `ListAllMyBuckets`, esta política incluye la última advertencia porque la acción no admite permisos en el nivel de recursos ni la clave de condición `s3:x-amz-acl`. Si soluciona el problema de recursos o el de condición, el problema que quede pendiente aparece en una advertencia detallada.

b. Advertencias de la condición de solicitud - En el caso de servicios que no conceden permisos para todas las condiciones incluidas, verá una de las siguientes advertencias en la columna Condición de la solicitud de la tabla:

   - 

     One or more actions do not have an applicable condition. (Una o más acciones no tienen una condición aplicable.) - Esto significa que el servicio ha definido acciones, pero que algunas de ellas no incluyen una condición admitida.

- 

  One or more conditions do not have an applicable action. (Una o más condiciones no tienen una acción aplicable.) - Esto significa que el servicio ha definido condiciones, pero que algunas de ellas no incluyen una acción admitida.

c. Multiple (Múltiples) |

One or more actions do not have an applicable resource. (Una o más acciones no tienen un recurso aplicable.) - La instrucción `Deny` para Amazon S3 incluye más de un recurso. También incluye más de una acción, de las cuales algunas admiten los recursos y otras no. Para consultar esta política, visite the section called “Documento de política JSON SummaryAllElements”. En este caso, la política incluye todas las acciones de Amazon S3 y se deniegan solo las acciones que pueden realizarse en un bucket o en un objeto de bucket.

d. 

No resources are defined (No hay recursos definidos) - El servicio incluye acciones definidas, pero la política no incluye recursos admitidos. Por lo tanto, el servicio no concede permisos. En este caso, la política incluye acciones de CodeCommit pero ningún recurso de CodeCommit.

e. DeploymentGroupName | cadena como | Todas, región | cadena como | us-west-2 |

Una o varias acciones no tienen un recurso aplicable. - El servicio tiene una acción definida y al menos una acción más que no tiene recurso de apoyo.

f. Ninguna |

Una o varias condiciones no tienen una acción aplicable. - El servicio tiene al menos una clave de condición que no tiene acción de apoyo.

Documento de política JSON SummaryAllElements

La política SummaryAllElements no ha sido creada para que defina permisos en su cuenta. En su lugar, se incluyen para demostrar los errores y las advertencias que podría encontrar al consultar un resumen de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "billing:Get*",
                "payments:List*",
                "payments:Update*",
                "account:Get*",
                "account:List*",
                "cur:GetUsage*"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::customer",
                "arn:aws:s3:::customer/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshots"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codedploy:*",
                "codecommit:*"
            ],
```

```
            "Resource": [
                "arn:aws:codedeploy:us-west-2:123456789012:deploymentgroup:*",
                "arn:aws:codebuild:us-east-1:123456789012:project/my-demo-project"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetObject",
                "s3:DeletObject",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::developer_bucket",
                "arn:aws:s3:::developer_bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Descripción de los niveles de acceso en los resúmenes de políticas

Resumen de nivel de acceso de AWS

Los resúmenes de políticas son resúmenes de niveles de acceso que describen los permisos de acciones definidos en cada servicio mencionado en la política en cuestión. Para obtener más información acerca de los resúmenes de políticas, consulte Permisos concedidos por una política. Los resúmenes de niveles de acceso indican si las acciones en cada nivel de acceso (`List`, `Read`, `Tagging`, `Write` y `Permissions management`) tienen permisos `Full` o `Limited` definidos en

la política. Para ver la clasificación de nivel de acceso que se asigna a cada acción de un servicio, consulte Acciones, Recursos y Claves de condición para servicios de AWS.

En el siguiente ejemplo se describe el acceso proporcionado por una política a unos servicios determinados. Para ver ejemplos de documentos de política JSON completos y los resúmenes de políticas relacionados, consulte Ejemplos de resúmenes de políticas.

Servicio	Nivel de acceso	Esta política proporciona lo siguiente
IAM	Acceso completo de	Acceso a todas las acciones dentro del servicio de IAM.
CloudWatch	Full (Completo): List (Enumerar)	Acceso a todas las acciones de CloudWatch en el nivel de acceso `List`, pero sin acceso a las acciones con la clasificación de nivel de acceso `Read`, `Write` o `Permissions management` .
Data Pipeline	Limited (Limitado): List, (Enumerar), Read (Lectura)	Acceso a al menos una pero no todas las acciones de AWS Data Pipeline con el nivel de acceso `List` y `Read`, pero sin acceso a las acciones `Write` o `Permissions management`
EC2	Full (Completo): List (Enumerar), Read (Lectura)Limited (Limitado): Write (Escritura)	Acceso a todas las acciones Amazon EC2 y `List` de `Read` y acceso a al menos una pero no a todas las acciones `Write` de Amazon EC2, pero ningún acceso a acciones con la clasificación de nivel de acceso `Permissions management` .
S3	Limited (Limitado): Read (Lectura), Write (Escritura), Permissions management (Administración de permisos)	Acceso a al menos una pero no a todas las acciones de Amazon S3 `Read`, `Write` y `Permissions management` .
CodeDeploy	(empty)	Acceso desconocido, porque IAM no reconoce este servicio.

Servicio	Nivel de acceso	Esta política proporciona lo siguiente
API Gateway	Ninguna	No hay accesos definidos en la política.
CodeBuild	 No hay acciones definidas.	No hay acceso porque no se definen las acciones para el servicio. Para obtener información sobre cómo comprender y resolver este problema, consulte the section called “Mi política no concede los permisos esperados”.

Tal como se ha mencionado anteriormente, el Full access (Acceso completo) indica que la política proporciona acceso a todas las acciones del servicio. Las políticas que proporcionan acceso a algunas pero no a todas las acciones de un servicio se agrupan en función de la clasificación del nivel de acceso. Esto se indica mediante una de las siguientes agrupaciones de nivel de acceso:

- Full (Completo): la política proporciona acceso a todas las acciones de la clasificación de nivel de acceso especificada.
- Limited (Limitado): la política proporciona acceso a una o varias, pero no todas, las acciones de la clasificación del nivel de acceso especificado.
- None (Ninguno): la política no proporciona ningún tipo de acceso.
- (vacío): IAM no reconoce este servicio. Si el nombre del servicio incluye un error tipográfico, entonces la política no proporcionará acceso al servicio. Si el nombre del servicio es correcto, el servicio podría no admitir resúmenes de políticas o podría estar en vista previa. En este caso, la política podría proporcionar acceso, pero dicho acceso no puede mostrarse en el resumen de la política. Para solicitar soporte de resumen de políticas para un servicio de disponibilidad general, consulte El servicio no admite resúmenes de políticas de IAM.

Los resúmenes de niveles de acceso que incluyen acceso (parcial) a acciones se agrupan utilizando las clasificaciones de nivel de acceso `List`, `Read`, `Tagging`, `Write` o `Permissions management` de AWS.

Niveles de acceso de AWS

AWS define las siguientes clasificaciones de nivel de acceso para las acciones en un servicio:

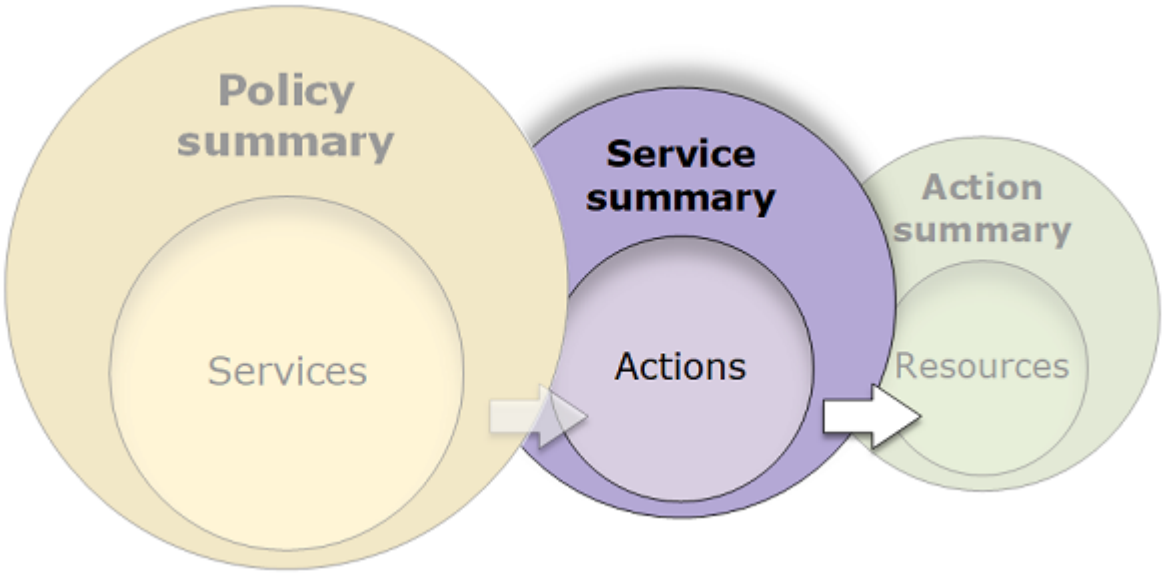
- List (Enumerar): permiso para enumerar los recursos dentro del servicio para determinar si existe un objeto. Las acciones con este nivel de acceso pueden enumerar objetos pero no pueden ver el contenido de un recurso. Por ejemplo, la acción de Amazon S3 `ListBucket` tiene el nivel de acceso Lista.
- Read (Lectura): permiso para leer, pero no editar, el contenido y los atributos de los recursos del servicio. Por ejemplo, las acciones de Amazon S3 `GetObject` y `GetBucketLocation` tienen el nivel de acceso Lectura.
- Etiquetado: permiso para realizar acciones que solo cambian el estado de etiquetas de recursos. Por ejemplo, las acciones de IAM `TagRole` y `UntagRole` tienen el nivel de acceso Etiquetado porque solo permiten etiquetar o quitar etiquetas de un rol. Sin embargo, la acción `CreateRole` permite etiquetar los recursos del rol al crear ese rol. Dado que la acción no solo añade una etiqueta, tiene el nivel de acceso `Write`.
- Write (Escritura): permiso para crear, eliminar o modificar los recursos del servicio. Por ejemplo, las acciones de Amazon S3 `CreateBucket`, `DeleteBucket` y `PutObject` tienen nivel de acceso Escritura. Las acciones `Write` también podrían permitir modificar una etiqueta de recurso. Sin embargo, una acción que solo permite cambios a etiquetas tiene el nivel de acceso `Tagging`.
- Permissions management (Administración de permisos): permiso para conceder o modificar permisos de recursos del servicio. Por ejemplo, la mayoría de las acciones de IAM y AWS Organizations, además de las acciones del tipo `PutBucketPolicy` y `DeleteBucketPolicy` de Amazon S3 tienen el nivel de acceso Administración de permisos.

> **Sugerencia**
>
> Para mejorar la seguridad de su cuenta de Cuenta de AWS, limite o monitoree periódicamente las políticas que incluyen la clasificación de nivel de acceso Permissions management (Administración de permisos).

Para ver la clasificación de nivel de acceso para todas las acciones de un servicio, consulte Acciones, Recursos y Claves de condición para servicios de AWS.

Resumen de servicios (lista de acciones)

Las políticas se resumen en tres tablas: el resumen de política, el resumen de servicio y el resumen de acción. La tabla resumen de servicio incluye una lista de acciones y resúmenes de los permisos definidos por la política del servicio elegido.



Puede ver un resumen de cada servicio enumerado en el resumen de política que concede los permisos. La tabla se agrupa en las secciones Uncategorized actions (Acciones sin categorizar), Uncategorized resource types (Tipos de recursos sin categorizar) y nivel de acceso. Si la política incluye una acción que IAM no reconoce, entonces la acción se incluye en la sección Acciones no categorizadas de la tabla. Si IAM reconoce la acción, entonces se incluye en una de las secciones de nivel de acceso (Enumerar, Leer, Escribir y Administración de permisos) de la tabla. Para ver la clasificación de nivel de acceso que se asigna a cada acción de un servicio, consulte Acciones, Recursos y Claves de condición para servicios de AWS.

Visualización de resúmenes de servicio

Puede ver el resumen de servicio de las políticas administradas en la página Políticas.

Para ver el resumen de servicio de una política administrada

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el nombre de la política que desea ver.
4. En la página Detalles de la política correspondiente a la política, consulte la pestaña Permisos para ver el resumen de la política.
5. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.

Para ver el resumen de servicio de una política asociada a un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Usuarios.
3. En la lista de usuarios, seleccione el nombre del usuario cuya política desea ver.
4. En la página Summary (Resumen) del usuario, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas directamente al usuario o desde un grupo.
5. En la tabla de políticas del usuario, elija el nombre de la política que desea ver.

   Si está en la página Usuarios y decide ver el resumen de servicio correspondiente a una política que está asociada a ese usuario, se le redirigirá a la página Políticas. Solo se pueden ver resúmenes de servicios en la página Políticas.
6. Seleccione Resumen. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.

> **Note**
>
> Si la política que selecciona es una política insertada que está asociada directamente con el usuario, aparecerá la tabla de resumen de servicio. Si la política es una política insertada asociada a un grupo, entonces accederá al documento de política JSON de ese grupo. Si la política es una política administrada, se le redirigirá al resumen de servicio de dicha política en la página Policies (Políticas).

Para ver el resumen de servicio de una política asociada a un rol

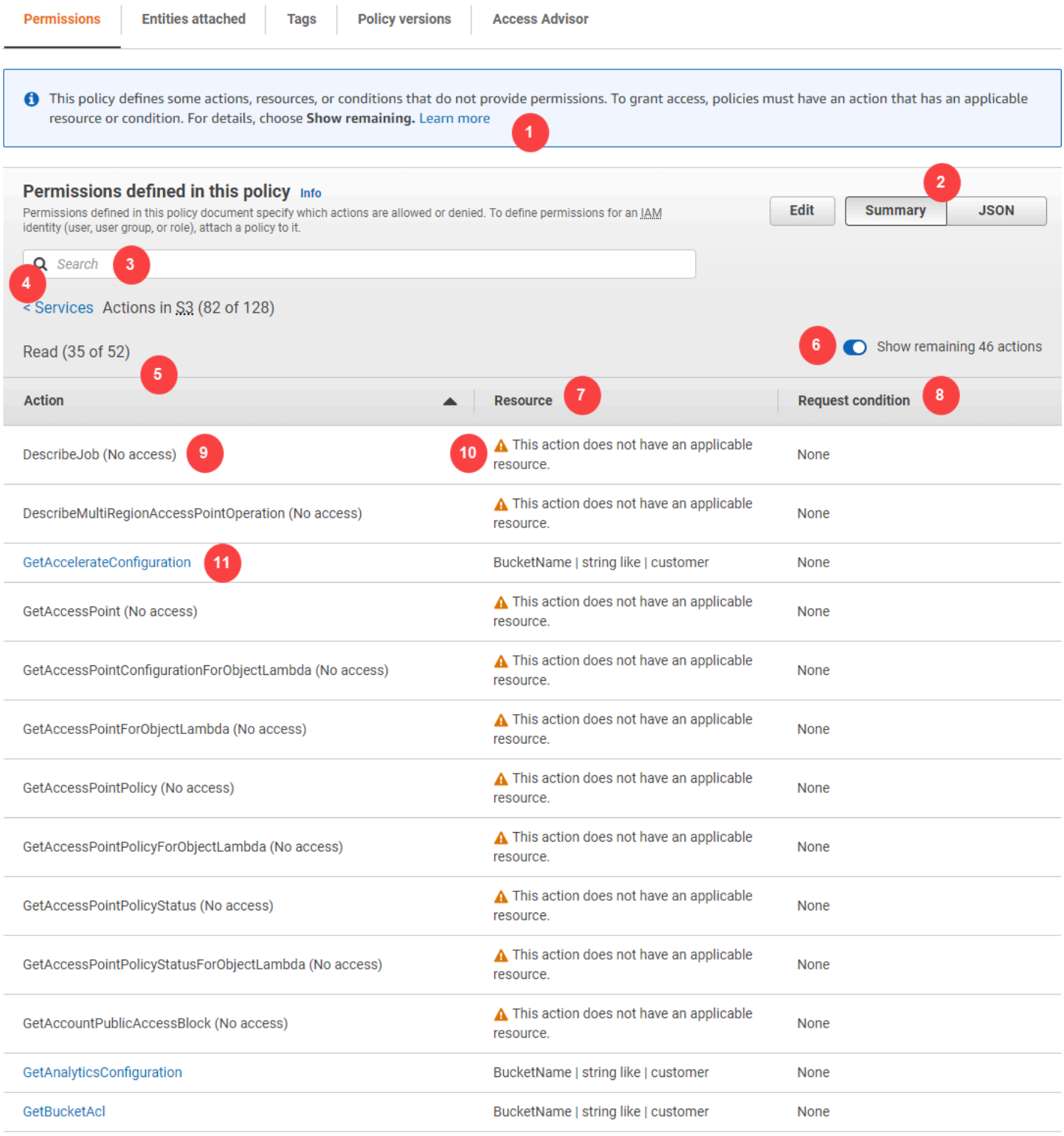
1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija la opción Roles en el panel de navegación.
3. En la lista de roles, seleccione el nombre del rol cuya política desea ver.
4. En la página Summary (Resumen) del rol, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas al rol.
5. En la tabla de políticas del rol, seleccione el nombre de la política que desee ver.

Si está en la página Roles y decide ver el resumen de servicio correspondiente a una política que está asociada a ese usuario, se le redirigirá a la página Políticas. Solo se pueden ver resúmenes de servicios en la página Políticas.

6. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.

Descripción de los elementos de un resumen de servicio

El ejemplo siguiente es el resumen de servicio correspondiente a acciones de Amazon S3 que están permitidas desde un resumen de política. Las acciones de este servicio se agrupan por nivel de acceso. Por ejemplo, se definen 35 acciones Leer de un total de 52 acciones Leer disponibles para el servicio.



La página de resumen de servicio de una política administrada incluye la siguiente información:

1. Si la política no concede permisos a todas las acciones, recursos y condiciones definidos por el servicio de la política, aparece un banner de advertencia en la parte superior de la página. El

resumen de servicio incluye información sobre el problema. Para obtener más información acerca de cómo los resúmenes de políticas pueden ayudarle a comprender y solucionar problemas de los permisos que concede su política, consulte the section called “Mi política no concede los permisos esperados”.

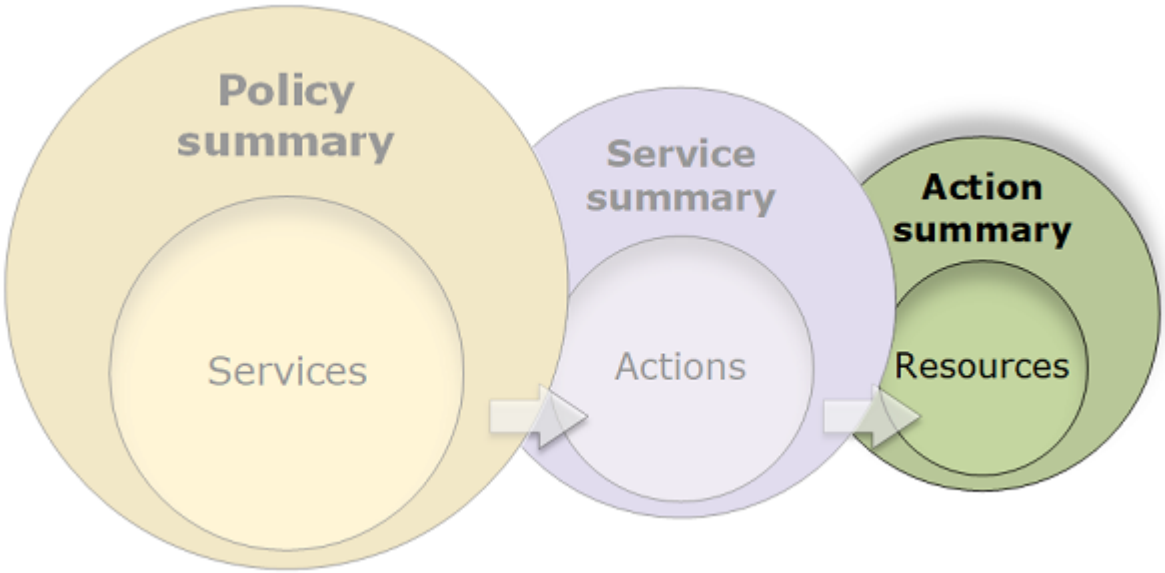
2. Seleccione JSON para ver detalles adicionales sobre la política. Puede hacerlo para ver todas las condiciones que se aplican a las acciones. (Si visualiza el resumen de servicio de una política insertada asociada directamente a un usuario, debe cerrar el cuadro de diálogo del resumen de servicio y volver al resumen de política para acceder al documento de la política JSON).
3. Para ver el resumen de una acción específica, escriba palabras clave en el cuadro Buscar, con el fin de reducir la lista de acciones disponibles.
4. Junto a la flecha de retroceso Servicios aparece el nombre del servicio (en este caso, S3). El resumen de servicio correspondiente a este servicio incluye la lista de acciones permitidas o denegadas que están definidas en la política. Si el servicio aparece en (Denegación explícita) en la pestaña Permisos, se deniegan explícitamente las acciones que figuran en la tabla de resumen de servicio. Si el servicio aparece en Permitir en la pestaña Permisos, se permiten las acciones que figuran en la tabla de resumen de servicio.
5. Acción: esta columna muestra las acciones que están definidas en la política y proporciona los recursos y condiciones para cada acción. Si la política concede o deniega permisos para la acción, el nombre de la acción enlaza a la tabla de resumen de acción. Esta tabla agrupa estas acciones en al menos una sección y hasta un máximo de cinco, en función del nivel de acceso que la política permita o deniegue. Las secciones son Enumerar, Leer, Escribir, Administración de permisos y Etiquetado. El recuento indica el número de acciones reconocidas que proporcionan permisos en cada nivel de acceso. El total es el número de acciones conocidas para el servicio. En este ejemplo, 35 acciones proporcionan permisos de un total de 52 acciones Leer de Amazon S3 conocidas. Para ver la clasificación de nivel de acceso que se asigna a cada acción de un servicio, consulte Acciones, Recursos y Claves de condición para servicios de AWS.
6. Mostrar acciones restantes: utilice este botón para ampliar u ocultar la tabla con el fin de incluir las acciones conocidas pero que no proporcionan permisos para este servicio. Al utilizar este botón también se muestran advertencias para los elementos que no proporcionan permisos.
7. Recurso - Esta columna muestra los recursos que la política define para el servicio. IAM no comprueba si el recurso se aplica a cada acción. En este ejemplo, las acciones del servicio Amazon S3 solo están permitidas en el recurso de bucket de Amazon S3 `developer_bucket`. En función de la información que el servicio proporcione a IAM, se puede mostrar un ARN, como `arn:aws:s3:::developer_bucket/*`, o el tipo de recurso definido, como `BucketName = developer_bucket`.

> **Note**
>
> Esta columna puede incluir un recurso de otro servicio. Si la instrucción de la política que incluye el recurso no incluye ambas acciones y recursos del mismo servicio, el servicio incluirá recursos erróneos. IAM no le avisa sobre recursos erróneos al crear una política o al visualizar una política en el resumen de servicio. IAM tampoco indica si la acción se aplica a los recursos, solo si coincide con el servicio. Si esta columna incluye un recurso no coincidente, debe comprobar que la política no tiene errores. Para comprender mejor las políticas, pruébelas siempre con el simulador de políticas.

8. Condición de solicitud - Esta columna muestra si las acciones asociadas al recurso están sujetas a las condiciones. Para obtener más información sobre estas condiciones, seleccione JSON para revisar el documento de política de JSON.
9. Sin acceso - Esta política incluye una acción que no proporciona permisos.
10. Advertencia de recursos - En el caso de acciones con recursos que no conceden permisos completos, verá una de las siguientes advertencias:
    - This action does not support resource-level permissions. (Esta acción no admite permisos por recursos). This requires a wildcard (*) for the resource. (Se necesita un comodín [*] para el recurso). - Esto significa que la política incluye permisos en el nivel de recursos, pero debe incluir `"Resource": ["*"]` para conceder permisos para esta acción.
    - This action does not have an applicable resource. (Esta acción no tiene un recurso aplicable.) - Esto significa que la acción está incluida en la política, pero sin un recurso admitido.
    - This action does not have an applicable resource and condition. (Esta acción no tienen un recurso y una condición aplicables.) - Esto significa que la acción está incluida en la política, pero sin un recurso ni una condición admitidos. En este caso, también existe una condición incluida en la política para este servicio, pero no hay condiciones aplicables a esta acción.
11. Entre las acciones que conceden los permisos se incluye un enlace al resumen de la acción.

Resumen de acción (lista de recursos)

Las políticas se resumen en tres tablas: el resumen de política, el resumen de servicio y el resumen de acción. La tabla de resumen de acción incluye una lista de recursos y las condiciones asociadas que son aplicables a la acción que elija.



Para ver un resumen de acción de cada acción que concede permisos, elija el enlace en el resumen del servicio. La tabla de resumen de acción incluye detalles sobre el recurso, incluidas las opciones Region (Región) y Account (Cuenta). También puede ver las condiciones que se aplican a cada recurso. Esto permite ver las condiciones que se aplican a algunos recursos, pero no a otros.

Visualización de resúmenes de acciones

Puede ver el resumen de acciones de las políticas administradas, cualquier política que esté asociada a un usuario y cualquier política que esté asociada a un rol en la página Políticas.

Para ver el resumen de acción de una política administrada

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el nombre de la política que desea ver.
4. En la página Detalles de la política correspondiente a la política, consulte la pestaña Permisos para ver el resumen de la política.
5. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.
6. En la lista de acciones del resumen de servicio, elija el nombre de la acción que desea ver.

Para ver el resumen de acción de una política asociada a un usuario

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija la opción Users (Usuarios) en el panel de navegación.
3. En la lista de usuarios, seleccione el nombre del usuario cuya política desea ver.
4. En la página Summary (Resumen) del usuario, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas directamente al usuario o desde un grupo.
5. En la tabla de políticas del usuario, elija el nombre de la política que desea ver.

   Si está en la página Usuarios y decide ver el resumen de servicio correspondiente a una política que está asociada a ese usuario, se le redirigirá a la página Políticas. Solo se pueden ver resúmenes de servicios en la página Políticas.
6. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.

   > **Note**
   >
   > Si la política que selecciona es una política insertada que está asociada directamente con el usuario, aparecerá la tabla de resumen de servicio. Si la política es una política insertada asociada a un grupo, entonces accederá al documento de política JSON de ese grupo. Si la política es una política administrada, se le redirigirá al resumen de servicio de dicha política en la página Policies (Políticas).
7. En la lista de acciones del resumen de servicio, elija el nombre de la acción que desea ver.

Para ver el resumen de acción de una política asociada a un rol

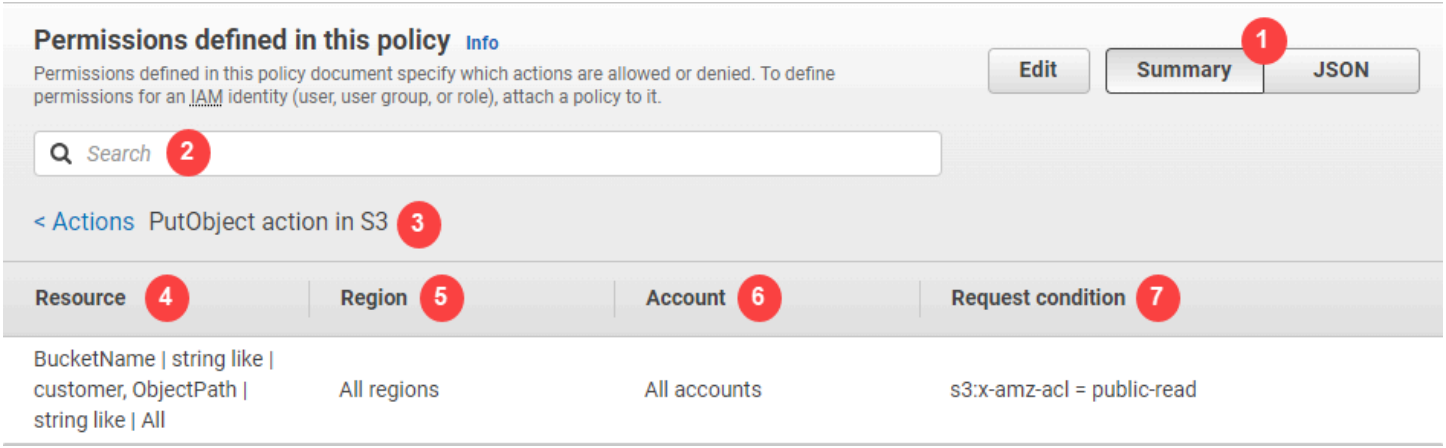
1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Roles en el panel de navegación.
3. En la lista de roles, seleccione el nombre del rol cuya política desea ver.
4. En la página Summary (Resumen) del rol, diríjase a la pestaña Permissions (Permisos) para ver la lista de políticas asociadas al rol.
5. En la tabla de políticas del rol, seleccione el nombre de la política que desee ver.

Si está en la página Roles y decide ver el resumen de servicio correspondiente a una política que está asociada a ese usuario, se le redirigirá a la página Políticas. Solo se pueden ver resúmenes de servicios en la página Políticas.

6. En la lista de servicios del resumen de política, elija el nombre del servicio que desea ver.
7. En la lista de acciones del resumen de servicio, elija el nombre de la acción que desea ver.

Descripción de los elementos de un resumen de acción

El ejemplo que aparece a continuación es el resumen de acción `PutObject` (escritura) del resumen de servicio de Amazon S3 (consulte Resumen de servicios (lista de acciones)). En esta acción, la política define varias condiciones en un único recurso.



La página de resumen de acción incluye la siguiente información:

1. Seleccione JSON para ver detalles adicionales sobre la política, tales como las diversas condiciones que se aplican a las acciones. (Si está viendo el resumen de la acción para una política insertada que se adjunta directamente a un usuario, los pasos son diferentes. Para acceder al documento de política JSON en ese caso, debe cerrar el cuadro de diálogo de resumen de acciones y volver al resumen de políticas).
2. Para ver el resumen correspondiente a un recurso específico, escriba palabras clave en el cuadro Buscar, con el fin de reducir la lista de recursos disponibles.
3. Junto a la flecha de retroceso Acciones aparece el nombre del servicio y la acción con el formato `action name action in service` (en este caso PutObject action in S3). El resumen de acción de este servicio incluye la lista de recursos que se define en la política.

4. Recurso - Esta columna enumera los recursos que la política define para el servicio que elija. En este ejemplo, la acción PutObject está permitida en todas las rutas de objetos, pero solo en el recurso de bucket de Amazon S3 `developer_bucket`. En función de la información que el servicio proporcione a IAM, se puede mostrar un ARN, como `arn:aws:s3:::developer_bucket/*`, o el tipo de recurso definido, como `BucketName = developer_bucket, ObjectPath = All`.
5. Región - Esta columna muestra la región en la que se define el recurso. Los recursos se pueden definir para todas las regiones o para una única región. No pueden existir en más de una región específica.
   - Todas las regiones: las acciones que están asociadas al recurso se aplican a todas las regiones. En este ejemplo, la acción pertenece a un servicio global, Amazon S3. Las acciones que pertenecen a los servicios globales se aplican a todas las regiones.
   - Texto de la región - Las acciones asociadas al recurso se aplican a una región. Por ejemplo, una política puede especificar la región `us-east-2` para un recurso.
6. Cuenta - Esta columna indica si los servicios o acciones asociados al recurso se aplican a una cuenta específica. Los recursos pueden existir en todas las cuentas o en una única cuenta. No pueden existir en más de una cuenta específica.
   - Todas las cuentas - Las acciones asociadas al recurso se aplican a todas las cuentas. En este ejemplo, la acción pertenece a un servicio global, Amazon S3. Las acciones que pertenecen a los servicios globales se aplican a todas las cuentas.
   - Esta cuenta: las acciones que están asociadas al recurso solo se aplican a la cuenta actual.
   - Número de cuenta - Las acciones asociadas al recurso se aplican a una cuenta (en la que no haya iniciado sesión actualmente). Por ejemplo, si una política especifica la cuenta `123456789012` para un recurso, el número de cuenta aparece en el resumen de política.
7. Condición de solicitud - Esta columna muestra si las acciones asociadas al recurso están sujetas a condiciones. En este ejemplo se incluye la condición `s3:x-amz-acl = public-read`. Para obtener más información sobre estas condiciones, seleccione JSON para revisar el documento de política de JSON.

Ejemplos de resúmenes de políticas

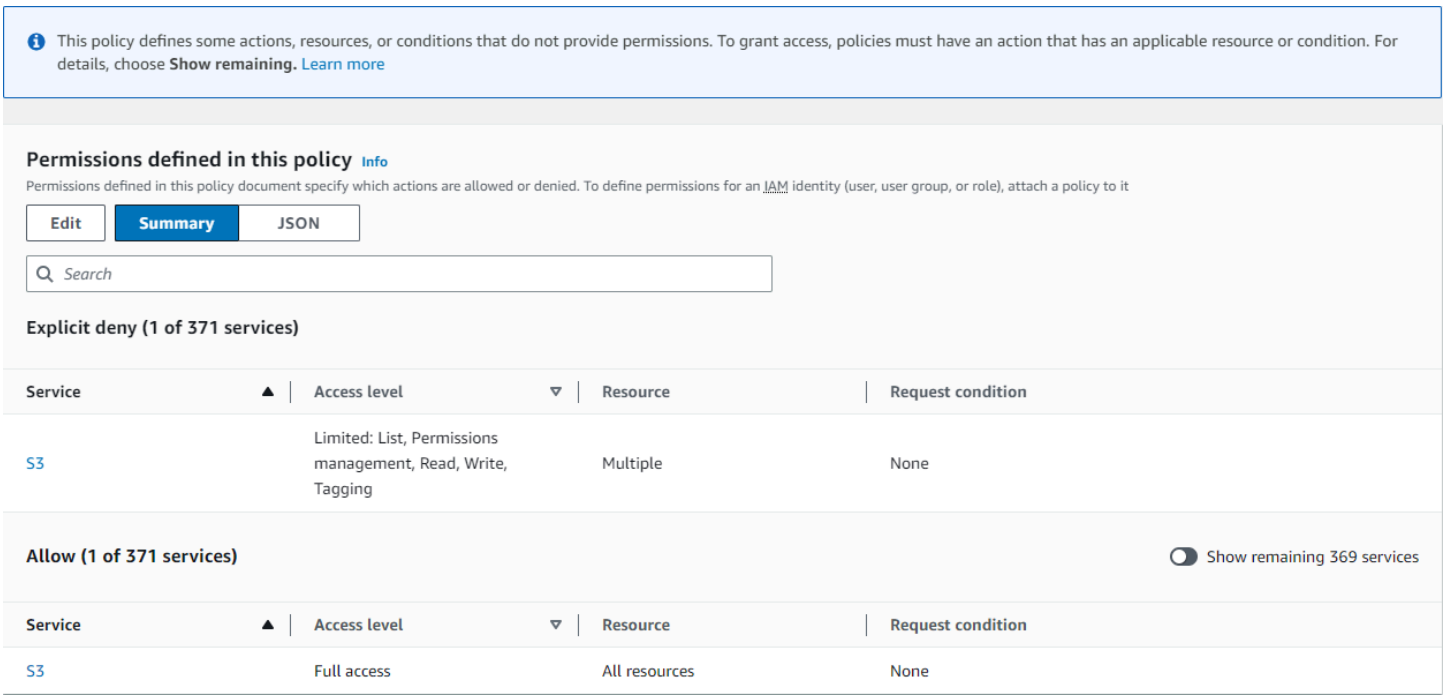
Los siguientes ejemplos muestran políticas JSON con sus resúmenes de política, resúmenes de servicio y resúmenes de acción asociados para ayudarle a comprender los permisos que se dan por medio de una política.

Política 1: DenyCustomerBucket

Esta política demuestra un permitir y un denegar para el mismo servicio.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccess",
            "Effect": "Allow",
            "Action": ["s3:*"],
            "Resource": ["*"]
        },
        {
            "Sid": "DenyCustomerBucket",
            "Action": ["s3:*"],
            "Effect": "Deny",
            "Resource": ["arn:aws:s3:::customer", "arn:aws:s3:::customer/*" ]
        }
    ]
}
```

Resumen de la política DenyCustomerBucket:



Resumen del servicio DenyCustomerBucket S3 (denegación explícita):

< Services Actions in S3 (82 of 130) Show remaining 48 actions

Read (35 of 53)

Action	Resource	Request condition
GetAccelerateConfiguration	BucketName| string like |customer	None
GetAnalyticsConfiguration	BucketName| string like |customer	None
GetBucketAcl	BucketName| string like |customer	None
GetBucketCORS	BucketName| string like |customer	None
GetBucketLocation	BucketName| string like |customer	None
GetBucketLogging	BucketName| string like |customer	None
GetBucketNotification	BucketName| string like |customer	None
GetBucketObjectLockConfiguration	BucketName| string like |customer	None
GetBucketOwnershipControls	BucketName| string like |customer	None
GetBucketPolicy	BucketName| string like |customer	None
GetBucketPolicyStatus	BucketName| string like |customer	None
GetBucketPublicAccessBlock	BucketName| string like |customer	None
GetBucketRequestPayment	BucketName| string like |customer	None
GetBucketTagging	BucketName| string like |customer	None
GetBucketVersioning	BucketName| string like |customer	None
GetBucketWebsite	BucketName| string like |customer	None

Resumen de la acción GetObject (lectura):

< Actions GetObject action in S3

Resource	Region	Account	Request condition
BucketName| string like |customer, ObjectPath| string like |All	-	All accounts	None

Política 2: DynamoDbRowCognitoID

Esta política proporciona acceso de nivel de filas a Amazon DynamoDB en función del ID de Amazon Cognito del usuario.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:DeleteItem",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-west-1:123456789012:table/myDynamoTable"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "dynamodb:LeadingKeys": [
                        "${cognito-identity.amazonaws.com:sub}"
                    ]
                }
            }
        }
    ]
}
```

Resumen de la política DynamoDbRowCognitoID:

Allow (1 of 370 services) — Show remaining 369 services

Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Resumen del servicio DynamoDbRowCognitoID DynamoDB (permitir):

< Services Actions in DynamoDB (4 of 65) Show remaining 61 actions

Read (1 of 26)

Action	Resource	Request condition
GetItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Write (3 of 33)

Action	Resource	Request condition
DeleteItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
PutItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}
UpdateItem	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

Resumen de la política GetItem (enumerar):

< Actions GetItem action in DynamoDB

Resource	Region	Account	Request condition
region| string like |us-west-1, TableName| string like |myDynamoTable	us-west-1	123456789012	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

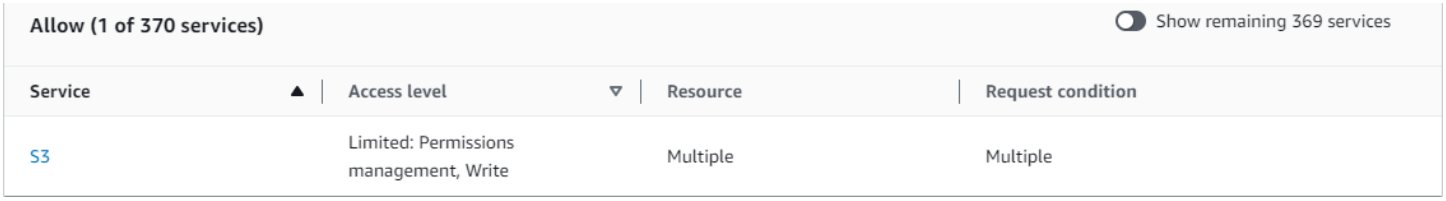
Política 3: MultipleResourceCondition

Esta política contiene varios recursos y condiciones.

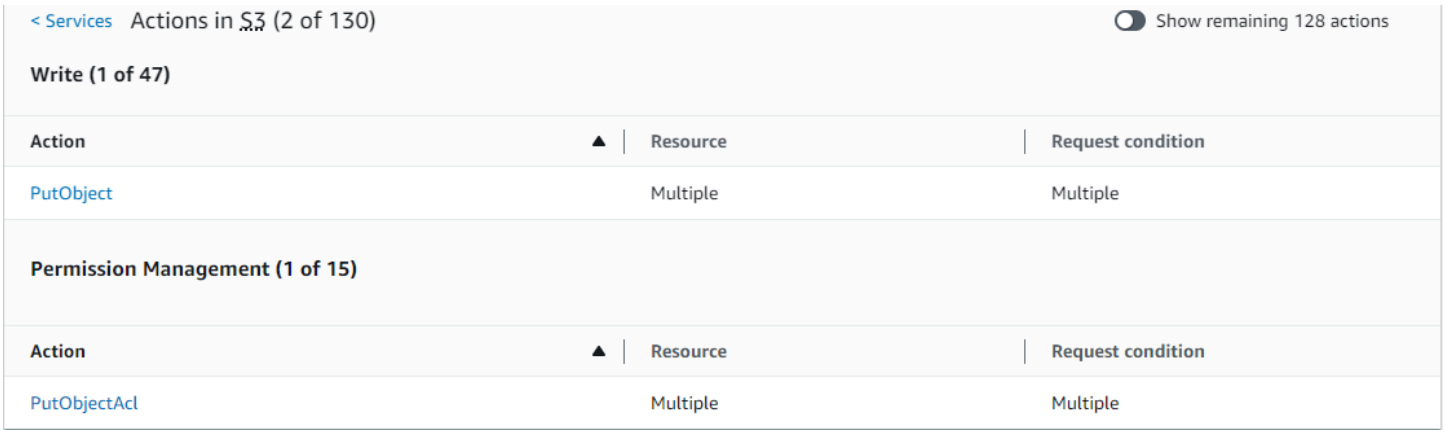
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["public-read"]}}
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Orange_bucket/*"],
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": ["custom"],
                "s3:x-amz-grant-full-control": ["1234"]
            }}
        }
    ]
}
```

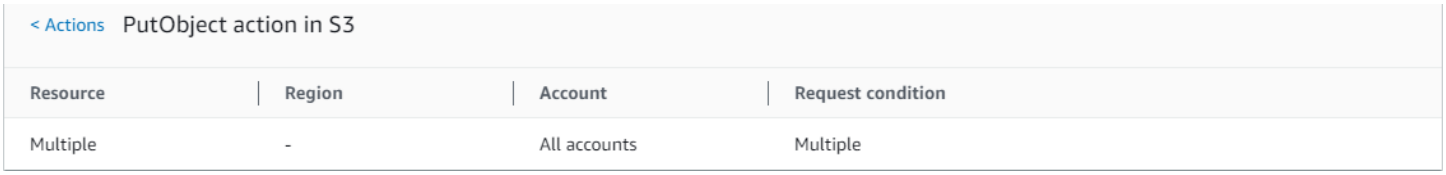
Resumen de la política MultipleResourceCondition:



Resumen del servicio MultipleResourceCondition S3 (permitir):



Resumen de la acción PutObject (escritura):

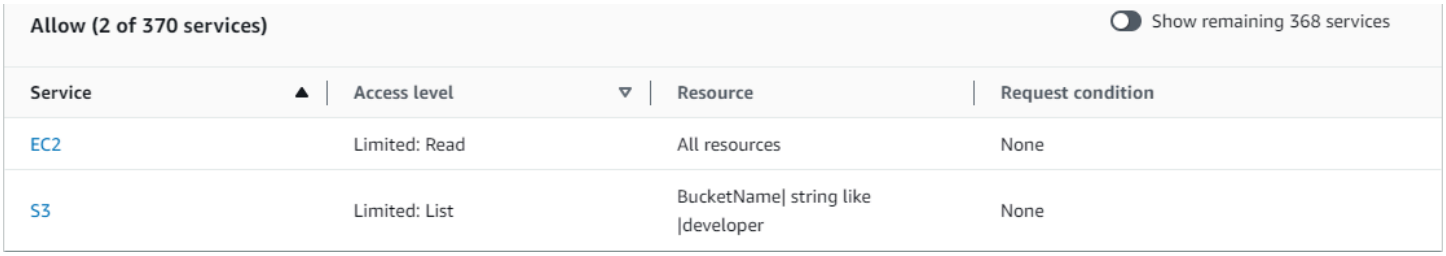


Política 4: EC2_troubleshoot

La siguiente política permite a los usuarios obtener una captura de pantalla de una instancia de Amazon EC2 en ejecución, lo que puede ayudarle a la resolución de problemas de EC2. Esta política también permite visualizar información sobre los elementos del bucket de desarrollador de Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

Resumen de la política EC2_Troubleshoot:



Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

Resumen de la política EC2_Troubleshoot S3 (permitir):

< Services Actions in S3 (1 of 130) Show remaining 129 actions

List (1 of 7)

Action	Resource	Request condition
ListBucket	BucketName| string like |developer	None

Resumen de la política ListBucket (enumerar):

< Actions ListBucket action in S3

Resource	Region	Account	Request condition
BucketName| string like |developer	-	All accounts	None

Política 5: CodeBuild_CodeCommit_CodeDeploy

Esta política proporciona acceso a recursos CodeBuild, CodeCommit y CodeDeploy específicos. Dado que estos recursos son específicos de cada servicio, aparecen únicamente con el servicio correspondiente. Si incluye un recurso que no coincide con ninguno de los servicios del elemento `Action`, el recurso aparece en todos los resúmenes de acción.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1487980617000",
            "Effect": "Allow",
            "Action": [
                "codebuild:*",
                "codecommit:*",
                "codedeploy:*"
            ],
            "Resource": [
                "arn:aws:codebuild:us-east-2:123456789012:project/my-demo-project",
                "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo",
                "arn:aws:codedeploy:us-east-2:123456789012:application:WordPress_App",
                "arn:aws:codedeploy:us-east-2:123456789012:instance/AssetTag*"
            ]
        }
    ]
}
```

Resumen de la política CodeBuild_CodeCommit_CodeDeploy:

Allow (3 of 370 services) Show remaining 367 services

Service	Access level	Resource	Request condition
CodeBuild	Full: Permissions management Limited: List, Read, Write	region| string like |us-east-2	None
CodeCommit	Full: Tagging Limited: List, Read, Write	ResourceSpecifier| string like |MyDemoRepo, region| string like |us-east-2	None
CodeDeploy	Full: Tagging Limited: List, Read, Write	Multiple	None

Resumen del servicio CodeBuild_CodeCommit_CodeDeploy CodeBuild (permitir):

< Services Actions in CodeBuild (24 of 53) Show remaining 29 actions

Read (4 of 9)

Action	Resource	Request condition
BatchGetBuildBatches	region| string like |us-east-2	None
BatchGetBuilds	region| string like |us-east-2	None
BatchGetProjects	region| string like |us-east-2	None
GetResourcePolicy	region| string like |us-east-2	None

Write (16 of 28)

Action	Resource	Request condition
BatchDeleteBuilds	region| string like |us-east-2	None
CreateProject	region| string like |us-east-2	None
CreateWebhook	region| string like |us-east-2	None
DeleteBuildBatch	region| string like |us-east-2	None
DeleteProject	region| string like |us-east-2	None
DeleteWebhook	region| string like |us-east-2	None
InvalidateProjectCache	region| string like |us-east-2	None
RetryBuild	region| string like |us-east-2	None
RetryBuildBatch	region| string like |us-east-2	None
StartBuild	region| string like |us-east-2	None
StartBuildBatch	region| string like |us-east-2	None
StopBuild	region| string like |us-east-2	None
StopBuildBatch	region| string like |us-east-2	None
UpdateProject	region| string like |us-east-2	None
UpdateProjectVisibility	region| string like |us-east-2	None
UpdateWebhook	region| string like |us-east-2	None

List (2 of 14)

Resumen de la acción CodeBuild_CodeCommit_CodeDeploy StartBuild (escritura):

< Actions StartBuild action in CodeBuild

Resource	Region	Account	Request condition
region| string like |us-east-2	us-east-2	123456789012	None

Permisos obligatorios para obtener acceso a recursos de IAM

Los recursos son objetos dentro de un servicio. Los recursos de IAM incluyen grupos, usuarios, roles y políticas. Si inicia sesión con las credenciales de Usuario raíz de la cuenta de AWS, no tiene restricciones para administrar credenciales de IAM o recursos de IAM. Sin embargo, los usuarios de IAM deben conceder permisos explícitamente para administrar credenciales o recursos de IAM. Puede hacerlo, adjuntando una política basada en identidades al usuario.

> **Note**
>
> En la documentación de AWS, cuando nos referimos a una política de IAM sin mencionar ninguna de las categorías específicas, nos referimos a una política administrada por el cliente basada en identidades. Para obtener información acerca de categorías de políticas, consulte the section called “Políticas y permisos”.

Permisos para administrar identidades de IAM

Los permisos obligatorios para administrar grupos, usuarios, roles y credenciales de IAM normalmente corresponden a las acciones de la API para la tarea. Por ejemplo, con el fin de crear usuarios IAM, debe tener el permiso `iam:CreateUser` que tiene el comando de API correspondiente: `CreateUser`. Para permitir que un usuario de IAM cree otros usuarios de IAM, puede conectar una política de IAM como la siguiente a dicho usuario:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

En una política, el valor del elemento `Resource` depende de la acción y de los recursos a los que la acción puede afectar. En el ejemplo anterior, la política permite a un usuario crear cualquier usuario (* es un comodín que coincide con todas las cadenas). En cambio, una política que permite a los usuarios cambiar únicamente sus propias claves de acceso (acciones de API `CreateAccessKey` y `UpdateAccessKey`) normalmente tiene un elemento `Resource`. En este caso, el ARN incluye una

variable (`${aws:username}`) que se resuelve en el nombre del usuario actual, como en el siguiente ejemplo:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListUsersForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "arn:aws:iam::*:*"
        },
        {
            "Sid": "ViewAndUpdateAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:UpdateAccessKey",
                "iam:CreateAccessKey",
                "iam:ListAccessKeys"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        }
    ]
}
```

En el ejemplo anterior, `${aws:username}` es una variable que se resuelve en el nombre de usuario del usuario actual. Para obtener más información sobre las variables de las políticas, consulte Elementos de la política de IAM: variables y etiquetas.

El uso de un comodín (*) en el nombre de acción a menudo facilita la concesión de permisos para todas las acciones relacionadas con una tarea específica. Por ejemplo, para permitir que los usuarios realicen cualquier acción de IAM, puede utilizar `iam:*` para la acción. Para permitir que los usuarios realicen cualquier acción relacionada solo con claves de acceso, puede utilizar `iam:*AccessKey*` en el elemento `Action` de la instrucción de una política. Esto da al usuario permiso para realizar las acciones `CreateAccessKey`, `DeleteAccessKey`, `GetAccessKeyLastUsed`, `ListAccessKeys` y `UpdateAccessKey`. (Si una acción se añade a IAM en el futuro con "AccessKey" en el nombre, si usa `iam:*AccessKey*` para el elemento `Action` también dará al usuario permiso para esa nueva acción). En el ejemplo siguiente se muestra una política que permite a los usuarios llevar a cabo todas las acciones que pertenecen a sus propias claves de acceso (se sustituye *`account-id`* por su ID de cuenta de Cuenta de AWS):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/${aws:username}"
  }
}
```

Algunas tareas, como, por ejemplo, eliminar un grupo, implican varias acciones: primero eliminar los usuarios del grupo, después separar o eliminar las políticas del grupo y, por último, eliminar el grupo. Si quiere que un usuario pueda eliminar un grupo, debe estar seguro de dar al usuario permisos para llevar a cabo todas las acciones relacionadas.

Permisos para trabajar en la AWS Management Console.

En los ejemplos anteriores se muestran políticas que permiten a un usuario realizar las acciones con la AWS CLI o los SDK de AWS.

A medida que los usuarios trabajan con la consola, esta genera solicitudes para IAM para enumerar grupos, usuarios, roles y políticas, y para obtener las políticas asociadas a un grupo, usuario o rol. La consola también envía solicitudes para obtener información de cuenta de Cuenta de AWS e información sobre la entidad principal. La entidad principal es el usuario que realiza solicitudes en la consola.

En general, para realizar una acción, debe tener únicamente la acción coincidente incluida en una política. Para crear un usuario, necesita permiso para llamar a la acción `CreateUser`. A menudo, cuando utiliza la consola para realizar una acción, debe disponer de permisos para mostrar, enumerar, obtener o consultar recursos en la consola. Esto es necesario para que pueda navegar a través de la consola para realizar la acción especificada. Por ejemplo, si el usuario Jorge desea utilizar la consola para cambiar sus propias claves de acceso, va a la consola de IAM y elige Users. Esta acción hace que la consola genere una solicitud `ListUsers`. Si Jorge no tiene permiso para la acción `iam:ListUsers`, se deniega el acceso a la consola cuando esta intenta enumerar los usuarios. Como resultado, Jorge no puede ir a su propio nombre ni a sus propias claves de acceso, aunque tenga permisos para las acciones `CreateAccessKey` y `UpdateAccessKey`.

Si quiere conceder a los usuarios permisos para administrar grupos, usuarios, roles, políticas y credenciales con la AWS Management Console, debe incluir permisos para las acciones que realiza

la consola. Para ver algunos ejemplos de políticas que puede utilizar para conceder a un usuario estos permisos, consulte Ejemplos de políticas para administrar recursos de IAM.

Conceder permisos sobre cuentas de AWS.

Puede conceder directamente a los usuarios de IAM de su propia cuenta acceso a los recursos. Si hay usuarios de otra cuenta que necesitan obtener acceso a sus recursos, puede crear un rol de IAM, que es una entidad que contiene permisos, pero que no está asociada a un usuario concreto. Los usuarios de otras cuentas pueden utilizar el rol y obtener acceso a recursos en función de los permisos que haya asignado al rol. Para obtener más información, consulte Proporcionar acceso a un usuario de IAM a otra Cuenta de AWS propia.

> **Note**
>
> Algunos servicios de admiten políticas basadas en recursos como se describe en Políticas basadas en identidad y políticas basadas en recursos (como Amazon S3, Amazon SNS y Amazon SQS). Para esos servicios, una alternativa al uso de roles es adjuntar una política al recurso (bucket, tema o cola) que desea compartir. La política basada en recursos puede especificar la cuenta de AWS que tenga permisos para obtener acceso al recurso.

Permisos para que un servicio obtenga acceso a otro.

Muchos servicios de AWS obtienen acceso a otros servicios de AWS. Por ejemplo, varios servicios de AWS, como Amazon EMR, Elastic Load Balancing y Amazon EC2 Auto Scaling administran instancias de Amazon EC2. Otros servicios de AWS utilizan buckets de Amazon S3, temas de Amazon SNS, colas de Amazon SQS, etc.

El escenario de administración de permisos en estos casos varía según el servicio. He aquí algunos ejemplos de cómo se gestionan los permisos para diferentes servicios:

- En Amazon EC2 Auto Scaling, los usuarios deben tener permiso para utilizar Auto Scaling, pero no necesitan que se les concedan de forma explícita permiso para administrar instancias Amazon EC2.
- En AWS Data Pipeline, un rol de IAM determina qué puede hacer una canalización; los usuarios necesitan permiso para asumir el rol. (Para más detalles, consulte Concesión de permisos a pipelines con IAM en la Guía del desarrollador de AWS Data Pipeline).

Para obtener información detallada sobre cómo configurar permisos de forma adecuada, de forma que un servicio de AWS sea capaz de realizar las tareas que usted se propone, consulte la documentación del servicio al que llama. Para obtener información sobre cómo crear un rol para un servicio, consulte Creación de un rol para delegar permisos a un servicio de AWS.

Configuración de un servicio con un rol de IAM para que trabaje en su nombre

Cuando quiere configurar un servicio de AWS para que trabaje en su nombre, normalmente proporciona el ARN de un rol de IAM que define qué puede hacer el servicio. AWS realiza una comprobación para asegurarse de que tiene permisos para transferir un rol a un servicio. Para obtener más información, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS.

Acciones obligatorias

Las acciones son las cosas que puede hacer en un recurso, como visualizar, crear, editar y eliminar dicho recurso. Las acciones se definen por medio de cada servicio de AWS.

Para permitir a alguien realizar una acción, debe incluir las acciones necesarias en una política que se aplica a la identidad que realiza la llamada o al recurso afectado. En general, para proporcionar el permiso obligatorio para realizar una acción, deberá incluir dicha acción en la política. Por ejemplo, para crear un usuario, tiene que añadir la acción CreateUser a su política.

En algunos casos, una acción podría requerir que incluya acciones relacionadas en su política. Por ejemplo, para proporcionar permiso para que alguien cree un directorio en AWS Directory Service utilizando la operación `ds:CreateDirectory`, deberá incluir las siguientes acciones en su política:

- `ds:CreateDirectory`
- `ec2:DescribeSubnets`
- `ec2:DescribeVpcs`
- `ec2:CreateSecurityGroup`
- `ec2:CreateNetworkInterface`
- `ec2:DescribeNetworkInterfaces`
- `ec2:AuthorizeSecurityGroupIngress`
- `ec2:AuthorizeSecurityGroupEgress`

Al crear o editar una política con el editor visual, recibe advertencias y avisos que le ayudarán a elegir todas las acciones necesarias para su política.

Para obtener más información acerca de los permisos necesarios para crear un directorio en AWS Directory Service, consulte Ejemplo 2: Permitir a un usuario crear un directorio.

Ejemplos de políticas para administrar recursos de IAM

A continuación se muestran ejemplos de políticas de IAM que permiten a los usuarios realizar tareas asociadas a la administración de usuarios, grupos y credenciales de IAM. Esto incluye políticas que permiten a los usuarios administrar sus propias contraseñas, claves de acceso y dispositivos de autenticación multifactor (MFA).

Para ver ejemplos de políticas que permiten a los usuarios realizar tareas con otros servicios de AWS, como Amazon S3, Amazon EC2, y DynamoDB consulte Ejemplos de políticas basadas en identidad de IAM.

Temas

- Permitir a un usuario elaborar una lista con los grupos, los usuarios y las políticas de una cuenta, y más información para realizar informes
- Permitir a un usuario administrar la suscripción a un grupo
- Permitir a un usuario administrar usuarios de IAM
- Permitir a los usuarios definir la política de contraseñas de la cuenta
- Permitir a los usuarios generar y recuperar informes de credenciales de IAM
- Permitir todas las acciones de IAM (acceso de administrador)

Permitir a un usuario elaborar una lista con los grupos, los usuarios y las políticas de una cuenta, y más información para realizar informes

La siguiente política permite a los usuarios llamar a cualquier acción de IAM que empiece con la cadena `Get` o `List`, y generar informes. Para ver la política de ejemplo, consulte IAM: ermite el acceso de solo lectura a la consola de IAM.

Permitir a un usuario administrar la suscripción a un grupo

La siguiente política permite al usuario actualizar la suscripción al grupo denominado MarketingGroup. Para ver la política de ejemplo, consulte IAM: permite administrar la pertenencia a un grupo mediante programación y en la consola.

Permitir a un usuario administrar usuarios de IAM

La siguiente política permite a un usuario realizar todas las tareas asociadas con la administración de usuarios de IAM, pero no para realizar acciones en otras entidades, como, por ejemplo, crear grupos o políticas. Las acciones permitidas son:

- Crear el usuario (la acción `CreateUser`).
- Eliminar el usuario. Esta tarea requiere permisos para llevar a cabo las acciones siguientes: `DeleteSigningCertificate`, `DeleteLoginProfile`, `RemoveUserFromGroup` y `DeleteUser`.
- Realizar una lista de los usuarios de la cuenta y de los grupos (las acciones `GetUser`, `ListUsers` y `ListGroupsForUser`).
- Realizar una lista y eliminar las políticas del usuario (las acciones `ListUserPolicies`, `ListAttachedUserPolicies`, `DetachUserPolicy`, `DeleteUserPolicy`).
- Cambiar de nombre o cambiar la ruta para el usuario (la acción `UpdateUser`). El elemento `Resource` debe incluir un ARN que cubra la ruta de origen y la ruta de destino. Para obtener más información acerca de las rutas, consulte Nombres fáciles de recordar y rutas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUsersToPerformUserActions",
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicies",
                "iam:GetPolicy",
                "iam:UpdateUser",
                "iam:AttachUserPolicy",
                "iam:ListEntitiesForPolicy",
                "iam:DeleteUserPolicy",
                "iam:DeleteUser",
                "iam:ListUserPolicies",
                "iam:CreateUser",
                "iam:RemoveUserFromGroup",
                "iam:AddUserToGroup",
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:PutUserPolicy",
```

```
                "iam:ListAttachedUserPolicies",
                "iam:ListUsers",
                "iam:GetUser",
                "iam:DetachUserPolicy"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowUsersToSeeStatsOnIAMConsoleDashboard",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccount*",
                "iam:ListAccount*"
            ],
            "Resource": "*"
        }
    ]
}
```

Varios permisos incluidos en la política anterior permiten al usuario realizar tareas en la AWS Management Console. Los usuarios que realizan tareas relacionadas con el usuario solo desde la AWS CLI, los SDK de AWS o la API de consulta HTTP de IAM no necesitarán determinados permisos. Por ejemplo, si los usuarios ya conocen el ARN de las políticas a separar de un usuario, no necesitarán el permiso `iam:ListAttachedUserPolicies`. La lista exacta de permisos que un usuario requiere depende de las tareas que el usuario debe realizar mientras administra otros usuarios.

Los siguientes permisos de la política otorgan acceso al usuario a las tareas mediante la AWS Management Console:

- `iam:GetAccount*`
- `iam:ListAccount*`

Permitir a los usuarios definir la política de contraseñas de la cuenta

Puede otorgar permisos a algunos usuarios para obtener y actualizar la política de contraseñas de su cuenta de Cuenta de AWS. Para ver la política de ejemplo, consulte IAM: permite establecer los requisitos de contraseña de la cuenta, mediante programación y en la consola.

Permitir a los usuarios generar y recuperar informes de credenciales de IAM

Puede otorgar permiso a los usuarios para generar y descargar un informe que contenga una lista de todos los usuarios de su cuenta de Cuenta de AWS. El informe también muestra el estado de las distintas credenciales del usuario, tales como las contraseñas, las claves de acceso, los dispositivos MFA y los certificados de firma. Para obtener más información sobre los informes de credenciales, consulte Obtención de informes de credenciales para su cuenta de Cuenta de AWS. Para ver la política de ejemplo, consulte IAM: generar y recuperar de informes de credenciales de IAM.

Permitir todas las acciones de IAM (acceso de administrador)

Es posible asignar a algunos usuarios permisos administrativos para llevar a cabo todas las acciones en IAM, incluida la administración de contraseñas, claves de acceso, dispositivos MFA y certificados de usuario. La siguiente política de ejemplo concede estos permisos.

> **Warning**
>
> Al ofrecer a un usuario acceso completo a IAM, no existe ningún límite en lo que respecta a los permisos que un usuario se puede otorgar a sí mismo o a otros. El usuario puede crear entidades de IAM (usuarios o roles) nuevas y otorgarles acceso completo a todos los recursos en su cuenta de Cuenta de AWS. Al ofrecer a un usuario acceso completo a IAM, está otorgándole acceso completo de forma eficaz a todos los recursos de su cuenta de Cuenta de AWS. Esto incluye el acceso para eliminar todos los recursos. Debe conceder estos permisos únicamente a los administradores de confianza y debe forzar la autenticación multifactor (MFA) para estos administradores.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*",
    "Resource": "*"
  }
}
```

Ejemplos de código de IAM con SDK de AWS

Los siguientes ejemplos de código muestran cómo utilizar IAM con un kit de desarrollo de software (SDK) de AWS.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código

- Eliminar un rol de IAM con un SDK de AWS
- Eliminar una política de rol de IAM con un SDK de AWS
- Eliminar un certificado de servidor de IAM con un SDK de AWS
- Eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Eliminar un usuario de IAM con un SDK de AWS
- Eliminar una clave de acceso de IAM con un SDK de AWS
- Eliminar un alias de cuenta de IAM con un SDK de AWS
- Eliminar una política de IAM insertada de un usuario con un SDK de AWS
- Eliminar un perfil de instancia de IAM mediante un AWS SDK
- Desasociar una política de IAM de un rol con un SDK de AWS
- Desasociar una política de IAM de un usuario con un SDK de AWS
- Generar un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de autorización de IAM detallado de la cuenta con un SDK de AWS
- Obtener la política de IAM con un SDK de AWS
- Obtener una versión de la política de IAM con un SDK de AWS
- Obtener un rol de IAM con un SDK de AWS
- Obtener un certificado de servidor de IAM con un SDK de AWS
- Obtención de un estado de eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Obtener un resumen del uso de la cuenta de IAM con un SDK de AWS
- Obtener un usuario de IAM con un SDK de AWS
- Obtener datos sobre el último uso de una clave de acceso de IAM con un SDK de AWS
- Obtener la política de contraseñas de la cuenta de IAM con un SDK de AWS
- Enumerar proveedores de SAML para IAM con un SDK de AWS
- Enumerar las claves de acceso de IAM de un usuario con un SDK de AWS
- Enumerar los alias de cuenta de IAM con un SDK de AWS
- Enumerar grupos de IAM con un SDK de AWS
- Enumerar políticas insertadas para un rol de IAM con un SDK de AWS
- Enumerar políticas de IAM insertadas para un rol de IAM con un AWS SDK

Ejemplos de código de IAM con SDK de AWS

Los siguientes ejemplos de código muestran cómo utilizar IAM con un kit de desarrollo de software (SDK) de AWS.

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las funciones de servicio individuales, es posible ver las acciones en contexto en los escenarios relacionados y en los ejemplos entre servicios.

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica llamando a varias funciones dentro del mismo servicio.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Introducción

Hola, IAM

En los siguientes ejemplos de código se muestra cómo empezar a utilizar IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace IAMActions;

public class HelloIAM
{
    static async Task Main(string[] args)
    {
        // Getting started with AWS Identity and Access Management (IAM). List
        // the policies for the account.
        var iamClient = new AmazonIdentityManagementServiceClient();
```

```
        var listPoliciesPaginator = iamClient.Paginators.ListPolicies(new
 ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        Console.WriteLine("Here are the policies defined for your account:\n");
        policies.ForEach(policy =>
        {
            Console.WriteLine($"Created:
 {policy.CreateDate}\t{policy.PolicyName}\t{policy.Description}");
        });
    }
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for .NET.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Código del archivo de CMake CMakeLists.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS iam)
```

```
# Set this project's name.
project("hello_iam")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD)
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
    # and set the proper subdirectory to the executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_iam.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Código del archivo de origen iam.cpp.

```
#include <aws/core/Aws.h>
#include <aws/iam/IAMClient.h>
```

```
#include <aws/iam/model/ListPoliciesRequest.h>
#include <iostream>
#include <iomanip>

/*
 *  A "Hello IAM" starter application which initializes an AWS Identity and
 Access Management (IAM) client
 *  and lists the IAM policies.
 *
 *  main function
 *
 *  Usage: 'hello_iam'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
    {
        const Aws::String DATE_FORMAT("%Y-%m-%d");
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::IAM::IAMClient iamClient(clientConfig);
        Aws::IAM::Model::ListPoliciesRequest request;

        bool done = false;
        bool header = false;
        while (!done) {
            auto outcome = iamClient.ListPolicies(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Failed to list iam policies: " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }

            if (!header) {
                std::cout << std::left << std::setw(55) << "Name" <<
                          std::setw(30) << "ID" << std::setw(80) << "Arn" <<
```

```
                            std::setw(64) << "Description" << std::setw(12) <<
                            "CreateDate" << std::endl;
                    header = true;
                }

                const auto &policies = outcome.GetResult().GetPolicies();
                for (const auto &policy: policies) {
                    std::cout << std::left << std::setw(55) <<
                              policy.GetPolicyName() << std::setw(30) <<
                              policy.GetPolicyId() << std::setw(80) <<
 policy.GetArn() <<
                              std::setw(64) << policy.GetDescription() <<
 std::setw(12) <<
                              policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
 <<
                              std::endl;
                }

                if (outcome.GetResult().GetIsTruncated()) {
                    request.SetMarker(outcome.GetResult().GetMarker());
                } else {
                    done = true;
                }
            }
        }


        Aws::ShutdownAPI(options); // Should only be called once.
        return result;
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for C++.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
package main

import (
 "context"
 "fmt"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/iam"
)

// main uses the AWS SDK for Go (v2) to create an AWS Identity and Access
 Management (IAM)
// client and list up to 10 policies in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 iamClient := iam.NewFromConfig(sdkConfig)
 const maxPols = 10
 fmt.Printf("Let's list up to %v policies for your account.\n", maxPols)
 result, err := iamClient.ListPolicies(context.TODO(), &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPols),
 })
 if err != nil {
```

```
  fmt.Printf("Couldn't list policies for your account. Here's why: %v\n", err)
  return
 }
 if len(result.Policies) == 0 {
  fmt.Println("You don't have any policies!")
 } else {
  for _, policy := range result.Policies {
   fmt.Printf("\t%v\n", *policy.PolicyName)
  }
 }
}
```

- Para obtener información acerca de la API, consulte ListPolicies en la referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.ListPoliciesResponse;
import software.amazon.awssdk.services.iam.model.Policy;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
```

```
 */
public class HelloIAM {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listPolicies(iam);
    }

    public static void listPolicies(IamClient iam) {
        ListPoliciesResponse response = iam.listPolicies();
        List<Policy> polList = response.policies();
        polList.forEach(policy -> {
            System.out.println("Policy Name: " + policy.policyName());
        });
    }
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { IAMClient, paginateListPolicies } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listLocalPolicies = async () => {
  /**
```

```
   * In v3, the clients expose paginateOperationName APIs that are written using
 async generators so that you can use async iterators in a for await..of loop.
   * https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/index.html#paginators
   */
  const paginator = paginateListPolicies(
    { client, pageSize: 10 },
    // List only customer managed policies.
    { Scope: "Local" },
  );

  console.log("IAM policies defined in your account:");
  let policyCount = 0;
  for await (const page of paginator) {
    if (page.Policies) {
      page.Policies.forEach((p) => {
        console.log(`${p.PolicyName}`);
        policyCount++;
      });
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for JavaScript.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

De src/bin/hello.rs.

```
use aws_sdk_iam::error::SdkError;
use aws_sdk_iam::operation::list_policies::ListPoliciesError;
```

```
use clap::Parser;

const PATH_PREFIX_HELP: &str = "The path prefix for filtering the results.";

#[derive(Debug, clap::Parser)]
#[command(about)]
struct HelloScenarioArgs {
    #[arg(long, default_value="/", help=PATH_PREFIX_HELP)]
    pub path_prefix: String,
}

#[tokio::main]
async fn main() -> Result<(), SdkError<ListPoliciesError>> {
    let sdk_config = aws_config::load_from_env().await;
    let client = aws_sdk_iam::Client::new(&sdk_config);

    let args = HelloScenarioArgs::parse();

    iam_service::list_policies(client, args.path_prefix).await?;

    Ok(())
}
```

De src/iam-service-lib.rs.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
```

```
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API del AWSSDK para Rust.

Ejemplos de código

- Acciones de IAM con SDK de AWS
  - Adición de un usuario de IAM a un grupo mediante un SDK de AWS
  - Asociación de una política de IAM a un rol con un SDK de AWS
  - Asociación de una política de IAM a un usuario con un SDK de AWS
  - Asociación de una política insertada a un rol de IAM con un SDK de AWS
  - Creación de un proveedor SAML de IAM con un SDK de AWS
  - Creación de un grupo de IAM con un SDK de AWS
  - Crear una política de IAM con un SDK de AWS
  - Crear una versión de la política de IAM con un SDK de AWS
  - Crear un rol de IAM con un SDK de AWS
  - Creación de un rol vinculado al servicio de IAM con un SDK de AWS
  - Crear un usuario de IAM con un SDK de AWS
  - Crear una clave de acceso de IAM con un SDK de AWS
  - Crear un alias para una cuenta de IAM con un SDK de AWS
  - Creación de una política de IAM insertada para un grupo con un SDK de AWS
  - Crear una política de IAM insertada para un usuario con un SDK AWS
  - Crear un perfil de instancia de IAM mediante un AWS SDK
  - Eliminación de un proveedor SAML de IAM con un SDK de AWS

- Eliminación de un grupo de IAM con un SDK de AWS
- Eliminación de una política de grupo de IAM con un SDK de AWS
- Eliminar una política de IAM con un SDK de AWS
- Eliminar un rol de IAM con un SDK de AWS
- Eliminar una política de rol de IAM con un SDK de AWS
- Eliminar un certificado de servidor de IAM con un SDK de AWS
- Eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Eliminar un usuario de IAM con un SDK de AWS
- Eliminar una clave de acceso de IAM con un SDK de AWS
- Eliminar un alias de cuenta de IAM con un SDK de AWS
- Eliminar una política de IAM insertada de un usuario con un SDK de AWS
- Eliminar un perfil de instancia de IAM mediante un AWS SDK
- Desasociar una política de IAM de un rol con un SDK de AWS
- Desasociar una política de IAM de un usuario con un SDK de AWS
- Generar un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de autorización de IAM detallado de la cuenta con un SDK de AWS
- Obtener la política de IAM con un SDK de AWS
- Obtener una versión de la política de IAM con un SDK de AWS
- Obtener un rol de IAM con un SDK de AWS
- Obtener un certificado de servidor de IAM con un SDK de AWS
- Obtención de un estado de eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Obtener un resumen del uso de la cuenta de IAM con un SDK de AWS
- Obtener un usuario de IAM con un SDK de AWS
- Obtener datos sobre el último uso de una clave de acceso de IAM con un SDK de AWS
- Obtener la política de contraseñas de la cuenta de IAM con un SDK de AWS
- Enumerar proveedores de SAML para IAM con un SDK de AWS
- Enumerar las claves de acceso de IAM de un usuario con un SDK de AWS
- Enumerar los alias de cuenta de IAM con un SDK de AWS

- Enumerar grupos de IAM con un SDK de AWS
- Enumerar políticas insertadas para un rol de IAM con un SDK de AWS
- Enumerar políticas de IAM insertadas para un rol de IAM con un AWS SDK
- Enumerar políticas de IAM con un SDK de AWS
- Enumerar las políticas asociadas a un rol de IAM con un SDK de AWS
- Enumerar roles de IAM con un SDK de AWS
- Enumerar certificados de servidor de IAM con un SDK de AWS
- Enumerar usuarios de IAM con un SDK de AWS
- Eliminación de un usuario de IAM de un grupo mediante un SDK de AWS
- Actualizar un certificado de servidor de IAM con un SDK de AWS
- Actualizar un usuario de IAM con un SDK de AWS
- Actualizar una clave de acceso de IAM con un SDK de AWS
- Carga de un certificado de servidor de IAM con un SDK de AWS

- Situaciones de IAM con SDK de AWS
  - Cree y gestione un servicio resiliente mediante un SDK de AWS
  - Creación de un grupo de IAM y adición de un usuario a un grupo mediante un SDK de AWS
  - Crear un usuario de IAM y asumir un rol con AWS STS con un SDK de AWS
  - Creación de usuarios de IAM de solo lectura y lectura y escritura con un SDK de AWS
  - Administrar claves de acceso de IAM con un SDK de AWS
  - Administrar políticas de IAM con un SDK de AWS
  - Administrar roles de IAM con un SDK de AWS
  - Administrar la cuenta de IAM con un SDK de AWS
  - Revertir una versión de la política de IAM con un SDK de AWS
  - Trabajar con la API del creador de políticas de IAM mediante un SDK de AWS

Acciones de IAM con SDK de AWS

Los siguientes ejemplos de código muestran cómo llevar a cabo acciones individuales de IAM con SDK de AWS. Estos fragmentos llaman a la API de IAM y son fragmentos de código de programas más grandes que deben ejecutarse en contexto. En cada ejemplo se incluye un enlace a GitHub, con instrucciones de configuración y ejecución del código.

Los ejemplos siguientes incluyen solo las acciones que se utilizan con mayor frecuencia. Para ver una lista completa, consulte la Referencia de la API de AWS Identity and Access Management (IAM).

Ejemplos

- Adición de un usuario de IAM a un grupo mediante un SDK de AWS
- Asociación de una política de IAM a un rol con un SDK de AWS
- Asociación de una política de IAM a un usuario con un SDK de AWS
- Asociación de una política insertada a un rol de IAM con un SDK de AWS
- Creación de un proveedor SAML de IAM con un SDK de AWS
- Creación de un grupo de IAM con un SDK de AWS
- Crear una política de IAM con un SDK de AWS
- Crear una versión de la política de IAM con un SDK de AWS
- Crear un rol de IAM con un SDK de AWS
- Creación de un rol vinculado al servicio de IAM con un SDK de AWS
- Crear un usuario de IAM con un SDK de AWS
- Crear una clave de acceso de IAM con un SDK de AWS
- Crear un alias para una cuenta de IAM con un SDK de AWS
- Creación de una política de IAM insertada para un grupo con un SDK de AWS
- Crear una política de IAM insertada para un usuario con un SDK AWS
- Crear un perfil de instancia de IAM mediante un AWS SDK
- Eliminación de un proveedor SAML de IAM con un SDK de AWS
- Eliminación de un grupo de IAM con un SDK de AWS
- Eliminación de una política de grupo de IAM con un SDK de AWS
- Eliminar una política de IAM con un SDK de AWS
- Eliminar un rol de IAM con un SDK de AWS
- Eliminar una política de rol de IAM con un SDK de AWS
- Eliminar un certificado de servidor de IAM con un SDK de AWS
- Eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Eliminar un usuario de IAM con un SDK de AWS
- Eliminar una clave de acceso de IAM con un SDK de AWS

- Eliminar un alias de cuenta de IAM con un SDK de AWS
- Eliminar una política de IAM insertada de un usuario con un SDK de AWS
- Eliminar un perfil de instancia de IAM mediante un AWS SDK
- Desasociar una política de IAM de un rol con un SDK de AWS
- Desasociar una política de IAM de un usuario con un SDK de AWS
- Generar un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de credencial de IAM con un SDK de AWS
- Obtener un informe de autorización de IAM detallado de la cuenta con un SDK de AWS
- Obtener la política de IAM con un SDK de AWS
- Obtener una versión de la política de IAM con un SDK de AWS
- Obtener un rol de IAM con un SDK de AWS
- Obtener un certificado de servidor de IAM con un SDK de AWS
- Obtención de un estado de eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS
- Obtener un resumen del uso de la cuenta de IAM con un SDK de AWS
- Obtener un usuario de IAM con un SDK de AWS
- Obtener datos sobre el último uso de una clave de acceso de IAM con un SDK de AWS
- Obtener la política de contraseñas de la cuenta de IAM con un SDK de AWS
- Enumerar proveedores de SAML para IAM con un SDK de AWS
- Enumerar las claves de acceso de IAM de un usuario con un SDK de AWS
- Enumerar los alias de cuenta de IAM con un SDK de AWS
- Enumerar grupos de IAM con un SDK de AWS
- Enumerar políticas insertadas para un rol de IAM con un SDK de AWS
- Enumerar políticas de IAM insertadas para un rol de IAM con un AWS SDK
- Enumerar políticas de IAM con un SDK de AWS
- Enumerar las políticas asociadas a un rol de IAM con un SDK de AWS
- Enumerar roles de IAM con un SDK de AWS
- Enumerar certificados de servidor de IAM con un SDK de AWS
- Enumerar usuarios de IAM con un SDK de AWS
- Eliminación de un usuario de IAM de un grupo mediante un SDK de AWS

- Actualizar un certificado de servidor de IAM con un SDK de AWS
- Actualizar un usuario de IAM con un SDK de AWS
- Actualizar una clave de acceso de IAM con un SDK de AWS
- Carga de un certificado de servidor de IAM con un SDK de AWS

Adición de un usuario de IAM a un grupo mediante un SDK de AWS

En el siguiente ejemplo de código se muestra cómo agregar un usuario a un grupo de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <summary>
   /// Add an existing IAM user to an existing IAM group.
   /// </summary>
   /// <param name="userName">The username of the user to add.</param>
   /// <param name="groupName">The name of the group to add the user to.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
   {
       var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
       {
           GroupName = groupName,
           UserName = userName,
       });
```

```
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte AddUserToGroup en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo añadir un usuario a un grupo de IAM

El siguiente comando `add-user-to-group` añade un usuario de IAM denominado `Bob` al grupo de IAM denominado `Admins`.

```
aws iam add-user-to-group \
    --user-name Bob \
    --group-name Admins
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Adición y eliminación de usuarios de un grupo de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte AddUserToGroup en la Referencia de comandos de la AWS CLI.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Asociación de una política de IAM a un rol con un SDK de AWS

Los siguientes ejemplos de código muestran cómo asociar una política de IAM a un rol.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Administrar roles

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for .NETAPI Reference (Referencia de la API de).

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
```

```
  echo "  -p policy_ARN -- The IAM policy document ARN."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) role_name="${OPTARG}" ;;
    p) policy_arn="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$role_name" ]]; then
  errecho "ERROR: You must provide a role name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_arn" ]]; then
  errecho "ERROR: You must provide a policy ARN with the -p parameter."
  usage
  return 1
fi

response=$(aws iam attach-role-policy \
  --role-name "$role_name" \
  --policy-arn "$policy_arn")

local error_code=${?}

if [[ $error_code -ne 0 ]]; then
  aws_cli_error_log $error_code
  errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
  return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte AttachRolePolicy en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::attachRolePolicy(const Aws::String &roleName,
                                   const Aws::String &policyArn,
                                   const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::ListAttachedRolePoliciesRequest list_request;
    list_request.SetRoleName(roleName);

    bool done = false;
    while (!done) {
        auto list_outcome = iam.ListAttachedRolePolicies(list_request);
        if (!list_outcome.IsSuccess()) {
            std::cerr << "Failed to list attached policies of role " <<
                      roleName << ": " << list_outcome.GetError().GetMessage() <<
                      std::endl;
            return false;
        }

        const auto &policies = list_outcome.GetResult().GetAttachedPolicies();
```

```
        if (std::any_of(policies.cbegin(), policies.cend(),
                        [=](const Aws::IAM::Model::AttachedPolicy &policy) {
                                return policy.GetPolicyArn() == policyArn;
                        })) {
            std::cout << "Policy " << policyArn <<
                      " is already attached to role " << roleName << std::endl;
            return true;
        }

        done = !list_outcome.GetResult().GetIsTruncated();
        list_request.SetMarker(list_outcome.GetResult().GetMarker());
    }

    Aws::IAM::Model::AttachRolePolicyRequest request;
    request.SetRoleName(roleName);
    request.SetPolicyArn(policyArn);

    Aws::IAM::Model::AttachRolePolicyOutcome outcome =
 iam.AttachRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Failed to attach policy " << policyArn << " to role " <<
                  roleName << ": " << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        std::cout << "Successfully attached policy " << policyArn << " to role "
 <<
                  roleName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información acerca de la API, consulte AttachRolePolicy en referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo asociar una política administrada a un rol de IAM

El siguiente comando `attach-role-policy` asocia la política administrada de AWS denominada `ReadOnlyAccess` al rol de IAM denominado `ReadOnlyRole`.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/ReadOnlyAccess \
    --role-name ReadOnlyRole
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Políticas administradas y políticas insertadas en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte AttachRolePolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
```

```
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for GoAPI Reference (Referencia de la API de).

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.AttachRolePolicyRequest;
import software.amazon.awssdk.services.iam.model.AttachedPolicy;
import software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesRequest;
import
 software.amazon.awssdk.services.iam.model.ListAttachedRolePoliciesResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AttachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleName = args[0];
        String policyArn = args[1];

        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        attachIAMRolePolicy(iam, roleName, policyArn);
        iam.close();
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
 ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
 iam.listAttachedRolePolicies(request);
```

```
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();

            // Ensure that the policy is not attached to this role
            String polArn = "";
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
 this role.");
                    return;
                }
            }

            AttachRolePolicyRequest attachRequest =
 AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);

            System.out.println("Successfully attached policy " + policyArn +
                    " to role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for Java 2.xAPI Reference (Referencia de la API de).

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Asocie la política.

```
import { AttachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const attachRolePolicy = (policyArn, roleName) => {
  const command = new AttachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        console.log(
          "AmazonDynamoDBFullAccess is already attached to this role."
        );
        process.exit();
      }
    });
    var params = {
      PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
      RoleName: process.argv[2],
    };
    iam.attachRolePolicy(params, function (err, data) {
      if (err) {
        console.log("Unable to attach policy to role", err);
      } else {
        console.log("Role attached successfully");
```

```
      }
    });
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun attachIAMRolePolicy(roleNameVal: String, policyArnVal: String) {

    val request = ListAttachedRolePoliciesRequest {
        roleName = roleNameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkList(attachedPolicies, policyArnVal)
            if (checkStatus == -1)
                return
        }

        val policyRequest = AttachRolePolicyRequest {
            roleName = roleNameVal
```

```
            policyArn = policyArnVal
        }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
    }
}

fun checkList(attachedPolicies: List<AttachedPolicy>, policyArnVal: String): Int
 {

    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}
```

- Para obtener información sobre la API, consulte AttachRolePolicy en la Referencia de la API del AWSSDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
```

```
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

    public function attachRolePolicy($roleName, $policyArn)
    {
        return $this->customWaiter(function () use ($roleName, $policyArn) {
            $this->iamClient->attachRolePolicy([
                'PolicyArn' => $policyArn,
                'RoleName' => $roleName,
            ]);
        });
    }
```

- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for PHPAPI Reference (Referencia de la API de).

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Asocie una política a un rol mediante el objeto Boto3 Policy.

```
def attach_to_role(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).attach_role(RoleName=role_name)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise
```

Asocie una política a un rol mediante el objeto Boto3 Role.

```
def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
```

```
            iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
            logger.info("Attached policy %s to role %s.", policy_arn, role_name)
        except ClientError:
            logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
    role_name)
            raise
```

- Para obtener información sobre la API, consulte AttachRolePolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este módulo de ejemplo, se enumeran, se crean, se adjuntan y se separan las políticas de roles.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
```

```
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
```

```
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte AttachRolePolicy en AWS SDK for RubyAPI Reference (Referencia de la API de).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn attach_role_policy(
    client: &iamClient,
    role: &Role,
    policy: &Policy,
) -> Result<AttachRolePolicyOutput, SdkError<AttachRolePolicyError>> {
    client
        .attach_role_policy()
        .role_name(role.role_name())
        .policy_arn(policy.arn().unwrap_or_default())
        .send()
        .await
}
```

- Para obtener información sobre la API, consulte AttachRolePolicy en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func attachRolePolicy(role: String, policyArn: String) async throws {
    let input = AttachRolePolicyInput(
        policyArn: policyArn,
        roleName: role
    )
    do {
        _ = try await client.attachRolePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Para obtener detalles sobre la API, consulte AttachRolePolicy en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Asociación de una política de IAM a un usuario con un SDK de AWS

Los siguientes ejemplos de código muestran cómo asociar una política de IAM a un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de usuarios de solo lectura, y lectura y escritura

CLI

AWS CLI

Cómo asociar una política administrada a un usuario de IAM

El siguiente comando `attach-user-policy` asocia la política administrada de AWS denominada `AdministratorAccess` al usuario de IAM denominado `Alice`.

```
aws iam attach-user-policy \
    --policy-arn arn:aws:iam::aws:policy/AdministratorAccess \
    --user-name Alice
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Políticas administradas y políticas insertadas en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte AttachUserPolicy en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
```

```
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
user_name)
        raise
```

- Para obtener información sobre la API, consulte AttachUserPolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Attaches a policy to a user
#
# @param user_name [String] The name of the user
# @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_user(user_name, policy_arn)
  @iam_client.attach_user_policy(
    user_name: user_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to user: #{e.message}")
  false
end
```

- Para obtener detalles sobre la API, consulte AttachUserPolicy en AWS SDK for Ruby API Reference (Referencia de la API de).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn attach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .attach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Para obtener detalles sobre la API, consulte AttachUserPolicy en la Referencia de la API del AWS SDK para Rust.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Asociación de una política insertada a un rol de IAM con un SDK de AWS

Los siguientes ejemplos de códigos muestran cómo asociar una política insertada a un rol de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte PutRolePolicy en la Referencia de la API de AWS SDK for .NET.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::putRolePolicy(
        const Aws::String &roleName,
        const Aws::String &policyName,
        const Aws::String &policyDocument,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient iamClient(clientConfig);
    Aws::IAM::Model::PutRolePolicyRequest request;

    request.SetRoleName(roleName);
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(policyDocument);

    Aws::IAM::Model::PutRolePolicyOutcome outcome =
 iamClient.PutRolePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error putting policy on role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully put the role policy." << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener detalles sobre la API, consulte PutRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

CLI

AWS CLI

Cómo asociar una política de permisos a un rol de IAM

El siguiente comando `put-role-policy` agrega una política de permisos al rol denominado `Test-Role`.

```
aws iam put-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Este comando no genera ninguna salida.

La política se define como un documento JSON en el archivo AdminPolicy.json. (El nombre y la extensión del archivo no son significativos).

Para asociar una política de confianza a un rol, utilice el comando `update-assume-role-policy`.

Para obtener más información, consulte Modificación de un rol en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte PutRolePolicy en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { PutRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const examplePolicyDocument = JSON.stringify({
```

```
  Version: "2012-10-17",
  Statement: [
    {
      Sid: "VisualEditor0",
      Effect: "Allow",
      Action: [
        "s3:ListBucketMultipartUploads",
        "s3:ListBucketVersions",
        "s3:ListBucket",
        "s3:ListMultipartUploadParts",
      ],
      Resource: "arn:aws:s3:::some-test-bucket",
    },
    {
      Sid: "VisualEditor1",
      Effect: "Allow",
      Action: [
        "s3:ListStorageLensConfigurations",
        "s3:ListAccessPointsForObjectLambda",
        "s3:ListAllMyBuckets",
        "s3:ListAccessPoints",
        "s3:ListJobs",
        "s3:ListMultiRegionAccessPoints",
      ],
      Resource: "*",
    },
  ],
});

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 * @param {string} policyName
 * @param {string} policyDocument
 */
export const putRolePolicy = async (roleName, policyName, policyDocument) => {
  const command = new PutRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
    PolicyDocument: policyDocument,
  });
```

```
  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener detalles sobre la API, consulte PutRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de un proveedor SAML de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo crear un proveedor SAML de AWS Identity and Access Management (IAM).

CLI

AWS CLI

Cómo crear un proveedor SAML

En este ejemplo se crea un nuevo proveedor SAML en IAM denominado `MySAMLProvider`. Se describe en el documento de metadatos de SAML que se encuentra en el archivo `SAMLMetaData.xml`.

```
aws iam create-saml-provider \
    --saml-metadata-document file://SAMLMetaData.xml \
    --name MySAMLProvider
```

Salida:

```
{
    "SAMLProviderArn": "arn:aws:iam::123456789012:saml-provider/MySAMLProvider"
}
```

Para obtener más información, consulte Creación de proveedores de identidad SAML de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateSAMLProvider en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import * as path from "path";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";

const client = new IAMClient({});

/**
 * This sample document was generated using Auth0.
 * For more information on generating this document,
 see https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_providers_create_saml.html#samlstep1.
 */
const sampleMetadataDocument = readFileSync(
  path.join(
    dirnameFromMetaUrl(import.meta.url),
    "../../../../resources/sample_files/sample_saml_metadata.xml",
  ),
);

/**
 *
 * @param {*} providerName
 * @returns
 */
export const createSAMLProvider = async (providerName) => {
  const command = new CreateSAMLProviderCommand({
    Name: providerName,
```

```
    SAMLMetadataDocument: sampleMetadataDocument.toString(),
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener detalles sobre la API, consulte CreateSAMLProvider en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de un grupo de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un grupo de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
```

```
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }
```

- Para obtener detalles sobre la API, consulte CreateGroup en la Referencia de la API de AWS SDK for JavaScript.

CLI

AWS CLI

Para crear un grupo de IAM

El siguiente comando `create-group` crea un grupo de IAM denominado `Admins`.

```
aws iam create-group \
    --group-name Admins
```

Salida:

```
{
    "Group": {
        "Path": "/",
        "CreateDate": "2015-03-09T20:30:24.940Z",
        "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:group/Admins",
        "GroupName": "Admins"
    }
}
```

Para obtener más información, consulte Creación de grupos de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateGroup en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { CreateGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const createGroup = async (groupName) => {
  const command = new CreateGroupCommand({ GroupName: groupName });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener detalles sobre la API, consulte CreateGroup en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear una política de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear una política de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura
- Administrar políticas
- Trabajar con la API del creador de políticas de IAM

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
```

```
  echo "function iam_create_policy"
  echo "Creates an AWS Identity and Access Management (IAM) policy."
  echo "  -n policy_name   The name of the IAM policy."
  echo "  -p policy_json -- The policy document."
  echo ""
}

# Retrieve the calling parameters.
while getopts "n:p:h" option; do
  case "${option}" in
    n) policy_name="${OPTARG}" ;;
    p) policy_document="${OPTARG}" ;;
    h)
      usage
      return 0
      ;;
    \?)
      echo "Invalid parameter"
      usage
      return 1
      ;;
  esac
done
export OPTIND=1

if [[ -z "$policy_name" ]]; then
  errecho "ERROR: You must provide a policy name with the -n parameter."
  usage
  return 1
fi

if [[ -z "$policy_document" ]]; then
  errecho "ERROR: You must provide a policy document with the -p parameter."
  usage
  return 1
fi

response=$(aws iam create-policy \
  --policy-name "$policy_name" \
  --policy-document "$policy_document" \
  --output text \
  --query Policy.Arn)

local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}
```

- Para obtener información de la API, consulte CreatePolicy en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
Aws::String AwsDoc::IAM::createPolicy(const Aws::String &policyName,
                                      const Aws::String &rsrcArn,
                                      const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreatePolicyRequest request;
    request.SetPolicyName(policyName);
    request.SetPolicyDocument(BuildSamplePolicyDocument(rsrcArn));

    Aws::IAM::Model::CreatePolicyOutcome outcome = iam.CreatePolicy(request);
    Aws::String result;
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating policy " << policyName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
```

```
        result = outcome.GetResult().GetPolicy().GetArn();
        std::cout << "Successfully created policy " << policyName <<
                  std::endl;
    }

    return result;
}

Aws::String AwsDoc::IAM::BuildSamplePolicyDocument(const Aws::String &rsrc_arn) {
    std::stringstream stringStream;
    stringStream << "{"
                 << "  \"Version\": \"2012-10-17\","
                 << "  \"Statement\": ["
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": \"logs:CreateLogGroup\","
                 << "        \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    },"
                 << "    {"
                 << "        \"Effect\": \"Allow\","
                 << "        \"Action\": ["
                 << "            \"dynamodb:DeleteItem\","
                 << "            \"dynamodb:GetItem\","
                 << "            \"dynamodb:PutItem\","
                 << "            \"dynamodb:Scan\","
                 << "            \"dynamodb:UpdateItem\""
                 << "       ],"
                 << "       \"Resource\": \""
                 << rsrc_arn
                 << "\""
                 << "    }"
                 << "   ]"
                 << "}";

    return stringStream.str();
}
```

- Para obtener información acerca de la API, consulte CreatePolicy en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Ejemplo 1: cómo crear una política administrada por el cliente

El siguiente comando crea una política administrada por el cliente denominada `my-policy`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy
```

El archivo `policy` es un documento JSON de la carpeta actual que concede acceso de solo lectura a la carpeta `shared` de un bucket de Amazon S3 denominado `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket/shared/*"
            ]
        }
    ]
}
```

Salida:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "CreateDate": "2015-06-01T19:31:18.620Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "ZXR6A36LTYANPAI7NJ5UV",
        "DefaultVersionId": "v1",
        "Path": "/",
```

```
        "Arn": "arn:aws:iam::0123456789012:policy/my-policy",
        "UpdateDate": "2015-06-01T19:31:18.620Z"
    }
}
```

Para obtener más información sobre el uso de archivos como entrada para los parámetros de cadena, consulte Especificar valores de parámetros para la CLI de AWS en la Guía del usuario de la CLI de AWS.

Ejemplo 2: cómo crear una política administrada por el cliente con una descripción

El siguiente comando crea una política administrada por el cliente denominada `my-policy` con una descripción inmutable:

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --description "This policy grants access to all Put, Get, and List actions
 for my-bucket"
```

El archivo `policy.json` es un documento JSON de la carpeta actual que concede acceso a todas las acciones poner, enumerar y obtener de un bucket de Amazon S3 denominado `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Salida:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::123456789012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T22:38:47+00:00",
        "UpdateDate": "2023-05-24T22:38:47+00:00"
    }
}
```

Para obtener más información acerca de las políticas basadas en identidades, consulte Políticas basadas en identidades y políticas basadas en recursos en la Guía del usuario de IAM de AWS.

Ejemplo 3: cómo crear una política administrada por el cliente con etiquetas

El siguiente comando crea una política administrada por el cliente denominada `my-policy` con etiquetas. En este ejemplo, se utiliza el indicador de parámetro `--tags` con las siguientes etiquetas con formato JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativamente, el indicador `--tags` se puede usar con indicadores con el formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

El archivo `policy.json` es un documento JSON de la carpeta actual que concede acceso a todas las acciones poner, enumerar y obtener de un bucket de Amazon S3 denominado `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Salida:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "ANPAWGSUGIDPEXAMPLE",
        "Arn": "arn:aws:iam::12345678012:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        "AttachmentCount": 0,
        "PermissionsBoundaryUsageCount": 0,
        "IsAttachable": true,
        "CreateDate": "2023-05-24T23:16:39+00:00",
        "UpdateDate": "2023-05-24T23:16:39+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
                "Key": "Location",
                "Value": "Seattle"
            {

        ]
    }
}
```

Para obtener más información sobre las políticas de etiquetado, consulte Políticas de etiquetado administradas por el cliente en la Guía del usuario de IAM de AWS.

- Para obtener información de la API, consulte CreatePolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
```

```
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreatePolicyRequest;
import software.amazon.awssdk.services.iam.model.CreatePolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyRequest;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
```

```
import software.amazon.awssdk.services.iam.waiters.IamWaiter;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreatePolicy {

    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"dynamodb:DeleteItem\"," +
            "            \"dynamodb:GetItem\"," +
            "            \"dynamodb:PutItem\"," +
            "            \"dynamodb:Scan\"," +
            "            \"dynamodb:UpdateItem\"" +
            "       ]," +
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

    public static void main(String[] args) {

        final String usage = """
                Usage:
                    CreatePolicy <policyName>\s

                Where:
                    policyName - A unique policy name.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String policyName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMPolicy(iam, policyName);
        System.out.println("Successfully created a policy with this ARN value: "
 + result);
        iam.close();
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument)
                    .build();

            CreatePolicyResponse response = iam.createPolicy(request);

            // Wait until the policy is created.
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
 iamWaiter.waitUntilPolicyExists(polRequest);

 waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree la política.

```
import { CreatePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyName
 */
export const createPolicy = (policyName) => {
  const command = new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: "*",
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var myManagedPolicy = {
  Version: "2012-10-17",
  Statement: [
    {
      Effect: "Allow",
      Action: "logs:CreateLogGroup",
      Resource: "RESOURCE_ARN",
    },
    {
      Effect: "Allow",
      Action: [
        "dynamodb:DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
      ],
      Resource: "RESOURCE_ARN",
    },
```

```
  ],
};

var params = {
  PolicyDocument: JSON.stringify(myManagedPolicy),
  PolicyName: "myDynamoDBPolicy",
};

iam.createPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun createIAMPolicy(policyNameVal: String?): String {

    val policyDocumentVal = "{" +
        "  \"Version\": \"2012-10-17\"," +
        "  \"Statement\": [" +
        "    {" +
        "        \"Effect\": \"Allow\"," +
        "        \"Action\": [" +
        "            \"dynamodb:DeleteItem\"," +
```

```
        "            \"dynamodb:GetItem\"," +
        "            \"dynamodb:PutItem\"," +
        "            \"dynamodb:Scan\"," +
        "            \"dynamodb:UpdateItem\"" +
        "        ]," +
        "        \"Resource\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API del AWSSDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
```

```
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

    public function createPolicy(string $policyName, string $policyDocument)
    {
        $result = $this->customWaiter(function () use ($policyName,
 $policyDocument) {
            return $this->iamClient->createPolicy([
                'PolicyName' => $policyName,
                'PolicyDocument' => $policyDocument,
            ]);
        });
        return $result['Policy'];
    }
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
```

```
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este módulo de ejemplo, se enumeran, se crean, se adjuntan y se separan las políticas de roles.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn create_policy(
    client: &iamClient,
    policy_name: &str,
    policy_document: &str,
) -> Result<Policy, iamError> {
    let policy = client
        .create_policy()
        .policy_name(policy_name)
        .policy_document(policy_document)
        .send()
        .await?;
    Ok(policy.policy.unwrap())
}
```

- Para obtener información sobre la API, consulte CreatePolicy en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func createPolicy(name: String, policyDocument: String) async throws -
> IAMClientTypes.Policy {
        let input = CreatePolicyInput(
            policyDocument: policyDocument,
            policyName: name
        )
        do {
            let output = try await iamClient.createPolicy(input: input)
            guard let policy = output.policy else {
                throw ServiceHandlerError.noSuchPolicy
            }
            return policy
        } catch {
            throw error
        }
    }
```

- Para obtener detalles sobre la API, consulte CreatePolicy en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear una versión de la política de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo crear una versión de la política de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administrar políticas

CLI

AWS CLI

Cómo crear una nueva versión de la política administrada

En este ejemplo, se crea una nueva versión v2 de la política de IAM cuyo ARN es `arn:aws:iam::123456789012:policy/MyPolicy` y la convierte en la versión predeterminada.

```
aws iam create-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --policy-document file://NewPolicyVersion.json \
    --set-as-default
```

Salida:

```
{
    "PolicyVersion": {
        "CreateDate": "2015-06-16T18:56:03.721Z",
        "VersionId": "v2",
        "IsDefaultVersion": true
    }
}
```

Para obtener más información, consulte Control de versiones de políticas de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreatePolicyVersion en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
```

```
        return policy_version
```

- Para obtener detalles sobre la API, consulte CreatePolicyVersion en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear un rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un rol de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Administrar roles

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
```

```
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_create_role
#
```

```
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
```

```
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información de la API, consulte CreateRole en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::createIamRole(
        const Aws::String &roleName,
        const Aws::String &policy,
        const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::CreateRoleRequest request;

    request.SetRoleName(roleName);
    request.SetAssumeRolePolicyDocument(policy);

    Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const Aws::IAM::Model::Role iamRole = outcome.GetResult().GetRole();
        std::cout << "Created role " << iamRole.GetRoleName() << "\n";
        std::cout << "ID: " << iamRole.GetRoleId() << "\n";
        std::cout << "ARN: " << iamRole.GetArn() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información acerca de la API, consulte CreateRole en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Ejemplo 1: cómo crear un rol de IAM

El siguiente comando `create-role` crean un rol denominado `Test-Role` y le asocia una política de confianza.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json
```

Salida:

```
{
    "Role": {
        "AssumeRolePolicyDocument": "<URL-encoded-JSON>",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "CreateDate": "2013-06-07T20:43:32.821Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

La política de confianza se define como un documento JSON en el archivo Test-Role-Trust-Policy.json. (El nombre y la extensión del archivo no son significativos). La política de confianza debe especificar una entidad principal.

Utilice el comando `put-role-policy` para asociar una política de permisos a un rol.

Para más información, consulte Creación de roles de IAM en la Guía del usuario de IAM de AWS.

Ejemplo 2: cómo crear un rol de IAM con una duración máxima de sesión especificada

El siguiente comando `create-role` crea un rol denominado `Test-Role` y establece una duración máxima de sesión de 7200 segundos (2 horas).

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
```

```
    --max-session-duration 7200
```

Salida:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:role/Test-Role",
        "CreateDate": "2023-05-24T23:50:25+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::12345678012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        }
    }
}
```

Para obtener más información, consulte Modificación de la duración máxima de sesión (API de AWS) en la Guía del usuario de IAM de AWS.

Ejemplo 3: cómo crear un rol de IAM con etiquetas

El siguiente comando crea un rol de IAM `Test-Role` con etiquetas. En este ejemplo, se utiliza el indicador de parámetro `--tags` con las siguientes etiquetas con formato JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativamente, el indicador `--tags` se puede usar con etiquetas en formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-role \
    --role-name Test-Role \
    --assume-role-policy-document file://Test-Role-Trust-Policy.json \
```

```
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Salida:

```
{
    "Role": {
        "Path": "/",
        "RoleName": "Test-Role",
        "RoleId": "AKIAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:role/Test-Role",
        "CreateDate": "2023-05-25T23:29:41+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Sid": "Statement1",
                    "Effect": "Allow",
                    "Principal": {
                        "AWS": "arn:aws:iam::123456789012:root"
                    },
                    "Action": "sts:AssumeRole"
                }
            ]
        },
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Para obtener más información, consulte Etiquetado de roles de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información de la API, consulte CreateRole en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Principal: map[string]string{"AWS": trustedUserArn},
   Action: []string{"sts:AssumeRole"},
  }},
 }
 policyBytes, err := json.Marshal(trustPolicy)
 if err != nil {
  log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
 trustedUserArn, err)
  return nil, err
```

```
 }
 result, err := wrapper.IamClient.CreateRole(context.TODO(),
 &iam.CreateRoleInput{
  AssumeRolePolicyDocument: aws.String(string(policyBytes)),
  RoleName:                 aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import org.json.simple.JSONObject;
import org.json.simple.parser.JSONParser;
import software.amazon.awssdk.services.iam.model.CreateRoleRequest;
import software.amazon.awssdk.services.iam.model.CreateRoleResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import java.io.FileReader;

/*
*   This example requires a trust policy document. For more information, see:
```

```
*   https://aws.amazon.com/blogs/security/how-to-use-trust-policies-with-iam-
roles/
*
*
*  In addition, set up your development environment, including your credentials.
*
*  For information, see this documentation topic:
*
*  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class CreateRole {
    public static void main(String[] args) throws Exception {
        final String usage = """
                Usage:
                    <rolename> <fileLocation>\s

                Where:
                    rolename - The name of the role to create.\s
                    fileLocation - The location of the JSON document that
 represents the trust policy.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String rolename = args[0];
        String fileLocation = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMRole(iam, rolename, fileLocation);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMRole(IamClient iam, String rolename, String
 fileLocation) throws Exception {
        try {
```

```
            JSONObject jsonObject = (JSONObject)
 readJsonSimpleDemo(fileLocation);
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(jsonObject.toJSONString())
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
 response.role().arn());

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static Object readJsonSimpleDemo(String filename) throws Exception {
        FileReader reader = new FileReader(filename);
        JSONParser jsonParser = new JSONParser();
        return jsonParser.parse(reader);
    }
}
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree el rol.

```
import { CreateRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const createRole = (roleName) => {
  const command = new CreateRoleCommand({
    AssumeRolePolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Principal: {
            Service: "lambda.amazonaws.com",
          },
          Action: "sts:AssumeRole",
        },
      ],
    }),
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
echo "Created role: {$assumeRoleRole['RoleName']}\n";

    /**
     * @param string $roleName
     * @param string $rolePolicyDocument
     * @return array
     * @throws AwsException
     */
    public function createRole(string $roleName, string $rolePolicyDocument)
    {
        $result = $this->customWaiter(function () use ($roleName,
 $rolePolicyDocument) {
            return $this->iamClient->createRole([
                'AssumeRolePolicyDocument' => $rolePolicyDocument,
                'RoleName' => $roleName,
            ]);
        });
        return $result['Role'];
    }
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Creates a role and attaches policies to it.
 #
 # @param role_name [String] The name of the role.
 # @param assume_role_policy_document [Hash] The trust relationship policy
document.
 # @param policy_arns [Array<String>] The ARNs of the policies to attach.
 # @return [String, nil] The ARN of the new role if successful, or nil if an
error occurred.
 def create_role(role_name, assume_role_policy_document, policy_arns)
   response = @iam_client.create_role(
     role_name: role_name,
     assume_role_policy_document: assume_role_policy_document.to_json
   )
   role_arn = response.role.arn

   policy_arns.each do |policy_arn|
     @iam_client.attach_role_policy(
       role_name: role_name,
       policy_arn: policy_arn
     )
   end

   role_arn
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error creating role: #{e.message}")
   nil
```

```
  end
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn create_role(
    client: &iamClient,
    role_name: &str,
    role_policy_document: &str,
) -> Result<Role, iamError> {
    let response: CreateRoleOutput = loop {
        if let Ok(response) = client
            .create_role()
            .role_name(role_name)
            .assume_role_policy_document(role_policy_document)
            .send()
            .await
        {
            break response;
        }
    };

    Ok(response.role.unwrap())
}
```

- Para obtener información sobre la API, consulte CreateRole en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func createRole(name: String, policyDocument: String) async throws ->
String {
        let input = CreateRoleInput(
            assumeRolePolicyDocument: policyDocument,
            roleName: name
        )
        do {
            let output = try await client.createRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            guard let id = role.roleId else {
                throw ServiceHandlerError.noSuchRole
            }
            return id
        } catch {
            throw error
        }
    }
```

- Para obtener información acerca de la API, consulte CreateRole en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de un rol vinculado al servicio de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un rol vinculado al servicio de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }
```

- Para obtener información acerca de la API, consulte CreateServiceLinkedRole en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo crear un rol vinculado a un servicio

En el siguiente ejemplo de `create-service-linked-role` se crea un rol vinculado a servicios para el servicio de AWS especificado y se adjunta la descripción especificada.

```
aws iam create-service-linked-role \
    --aws-service-name lex.amazonaws.com \
    --description "My service-linked role to support Lex"
```

Salida:

```
{
    "Role": {
        "Path": "/aws-service-role/lex.amazonaws.com/",
        "RoleName": "AWSServiceRoleForLexBots",
        "RoleId": "AROA1234567890EXAMPLE",
        "Arn": "arn:aws:iam::1234567890:role/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots",
        "CreateDate": "2019-04-17T20:34:14+00:00",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": [
                        "sts:AssumeRole"
                    ],
                    "Effect": "Allow",
                    "Principal": {
                        "Service": [
                            "lex.amazonaws.com"
                        ]
                    }
                }
            ]
        }
    }
}
```

Para obtener más información, consulte Uso de roles vinculados a servicios en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
```

```
 return role, err
}
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un rol vinculado al servicio.

```
import { CreateServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} serviceName
 */
export const createServiceLinkedRole = async (serviceName) => {
  const command = new CreateServiceLinkedRoleCommand({
    // For a list of AWS services that support service-linked roles,
    // see https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-
services-that-work-with-iam.html.
    //
    // For a list of AWS service endpoints, see https://docs.aws.amazon.com/
general/latest/gr/aws-service-information.html.
    AWSServiceName: serviceName,
  });

  const response = await client.send(command);
  console.log(response);
```

```
  return response;
};
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function createServiceLinkedRole($awsServiceName, $customSuffix = "",
 $description = "")
    {
        $createServiceLinkedRoleArguments = ['AWSServiceName' =>
 $awsServiceName];
        if ($customSuffix) {
            $createServiceLinkedRoleArguments['CustomSuffix'] = $customSuffix;
        }
        if ($description) {
            $createServiceLinkedRoleArguments['Description'] = $description;
        }
        return $this->iamClient-
>createServiceLinkedRole($createServiceLinkedRoleArguments);
    }
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_service_linked_role(service_name, description):
    """
    Creates a service-linked role.

    :param service_name: The name of the service that owns the role.
    :param description: A description to give the role.
    :return: The newly created role.
    """
    try:
        response = iam.meta.client.create_service_linked_role(
            AWSServiceName=service_name, Description=description
        )
        role = iam.Role(response["Role"]["RoleName"])
        logger.info("Created service-linked role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create service-linked role for %s.",
 service_name)
        raise
    else:
        return role
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Creates a service-linked role
 #
 # @param service_name [String] The service name to create the role for.
 # @param description [String] The description of the service-linked role.
 # @param suffix [String] Suffix for customizing role name.
 # @return [String] The name of the created role
 def create_service_linked_role(service_name, description, suffix)
   response = @iam_client.create_service_linked_role(
     aws_service_name: service_name, description: description, custom_suffix:
suffix,)
   role_name = response.role.role_name
   @logger.info("Created service-linked role #{role_name}.")
   role_name
 rescue Aws::Errors::ServiceError => e
   @logger.error("Couldn't create service-linked role for #{service_name}.
Here's why:")
   @logger.error("\t#{e.code}: #{e.message}")
   raise
 end
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn create_service_linked_role(
    client: &iamClient,
    aws_service_name: String,
    custom_suffix: Option<String>,
    description: Option<String>,
) -> Result<CreateServiceLinkedRoleOutput,
 SdkError<CreateServiceLinkedRoleError>> {
    let response = client
        .create_service_linked_role()
        .aws_service_name(aws_service_name)
        .set_custom_suffix(custom_suffix)
        .set_description(description)
        .send()
        .await?;

    Ok(response)
}
```

- Para obtener información sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   public func createServiceLinkedRole(service: String, suffix: String? = nil,
description: String?)
                   async throws -> IAMClientTypes.Role {
       let input = CreateServiceLinkedRoleInput(
           awsServiceName: service,
           customSuffix: suffix,
           description: description
       )
       do {
           let output = try await client.createServiceLinkedRole(input: input)
           guard let role = output.role else {
               throw ServiceHandlerError.noSuchRole
           }
           return role
       } catch {
           throw error
       }
   }
```

- Para obtener detalles sobre la API, consulte CreateServiceLinkedRole en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear un usuario de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un usuario de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
```

```
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi
```

```
  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateUserRequest create_request;
    create_request.SetUserName(userName);

    auto create_outcome = iam.CreateUser(create_request);
    if (!create_outcome.IsSuccess()) {
        std::cerr << "Error creating IAM user " << userName << ":" <<
                  create_outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created IAM user " << userName << std::endl;
    }

    return create_outcome.IsSuccess();
```

- Para obtener información acerca de la API, consulte CreateUser en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Ejemplo 1: cómo crear un usuario de IAM

El siguiente comando `create-user` crea un usuario de IAM denominado Bob en la cuenta actual.

```
aws iam create-user \
    --user-name Bob
```

Salida:

```
{
    "User": {
        "UserName": "Bob",
        "Path": "/",
        "CreateDate": "2023-06-08T03:20:41.270Z",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Bob"
```

```
    }
}
```

Para obtener más información, consulte Creación de un usuario de IAM en su cuenta de AWS en la Guía del usuario de IAM de AWS.

Ejemplo 2: cómo crear un usuario de IAM en una ruta específica

El siguiente comando `create-user` crea un usuario de IAM denominado Bob en la ruta especificada.

```
aws iam create-user \
    --user-name Bob \
    --path /division_abc/subdivision_xyz/
```

Salida:

```
{
    "User": {
        "Path": "/division_abc/subdivision_xyz/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/division_abc/subdivision_xyz/Bob",
        "CreateDate": "2023-05-24T18:20:17+00:00"
    }
}
```

Para obtener más información, consulte Identificadores de IAM en la Guía del usuario de IAM de AWS.

Ejemplo 3: cómo crear un usuario de IAM con etiquetas

El siguiente comando `create-user` crea un usuario de IAM denominado Bob con etiquetas. En este ejemplo, se utiliza el indicador de parámetro `--tags` con las siguientes etiquetas con formato JSON: `'{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location", "Value": "Seattle"}'`. Alternativamente, el indicador `--tags` se puede usar con etiquetas en formato abreviado: `'Key=Department,Value=Accounting Key=Location,Value=Seattle'`.

```
aws iam create-user \
```

```
    --user-name Bob \
    --tags '{"Key": "Department", "Value": "Accounting"}' '{"Key": "Location",
 "Value": "Seattle"}'
```

Salida:

```
{
    "User": {
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-25T17:14:21+00:00",
        "Tags": [
            {
                "Key": "Department",
                "Value": "Accounting"
            },
            {
                "Key": "Location",
                "Value": "Seattle"
            }
        ]
    }
}
```

Para obtener más información, consulte Etiquetado de usuarios de IAM en la Guía del usuario de AWS.

Ejemplo 3: cómo crear un usuario de IAM con un límite de permisos establecido

El siguiente comando `create-user` crea un usuario de IAM denominado Bob con el límite de permisos de AmazonS3FullAccess.

```
aws iam create-user \
    --user-name Bob \
    --permissions-boundary arn:aws:iam::aws:policy/AmazonS3FullAccess
```

Salida:

```
{
    "User": {
```

```
        "Path": "/",
        "UserName": "Bob",
        "UserId": "AIDAIOSFODNN7EXAMPLE",
        "Arn": "arn:aws:iam::12345678012:user/Bob",
        "CreateDate": "2023-05-24T17:50:53+00:00",
        "PermissionsBoundary": {
        "PermissionsBoundaryType": "Policy",
        "PermissionsBoundaryArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
        }
    }
}
```

Para obtener más información, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateUser en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
```

```
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.core.waiters.WaiterResponse;
import software.amazon.awssdk.services.iam.model.CreateUserRequest;
import software.amazon.awssdk.services.iam.model.CreateUserResponse;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.waiters.IamWaiter;
import software.amazon.awssdk.services.iam.model.GetUserRequest;
import software.amazon.awssdk.services.iam.model.GetUserResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
```

```
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username>\s

                Where:
                    username - The name of the user to create.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String result = createIAMUser(iam, username);
        System.out.println("Successfully created user: " + result);
        iam.close();
    }

    public static String createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();

            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();

            CreateUserResponse response = iam.createUser(request);

            // Wait until the user is created.
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
```

```
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
 iamWaiter.waitUntilUserExists(userRequest);

 waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user().userName();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree el usuario .

```
import { CreateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const createUser = (name) => {
  const command = new CreateUserCommand({ UserName: name });
```

```
  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    iam.createUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  } else {
    console.log(
      "User " + process.argv[2] + " already exists",
      data.User.UserId
```

```
    );
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {

    val request = CreateUserRequest {
        userName = usernameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API del AWSSDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";


    /**
     * @param string $name
     * @return array
     * @throws AwsException
     */
    public function createUser(string $name): array
    {
        $result = $this->iamClient->createUser([
            'UserName' => $name,
        ]);

        return $result['User'];
    }
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  # Creates a user and their login profile
  #
  # @param user_name [String] The name of the user
  # @param initial_password [String] The initial password for the user
  # @return [String, nil] The ID of the user if created, or nil if an error
 occurred
  def create_user(user_name, initial_password)
    response = @iam_client.create_user(user_name: user_name)
    @iam_client.wait_until(:user_exists, user_name: user_name)
    @iam_client.create_login_profile(
      user_name: user_name,
      password: initial_password,
      password_reset_required: true
    )
    @logger.info("User '#{user_name}' created successfully.")
    response.user.user_id
  rescue Aws::IAM::Errors::EntityAlreadyExists
    @logger.error("Error creating user '#{user_name}': user already exists.")
    nil
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating user '#{user_name}': #{e.message}")
    nil
  end
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn create_user(client: &iamClient, user_name: &str) -> Result<User,
 iamError> {
    let response = client.create_user().user_name(user_name).send().await?;
```

```
    Ok(response.user.unwrap())
}
```

- Para obtener información sobre la API, consulte CreateUser en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func createUser(name: String) async throws -> String {
        let input = CreateUserInput(
            userName: name
        )
        do {
            let output = try await client.createUser(input: input)
            guard let user = output.user else {
                throw ServiceHandlerError.noSuchUser
            }
            guard let id = user.userId else {
                throw ServiceHandlerError.noSuchUser
            }
            return id
        } catch {
            throw error
        }
```

```
    }
```

- Para obtener detalles sobre la API, consulte CreateUser en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear una clave de acceso de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear una clave de acceso de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura
- Administrar claves de acceso

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}
```

```
###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
```

```
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}
```

- Para obtener información de la API, consulte CreateAccessKey en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
Aws::String AwsDoc::IAM::createAccessKey(const Aws::String &userName,
                                         const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::CreateAccessKeyRequest request;
    request.SetUserName(userName);

    Aws::String result;
    Aws::IAM::Model::CreateAccessKeyOutcome outcome =
 iam.CreateAccessKey(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating access key for IAM user " << userName
                  << ":" << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &accessKey = outcome.GetResult().GetAccessKey();
        std::cout << "Successfully created access key for IAM user " <<
                  userName << std::endl << "  aws_access_key_id = " <<
                  accessKey.GetAccessKeyId() << std::endl <<
                  " aws_secret_access_key = " << accessKey.GetSecretAccessKey()
 <<
                  std::endl;
        result = accessKey.GetAccessKeyId();
    }

    return result;
}
```

- Para obtener información acerca de la API, consulte CreateAccessKey en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo crear una clave de acceso para un usuario de IAM

El siguiente comando `create-access-key` crea una clave de acceso (un ID de clave de acceso y una clave de acceso secreta) para el usuario de IAM denominado Bob.

```
aws iam create-access-key \
    --user-name Bob
```

Salida:

```
{
    "AccessKey": {
        "UserName": "Bob",
        "Status": "Active",
        "CreateDate": "2015-03-09T18:39:23.411Z",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
        "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
    }
}
```

Almacene la clave de acceso secreta en un lugar seguro. Si se pierde, no se puede recuperar y debe crear una nueva clave de acceso.

Para obtener más información, consulte Administración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información de la API, consulte CreateAccessKey en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.CreateAccessKeyResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                   <user>\s

                Where:
                   user - An AWS IAM user that you can obtain from the AWS
 Management Console.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String user = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        String keyId = createIAMAccessKey(iam, user);
        System.out.println("The Key Id is " + keyId);
        iam.close();
    }
```

```
    public static String createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey().accessKeyId();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree la clave de acceso.

```
import { CreateAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 */
```

```
export const createAccessKey = (userName) => {
  const command = new CreateAccessKeyCommand({ UserName: userName });
  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccessKey({ UserName: "IAM_USER_NAME" }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.AccessKey);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun createIAMAccessKey(user: String?): String {

    val request = CreateAccessKeyRequest {
        userName = user
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.
```

```
    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Este módulo de ejemplo muestra, crea, desactiva y elimina las claves de acceso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end
```

```
  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
```

```
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn create_access_key(client: &iamClient, user_name: &str) ->
 Result<AccessKey, iamError> {
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<CreateAccessKeyOutput, SdkError<CreateAccessKeyError>> =
 loop {
        match client.create_access_key().user_name(user_name).send().await {
            Ok(inner_response) => {
                break Ok(inner_response);
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    Ok(response.unwrap().access_key.unwrap())
}
```

- Para obtener información sobre la API, consulte CreateAccessKey en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   public func createAccessKey(userName: String) async throws ->
IAMClientTypes.AccessKey {
        let input = CreateAccessKeyInput(
            userName: userName
        )
        do {
            let output = try await iamClient.createAccessKey(input: input)
            guard let accessKey = output.accessKey else {
                throw ServiceHandlerError.keyError
            }
            return accessKey
        } catch {
            throw error
        }
    }
```

- Para obtener detalles sobre la API, consulte CreateAccessKey en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear un alias para una cuenta de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un alias para una cuenta de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Administre su cuenta

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::createAccountAlias(const Aws::String &aliasName,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::CreateAccountAliasRequest request;
    request.SetAccountAlias(aliasName);

    Aws::IAM::Model::CreateAccountAliasOutcome outcome = iam.CreateAccountAlias(
            request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error creating account alias " << aliasName << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully created account alias " << aliasName <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo crear un alias de una cuenta

El siguiente comando `create-account-alias` crea el alias `examplecorp` para su cuenta de AWS.

```
aws iam create-account-alias \
    --account-alias examplecorp
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Su ID de cuenta y alias de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.CreateAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateAccountAlias {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to create (for example,
 myawsaccount).\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        createIAMAccountAlias(iam, alias);
        iam.close();
        System.out.println("Done");
    }

    public static void createIAMAccountAlias(IamClient iam, String alias) {
        try {
            CreateAccountAliasRequest request =
 CreateAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.createAccountAlias(request);
            System.out.println("Successfully created account alias: " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree el alias de la cuenta.

```
import { CreateAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias - A unique name for the account alias.
 * @returns
 */
export const createAccountAlias = (alias) => {
  const command = new CreateAccountAliasCommand({
    AccountAlias: alias,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.createAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun createIAMAccountAlias(alias: String) {

    val request = CreateAccountAliasRequest {
        accountAlias = alias
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
```

```
        raise
```

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, cree y elimine los alias de la cuenta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
```

```
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte CreateAccountAlias en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de una política de IAM insertada para un grupo con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo crear una política de IAM insertada para un grupo.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte PutGroupPolicy en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo agregar una política a un grupo

El siguiente comando `put-group-policy` agrega una política al grupo de IAM denominado `Admins`.

```
aws iam put-group-policy \
    --group-name Admins \
    --policy-document file://AdminPolicy.json \
    --policy-name AdminRoot
```

Este comando no genera ninguna salida.

La política se define como un documento JSON en el archivo AdminPolicy.json. (El nombre y la extensión del archivo no son significativos).

Para obtener información, consulte Administración de políticas de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte PutGroupPolicy en la Referencia de comandos de la AWS CLI.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear una política de IAM insertada para un usuario con un SDK AWS

Los siguientes ejemplos de código muestran cómo crear una política de IAM en línea para un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Crear un usuario y asumir un rol

CLI

AWS CLI

Cómo asociar una política a un usuario de IAM

El siguiente comando `put-user-policy` asocia una política al usuario de IAM denominado Bob.

```
aws iam put-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy \
    --policy-document file://AdminPolicy.json
```

Este comando no genera ninguna salida.

La política se define como un documento JSON en el archivo AdminPolicy.json. (El nombre y la extensión del archivo no son significativos).

Para más información, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información acerca de la API, consulte PutUserPolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte PutUserPolicy en la Referencia de la API de AWS SDK for Go.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Creates an inline policy for a specified user.
# @param username [String] The name of the IAM user.
# @param policy_name [String] The name of the policy to create.
# @param policy_document [String] The JSON policy document.
# @return [Boolean]
def create_user_policy(username, policy_name, policy_document)
  @iam_client.put_user_policy({
    user_name: username,
    policy_name: policy_name,
    policy_document: policy_document
  })
  @logger.info("Policy #{policy_name} created for user #{username}.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  false
end
```

- Para obtener información sobre la API, consulte PutUserPolicy en la Referencia de la API de AWS SDK for Ruby.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   func putUserPolicy(policyDocument: String, policyName: String, user:
IAMClientTypes.User) async throws {
        let input = PutUserPolicyInput(
            policyDocument: policyDocument,
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.putUserPolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Para obtener información acerca de la API, consulte PutUserPolicy en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear un perfil de instancia de IAM mediante un AWS SDK

En los siguientes ejemplos de código, se muestra cómo crear un perfil de instancia de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Cree y gestione un servicio resiliente

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <summary>
   /// Create a policy, role, and profile that is associated with instances with
a specified name.
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {
```

```
        var assumeRoleDoc = "{" +
                                    "\"Version\": \"2012-10-17\"," +
                                    "\"Statement\": [{" +
                                         "\"Effect\": \"Allow\"," +
                                         "\"Principal\": {" +
                                         "\"Service\": [" +
                                             "\"ec2.amazonaws.com\"" +
                                         "]" +
                                         "}," +
                                    "\"Action\": \"sts:AssumeRole\"" +
                                    "}]" +
                                "}";

        var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

        var policyArn = "";

        try
        {
            var createPolicyResult = await _amazonIam.CreatePolicyAsync(
                new CreatePolicyRequest
                {
                    PolicyName = policyName,
                    PolicyDocument = policyDocument
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }
```

```
            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
```

```
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

- Para obtener información acerca de la API, consulte CreateInstanceProfile en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo crear un perfil de instancia

El siguiente comando `create-instance-profile` crea un perfil de instancia denominado `Webserver`.

```
aws iam create-instance-profile \
    --instance-profile-name Webserver
```

Salida:

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJMBYC7DLSPEXAMPLE",
        "Roles": [],
        "CreateDate": "2015-03-09T20:33:19.626Z",
        "InstanceProfileName": "Webserver",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/Webserver"
    }
}
```

Para añadir un rol a un perfil de instancia, utilice el comando `add-role-to-instance-profile`.

Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias de Amazon EC2 en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte CreateInstanceProfile en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
```

- Para obtener información acerca de la API, consulte CreateInstanceProfile en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este ejemplo, se crea una política, un rol y un perfil de instancia, luego, se vinculan todos entre sí.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
```

```
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
```

```
                                    use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
```

```
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn
```

- Para obtener información acerca de la API, consulte CreateInstanceProfile en la Referencia de la API de AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminación de un proveedor SAML de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo eliminar un proveedor SAML de AWS Identity and Access Management (IAM).

CLI

AWS CLI

Cómo eliminar un proveedor SAML

En este ejemplo se elimina el proveedor SAML 2.0 de IAM cuyo ARN es `arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider`.

```
aws iam delete-saml-provider \
--saml-provider-arn arn:aws:iam::123456789012:saml-provider/SAMLADFSProvider
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Creación de proveedores de identidad SAML de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteSAMLProvider en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { DeleteSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} providerArn
 * @returns
 */
export const deleteSAMLProvider = async (providerArn) => {
  const command = new DeleteSAMLProviderCommand({
    SAMLProviderArn: providerArn,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener detalles sobre la API, consulte DeleteSAMLProvider en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminación de un grupo de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar un grupo de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obtener detalles sobre la API, consulte DeleteGroup en la Referencia de la API de AWS SDK for JavaScript.

CLI

AWS CLI

Cómo eliminar un grupo de IAM

El siguiente comando `delete-group` elimina un grupo de IAM denominado `MyTestGroup`.

```
aws iam delete-group \
    --group-name MyTestGroup
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Eliminación de un grupo de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteGroup en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { DeleteGroupCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} groupName
 */
export const deleteGroup = async (groupName) => {
  const command = new DeleteGroupCommand({
    GroupName: groupName,
  });

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener detalles sobre la API, consulte DeleteGroup en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminación de una política de grupo de IAM con un SDK de AWS

En el siguiente ejemplo de código se muestra cómo eliminar una política de grupo de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
```

```
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DeleteGroupPolicy en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo eliminar una política de un grupo de IAM

El siguiente comando `delete-group-policy` elimina la política denominada `ExamplePolicy` del grupo denominado `Admins`.

```
aws iam delete-group-policy \
    --group-name Admins \
    --policy-name ExamplePolicy
```

Este comando no genera ninguna salida.

Para ver las políticas asociadas a un grupo, utilice el comando `list-group-policies`.

Para obtener más información, consulte Administración de políticas de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteGroupPolicy en la Referencia de comandos de la AWS CLI.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar una política de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar una política de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura
- Administrar políticas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function iam_delete_policy() {
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}
```

```
  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::deletePolicy(const Aws::String &policyArn,
                               const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeletePolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.DeletePolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting policy with arn " << policyArn << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted policy with arn " << policyArn
```

```
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información acerca de la API, consulte DeletePolicy en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo eliminar la política de IAM

En este ejemplo se elimina la política cuyo ARN es `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam delete-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Políticas y permisos en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.DeletePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeletePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <policyARN>\s

                Where:
                    policyARN - A policy ARN value to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String policyARN = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMPolicy(iam, policyARN);
        iam.close();
    }

    public static void deleteIAMPolicy(IamClient iam, String policyARN) {
        try {
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(policyARN)
                    .build();

            iam.deletePolicy(request);
            System.out.println("Successfully deleted the policy");
```

```
        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine la directiva.

```
import { DeletePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const deletePolicy = (policyArn) => {
  const command = new DeletePolicyCommand({ PolicyArn: policyArn });
  return client.send(command);
};
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {

    val request = DeletePolicyRequest {
        policyArn = policyARNVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def delete_policy(policy_arn):
    """
    Deletes a policy.
```

```
    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Para obtener información sobre la API, consulte DeletePolicy en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
    let input = DeletePolicyInput(
        policyArn: policy.arn
    )
    do {
        _ = try await iamClient.deletePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Para obtener información acerca de la API, consulte DeletePolicy en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar un rol de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Crear un usuario y asumir un rol
- Administrar roles

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
{ RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DeleteRole en AWS SDK for .NETAPI Reference (Referencia de la API de).

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function iecho
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
##############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
```

```
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}
```

- Para obtener información sobre la API, consulte DeleteRole en la Referencia de comandos de la AWS CLI.

CLI

AWS CLI

Cómo eliminar un rol de IAM

El siguiente comando `delete-role` elimina el rol denominado `Test-Role`.

```
aws iam delete-role \
    --role-name Test-Role
```

Este comando no genera ninguna salida.

Antes de poder eliminar un rol, debe eliminarlo de cualquier perfil de instancia (`remove-role-from-instance-profile`), separar cualquier política administrada (`detach-role-policy`) y eliminar cualquier política insertada que esté asociada al rol (`delete-role-policy`).

Para obtener más información, consulte Creación de roles de IAM y Uso de perfiles de instancia en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteRole en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeleteRole en AWS SDK for GoAPI Reference (Referencia de la API de).

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine el rol.

```
import { DeleteRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const deleteRole = (roleName) => {
  const command = new DeleteRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Para obtener información sobre la API, consulte DeleteRole en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def delete_role(role_name):
```

```
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

- Para obtener información sobre la API, consulte DeleteRole en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    begin
      # Detach and delete attached policies
      @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
        response.attached_policies.each do |policy|
          @iam_client.detach_role_policy({
                                           role_name: role_name,
                                           policy_arn: policy.policy_arn
                                         })
```

```
          # Check if the policy is a customer managed policy (not AWS managed)
          unless policy.policy_arn.include?("aws:policy/")
            @iam_client.delete_policy({ policy_arn: policy.policy_arn })
            @logger.info("Deleted customer managed policy
#{policy.policy_name}.")
          end
        end
      end

      # Delete the role
      @iam_client.delete_role({ role_name: role_name })
      @logger.info("Deleted role #{role_name}.")
    rescue Aws::IAM::Errors::ServiceError => e
      @logger.error("Couldn't detach policies and delete role #{role_name}.
Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Para obtener información sobre la API, consulte DeleteRole en AWS SDK for RubyAPI Reference (Referencia de la API de).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_role(client: &iamClient, role: &Role) -> Result<(), iamError>
 {
    let role = role.clone();
    while client
        .delete_role()
        .role_name(role.role_name())
        .send()
```

```
        .await
        .is_err()
    {
        sleep(Duration::from_secs(2)).await;
    }
    Ok(())
}
```

- Para obtener información sobre la API, consulte DeleteRole en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func deleteRole(role: IAMClientTypes.Role) async throws {
        let input = DeleteRoleInput(
            roleName: role.roleName
        )
        do {
            _ = try await iamClient.deleteRole(input: input)
        } catch {
            throw error
        }
    }
```

- Para obtener información acerca de la API, consulte DeleteRole en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar una política de rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar una política de rol de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener detalles sobre la API, consulte DeleteRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

CLI

AWS CLI

Cómo eliminar una política de un rol de IAM

El siguiente comando `delete-role-policy` elimina la política denominada `ExamplePolicy` del rol denominado `Test-Role`.

```
aws iam delete-role-policy \
    --role-name Test-Role \
    --policy-name ExamplePolicy
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Modificación de un rol en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteRolePolicy en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { DeleteRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} roleName
 * @param {string} policyName
 */
export const deleteRolePolicy = (roleName, policyName) => {
  const command = new DeleteRolePolicyCommand({
    RoleName: roleName,
    PolicyName: policyName,
  });
  return client.send(command);
};
```

- Para obtener detalles sobre la API, consulte DeleteRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un certificado de servidor de IAM con un SDK de AWS

El siguiente ejemplo de código muestra cómo eliminar un certificado de servidor de IAM.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::deleteServerCertificate(const Aws::String &certificateName,
                                          const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::DeleteServerCertificateRequest request;
    request.SetServerCertificateName(certificateName);
```

```
    const auto outcome = iam.DeleteServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error deleting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        std::cout << "Successfully deleted server certificate " <<
 certificateName
                  << std::endl;
    }

    return result;
}
```

- Para obtener información sobre la API, consulte DeleteServerCertificate en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo eliminar un certificado de servidor de su cuenta de AWS

El siguiente comando `delete-server-certificate` elimina el certificado de servidor especificado de su cuenta de AWS.

```
aws iam delete-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Este comando no genera ninguna salida.

Para enumerar los certificados de servidor disponibles en su cuenta de AWS, utilice el comando `list-server-certificates`.

Para obtener más información, consulte Administración de certificados de servidor en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteServerCertificate en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine un certificado de servidor.

```
import { DeleteServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 */
export const deleteServerCertificate = (certName) => {
  const command = new DeleteServerCertificateCommand({
    ServerCertificateName: certName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.

- Para obtener información sobre la API, consulte DeleteServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, actualice y elimine certificados del servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
```

```
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Para obtener detalles sobre la API, consulte DeleteServerCertificate en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar un rol de IAM vinculado a un servicio.

CLI

AWS CLI

Cómo eliminar un rol vinculado a un servicio

En el siguiente ejemplo de `delete-service-linked-role`, se elimina el rol vinculado a un servicio especificado que ya no necesita. La eliminación se produce de forma asíncrona. Puede comprobar el estado de la eliminación y confirmar cuando se ha realizado con el comando `get-service-linked-role-deletion-status`.

```
aws iam delete-service-linked-role \
    --role-name AWSServiceRoleForLexBots
```

Salida:

```
{
    "DeletionTaskId": "task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE"
}
```

Para obtener más información, consulte Uso de los roles vinculados a servicios en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteServiceLinkedRole en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeleteServiceLinkedRole en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { DeleteServiceLinkedRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});
```

```
/**
 *
 * @param {string} roleName
 */
export const deleteServiceLinkedRole = (roleName) => {
  const command = new DeleteServiceLinkedRoleCommand({ RoleName: roleName });
  return client.send(command);
};
```

- Para obtener información sobre la API, consulte DeleteServiceLinkedRole en la Referencia de la API de AWS SDK for JavaScript.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  # Deletes a service-linked role.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_service_linked_role(role_name)
    response = @iam_client.delete_service_linked_role(role_name: role_name)
    task_id = response.deletion_task_id
    check_deletion_status(role_name, task_id)
  rescue Aws::Errors::ServiceError => e
    handle_deletion_error(e, role_name)
  end

  private

  # Checks the deletion status of a service-linked role
  #
  # @param role_name [String] The name of the role being deleted
  # @param task_id [String] The task ID for the deletion process
```

```
  def check_deletion_status(role_name, task_id)
    loop do
      response = @iam_client.get_service_linked_role_deletion_status(
        deletion_task_id: task_id)
      status = response.status
      @logger.info("Deletion of #{role_name} #{status}.")
      break if %w[SUCCEEDED FAILED].include?(status)
      sleep(3)
    end
  end

  # Handles deletion error
  #
  # @param e [Aws::Errors::ServiceError] The error encountered during deletion
  # @param role_name [String] The name of the role attempted to delete
  def handle_deletion_error(e, role_name)
    unless e.code == "NoSuchEntity"
      @logger.error("Couldn't delete #{role_name}. Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Para obtener información sobre la API, consulte DeleteServiceLinkedRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_service_linked_role(
    client: &iamClient,
    role_name: &str,
) -> Result<(), iamError> {
```

```
    client
        .delete_service_linked_role()
        .role_name(role_name)
        .send()
        .await?;

    Ok(())
}
```

- Para obtener detalles sobre la API, consulte DeleteServiceLinkedRole en la Referencia de la API del AWS SDK para Rust.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un usuario de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar un usuario de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for .NETAPI Reference (Referencia de la API de).

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iecho
```

```
#
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
#############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

#############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
#############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

#############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
#############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_user"
    echo "Deletes an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user does not exist, we don't want to try to delete it.
  if (! iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name does not exist in the account."
    return 1
  fi

  response=$(aws iam delete-user \
    --user-name "$user_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi
```

```
  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Para obtener información sobre la API, consulte DeleteUser en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteUserRequest request;
    request.SetUserName(userName);
    auto outcome = iam.DeleteUser(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting IAM user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;;
    }
    else {
        std::cout << "Successfully deleted IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
```

- Para obtener información acerca de la API, consulte DeleteUser en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo eliminar un usuario de IAM

El siguiente comando `delete-user` elimina el usuario de IAM denominado Bob de la cuenta actual.

```
aws iam delete-user \
    --user-name Bob
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Eliminación de un usuario de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteUser en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteUser deletes a user.
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
```

```
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for GoAPI Reference (Referencia de la API de).

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteUserRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteUser {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMUser(iam, userName);
        System.out.println("Done");
        iam.close();
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
                    .build();

            iam.deleteUser(request);
            System.out.println("Successfully deleted IAM user " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for Java 2.xAPI Reference (Referencia de la API de).

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine el usuario.

```
import { DeleteUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} name
 */
export const deleteUser = (name) => {
  const command = new DeleteUserCommand({ UserName: name });
  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
```

```
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
};

iam.getUser(params, function (err, data) {
  if (err && err.code === "NoSuchEntity") {
    console.log("User " + process.argv[2] + " does not exist.");
  } else {
    iam.deleteUser(params, function (err, data) {
      if (err) {
        console.log("Error", err);
      } else {
        console.log("Success", data);
      }
    });
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteIAMUser(userNameVal: String) {
```

```
    val request = DeleteUserRequest {
        userName = userNameVal
    }

    // To delete a user, ensure that the user's access keys are deleted first.
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Para obtener información sobre la API, consulte DeleteUser en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise
```

- Para obtener información sobre la API, consulte DeleteUser en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Deletes a user and their associated resources
 #
 # @param user_name [String] The name of the user to delete
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
   user.each do |key|
     @iam_client.delete_access_key({ access_key_id: key.access_key_id,
user_name: user_name })
     @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
   end

   @iam_client.delete_user(user_name: user_name)
   @logger.info("Deleted user '#{user_name}'.")
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error deleting user '#{user_name}': #{e.message}")
 end
```

- Para obtener información sobre la API, consulte DeleteUser en AWS SDK for RubyAPI Reference (Referencia de la API de).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_user(client: &iamClient, user: &User) -> Result<(),
 SdkError<DeleteUserError>> {
    let user = user.clone();
    let mut tries: i32 = 0;
    let max_tries: i32 = 10;

    let response: Result<(), SdkError<DeleteUserError>> = loop {
        match client
            .delete_user()
            .user_name(user.user_name())
            .send()
            .await
        {
            Ok(_) => {
                break Ok(());
            }
            Err(e) => {
                tries += 1;
                if tries > max_tries {
                    break Err(e);
                }
                sleep(Duration::from_secs(2)).await;
            }
        }
    };

    response
}
```

- Para obtener información sobre la API, consulte DeleteUser en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func deleteUser(user: IAMClientTypes.User) async throws {
    let input = DeleteUserInput(
        userName: user.userName
    )
    do {
        _ = try await iamClient.deleteUser(input: input)
    } catch {
        throw error
    }
}
```

- Para obtener información acerca de la API, consulte DeleteUser en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar una clave de acceso de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar una clave de acceso de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Creación de un grupo y adición de un usuario
- Crear un usuario y asumir un rol
- Creación de usuarios de solo lectura, y lectura y escritura
- Administrar claves de acceso

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
    {
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
```

```
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
#
# Returns:
#       0 - If successful.
```

```
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
```

```
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}
```

- Para obtener información de sobre la API, consulte DeleteAccessKey en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::deleteAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
```

```
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);

    auto outcome = iam.DeleteAccessKey(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting access key " << accessKeyID << " from user "
                  << userName << ": " << outcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully deleted access key " << accessKeyID
                  << " for IAM user " << userName << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información acerca de la API, consulte DeleteAccessKey en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo eliminar una clave de acceso para un usuario de IAM

El siguiente comando `delete-access-key` elimina la clave de acceso especificada (ID de clave de acceso y clave de acceso secreta) para el usuario de IAM denominado `Bob`.

```
aws iam delete-access-key \
    --access-key-id AKIDPMS9RO4H3FEXAMPLE \
    --user-name Bob
```

Este comando no genera ninguna salida.

Para enumerar las claves de acceso definidas por un usuario de IAM, utilice el comando `list-access-keys`.

Para obtener más información, consulte Administración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información de sobre la API, consulte DeleteAccessKey en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.DeleteAccessKeyRequest;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccessKey {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <username> <accessKey>\s

                Where:
                    username - The name of the user.\s
                    accessKey - The access key ID for the secret access key you
 want to delete.\s
                """;
```

```
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessKey = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        deleteKey(iam, username, accessKey);
        iam.close();
    }

    public static void deleteKey(IamClient iam, String username, String
 accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine la clave de acceso.

```
import { DeleteAccessKeyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
 */
export const deleteAccessKey = (userName, accessKeyId) => {
  const command = new DeleteAccessKeyCommand({
    AccessKeyId: accessKeyId,
    UserName: userName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  UserName: "USER_NAME",
};

iam.deleteAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteKey(userNameVal: String, accessKey: String) {

    val request = DeleteAccessKeyRequest {
        accessKeyId = accessKey
        userName = userNameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.
```

```
    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Este módulo de ejemplo muestra, crea, desactiva y elimina las claves de acceso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
```

```
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
```

```
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_access_key(
    client: &iamClient,
    user: &User,
```

```
    key: &AccessKey,
) -> Result<(), iamError> {
    loop {
        match client
            .delete_access_key()
            .user_name(user.user_name())
            .access_key_id(key.access_key_id())
            .send()
            .await
        {
            Ok(_) => {
                break;
            }
            Err(e) => {
                println!("Can't delete the access key: {:?}", e);
                sleep(Duration::from_secs(2)).await;
            }
        }
    }
    Ok(())
}
```

- Para obtener información sobre la API, consulte DeleteAccessKey en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func deleteAccessKey(user: IAMClientTypes.User? = nil,
                                key: IAMClientTypes.AccessKey) async throws {
        let userName: String?

        if user != nil {
            userName = user!.userName
        } else {
            userName = nil
        }

        let input = DeleteAccessKeyInput(
            accessKeyId: key.accessKeyId,
            userName: userName
        )
        do {
            _ = try await iamClient.deleteAccessKey(input: input)
        } catch {
            throw error
        }
    }
```

- Para obtener información acerca de la API, consulte DeleteAccessKey en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un alias de cuenta de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar un alias de cuenta de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Administre su cuenta

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::deleteAccountAlias(const Aws::String &accountAlias,
                                     const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DeleteAccountAliasRequest request;
    request.SetAccountAlias(accountAlias);

    const auto outcome = iam.DeleteAccountAlias(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error deleting account alias " << accountAlias << ": "
                  << outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Successfully deleted account alias " << accountAlias <<
                  std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo eliminar un alias de la cuenta

El siguiente comando `delete-account-alias` elimina el alias `mycompany` de la cuenta actual.

```
aws iam delete-account-alias \
    --account-alias mycompany
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Su ID de cuenta y alias de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.DeleteAccountAliasRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DeleteAccountAlias {
    public static void main(String[] args) {
        final String usage = """
```

```
                Usage:
                    <alias>\s

                Where:
                    alias - The account alias to delete.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String alias = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        deleteIAMAccountAlias(iam, alias);
        iam.close();
    }

    public static void deleteIAMAccountAlias(IamClient iam, String alias) {
        try {
            DeleteAccountAliasRequest request =
 DeleteAccountAliasRequest.builder()
                    .accountAlias(alias)
                    .build();

            iam.deleteAccountAlias(request);
            System.out.println("Successfully deleted account alias " + alias);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Done");
    }
}
```

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Elimine el alias de cuenta.

```
import { DeleteAccountAliasCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} alias
 */
export const deleteAccountAlias = (alias) => {
  const command = new DeleteAccountAliasCommand({ AccountAlias: alias });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.deleteAccountAlias({ AccountAlias: process.argv[2] }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {

    val request = DeleteAccountAliasRequest {
        accountAlias = alias
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
}
```

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise
```

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, cree y elimine los alias de la cuenta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end
```

```
  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte DeleteAccountAlias en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar una política de IAM insertada de un usuario con un SDK de AWS

Los siguientes ejemplos de código muestran cómo eliminar una política de IAM insertada de un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Crear un usuario y asumir un rol

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información acerca de la API, consulte DeleteUserPolicy en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo eliminar una política de un usuario de IAM

El siguiente comando `delete-user-policy` elimina la política especificada del usuario de IAM denominado Bob.

```
aws iam delete-user-policy \
    --user-name Bob \
    --policy-name ExamplePolicy
```

Este comando no genera ninguna salida.

Para obtener una lista de las políticas para un usuario de IAM, utilice el comando `list-user-policies`.

Para obtener más información, consulte Creación del usuario de IAM en su cuenta de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información de sobre la API, consulte DeleteUserPolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
```

```
 IamClient *iam.Client
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DeleteUserPolicy en la Referencia de la API de AWS SDK for Go.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  # Deletes a user and their associated resources
  #
  # @param user_name [String] The name of the user to delete
  def delete_user(user_name)
    user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
    user.each do |key|
```

```
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
```

- Para obtener información sobre la API, consulte DeleteUserPolicy en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn delete_user_policy(
    client: &iamClient,
    user: &User,
    policy_name: &str,
) -> Result<(), SdkError<DeleteUserPolicyError>> {
    client
        .delete_user_policy()
        .user_name(user.user_name())
        .policy_name(policy_name)
        .send()
        .await?;

    Ok(())
}
```

- Para obtener información sobre la API, consulte DeleteUserPolicy en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    func deleteUserPolicy(user: IAMClientTypes.User, policyName: String) async
throws {
        let input = DeleteUserPolicyInput(
            policyName: policyName,
            userName: user.userName
        )
        do {
            _ = try await iamClient.deleteUserPolicy(input: input)
        } catch {
            throw error
        }
    }
```

- Para obtener información acerca de la API, consulte DeleteUserPolicy en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminar un perfil de instancia de IAM mediante un AWS SDK

En los siguientes ejemplos de código, se muestra cómo eliminar un perfil de instancia de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Cree y gestione un servicio resiliente

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <summary>
   /// Detaches a role from an instance profile, detaches policies from the
role,
   /// and deletes all the resources.
   /// </summary>
   /// <param name="profileName">The name of the profile to delete.</param>
   /// <param name="roleName">The name of the role to delete.</param>
   /// <returns>Async task.</returns>
   public async Task DeleteInstanceProfile(string profileName, string roleName)
   {
       try
       {
           await _amazonIam.RemoveRoleFromInstanceProfileAsync(
               new RemoveRoleFromInstanceProfileRequest()
               {
                   InstanceProfileName = profileName,
                   RoleName = roleName
               });
           await _amazonIam.DeleteInstanceProfileAsync(
               new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
           var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
```

```
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }
```

- Para obtener información acerca de la API, consulte DeleteInstanceProfile en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo eliminar un perfil de instancia

El siguiente comando `delete-instance-profile` elimina el perfil de instancia denominado `ExampleInstanceProfile`.

```
aws iam delete-instance-profile \
```

```
    --instance-profile-name ExampleInstanceProfile
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Uso de perfiles de instancias en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DeleteInstanceProfile en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Para obtener información acerca de la API, consulte DeleteInstanceProfile en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este ejemplo, se elimina el rol del perfil de instancia, se desvinculan todas las políticas asociadas al rol y se eliminan todos los recursos.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
 created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
 that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
```

```
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )
```

- Para obtener información acerca de la API, consulte DeleteInstanceProfile en la Referencia de la API de AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Desasociar una política de IAM de un rol con un SDK de AWS

Los siguientes ejemplos de código muestran cómo desasociar una política de IAM de un rol.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Crear un usuario y asumir un rol
- Administrar roles

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <summary>
   /// Detach an IAM policy from an IAM role.
   /// </summary>
   /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
   /// <param name="roleName">The name of the IAM role.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
   {
```

```
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
```

```
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi
```

```
  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::DetachRolePolicyRequest detachRequest;
    detachRequest.SetRoleName(roleName);
```

```
    detachRequest.SetPolicyArn(policyArn);

    auto detachOutcome = iam.DetachRolePolicy(detachRequest);
    if (!detachOutcome.IsSuccess()) {
        std::cerr << "Failed to detach policy " << policyArn << " from role "
                  << roleName << ": " << detachOutcome.GetError().GetMessage() <<
                  std::endl;
    }
    else {
        std::cout << "Successfully detached policy " << policyArn << " from role
"
                  << roleName << std::endl;
    }

    return detachOutcome.IsSuccess();
```

- Para obtener información acerca de la API, consulte DetachRolePolicy en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo desasociar una política de un rol

En este ejemplo se elimina la política administrada con la `arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy` de ARN del rol denominado `FedTesterRole`.

```
aws iam detach-role-policy \
    --role-name FedTesterRole \
    --policy-arn arn:aws:iam::123456789012:policy/FederatedTesterAccessPolicy
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Modificación de un rol en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.DetachRolePolicyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DetachRolePolicy {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleName> <policyArn>\s

                Where:
                    roleName - A role name that you can obtain from the AWS
 Management Console.\s
                    policyArn - A policy ARN that you can obtain from the AWS
 Management Console.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String roleName = args[0];
        String policyArn = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();
        detachPolicy(iam, roleName, policyArn);
        System.out.println("Done");
        iam.close();
    }

    public static void detachPolicy(IamClient iam, String roleName, String
 policyArn) {
        try {
            DetachRolePolicyRequest request = DetachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.detachRolePolicy(request);
            System.out.println("Successfully detached policy " + policyArn +
                    " from role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Desasocie la política.

```
import { DetachRolePolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 * @param {string} roleName
 */
export const detachRolePolicy = (policyArn, roleName) => {
  const command = new DetachRolePolicyCommand({
    PolicyArn: policyArn,
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var paramsRoleList = {
  RoleName: process.argv[2],
};

iam.listAttachedRolePolicies(paramsRoleList, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var myRolePolicies = data.AttachedPolicies;
    myRolePolicies.forEach(function (val, index, array) {
      if (myRolePolicies[index].PolicyName === "AmazonDynamoDBFullAccess") {
        var params = {
          PolicyArn: "arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess",
          RoleName: process.argv[2],
        };
        iam.detachRolePolicy(params, function (err, data) {
          if (err) {
            console.log("Unable to detach policy from role", err);
          } else {
            console.log("Policy detached from role successfully");
            process.exit();
          }
        });
      }
    });
  }
```

```
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun detachPolicy(roleNameVal: String, policyArnVal: String) {

    val request = DetachRolePolicyRequest {
        roleName = roleNameVal
        policyArn = policyArnVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Desasocie una política de un rol mediante el objeto Boto3 Policy.

```
def detach_from_role(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Policy(policy_arn).detach_role(RoleName=role_name)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

Desasocie una política de un rol mediante el objeto Boto3 Role.

```
def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
```

```
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
        )
        raise
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este módulo de ejemplo, se enumeran, se crean, se adjuntan y se separan las políticas de roles.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
```

```
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
    raise
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn detach_role_policy(
    client: &iamClient,
    role_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_role_policy()
        .role_name(role_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Para obtener información sobre la API, consulte DetachRolePolicy en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   public func detachRolePolicy(policy: IAMClientTypes.Policy, role:
IAMClientTypes.Role) async throws {
       let input = DetachRolePolicyInput(
           policyArn: policy.arn,
           roleName: role.roleName
       )

       do {
           _ = try await iamClient.detachRolePolicy(input: input)
       } catch {
           throw error
       }
   }
```

- Para obtener detalles sobre la API, consulte DetachRolePolicy en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Desasociar una política de IAM de un usuario con un SDK de AWS

Los siguientes ejemplos de código muestran cómo desasociar una política de IAM de un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de usuarios de solo lectura, y lectura y escritura

CLI

AWS CLI

Cómo desasociar una política de un usuario

En este ejemplo se elimina la política administrada con la `arn:aws:iam::123456789012:policy/TesterPolicy` de ARN del usuario Bob.

```
aws iam detach-user-policy \
    --user-name Bob \
    --policy-arn arn:aws:iam::123456789012:policy/TesterPolicy
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Cambio de los permisos para un usuario de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte DetachUserPolicy en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
```

```
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

- Para obtener información sobre la API, consulte DetachUserPolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Detaches a policy from a user
 #
 # @param user_name [String] The name of the user
 # @param policy_arn [String] The ARN of the policy to detach
 # @return [Boolean] true if the policy was successfully detached, false
otherwise
 def detach_user_policy(user_name, policy_arn)
   @iam_client.detach_user_policy(
     user_name: user_name,
     policy_arn: policy_arn
   )
   @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
   true
```

```
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Error detaching policy: Policy or user does not exist.")
    false
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from user '#{user_name}':
 #{e.message}")
    false
  end
```

- Para detalles sobre la API, consulte DetachUserPolicy en Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn detach_user_policy(
    client: &iamClient,
    user_name: &str,
    policy_arn: &str,
) -> Result<(), iamError> {
    client
        .detach_user_policy()
        .user_name(user_name)
        .policy_arn(policy_arn)
        .send()
        .await?;

    Ok(())
}
```

- Para obtener detalles sobre la API, consulte DetachUserPolicy en la Referencia de la API del AWS SDK para Rust.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Generar un informe de credencial de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo generar un informe de credencial desde IAM para la cuenta actual. Una vez generado el informe, obténgalo mediante la acción GetCredentialReport.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administre su cuenta

CLI

AWS CLI

Cómo generar un informe de credenciales

En el siguiente ejemplo se intenta generar un informe de credenciales para la cuenta de AWS.

```
aws iam generate-credential-report
```

Salida:

```
{
    "State":  "STARTED",
    "Description": "No report exists. Starting a new report generation task"
}
```

Para obtener más información, consulte Obtención de informes de credenciales para su cuenta de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GenerateCredentialReport en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response
```

- Para obtener detalles sobre la API, consulte GenerateCredentialReport en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un informe de credencial de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo obtener el último informe de credencial generado por IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administre su cuenta

CLI

AWS CLI

Cómo obtener un informe de credencial

En este ejemplo se abre el informe devuelto y se envía a la canalización como una matriz de líneas de texto.

```
aws iam get-credential-report
```

Salida:

```
{
    "GeneratedTime":  "2015-06-17T19:11:50Z",
    "ReportFormat": "text/csv"
}
```

Para obtener más información, consulte Obtención de informes de credenciales para su cuenta de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetCredentialReport en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]
```

- Para obtener detalles sobre la API, consulte GetCredentialReport en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un informe de autorización de IAM detallado de la cuenta con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo obtener un informe detallado de autorización de IAM para su cuenta.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administre su cuenta

CLI

AWS CLI

Cómo enumerar usuarios, grupos, roles y políticas de IAM de cuentas de AWS

El siguiente comando `get-account-authorization-details` devuelve información sobre todos los usuarios, grupos, roles y políticas de IAM en la cuenta de AWS.

```
aws iam get-account-authorization-details
```

Salida:

```
{
    "RoleDetailList": [
        {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "RoleId": "AROA1234567890EXAMPLE",
            "CreateDate": "2014-07-30T17:09:20Z",
```

```
            "InstanceProfileList": [
                {
                    "InstanceProfileId": "AIPA1234567890EXAMPLE",
                    "Roles": [
                        {
                            "AssumeRolePolicyDocument": {
                                "Version": "2012-10-17",
                                "Statement": [
                                    {
                                        "Sid": "",
                                        "Effect": "Allow",
                                        "Principal": {
                                            "Service": "ec2.amazonaws.com"
                                        },
                                        "Action": "sts:AssumeRole"
                                    }
                                ]
                            },
                            "RoleId": "AROA1234567890EXAMPLE",
                            "CreateDate": "2014-07-30T17:09:20Z",
                            "RoleName": "EC2role",
                            "Path": "/",
                            "Arn": "arn:aws:iam::123456789012:role/EC2role"
                        }
                    ],
                    "CreateDate": "2014-07-30T17:09:20Z",
                    "InstanceProfileName": "EC2role",
                    "Path": "/",
                    "Arn": "arn:aws:iam::123456789012:instance-profile/EC2role"
                }
            ],
            "RoleName": "EC2role",
            "Path": "/",
            "AttachedManagedPolicies": [
                {
                    "PolicyName": "AmazonS3FullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/AmazonS3FullAccess"
                },
                {
                    "PolicyName": "AmazonDynamoDBFullAccess",
                    "PolicyArn": "arn:aws:iam::aws:policy/
AmazonDynamoDBFullAccess"
                }
            ],
```

```
            "RoleLastUsed": {
                "Region": "us-west-2",
                "LastUsedDate": "2019-11-13T17:30:00Z"
            },
            "RolePolicyList": [],
            "Arn": "arn:aws:iam::123456789012:role/EC2role"
        }
    ],
    "GroupDetailList": [
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "AdministratorAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/AdministratorAccess"
            },
            "GroupName": "Admins",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "CreateDate": "2013-10-14T18:32:24Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": {
                "PolicyName": "PowerUserAccess",
                "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
            },
            "GroupName": "Dev",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Dev",
            "CreateDate": "2013-10-14T18:33:55Z",
            "GroupPolicyList": []
        },
        {
            "GroupId": "AIDA1234567890EXAMPLE",
            "AttachedManagedPolicies": [],
            "GroupName": "Finance",
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:group/Finance",
            "CreateDate": "2013-10-14T18:57:48Z",
            "GroupPolicyList": [
                {
                    "PolicyName": "policygen-201310141157",
                    "PolicyDocument": {
```

```
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Action": "aws-portal:*",
                            "Sid": "Stmt1381777017000",
                            "Resource": "*",
                            "Effect": "Allow"
                        }
                    ]
                }
            }
        ]
    }
],
"UserDetailList": [
    {
        "UserName": "Alice",
        "GroupList": [
            "Admins"
        ],
        "CreateDate": "2013-10-14T18:32:24Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Alice"
    },
    {
        "UserName": "Bob",
        "GroupList": [
            "Admins"
        ],
        "CreateDate": "2013-10-14T18:32:25Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [
            {
                "PolicyName": "DenyBillingAndIAMPolicy",
                "PolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": {
                        "Effect": "Deny",
                        "Action": [
                            "aws-portal:*",
                            "iam:*"
```

```
                        ],
                        "Resource": "*"
                    }
                }
            }
        ],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Bob"
    },
    {
        "UserName": "Charlie",
        "GroupList": [
            "Dev"
        ],
        "CreateDate": "2013-10-14T18:33:56Z",
        "UserId": "AIDA1234567890EXAMPLE",
        "UserPolicyList": [],
        "Path": "/",
        "AttachedManagedPolicies": [],
        "Arn": "arn:aws:iam::123456789012:user/Charlie"
    }
],
"Policies": [
    {
        "PolicyName": "create-update-delete-set-managed-policies",
        "CreateDate": "2015-02-06T19:58:34Z",
        "AttachmentCount": 1,
        "IsAttachable": true,
        "PolicyId": "ANPA1234567890EXAMPLE",
        "DefaultVersionId": "v1",
        "PolicyVersionList": [
            {
                "CreateDate": "2015-02-06T19:58:34Z",
                "VersionId": "v1",
                "Document": {
                    "Version": "2012-10-17",
                    "Statement": {
                        "Effect": "Allow",
                        "Action": [
                            "iam:CreatePolicy",
                            "iam:CreatePolicyVersion",
                            "iam:DeletePolicy",
                            "iam:DeletePolicyVersion",
```

```
                                "iam:GetPolicy",
                                "iam:GetPolicyVersion",
                                "iam:ListPolicies",
                                "iam:ListPolicyVersions",
                                "iam:SetDefaultPolicyVersion"
                            ],
                            "Resource": "*"
                        }
                    },
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/create-update-delete-set-
managed-policies",
            "UpdateDate": "2015-02-06T19:58:34Z"
        },
        {
            "PolicyName": "S3-read-only-specific-bucket",
            "CreateDate": "2015-01-21T21:39:41Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2015-01-21T21:39:41Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "s3:Get*",
                                    "s3:List*"
                                ],
                                "Resource": [
                                    "arn:aws:s3:::example-bucket",
                                    "arn:aws:s3:::example-bucket/*"
                                ]
                            }
                        ]
                    },
```

```
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::123456789012:policy/S3-read-only-specific-
bucket",
            "UpdateDate": "2015-01-21T23:39:41Z"
        },
        {
            "PolicyName": "AmazonEC2FullAccess",
            "CreateDate": "2015-02-06T18:40:15Z",
            "AttachmentCount": 1,
            "IsAttachable": true,
            "PolicyId": "ANPA1234567890EXAMPLE",
            "DefaultVersionId": "v1",
            "PolicyVersionList": [
                {
                    "CreateDate": "2014-10-30T20:59:46Z",
                    "VersionId": "v1",
                    "Document": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Action": "ec2:*",
                                "Effect": "Allow",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "elasticloadbalancing:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "cloudwatch:*",
                                "Resource": "*"
                            },
                            {
                                "Effect": "Allow",
                                "Action": "autoscaling:*",
                                "Resource": "*"
                            }
                        ]
                    },
```

```
                    "IsDefaultVersion": true
                }
            ],
            "Path": "/",
            "Arn": "arn:aws:iam::aws:policy/AmazonEC2FullAccess",
            "UpdateDate": "2015-02-06T18:40:15Z"
        }
    ],
    "Marker": "EXAMPLEkakv9BCuUNFDtxWSyfzetYwEx2ADc8dnzfvERF5S6YMvXKx41t6gCl/
eeaCX3Jo94/bKqezEAg8TEVS99EKFLxm3jtbpl25FDWEXAMPLE",
    "IsTruncated": true
}
```

Para obtener más información, consulte Pautas de auditoría de seguridad de AWS en la Guía del usuario de IAM de AWS.

- Para obtener detalles sobre la API, consulte GetAccountAuthorizationDetails en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
```

```
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

- Para obtener detalles sobre la API, consulte GetAccountAuthorizationDetails en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener la política de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo obtener una política de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Trabajar con la API del creador de políticas de IAM

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// Get information about an IAM policy.
```

```
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for .NET.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::getPolicy(const Aws::String &policyArn,
                            const Aws::Client::ClientConfiguration &clientConfig)
 {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetPolicyRequest request;
    request.SetPolicyArn(policyArn);

    auto outcome = iam.GetPolicy(request);
    if (!outcome.IsSuccess()) {
        std::cerr << "Error getting policy " << policyArn << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        const auto &policy = outcome.GetResult().GetPolicy();
```

```
        std::cout << "Name: " << policy.GetPolicyName() << std::endl <<
                  "ID: " << policy.GetPolicyId() << std::endl << "Arn: " <<
                  policy.GetArn() << std::endl << "Description: " <<
                  policy.GetDescription() << std::endl << "CreateDate: " <<

 policy.GetCreateDate().ToGmtString(Aws::Utils::DateFormat::ISO_8601)
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información acerca de la API, consulte GetPolicy en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo recuperar información sobre una política administrada especificada

En este ejemplo se devuelven detalles sobre la política administrada cuyo ARN es `arn:aws:iam::123456789012:policy/MySamplePolicy`.

```
aws iam get-policy \
    --policy-arn arn:aws:iam::123456789012:policy/MySamplePolicy
```

Salida:

```
{
    "Policy": {
        "PolicyName": "MySamplePolicy",
        "CreateDate": "2015-06-17T19:23;32Z",
        "AttachmentCount": 0,
        "IsAttachable": true,
        "PolicyId": "Z27SI6FQMGNQ2EXAMPLE1",
        "DefaultVersionId": "v1",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:policy/MySamplePolicy",
        "UpdateDate": "2015-06-17T19:23:32Z"
```

```
    }
}
```

Para obtener más información, consulte Políticas y permisos en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
```

```
}
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga la política.

```
import { GetPolicyCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} policyArn
 */
export const getPolicy = (policyArn) => {
  const command = new GetPolicyCommand({
    PolicyArn: policyArn,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  PolicyArn: "arn:aws:iam::aws:policy/AWSLambdaExecute",
};

iam.getPolicy(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data.Policy.Description);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {

    val request = GetPolicyRequest {
        policyArn = policyArnVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API del AWSSDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getPolicy($policyArn)
```

```
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Fetches an IAM policy by its ARN
 # @param policy_arn [String] the ARN of the IAM policy to retrieve
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end
```

- Para obtener información sobre la API, consulte GetPolicy en la Referencia de la API de AWS SDK for Ruby.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func getPolicy(arn: String) async throws -> IAMClientTypes.Policy {
    let input = GetPolicyInput(
        policyArn: arn
    )
    do {
        let output = try await client.getPolicy(input: input)
        guard let policy = output.policy else {
            throw ServiceHandlerError.noSuchPolicy
        }
        return policy
    } catch {
        throw error
    }
}
```

- Para obtener detalles sobre la API, consulte GetPolicy en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener una versión de la política de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo obtener una versión de la política de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en los siguientes ejemplos de código:

- Administrar políticas
- Trabajar con la API del creador de políticas de IAM

CLI

AWS CLI

Cómo recuperar información acerca de la versión especificada de la política administrada especificada

En este ejemplo se devuelve el documento de la política para la versión v2 de la política cuyo ARN es `arn:aws:iam::123456789012:policy/MyManagedPolicy`.

```
aws iam get-policy-version \
    --policy-arn arn:aws:iam::123456789012:policy/MyPolicy \
    --version-id v2
```

Salida:

```
{
    "PolicyVersion": {
        "Document": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Action": "iam:*",
                    "Resource": "*"
                }
            ]
        },
        "VersionId": "v2",
        "IsDefaultVersion": true,
        "CreateDate": "2023-04-11T00:22:54+00:00"
```

```
    }
}
```

Para obtener más información, consulte Políticas y permisos en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetPolicyVersion en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.

    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement
```

- Para obtener detalles sobre la API, consulte GetPolicyVersion en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo obtener un rol de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// Get information about an IAM role.
/// </summary>
/// <param name="roleName">The name of the IAM role to retrieve information
/// for.</param>
/// <returns>The IAM role that was retrieved.</returns>
public async Task<Role> GetRoleAsync(string roleName)
{
    var response = await _IAMService.GetRoleAsync(new GetRoleRequest
    {
        RoleName = roleName,
    });

    return response.Role;
}
```

- Para obtener información acerca de la API, consulte GetRole en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo obtener información acerca de un rol de IAM

El siguiente comando `get-role` obtiene información sobre el rol denominado `Test-Role`.

```
aws iam get-role \
    --role-name Test-Role
```

Salida:

```
{
    "Role": {
        "Description": "Test Role",
        "AssumeRolePolicyDocument":"<URL-encoded-JSON>",
        "MaxSessionDuration": 3600,
        "RoleId": "AROA1234567890EXAMPLE",
        "CreateDate": "2019-11-13T16:45:56Z",
        "RoleName": "Test-Role",
        "Path": "/",
        "RoleLastUsed": {
            "Region": "us-east-1",
            "LastUsedDate": "2019-11-13T17:14:00Z"
        },
        "Arn": "arn:aws:iam::123456789012:role/Test-Role"
    }
}
```

El comando muestra la política de confianza asociada al rol. Utilice el comando `list-role-policies` para enumerar las políticas de permisos asociadas al rol.

Para más información, consulte Creación de roles de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetRole en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga el rol.

```
import { GetRoleCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} roleName
 */
export const getRole = (roleName) => {
  const command = new GetRoleCommand({
    RoleName: roleName,
  });

  return client.send(command);
};
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_role(role_name):
    """
    Gets a role by name.

    :param role_name: The name of the role to retrieve.
    :return: The specified role.
    """
    try:
        role = iam.Role(role_name)
        role.load()  # calls GetRole to load attributes
        logger.info("Got role with arn %s.", role.arn)
    except ClientError:
        logger.exception("Couldn't get role named %s.", role_name)
        raise
    else:
        return role
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
                               role_name: name,
                             }).role
  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  role
end
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn get_role(
    client: &iamClient,
    role_name: String,
) -> Result<GetRoleOutput, SdkError<GetRoleError>> {
    let response = client.get_role().role_name(role_name).send().await?;
    Ok(response)
}
```

- Para obtener información sobre la API, consulte GetRole en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func getRole(name: String) async throws -> IAMClientTypes.Role {
```

```
        let input = GetRoleInput(
            roleName: name
        )
        do {
            let output = try await client.getRole(input: input)
            guard let role = output.role else {
                throw ServiceHandlerError.noSuchRole
            }
            return role
        } catch {
            throw error
        }
    }
```

- Para obtener detalles sobre la API, consulte GetRole en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un certificado de servidor de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo obtener un certificado de servidor de IAM.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::getServerCertificate(const Aws::String &certificateName,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetServerCertificateRequest request;
```

```
    request.SetServerCertificateName(certificateName);

    auto outcome = iam.GetServerCertificate(request);
    bool result = true;
    if (!outcome.IsSuccess()) {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error getting server certificate " << certificateName
 <<
                      ": " << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << certificateName
                      << "' not found." << std::endl;
        }
    }
    else {
        const auto &certificate = outcome.GetResult().GetServerCertificate();
        std::cout << "Name: " <<

 certificate.GetServerCertificateMetadata().GetServerCertificateName()
                  << std::endl << "Body: " << certificate.GetCertificateBody() <<
                  std::endl << "Chain: " << certificate.GetCertificateChain() <<
                  std::endl;
    }

    return result;
}
```

- Para obtener información sobre la API, consulte GetServerCertificate en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo obtener detalles sobre un certificado de servidor en su cuenta de AWS

El siguiente comando `get-server-certificate` recupera todos los detalles sobre el certificado de servidor especificado en su cuenta de AWS.

```
aws iam get-server-certificate \
    --server-certificate-name myUpdatedServerCertificate
```

Salida:

```
{
    "ServerCertificate": {
        "ServerCertificateMetadata": {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        "CertificateBody": "-----BEGIN CERTIFICATE-----
            MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
            VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
            b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
            BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
            MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
            VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
            b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
            YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
            21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
            rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
            Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
            nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
            FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
            NYiytVbZPQUQ5Yaxu2jXnimvrszlaEXAMPLE=-----END CERTIFICATE-----",
        "CertificateChain": "-----BEGIN CERTIFICATE-----\nMIICiTCCAfICCQD6md
            7oRw0uXOjANBgkqhkiG9w0BAqQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgT
            AldBMRAwDgYDVQQHEwdTZWF0drGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAs
            TC0lBTSBDb25zb2xlMRIwEAYDVsQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQ
            jb20wHhcNMTEwNDI1MjA0NTIxWhtcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
            MCVVMxCzAJBgNVBAgTAldBMRAwDgsYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
            WF6b24xFDASBgNVBAsTC0lBTSBDb2d5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
            HzAdBgkqhkiG9w0BCQEWEG5vb25lQGfFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
            BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIgWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
            k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8mh9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
            ITxOUSQv7c7ugFFDzQGBzZswY6786m86gjpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
            AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCku4nUhVVxYUntneD9+h8Mg9q6q+auN
```

```
            KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FlkbFFBjvSfpJIlJ00zbhNYS5f6Guo
            EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjS;TbNYiytVbZPQUQ5Yaxu2jXnimvw
            3rrszlaEWEG5vb25lQGFtsYXpvbiEXAMPLE=\n-----END CERTIFICATE-----"
    }
}
```

Para enumerar los certificados de servidor disponibles en su cuenta de AWS, utilice el comando `list-server-certificates`.

Para obtener más información, consulte Administración de certificados de servidor en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetServerCertificate en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga un certificado de servidor.

```
import { GetServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} certName
 * @returns
 */
export const getServerCertificate = async (certName) => {
  const command = new GetServerCertificateCommand({
    ServerCertificateName: certName,
  });
```

```
  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte GetServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getServerCertificate(
  { ServerCertificateName: "CERTIFICATE_NAME" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data);
    }
  }
);
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.

- Para obtener detalles sobre la API, consulte GetServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtención de un estado de eliminación de un rol de IAM vinculado a un servicio con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo obtener un estado de eliminación de un rol de vinculado a un servicio de AWS Identity and Access Management (IAM).

CLI

AWS CLI

Cómo comprobar el estado de una solicitud de eliminación de un rol vinculado a un servicio

En el siguiente ejemplo de `get-service-linked-role-deletion-status`, se muestra el estado de una solicitud anterior para eliminar un rol vinculado a un servicio. La operación de eliminación se produce de forma asíncrona. Al realizar la solicitud, se obtiene un valor de `DeletionTaskId` que proporciona como un parámetro para este comando.

```
aws iam get-service-linked-role-deletion-status \
    --deletion-task-id task/aws-service-role/lex.amazonaws.com/
AWSServiceRoleForLexBots/1a2b3c4d-1234-abcd-7890-abcdeEXAMPLE
```

Salida:

```
{
"Status": "SUCCEEDED"
}
```

Para obtener más información, consulte Uso de roles vinculados a servicios en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetServiceLinkedRoleDeletionStatus en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import {
  GetServiceLinkedRoleDeletionStatusCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} deletionTaskId
 */
export const getServiceLinkedRoleDeletionStatus = (deletionTaskId) => {
  const command = new GetServiceLinkedRoleDeletionStatusCommand({
    DeletionTaskId: deletionTaskId,
  });

  return client.send(command);
};
```

- Para obtener detalles sobre la API, consulte GetServiceLinkedRoleDeletionStatus en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un resumen del uso de la cuenta de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo obtener un resumen del uso de la cuenta desde IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administre su cuenta

CLI

AWS CLI

Cómo obtener información sobre el uso de la entidad de IAM y las cuotas de IAM en la cuenta actual

El siguiente comando `get-account-summary` devuelve información sobre el uso actual de la entidad de IAM y las cuotas actuales de entidades de IAM en la cuenta.

```
aws iam get-account-summary
```

Salida:

```
{
    "SummaryMap": {
        "UsersQuota": 5000,
        "GroupsQuota": 100,
        "InstanceProfiles": 6,
        "SigningCertificatesPerUserQuota": 2,
        "AccountAccessKeysPresent": 0,
        "RolesQuota": 250,
        "RolePolicySizeQuota": 10240,
        "AccountSigningCertificatesPresent": 0,
        "Users": 27,
        "ServerCertificatesQuota": 20,
        "ServerCertificates": 0,
        "AssumeRolePolicySizeQuota": 2048,
        "Groups": 7,
        "MFADevicesInUse": 1,
        "Roles": 3,
        "AccountMFAEnabled": 1,
        "MFADevices": 3,
        "GroupsPerUserQuota": 10,
        "GroupPolicySizeQuota": 5120,
        "InstanceProfilesQuota": 100,
        "AccessKeysPerUserQuota": 2,
```

```
        "Providers": 0,
        "UserPolicySizeQuota": 2048
    }
}
```

Para obtener más información sobre las limitaciones de las entidades, consulte Cuotas de IAM y AWS STS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetAccountSummary en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
        raise
    else:
        return summary.summary_map
```

- Para obtener detalles sobre la API, consulte GetAccountSummary en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un usuario de IAM con un SDK de AWS

En los siguientes ejemplos de código, se muestra cómo obtener un usuario de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }
```

- Para obtener información de la API, consulte GetUser en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)
```

```
  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}
```

- Para obtener información sobre la API, consulte GetUser en la Referencia de comandos de la AWS CLI.

CLI

AWS CLI

Cómo obtener información acerca de un usuario de IAM

El siguiente comando `get-user` obtiene información sobre el usuario de IAM denominado `Paulo`.

```
aws iam get-user \
    --user-name Paulo
```

Salida:

```
{
    "User": {
        "UserName": "Paulo",
        "Path": "/",
        "CreateDate": "2019-09-21T23:03:13Z",
        "UserId": "AIDA123456789EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/Paulo"
    }
}
```

Para obtener más información, consulte Administración de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetUser en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
```

```
 } else {
  user = result.User
 }
 return user, err
}
```

- Para obtener información de la API, consulte GetUser en la Referencia de la API de AWS SDK for Go.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Retrieves a user's details
 #
 # @param user_name [String] The name of the user to retrieve
 # @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
 def get_user(user_name)
   response = @iam_client.get_user(user_name: user_name)
   response.user
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("User '#{user_name}' not found.")
   nil
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error retrieving user '#{user_name}': #{e.message}")
   nil
 end
```

- Para obtener detalles de la API, consulte GetItem en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener datos sobre el último uso de una clave de acceso de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo obtener datos sobre el último uso de una clave de acceso de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Administrar claves de acceso

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::accessKeyLastUsed(const Aws::String &secretKeyID,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::GetAccessKeyLastUsedRequest request;

    request.SetAccessKeyId(secretKeyID);
```

```
    Aws::IAM::Model::GetAccessKeyLastUsedOutcome outcome =
 iam.GetAccessKeyLastUsed(
            request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error querying last used time for access key " <<
                  secretKeyID << ":" << outcome.GetError().GetMessage() <<
 std::endl;
    }
    else {
        Aws::String lastUsedTimeString =
                outcome.GetResult()
                        .GetAccessKeyLastUsed()
                        .GetLastUsedDate()
                        .ToGmtString(Aws::Utils::DateFormat::ISO_8601);
        std::cout << "Access key " << secretKeyID << " last used at time " <<
                  lastUsedTimeString << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte GetAccessKeyLastUsed en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo recuperar información acerca de cuándo se utilizó por última vez la clave de acceso especificada

En el siguiente ejemplo se recupera información acerca de cuándo se utilizó por última vez la clave de acceso ABCDEXAMPLE.

```
aws iam get-access-key-last-used \
    --access-key-id ABCDEXAMPLE
```

Salida:

```
{
    "UserName":  "Bob",
    "AccessKeyLastUsed": {
        "Region": "us-east-1",
        "ServiceName": "iam",
        "LastUsedDate": "2015-06-16T22:45:00Z"
    }
}
```

Para obtener más información, consulte Administración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener más información sobre la API, consulte GetAccessKeyLastUsed en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga una clave de acceso.

```
import { GetAccessKeyLastUsedCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} accessKeyId
 */
export const getAccessKeyLastUsed = async (accessKeyId) => {
  const command = new GetAccessKeyLastUsedCommand({
    AccessKeyId: accessKeyId,
  });

  const response = await client.send(command);
```

```
  if (response.AccessKeyLastUsed?.LastUsedDate) {
    console.log(`
    ${accessKeyId} was last used by ${response.UserName} via
    the ${response.AccessKeyLastUsed.ServiceName} service on
    ${response.AccessKeyLastUsed.LastUsedDate.toISOString()}
    `);
  }

  return response;
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte GetAccessKeyLastUsed en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.getAccessKeyLastUsed(
  { AccessKeyId: "ACCESS_KEY_ID" },
  function (err, data) {
    if (err) {
      console.log("Error", err);
    } else {
      console.log("Success", data.AccessKeyLastUsed);
    }
```

```
  }
);
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte GetAccessKeyLastUsed en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
```

```
        return response
```

- Para obtener detalles sobre la API, consulte GetAccessKeyLastUsed en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener la política de contraseñas de la cuenta de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo obtener la política de contraseñas de la cuenta de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }
```

- Para obtener información acerca de la API, consulte GetAccountPasswordPolicy en la Referencia de la APID deAWS SDK for .NET.

CLI

AWS CLI

Cómo ver la política de contraseñas de la cuenta actual

El siguiente comando `get-account-password-policy` muestra detalles sobre la política de contraseñas de la cuenta actual.

```
aws iam get-account-password-policy
```

Salida:

```
{
    "PasswordPolicy": {
        "AllowUsersToChangePassword": false,
        "RequireLowercaseCharacters": false,
        "RequireUppercaseCharacters": false,
        "MinimumPasswordLength": 8,
        "RequireNumbers": true,
        "RequireSymbols": true
    }
}
```

Si no se ha definido una política de contraseñas para la cuenta, el comando devuelve un error `NoSuchEntity`.

Para obtener más información, consulte Configuración de una política de contraseñas de cuentas para los usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en AWS SDK for GoAPI Reference (Referencia de la API de).

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga la política de contraseñas de la cuenta.

```
import {
  GetAccountPasswordPolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const getAccountPasswordPolicy = async () => {
  const command = new GetAccountPasswordPolicyCommand({});

  const response = await client.send(command);
  console.log(response.PasswordPolicy);
  return response;
};
```

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en AWS SDK for PHPAPI Reference (Referencia de la API de).

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def print_password_policy():
    """
    Prints the password policy for the account.
    """
    try:
        pw_policy = iam.AccountPasswordPolicy()
        print("Current account password policy:")
        print(
            f"\tallow_users_to_change_password:
 {pw_policy.allow_users_to_change_password}"
        )
        print(f"\texpire_passwords: {pw_policy.expire_passwords}")
        print(f"\thard_expiry: {pw_policy.hard_expiry}")
        print(f"\tmax_password_age: {pw_policy.max_password_age}")
        print(f"\tminimum_password_length: {pw_policy.minimum_password_length}")
        print(f"\tpassword_reuse_prevention:
 {pw_policy.password_reuse_prevention}")
        print(
```

```
            f"\trequire_lowercase_characters:
 {pw_policy.require_lowercase_characters}"
        )
        print(f"\trequire_numbers: {pw_policy.require_numbers}")
        print(f"\trequire_symbols: {pw_policy.require_symbols}")
        print(
            f"\trequire_uppercase_characters:
 {pw_policy.require_uppercase_characters}"
        )
        printed = True
    except ClientError as error:
        if error.response["Error"]["Code"] == "NoSuchEntity":
            print("The account does not have a password policy set.")
        else:
            logger.exception("Couldn't get account password policy.")
            raise
    else:
        return printed
```

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
```

```
    @logger = logger
    @logger.progname = "IAMPolicyManager"
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    begin
      response = @iam_client.get_account_password_policy
      @logger.info("The account password policy is:
 #{response.password_policy.to_h}")
    rescue Aws::IAM::Errors::NoSuchEntity
      @logger.info("The account does not have a password policy.")
    rescue Aws::Errors::ServiceError => e
      @logger.error("Couldn't print the account password policy. Error: #{e.code}
 - #{e.message}")
      raise
    end
  end
end
```

- Para obtener información sobre la API, consulte GetAccountPasswordPolicy en AWS SDK for RubyAPI Reference (Referencia de la API de).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
```

```
}
```

- Para obtener detalles sobre la API, consulte GetAccountPasswordPolicy en la Referencia de la API del AWS SDK para Rust.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar proveedores de SAML para IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar proveedores de SAML para IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }
```

- Para obtener información acerca de la API, consulte ListSAMLProviders en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo enumerar los proveedores de SAML en la cuenta de AWS

En este ejemplo se recupera la lista de proveedores de SAML 2.0 creada en la cuenta actual de AWS.

```
aws iam list-saml-providers
```

Salida:

```
{
    "SAMLProviderList": [
        {
            "Arn": "arn:aws:iam::123456789012:saml-provider/SAML-ADFS",
            "ValidUntil": "2015-06-05T22:45:14Z",
            "CreateDate": "2015-06-05T22:45:14Z"
        }
    ]
}
```

Para obtener más información, consulte Creación de proveedores de identidad SAML de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los proveedores de SAML.

```
import { ListSAMLProvidersCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

export const listSamlProviders = async () => {
  const command = new ListSAMLProvidersCommand({});

  const response = await client.send(command);
  console.log(response);
  return response;
};
```

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_saml_providers(count):
    """
    Lists the SAML providers for the account.

    :param count: The maximum number of providers to list.
    """
    try:
        found = 0
        for provider in iam.saml_providers.limit(count):
            logger.info("Got SAML provider %s.", provider.arn)
            found += 1
        if found == 0:
            logger.info("Your account has no SAML providers.")
    except ClientError:
        logger.exception("Couldn't list SAML providers.")
        raise
```

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
class SamlProviderLister
  # Initializes the SamlProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Para obtener información sobre la API, consulte ListSAMLProviders en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Para obtener detalles sobre la API, consulte ListSAMLProviders en la Referencia de la API del AWS SDK para Rust.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar las claves de acceso de IAM de un usuario con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar las claves de acceso de IAM de un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administrar claves de acceso

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_access_keys
#
# This function lists the access keys for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#
# Returns:
#       access_key_ids
#     And:
#       0 - If successful.
#       1 - If it fails.
##############################################################################
function iam_list_access_keys() {

  # bashsupport disable=BP5008
```

```
 function usage() {
   echo "function iam_list_access_keys"
   echo "Lists the AWS Identity and Access Management (IAM) access key IDs for
the specified user."
   echo "  -u user_name   The name of the IAM user."
   echo ""
 }

 local user_name response
 local option OPTARG # Required to use getopts command in a function.
 # Retrieve the calling parameters.
 while getopts "u:h" option; do
   case "${option}" in
     u) user_name="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$user_name" ]]; then
   errecho "ERROR: You must provide a username with the -u parameter."
   usage
   return 1
 fi

 response=$(aws iam list-access-keys \
   --user-name "$user_name" \
   --output text \
   --query 'AccessKeyMetadata[].AccessKeyId')

 local error_code=${?}

 if [[ $error_code -ne 0 ]]; then
   aws_cli_error_log $error_code
   errecho "ERROR: AWS reports list-access-keys operation failed.$response"
   return 1
```

```
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::listAccessKeys(const Aws::String &userName,
                                 const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccessKeysRequest request;
    request.SetUserName(userName);

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListAccessKeys(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list access keys for user " << userName
                      << ": " << outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "UserName" <<
                      std::setw(30) << "KeyID" << std::setw(20) << "Status" <<
```

```
                        std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &keys = outcome.GetResult().GetAccessKeyMetadata();
        const Aws::String DATE_FORMAT = "%Y-%m-%d";

        for (const auto &key: keys) {
            Aws::String statusString =
                    Aws::IAM::Model::StatusTypeMapper::GetNameForStatusType(
                            key.GetStatus());
            std::cout << std::left << std::setw(32) << key.GetUserName() <<
                      std::setw(30) << key.GetAccessKeyId() << std::setw(20) <<
                      statusString << std::setw(20) <<
                      key.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
 std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Para obtener información acerca de la API, consulte ListAccessKeys en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo enumerar los ID de las claves de acceso de un usuario de IAM

El siguiente comando `list-access-keys` muestra los ID de las claves de acceso del usuario de IAM denominado Bob.

```
aws iam list-access-keys \
```

```
    --user-name Bob
```

Salida:

```
{
    "AccessKeyMetadata": [
        {
            "UserName": "Bob",
            "Status": "Active",
            "CreateDate": "2013-06-04T18:17:34Z",
            "AccessKeyId": "AKIAIOSFODNN7EXAMPLE"
        },
        {
            "UserName": "Bob",
            "Status": "Inactive",
            "CreateDate": "2013-06-06T20:42:26Z",
            "AccessKeyId": "AKIAI44QH8DHBEXAMPLE"
        }
    ]
}
```

No puede enumerar las claves de acceso secretas para los usuarios de IAM. Si se pierden las claves de acceso secretas, debe crear nuevas claves de acceso mediante el comando `create-access-keys`.

Para obtener más información, consulte Administración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.AccessKeyMetadata;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccessKeysRequest;
import software.amazon.awssdk.services.iam.model.ListAccessKeysResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccessKeys {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <userName>\s

                Where:
                    userName - The name of the user for which access keys are
 retrieved.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userName = args[0];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listKeys(iam, userName);
        System.out.println("Done");
        iam.close();
    }
```

```
   public static void listKeys(IamClient iam, String userName) {
       try {
           boolean done = false;
           String newMarker = null;

           while (!done) {
               ListAccessKeysResponse response;

               if (newMarker == null) {
                   ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                           .userName(userName)
                           .build();

                   response = iam.listAccessKeys(request);

               } else {
                   ListAccessKeysRequest request =
ListAccessKeysRequest.builder()
                           .userName(userName)
                           .marker(newMarker)
                           .build();

                   response = iam.listAccessKeys(request);
               }

               for (AccessKeyMetadata metadata : response.accessKeyMetadata()) {
                   System.out.format("Retrieved access key %s",
metadata.accessKeyId());
               }

               if (!response.isTruncated()) {
                   done = true;
               } else {
                   newMarker = response.marker();
               }
           }

       } catch (IamException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

```
}
```

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere las claves de acceso.

```
import { ListAccessKeysCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} userName
 */
export async function* listAccessKeys(userName) {
  const command = new ListAccessKeysCommand({
    MaxItems: 5,
    UserName: userName,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListAccessKeysCommandOutput |
 undefined}
   */
  let response = await client.send(command);
```

```
  while (response?.AccessKeyMetadata?.length) {
    for (const key of response.AccessKeyMetadata) {
      yield key;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccessKeysCommand({
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 5,
```

```
  UserName: "IAM_USER_NAME",
};

iam.listAccessKeys(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun listKeys(userNameVal: String?) {

    val request = ListAccessKeysRequest {
        userName = userNameVal
    }
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys
```

- Para obtener información sobre la API, consulte ListAccessKeys en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Este módulo de ejemplo muestra, crea, desactiva y elimina las claves de acceso.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
```

```
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create
more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end

  # Deactivates an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def deactivate_access_key(user_name, access_key_id)
    @iam_client.update_access_key(
      user_name: user_name,
      access_key_id: access_key_id,
      status: "Inactive"
    )
    true
  rescue StandardError => e
    @logger.error("Error deactivating access key: #{e.message}")
    false
  end

  # Deletes an access key
  #
  # @param user_name [String] The name of the user.
  # @param access_key_id [String] The ID for the access key.
  # @return [Boolean]
  def delete_access_key(user_name, access_key_id)
    @iam_client.delete_access_key(
      user_name: user_name,
      access_key_id: access_key_id
    )
    true
  rescue StandardError => e
    @logger.error("Error deleting access key: #{e.message}")
```

```
    false
  end
end
```

- Para obtener detalles sobre la API, consulte ListAccessKeys en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar los alias de cuenta de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar los alias de cuentas de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Administre su cuenta

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool
AwsDoc::IAM::listAccountAliases(const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListAccountAliasesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
```

```
        auto outcome = iam.ListAccountAliases(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list account aliases: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        const auto &aliases = outcome.GetResult().GetAccountAliases();
        if (!header) {
            if (aliases.size() == 0) {
                std::cout << "Account has no aliases" << std::endl;
                break;
            }
            std::cout << std::left << std::setw(32) << "Alias" << std::endl;
            header = true;
        }

        for (const auto &alias: aliases) {
            std::cout << std::left << std::setw(32) << alias << std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo enumerar los alias de una cuenta

El siguiente comando `list-account-aliases` enumera los alias de la cuenta actual.

```
aws iam list-account-aliases
```

Salida:

```
{
    "AccountAliases": [
    "mycompany"
    ]
}
```

Para obtener más información, consulte Su ID de cuenta y alias de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListAccountAliasesResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListAccountAliases {
```

```
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAliases(iam);
        System.out.println("Done");
        iam.close();
    }

    public static void listAliases(IamClient iam) {
        try {
            ListAccountAliasesResponse response = iam.listAccountAliases();
            for (String alias : response.accountAliases()) {
                System.out.printf("Retrieved account alias %s", alias);
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los alias de cuenta.

```
import { ListAccountAliasesCommand, IAMClient } from "@aws-sdk/client-iam";
```

```
const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listAccountAliases() {
  const command = new ListAccountAliasesCommand({ MaxItems: 5 });

  let response = await client.send(command);

  while (response.AccountAliases?.length) {
    for (const alias of response.AccountAliases) {
      yield alias;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAccountAliasesCommand({
          Marker: response.Marker,
          MaxItems: 5,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listAccountAliases({ MaxItems: 10 }, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun listAliases() {

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API del AWSSDK para Kotlin).

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
```

```
    else:
        return response["AccountAliases"]
```

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, cree y elimine los alias de la cuenta.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("
 #{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte ListAccountAliases en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar grupos de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar los grupos de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }
```

- Para obtener información acerca de la API, consulte ListGroups en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo enumerar los grupos de IAM de la cuenta actual

El siguiente comando `list-groups` enumera los grupos de IAM de la cuenta actual.

```
aws iam list-groups
```

Salida:

```
{
    "Groups": [
        {
            "Path": "/",
            "CreateDate": "2013-06-04T20:27:27.972Z",
            "GroupId": "AIDACKCEVSQ6C2EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/Admins",
            "GroupName": "Admins"
        },
        {
            "Path": "/",
            "CreateDate": "2013-04-16T20:30:42Z",
            "GroupId": "AIDGPMS9RO4H3FEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:group/S3-Admins",
            "GroupName": "S3-Admins"
        }
    ]
}
```

Para obtener más información, consulte Administración de grupos de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListGroups en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// GroupWrapper encapsulates AWS Identity and Access Management (IAM) group
 actions
// used in the examples.
// It contains an IAM service client that is used to perform group actions.
type GroupWrapper struct {
 IamClient *iam.Client
}



// ListGroups lists up to maxGroups number of groups.
func (wrapper GroupWrapper) ListGroups(maxGroups int32) ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(context.TODO(),
 &iam.ListGroupsInput{
  MaxItems: aws.Int32(maxGroups),
 })
 if err != nil {
  log.Printf("Couldn't list groups. Here's why: %v\n", err)
 } else {
  groups = result.Groups
 }
 return groups, err
}
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los grupos.

```
import { ListGroupsCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 */
export async function* listGroups() {
  const command = new ListGroupsCommand({
    MaxItems: 10,
  });

  let response = await client.send(command);

  while (response.Groups?.length) {
    for (const group of response.Groups) {
      yield group;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListGroupsCommand({
          Marker: response.Marker,
          MaxItems: 10,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listGroups($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listGroupsArguments = [];
        if ($pathPrefix) {
            $listGroupsArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listGroupsArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listGroupsArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listGroups($listGroupsArguments);
    }
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_groups(count):
    """
    Lists the specified number of groups for the account.

    :param count: The number of groups to list.
    """
    try:
        for group in iam.groups.limit(count):
            logger.info("Group: %s", group.name)
    except ClientError:
        logger.exception("Couldn't list groups for the account.")
        raise
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# A class to manage IAM operations via the AWS SDK client
```

```
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_groups(
    client: &iamClient,
    path_prefix: Option<String>,
```

```
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListGroupsOutput, SdkError<ListGroupsError>> {
    let response = client
        .list_groups()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Para obtener información sobre la API, consulte ListGroups en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func listGroups() async throws -> [String] {
        var groupList: [String] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
```

```
            let input = ListGroupsInput(marker: marker)
            let output = try await client.listGroups(input: input)

            guard let groups = output.groups else {
                return groupList
            }

            for group in groups {
                if let name = group.groupName {
                    groupList.append(name)
                }
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return groupList
    }
```

- Para obtener detalles sobre la API, consulte ListGroups en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar políticas insertadas para un rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar políticas insertadas para un rol de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
```

```
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }
```

- Para obtener información sobre la API, consulte ListRolePolicies en API Reference (Referencia de la API de AWS SDK for .NET).

CLI

AWS CLI

Cómo enumerar políticas asociadas a un rol de IAM

El siguiente comando `list-role-policies` enumera los nombres de las políticas de permisos para el rol de IAM especificado.

```
aws iam list-role-policies \
    --role-name Test-Role
```

Salida:

```
{
    "PolicyNames": [
        "ExamplePolicy"
```

```
    ]
}
```

Para ver la política de confianza asociada a un rol, utilice el comando `get-role`. Para ver los detalles de una política de permisos, utilice el comando `get-role-policy`.

Para más información, consulte Creación de roles de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListRolePolicies en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
```

```
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Para obtener información sobre la API, consulte ListRolePolicies en API Reference (Referencia de la API de AWS SDK for Go).

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere las políticas.

```
import { ListRolePoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 * @param {string} roleName
 */
export async function* listRolePolicies(roleName) {
  const command = new ListRolePoliciesCommand({
    RoleName: roleName,
    MaxItems: 10,
  });
```

```
  let response = await client.send(command);

  while (response.PolicyNames?.length) {
    for (const policyName of response.PolicyNames) {
      yield policyName;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolePoliciesCommand({
          RoleName: roleName,
          MaxItems: 10,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información sobre la API, consulte ListRolePolicies en API Reference (Referencia de la API de AWS SDK for JavaScript).

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listRolePolicies($roleName, $marker = "", $maxItems = 0)
    {
        $listRolePoliciesArguments = ['RoleName' => $roleName];
```

```
        if ($marker) {
            $listRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->customWaiter(function () use ($listRolePoliciesArguments) {
            return $this->iamClient-
>listRolePolicies($listRolePoliciesArguments);
        });
    }
```

- Para obtener información sobre la API, consulte ListRolePolicies en API Reference (Referencia de la API de AWS SDK for PHP).

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_policies(role_name):
    """
    Lists inline policies for a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.policies.all():
            logger.info("Got inline policy %s.", policy.name)
    except ClientError:
        logger.exception("Couldn't list inline policies for %s.", role_name)
        raise
```

- Para obtener información sobre la API, consulte ListRolePolicies en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Para obtener información sobre la API, consulte ListRolePolicies en API Reference (Referencia de la API de AWS SDK for Ruby).

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_role_policies(
    client: &iamClient,
    role_name: &str,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolePoliciesOutput, SdkError<ListRolePoliciesError>> {
    let response = client
        .list_role_policies()
        .role_name(role_name)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Para obtener información sobre la API, consulte ListRolePolicies en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func listRolePolicies(role: String) async throws -> [String] {
        var policyList: [String] = []
        var marker: String? = nil
```

```
        var isTruncated: Bool

        repeat {
            let input = ListRolePoliciesInput(
                marker: marker,
                roleName: role
            )
            let output = try await client.listRolePolicies(input: input)

            guard let policies = output.policyNames else {
                return policyList
            }

            for policy in policies {
                policyList.append(policy)
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Para obtener detalles sobre la API, consulte ListRolePolicies en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar políticas de IAM insertadas para un rol de IAM con un AWS SDK

En los siguientes ejemplos de código se muestra cómo enumerar políticas de IAM insertadas para un usuario.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

CLI

AWS CLI

Cómo enumerar las políticas de un usuario de IAM

El siguiente comando `list-user-policies` enumera las políticas asociadas al usuario de IAM denominado Bob.

```
aws iam list-user-policies \
    --user-name Bob
```

Salida:

```
{
    "PolicyNames": [
        "ExamplePolicy",
        "TestPolicy"
    ]
}
```

Para obtener más información, consulte Creación de un usuario de IAM en su cuenta de AWS en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListUserPolicies en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
```

```
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}
```

- Para obtener detalles sobre la API, consulte ListRolePolicies en la Referencia de la API de AWS SDK for Go.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar políticas de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar las políticas de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Administrar políticas

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for .NET.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::listPolicies(const Aws::Client::ClientConfiguration
 &clientConfig) {
    const Aws::String DATE_FORMAT("%Y-%m-%d");
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListPoliciesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListPolicies(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam policies: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(64) << "Description" << std::setw(12) <<
                      "CreateDate" << std::endl;
            header = true;
        }

        const auto &policies = outcome.GetResult().GetPolicies();
        for (const auto &policy: policies) {
            std::cout << std::left << std::setw(55) <<
                      policy.GetPolicyName() << std::setw(30) <<
                      policy.GetPolicyId() << std::setw(80) << policy.GetArn() <<
                      std::setw(64) << policy.GetDescription() << std::setw(12)
 <<
                      policy.GetCreateDate().ToGmtString(DATE_FORMAT.c_str()) <<
```

```
                        std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Para obtener información acerca de la API, consulte ListPolicies en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo enumerar las políticas administradas que están disponibles para su cuenta de AWS

En este ejemplo, se devuelve un conjunto de las dos primeras políticas administradas disponibles en la cuenta actual de AWS.

```
aws iam list-policies \
    --max-items 3
```

Salida:

```
{
    "Policies": [
        {
            "PolicyName": "AWSCloudTrailAccessPolicy",
            "PolicyId": "ANPAXQE2B5PJ7YEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:policy/AWSCloudTrailAccessPolicy",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 0,
```

```
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2019-09-04T17:43:42+00:00",
            "UpdateDate": "2019-09-04T17:43:42+00:00"
        },
        {
            "PolicyName": "AdministratorAccess",
            "PolicyId": "ANPAIWMBCKSKIEE64ZLYK",
            "Arn": "arn:aws:iam::aws:policy/AdministratorAccess",
            "Path": "/",
            "DefaultVersionId": "v1",
            "AttachmentCount": 6,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:46+00:00",
            "UpdateDate": "2015-02-06T18:39:46+00:00"
        },
        {
            "PolicyName": "PowerUserAccess",
            "PolicyId": "ANPAJYRXTHIB4FOVS3ZXS",
            "Arn": "arn:aws:iam::aws:policy/PowerUserAccess",
            "Path": "/",
            "DefaultVersionId": "v5",
            "AttachmentCount": 1,
            "PermissionsBoundaryUsageCount": 0,
            "IsAttachable": true,
            "CreateDate": "2015-02-06T18:39:47+00:00",
            "UpdateDate": "2023-07-06T22:04:00+00:00"
        }
    ],
    "NextToken": "EXAMPLErZXIiOiBudWxsLCAiYm90b190cnVuY2F0ZV9hbW91bnQiOiA4fQ=="
}
```


Para obtener más información, consulte Políticas y permisos en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere las políticas.

```
import { ListPoliciesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listPolicies() {
  const command = new ListPoliciesCommand({
    MaxItems: 10,
    OnlyAttached: false,
    // List only the customer managed policies in your Amazon Web Services
 account.
    Scope: "Local",
  });

  let response = await client.send(command);

  while (response.Policies?.length) {
    for (const policy of response.Policies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListPoliciesCommand({
          Marker: response.Marker,
```

```
          MaxItems: 10,
          OnlyAttached: false,
          Scope: "Local",
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listPolicies($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listPoliciesArguments = [];
        if ($pathPrefix) {
            $listPoliciesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listPoliciesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listPoliciesArguments["MaxItems"] = $maxItems;
        }
```

```
        return $this->iamClient->listPolicies($listPoliciesArguments);
    }
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
        return policies
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este módulo de ejemplo, se enumeran, se crean, se adjuntan y se separan las políticas de roles.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
```

```
 # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
 def get_policy(policy_arn)
   response = @iam_client.get_policy(policy_arn: policy_arn)
   policy = response.policy
   @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
   policy
 rescue Aws::IAM::Errors::NoSuchEntity
   @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
exist.")
   raise
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
   raise
 end

 # Attaches a policy to a role
 #
 # @param role_name [String] The name of the role
 # @param policy_arn [String] The policy ARN
 # @return [Boolean] true if successful, false otherwise
 def attach_policy_to_role(role_name, policy_arn)
   @iam_client.attach_role_policy(
     role_name: role_name,
     policy_arn: policy_arn
   )
   true
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error attaching policy to role: #{e.message}")
   false
 end

 # Lists policy ARNs attached to a role
 #
 # @param role_name [String] The name of the role
 # @return [Array<String>] List of policy ARNs
 def list_attached_policy_arns(role_name)
   response = @iam_client.list_attached_role_policies(role_name: role_name)
   response.attached_policies.map(&:policy_arn)
 rescue Aws::IAM::Errors::ServiceError => e
   @logger.error("Error listing policies attached to role: #{e.message}")
   []
 end
```

```
  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_policies(
    client: iamClient,
    path_prefix: String,
) -> Result<Vec<String>, SdkError<ListPoliciesError>> {
    let list_policies = client
        .list_policies()
        .path_prefix(path_prefix)
        .scope(PolicyScopeType::Local)
        .into_paginator()
```

```
        .items()
        .send()
        .try_collect()
        .await?;

    let policy_names = list_policies
        .into_iter()
        .map(|p| {
            let name = p
                .policy_name
                .unwrap_or_else(|| "Missing Policy Name".to_string());
            println!("{}", name);
            name
        })
        .collect();

    Ok(policy_names)
}
```

- Para obtener información sobre la API, consulte ListPolicies en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    public func listPolicies() async throws -> [MyPolicyRecord] {
```

```
        var policyList: [MyPolicyRecord] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListPoliciesInput(marker: marker)
            let output = try await client.listPolicies(input: input)

            guard let policies = output.policies else {
                return policyList
            }

            for policy in policies {
                guard   let name = policy.policyName,
                        let id = policy.policyId,
                        let arn = policy.arn else {
                    throw ServiceHandlerError.noSuchPolicy
                }
                policyList.append(MyPolicyRecord(name: name, id: id, arn: arn))
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Para obtener detalles sobre la API, consulte ListPolicies en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar las políticas asociadas a un rol de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar las políticas asociadas a un rol de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }
```

- Para obtener información acerca de la API, consulte ListAttachedRolePolicies en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo enumerar todas las políticas administradas que se asocian al rol especificado

Este comando devuelve los nombres y los ARN de las políticas administradas asociadas al rol de IAM denominado `SecurityAuditRole` en la cuenta de AWS.

```
aws iam list-attached-role-policies \
    --role-name SecurityAuditRole
```

Salida:

```
{
    "AttachedPolicies": [
        {
            "PolicyName": "SecurityAudit",
            "PolicyArn": "arn:aws:iam::aws:policy/SecurityAudit"
        }
    ],
    "IsTruncated": false
}
```

Para obtener más información, consulte Políticas y permisos en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
```

```
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere las políticas que están asociadas a un rol.

```
import {
  ListAttachedRolePoliciesCommand,
```

```
  IAMClient,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 * @param {string} roleName
 */
export async function* listAttachedRolePolicies(roleName) {
  const command = new ListAttachedRolePoliciesCommand({
    RoleName: roleName,
  });

  let response = await client.send(command);

  while (response.AttachedPolicies?.length) {
    for (const policy of response.AttachedPolicies) {
      yield policy;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListAttachedRolePoliciesCommand({
          RoleName: roleName,
          Marker: response.Marker,
        }),
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function listAttachedRolePolicies($roleName, $pathPrefix = "", $marker
 = "", $maxItems = 0)
    {
        $listAttachRolePoliciesArguments = ['RoleName' => $roleName];
        if ($pathPrefix) {
            $listAttachRolePoliciesArguments['PathPrefix'] = $pathPrefix;
        }
        if ($marker) {
            $listAttachRolePoliciesArguments['Marker'] = $marker;
        }
        if ($maxItems) {
            $listAttachRolePoliciesArguments['MaxItems'] = $maxItems;
        }
        return $this->iamClient-
>listAttachedRolePolicies($listAttachRolePoliciesArguments);
    }
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_attached_policies(role_name):
    """
    Lists policies attached to a role.

    :param role_name: The name of the role to query.
    """
    try:
        role = iam.Role(role_name)
        for policy in role.attached_policies.all():
            logger.info("Got policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't list attached policies for %s.", role_name)
        raise
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En este módulo de ejemplo, se enumeran, se crean, se adjuntan y se separan las políticas de roles.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
    @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
 #{policy.policy_id}.")
    policy
  rescue Aws::IAM::Errors::NoSuchEntity
    @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not
 exist.")
    raise
```

```
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
    raise
  end

  # Attaches a policy to a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def attach_policy_to_role(role_name, policy_arn)
    @iam_client.attach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error attaching policy to role: #{e.message}")
    false
  end

  # Lists policy ARNs attached to a role
  #
  # @param role_name [String] The name of the role
  # @return [Array<String>] List of policy ARNs
  def list_attached_policy_arns(role_name)
    response = @iam_client.list_attached_role_policies(role_name: role_name)
    response.attached_policies.map(&:policy_arn)
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing policies attached to role: #{e.message}")
    []
  end

  # Detaches a policy from a role
  #
  # @param role_name [String] The name of the role
  # @param policy_arn [String] The policy ARN
  # @return [Boolean] true if successful, false otherwise
  def detach_policy_from_role(role_name, policy_arn)
    @iam_client.detach_role_policy(
      role_name: role_name,
      policy_arn: policy_arn
    )
```

```
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error detaching policy from role: #{e.message}")
    false
  end
end
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_attached_role_policies(
    client: &iamClient,
    role_name: String,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListAttachedRolePoliciesOutput,
 SdkError<ListAttachedRolePoliciesError>> {
    let response = client
        .list_attached_role_policies()
        .role_name(role_name)
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;

    Ok(response)
}
```

- Para obtener información sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// Returns a list of AWS Identity and Access Management (IAM) policies
    /// that are attached to the role.
    ///
    /// - Parameter role: The IAM role to return the policy list for.
    ///
    /// - Returns: An array of `IAMClientTypes.AttachedPolicy` objects
    ///   describing each managed policy that's attached to the role.
    public func listAttachedRolePolicies(role: String) async throws ->
[IAMClientTypes.AttachedPolicy] {
        var policyList: [IAMClientTypes.AttachedPolicy] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListAttachedRolePoliciesInput(
                marker: marker,
                roleName: role
            )
            let output = try await client.listAttachedRolePolicies(input: input)
```

```
            guard let attachedPolicies = output.attachedPolicies else {
                return policyList
            }

            for attachedPolicy in attachedPolicies {
                policyList.append(attachedPolicy)
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Para obtener detalles sobre la API, consulte ListAttachedRolePolicies en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar roles de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar roles de IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
/// <summary>
/// List IAM roles.
/// </summary>
/// <returns>A list of IAM roles.</returns>
public async Task<List<Role>> ListRolesAsync()
{
```

```
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }
```

- Para obtener información acerca de la API, consulte ListRoles en la referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo enumerar los roles de IAM para la cuenta actual

El siguiente comando `list-roles` muestra los roles de IAM para la cuenta actual.

```
aws iam list-roles
```

Salida:

```
{
    "Roles": [
        {
            "Path": "/",
            "RoleName": "ExampleRole",
            "RoleId": "AROAJ52OTH4H7LEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/ExampleRole",
            "CreateDate": "2017-09-12T19:23:36+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
```

```
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        },
        {
            "Path": "/example_path/",
            "RoleName": "ExampleRoleWithPath",
            "RoleId": "AROAI4QRP7UFT7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:role/example_path/
ExampleRoleWithPath",
            "CreateDate": "2023-09-21T20:29:38+00:00",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Sid": "",
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "MaxSessionDuration": 3600
        }
    ]
}
```

Para más información, consulte Creación de roles de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListRoles en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API de AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los roles.

```
import { ListRolesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listRoles() {
  const command = new ListRolesCommand({
    MaxItems: 10,
  });

  /**
   * @type {import("@aws-sdk/client-iam").ListRolesCommandOutput | undefined}
   */
  let response = await client.send(command);

  while (response?.Roles?.length) {
    for (const role of response.Roles) {
      yield role;
    }

    if (response.IsTruncated) {
      response = await client.send(
        new ListRolesCommand({
          Marker: response.Marker,
        }),
```

```
      );
    } else {
      break;
    }
  }
}
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API de AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();

    /**
     * @param string $pathPrefix
     * @param string $marker
     * @param int $maxItems
     * @return Result
     * $roles = $service->listRoles();
     */
    public function listRoles($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listRolesArguments = [];
        if ($pathPrefix) {
            $listRolesArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listRolesArguments["Marker"] = $marker;
        }
        if ($maxItems) {
```

```
            $listRolesArguments["MaxItems"] = $maxItems;
        }
        return $this->iamClient->listRoles($listRolesArguments);
    }
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_roles(count):
    """
    Lists the specified number of roles for the account.

    :param count: The number of roles to list.
    """
    try:
        roles = list(iam.roles.limit(count=count))
        for role in roles:
            logger.info("Role: %s", role.name)
    except ClientError:
        logger.exception("Couldn't list roles for the account.")
        raise
    else:
        return roles
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count
      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
  end

  role_names
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't list roles for the account. Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  raise
end
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_roles(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListRolesOutput, SdkError<ListRolesError>> {
    let response = client
        .list_roles()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Para obtener información sobre la API, consulte ListRoles en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func listRoles() async throws -> [String] {
    var roleList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolesInput(marker: marker)
        let output = try await client.listRoles(input: input)

        guard let roles = output.roles else {
            return roleList
        }

        for role in roles {
            if let name = role.roleName {
                roleList.append(name)
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return roleList
}
```

- Para obtener detalles sobre la API, consulte ListRoles en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar certificados de servidor de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar certificados de servidor de IAM.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::listServerCertificates(
        const Aws::Client::ClientConfiguration &clientConfig) {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";

    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListServerCertificatesRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListServerCertificates(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list server certificates: " <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(55) << "Name" <<
                      std::setw(30) << "ID" << std::setw(80) << "Arn" <<
                      std::setw(14) << "UploadDate" << std::setw(14) <<
                      "ExpirationDate" << std::endl;
            header = true;
        }

        const auto &certificates =
                outcome.GetResult().GetServerCertificateMetadataList();

        for (const auto &certificate: certificates) {
            std::cout << std::left << std::setw(55) <<
                      certificate.GetServerCertificateName() << std::setw(30) <<
                      certificate.GetServerCertificateId() << std::setw(80) <<
```

```
                    certificate.GetArn() << std::setw(14) <<

 certificate.GetUploadDate().ToGmtString(DATE_FORMAT.c_str()) <<
                    std::setw(14) <<

 certificate.GetExpiration().ToGmtString(DATE_FORMAT.c_str()) <<
                    std::endl;
        }

        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Para obtener información sobre la API, consulte ListServerCertificates en AWS SDK for C++API Reference (Referencia de la API de).

CLI

AWS CLI

Cómo enumerar los certificados de servidor en su cuenta de AWS

El siguiente comando `list-server-certificates` enumera todos los certificados de servidor almacenados y disponibles para su uso en su cuenta de AWS.

```
aws iam list-server-certificates
```

Salida:

```
{
    "ServerCertificateMetadataList": [
        {
            "Path": "/",
            "ServerCertificateName": "myUpdatedServerCertificate",
```

```
            "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/
myUpdatedServerCertificate",
            "UploadDate": "2019-04-22T21:13:44+00:00",
            "Expiration": "2019-10-15T22:23:16+00:00"
        },
        {
            "Path": "/cloudfront/",
            "ServerCertificateName": "MyTestCert",
            "ServerCertificateId": "ASCAEXAMPLE456EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:server-certificate/Org1/Org2/
MyTestCert",
            "UploadDate": "2015-04-21T18:14:16+00:00",
            "Expiration": "2018-01-14T17:52:36+00:00"
        }
    ]
}
```

Para obtener más información, consulte Administración de certificados de servidor en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListServerCertificates en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los certificados.

```
import { ListServerCertificatesCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 * A generator function that handles paginated results.
```

```
 * The AWS SDK for JavaScript (v3) provides {@link https://docs.aws.amazon.com/
AWSJavaScriptSDK/v3/latest/index.html#paginators | paginator} functions to
 simplify this.
 *
 */
export async function* listServerCertificates() {
  const command = new ListServerCertificatesCommand({});
  let response = await client.send(command);

  while (response.ServerCertificateMetadataList?.length) {
    for await (const cert of response.ServerCertificateMetadataList) {
      yield cert;
    }

    if (response.IsTruncated) {
      response = await client.send(new ListServerCertificatesCommand({}));
    } else {
      break;
    }
  }
}
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListServerCertificates en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
```

```
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

iam.listServerCertificates({}, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListServerCertificates en AWS SDK for JavaScriptAPI Reference (Referencia de la API de).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, actualice y elimine certificados del servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
```

```
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
```

```
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Para obtener detalles sobre la API, consulte ListServerCertificates en la Referencia de la API deAWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Enumerar usuarios de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo enumerar usuarios de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de usuarios de solo lectura, y lectura y escritura

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
##############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
##############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

##############################################################################
# function iam_list_users
#
# List the IAM users in the account.
#
# Returns:
#       The list of users names
#    And:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
##############################################################################
function iam_list_users() {
  local option OPTARG # Required to use getopts command in a function.
  local error_code
  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_list_users"
    echo "Lists the AWS Identity and Access Management (IAM) user in the
 account."
    echo ""
  }

  # Retrieve the calling parameters.
```

```
  while getopts "h" option; do
    case "${option}" in
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  local response

  response=$(aws iam list-users \
    --output text \
    --query "Users[].UserName")
  error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports list-users operation failed.$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::listUsers(const Aws::Client::ClientConfiguration &clientConfig)
 {
    const Aws::String DATE_FORMAT = "%Y-%m-%d";
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::ListUsersRequest request;

    bool done = false;
    bool header = false;
    while (!done) {
        auto outcome = iam.ListUsers(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Failed to list iam users:" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }

        if (!header) {
            std::cout << std::left << std::setw(32) << "Name" <<
                      std::setw(30) << "ID" << std::setw(64) << "Arn" <<
                      std::setw(20) << "CreateDate" << std::endl;
            header = true;
        }

        const auto &users = outcome.GetResult().GetUsers();
        for (const auto &user: users) {
            std::cout << std::left << std::setw(32) << user.GetUserName() <<
                      std::setw(30) << user.GetUserId() << std::setw(64) <<
                      user.GetArn() << std::setw(20) <<
                      user.GetCreateDate().ToGmtString(DATE_FORMAT.c_str())
                      << std::endl;
        }
```

```
        if (outcome.GetResult().GetIsTruncated()) {
            request.SetMarker(outcome.GetResult().GetMarker());
        }
        else {
            done = true;
        }
    }

    return true;
}
```

- Para obtener información acerca de la API, consulte ListUsers en la referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo enumerar usuarios de IAM

El siguiente comando `list-users` enumera los usuarios de IAM en la cuenta actual.

```
aws iam list-users
```

Salida:

```
{
    "Users": [
        {
            "UserName": "Adele",
            "Path": "/",
            "CreateDate": "2013-03-07T05:14:48Z",
            "UserId": "AKIAI44QH8DHBEXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Adele"
        },
        {
            "UserName": "Bob",
            "Path": "/",
            "CreateDate": "2012-09-21T23:03:13Z",
            "UserId": "AKIAIOSFODNN7EXAMPLE",
            "Arn": "arn:aws:iam::123456789012:user/Bob"
```

```
        }
    ]
}
```

Para más información, consulte Enumeración de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte ListUsers en la Referencia de comandos de la AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.AttachedPermissionsBoundary;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.ListUsersRequest;
import software.amazon.awssdk.services.iam.model.ListUsersResponse;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.User;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListUsers {
    public static void main(String[] args) {
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        listAllUsers(iam);
```

```
        System.out.println("Done");
        iam.close();
    }

    public static void listAllUsers(IamClient iam) {
        try {
            boolean done = false;
            String newMarker = null;
            while (!done) {
                ListUsersResponse response;
                if (newMarker == null) {
                    ListUsersRequest request =
ListUsersRequest.builder().build();
                    response = iam.listUsers(request);
                } else {
                    ListUsersRequest request = ListUsersRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = iam.listUsers(request);
                }

                for (User user : response.users()) {
                    System.out.format("\n Retrieved user %s", user.userName());
                    AttachedPermissionsBoundary permissionsBoundary =
user.permissionsBoundary();
                    if (permissionsBoundary != null)
                        System.out.format("\n Permissions boundary details %s",

permissionsBoundary.permissionsBoundaryTypeAsString());
                }

                if (!response.isTruncated()) {
                    done = true;
                } else {
                    newMarker = response.marker();
                }
            }

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

```
}
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere los usuarios.

```
import { ListUsersCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

export const listUsers = async () => {
  const command = new ListUsersCommand({ MaxItems: 10 });

  const response = await client.send(command);
  response.Users?.forEach(({ UserName, CreateDate }) => {
    console.log(`${UserName} created on: ${CreateDate}`);
  });
  return response;
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  MaxItems: 10,
};

iam.listUsers(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    var users = data.Users || [];
    users.forEach(function (user) {
      console.log("User " + user.UserName + " created", user.CreateDate);
    });
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun listAllUsers() {

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listUsers(ListUsersRequest { })
        response.users?.forEach { user ->
            println("Retrieved user ${user.userName}")
            val permissionsBoundary = user.permissionsBoundary
            if (permissionsBoundary != null)
                println("Permissions boundary details
 ${permissionsBoundary.permissionsBoundaryType}")
        }
    }
}
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API del AWSSDK para Kotlin.

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
$uuid = uniqid();
$service = new IAMService();
```

```
    public function listUsers($pathPrefix = "", $marker = "", $maxItems = 0)
    {
        $listUsersArguments = [];
        if ($pathPrefix) {
            $listUsersArguments["PathPrefix"] = $pathPrefix;
        }
        if ($marker) {
            $listUsersArguments["Marker"] = $marker;
        }
        if ($maxItems) {
            $listUsersArguments["MaxItems"] = $maxItems;
        }

        return $this->iamClient->listUsers($listUsersArguments);
    }
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for PHP.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
```

```
        logger.exception("Couldn't get users.")
        raise
    else:
        return users
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
# Lists all users in the AWS account
#
# @return [Array<Aws::IAM::Types::User>] An array of user objects
def list_users
  users = []
  @iam_client.list_users.each_page do |page|
    page.users.each do |user|
      users << user
    end
  end
  users
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing users: #{e.message}")
  []
end
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
pub async fn list_users(
    client: &iamClient,
    path_prefix: Option<String>,
    marker: Option<String>,
    max_items: Option<i32>,
) -> Result<ListUsersOutput, SdkError<ListUsersError>> {
    let response = client
        .list_users()
        .set_path_prefix(path_prefix)
        .set_marker(marker)
        .set_max_items(max_items)
        .send()
        .await?;
    Ok(response)
}
```

- Para obtener información sobre la API, consulte ListUsers en la Referencia de la API del AWSSDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func listUsers() async throws -> [MyUserRecord] {
    var userList: [MyUserRecord] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListUsersInput(marker: marker)
        let output = try await client.listUsers(input: input)

        guard let users = output.users else {
            return userList
        }

        for user in users {
            if let id = user.userId, let name = user.userName {
                userList.append(MyUserRecord(id: id, name: name))
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    return userList
}
```

- Para obtener información acerca de la API, consulte ListUsers en la Referencia de la API del AWS SDK para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Eliminación de un usuario de IAM de un grupo mediante un SDK de AWS

En los siguientes ejemplos de código se muestra cómo eliminar un usuario de un grupo de IAM.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Creación de un grupo y adición de un usuario

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
   /// <summary>
   /// Remove a user from an IAM group.
   /// </summary>
   /// <param name="userName">The username of the user to remove.</param>
   /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
   {
       // Remove the user from the group.
       var removeUserRequest = new RemoveUserFromGroupRequest()
       {
           UserName = userName,
           GroupName = groupName,
       };

       var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
       return response.HttpStatusCode == HttpStatusCode.OK;
   }
```

- Para obtener información sobre la API, consulte RemoveUserFromGroup en la Referencia de la API de AWS SDK for .NET.

CLI

AWS CLI

Cómo eliminar un usuario de un grupo de IAM

El siguiente comando `remove-user-from-group` elimina al usuario denominado Bob del grupo de IAM denominado`Admins`.

```
aws iam remove-user-from-group \
    --user-name Bob \
    --group-name Admins
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Adición y eliminación de usuarios de un grupo de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte RemoveUserFromGroup en la Referencia de comandos de la AWS CLI.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Actualizar un certificado de servidor de IAM con un SDK de AWS

Los siguientes ejemplos de código muestran cómo actualizar un certificado de servidor de IAM.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::updateServerCertificate(const Aws::String
 &currentCertificateName,
```

```
                                                const Aws::String &newCertificateName,
                                                const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateServerCertificateRequest request;
    request.SetServerCertificateName(currentCertificateName);
    request.SetNewServerCertificateName(newCertificateName);

    auto outcome = iam.UpdateServerCertificate(request);
    bool result = true;
    if (outcome.IsSuccess()) {
        std::cout << "Server certificate " << currentCertificateName
                  << " successfully renamed as " << newCertificateName
                  << std::endl;
    }
    else {
        if (outcome.GetError().GetErrorType() !=
 Aws::IAM::IAMErrors::NO_SUCH_ENTITY) {
            std::cerr << "Error changing name of server certificate " <<
                      currentCertificateName << " to " << newCertificateName <<
 ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Certificate '" << currentCertificateName
                      << "' not found." << std::endl;
        }
    }

    return result;
}
```

- Para obtener información sobre la API, consulte UpdateServerCertificate en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo cambiar la ruta o el nombre de un certificado de servidor en su cuenta de AWS

El siguiente comando `update-server-certificate` cambia el nombre del certificado de `myServerCertificate` a `myUpdatedServerCertificate`. También cambia la ruta a `/cloudfront/` para que el servicio Amazon CloudFront pueda acceder a ella. Este comando no genera ninguna salida. Puede ver los resultados de la actualización al ejecutar el comando `list-server-certificates`.

```
aws-iam update-server-certificate \
    --server-certificate-name myServerCertificate \
    --new-server-certificate-name myUpdatedServerCertificate \
    --new-path /cloudfront/
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Administración de certificados de servidor en IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte UpdateServerCertificate en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Actualice un certificado de servidor.

```
import { UpdateServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentName
 * @param {string} newName
 */
export const updateServerCertificate = (currentName, newName) => {
```

```
  const command = new UpdateServerCertificateCommand({
    ServerCertificateName: currentName,
    NewServerCertificateName: newName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  ServerCertificateName: "CERTIFICATE_NAME",
  NewServerCertificateName: "NEW_CERTIFICATE_NAME",
};

iam.updateServerCertificate(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Enumere, actualice y elimine certificados del servidor.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
                                            server_certificate_name: name,
                                            certificate_body: certificate_body,
                                            private_key: private_key,
                                          })
    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
```

```
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
      @logger.info("No server certificates found.")
      return
    end

    response.server_certificate_metadata_list.each do |certificate_metadata|
      @logger.info("Certificate Name:
 #{certificate_metadata.server_certificate_name}")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing server certificates: #{e.message}")
  end

  # Updates the name of a server certificate.
  def update_server_certificate_name(current_name, new_name)
    @iam_client.update_server_certificate(
      server_certificate_name: current_name,
      new_server_certificate_name: new_name
    )
    @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error updating server certificate name: #{e.message}")
    false
  end

  # Deletes a server certificate.
  def delete_server_certificate(name)
    @iam_client.delete_server_certificate(server_certificate_name: name)
    @logger.info("Server certificate '#{name}' deleted.")
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting server certificate: #{e.message}")
    false
  end
end
```

- Para obtener detalles sobre la API, consulte UpdateServerCertificate en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Actualizar un usuario de IAM con un SDK de AWS

En los siguientes ejemplos de código, se muestra cómo actualizar un usuario de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Creación de usuarios de solo lectura, y lectura y escritura

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::updateUser(const Aws::String &currentUserName,
                             const Aws::String &newUserName,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
```

```
    Aws::IAM::IAMClient iam(clientConfig);

    Aws::IAM::Model::UpdateUserRequest request;
    request.SetUserName(currentUserName);
    request.SetNewUserName(newUserName);

    auto outcome = iam.UpdateUser(request);
    if (outcome.IsSuccess()) {
        std::cout << "IAM user " << currentUserName <<
                  " successfully updated with new user name " << newUserName <<
                  std::endl;
    }
    else {
        std::cerr << "Error updating user name for IAM user " << currentUserName
 <<
                  ":" << outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo cambiar el nombre de un usuario de IAM

El siguiente comando `update-user` cambia el nombre del usuario de IAM de `Bob` a `Robert`.

```
aws iam update-user \
    --user-name Bob \
    --new-user-name Robert
```

Este comando no genera ninguna salida.

Para obtener más información, consulte Cambio del nombre de un grupo de usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.UpdateUserRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateUser {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <curName> <newName>\s

                Where:
                    curName - The current user name.\s
                    newName - An updated user name.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
```

```
            System.exit(1);
        }

        String curName = args[0];
        String newName = args[1];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateIAMUser(iam, curName, newName);
        System.out.println("Done");
        iam.close();
    }

    public static void updateIAMUser(IamClient iam, String curName, String
 newName) {
        try {
            UpdateUserRequest request = UpdateUserRequest.builder()
                    .userName(curName)
                    .newUserName(newName)
                    .build();

            iam.updateUser(request);
            System.out.printf("Successfully updated user to username %s",
 newName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Actualice el usuario.

```
import { UpdateUserCommand, IAMClient } from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} currentUserName
 * @param {string} newUserName
 */
export const updateUser = (currentUserName, newUserName) => {
  const command = new UpdateUserCommand({
    UserName: currentUserName,
    NewUserName: newUserName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  UserName: process.argv[2],
  NewUserName: process.argv[3],
};

iam.updateUser(params, function (err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API de AWS SDK for JavaScript.

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
suspend fun updateIAMUser(curName: String?, newName: String?) {

    val request = UpdateUserRequest {
        userName = curName
        newUserName = newName
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API del AWSSDK para Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def update_user(user_name, new_user_name):
    """
    Updates a user's name.
```

```
    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user
```

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API del AWSSDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
  # Updates an IAM user's name
  #
  # @param current_name [String] The current name of the user
  # @param new_name [String] The new name of the user
  def update_user_name(current_name, new_name)
    @iam_client.update_user(user_name: current_name, new_user_name: new_name)
    true
  rescue StandardError => e
    @logger.error("Error updating user name from '#{current_name}' to
'#{new_name}': #{e.message}")
    false
```

```
end
```

- Para obtener información sobre la API, consulte UpdateUser en la Referencia de la API de AWS SDK for Ruby.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Actualizar una clave de acceso de IAM con un SDK de AWS

En los siguientes ejemplos de código, se muestra cómo actualizar una clave de acceso de IAM.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en el siguiente ejemplo de código:

- Administrar claves de acceso

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::IAM::updateAccessKey(const Aws::String &userName,
                                  const Aws::String &accessKeyID,
```

```
                                    Aws::IAM::Model::StatusType status,
                                    const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::IAM::IAMClient iam(clientConfig);
    Aws::IAM::Model::UpdateAccessKeyRequest request;
    request.SetUserName(userName);
    request.SetAccessKeyId(accessKeyID);
    request.SetStatus(status);

    auto outcome = iam.UpdateAccessKey(request);
    if (outcome.IsSuccess()) {
        std::cout << "Successfully updated status of access key "
                  << accessKeyID << " for user " << userName << std::endl;
    }
    else {
        std::cerr << "Error updated status of access key " << accessKeyID <<
                  " for user " << userName << ": " <<
                  outcome.GetError().GetMessage() << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte UpdateAccessKey en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo activar o desactivar una clave de acceso para un usuario de IAM

El siguiente comando `update-access-key` desactiva la clave de acceso especificada (ID de clave de acceso y clave de acceso secreta) para el usuario de IAM denominado `Bob`.

```
aws iam update-access-key \
    --access-key-id AKIAIOSFODNN7EXAMPLE \
    --status Inactive \
    --user-name Bob
```

Este comando no genera ninguna salida.

Si se desactiva la clave, no se puede utilizar para acceder mediante programación a AWS. Sin embargo, la clave sigue disponible y se puede reactivar.

Para obtener más información, consulte Administración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte UpdateAccessKey en la Referencia de la API de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.services.iam.model.IamException;
import software.amazon.awssdk.services.iam.model.StatusType;
import software.amazon.awssdk.services.iam.model.UpdateAccessKeyRequest;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class UpdateAccessKey {

    private static StatusType statusType;

    public static void main(String[] args) {
        final String usage = """

                Usage:
```

```
                    <username> <accessId> <status>\s

                Where:
                    username - The name of the user whose key you want to update.
\s
                    accessId - The access key ID of the secret access key you
 want to update.\s
                    status - The status you want to assign to the secret access
 key.\s
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String username = args[0];
        String accessId = args[1];
        String status = args[2];
        Region region = Region.AWS_GLOBAL;
        IamClient iam = IamClient.builder()
                .region(region)
                .build();

        updateKey(iam, username, accessId, status);
        System.out.println("Done");
        iam.close();
    }

    public static void updateKey(IamClient iam, String username, String accessId,
 String status) {
        try {
            if (status.toLowerCase().equalsIgnoreCase("active")) {
                statusType = StatusType.ACTIVE;
            } else if (status.toLowerCase().equalsIgnoreCase("inactive")) {
                statusType = StatusType.INACTIVE;
            } else {
                statusType = StatusType.UNKNOWN_TO_SDK_VERSION;
            }

            UpdateAccessKeyRequest request = UpdateAccessKeyRequest.builder()
                    .accessKeyId(accessId)
                    .userName(username)
                    .status(statusType)
```

```
                    .build();

            iam.updateAccessKey(request);
            System.out.printf("Successfully updated the status of access key %s
 to" +
                    "status %s for user %s", accessId, status, username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte UpdateAccessKey en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Actualice la clave de acceso.

```
import {
  UpdateAccessKeyCommand,
  IAMClient,
  StatusType,
} from "@aws-sdk/client-iam";

const client = new IAMClient({});

/**
 *
 * @param {string} userName
 * @param {string} accessKeyId
```

```
 */
export const updateAccessKey = (userName, accessKeyId) => {
  const command = new UpdateAccessKeyCommand({
    AccessKeyId: accessKeyId,
    Status: StatusType.Inactive,
    UserName: userName,
  });

  return client.send(command);
};
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateAccessKey en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create the IAM service object
var iam = new AWS.IAM({ apiVersion: "2010-05-08" });

var params = {
  AccessKeyId: "ACCESS_KEY_ID",
  Status: "Active",
  UserName: "USER_NAME",
};

iam.updateAccessKey(params, function (err, data) {
  if (err) {
    console.log("Error", err);
```

```
  } else {
    console.log("Success", data);
  }
});
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for JavaScript.
- Para obtener información sobre la API, consulte UpdateAccessKey en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
```

```
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise
```

- Para obtener detalles sobre la API, consulte UpdateAccessKey en la Referencia de la API del AWS SDK para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Carga de un certificado de servidor de IAM con un SDK de AWS

En los siguientes ejemplos de código se muestra cómo cargar un certificado de servidor de AWS Identity and Access Management (IAM).

CLI

AWS CLI

Cómo cargar un certificado de servidor en su cuenta de AWS

El siguiente comando upload-server-certificate carga un certificado de servidor en su cuenta de AWS. En este ejemplo, el certificado está en el archivo `public_key_cert_file.pem`, la clave privada asociada está en el archivo `my_private_key.pem` y la cadena de certificados proporcionada por la entidad de certificación (CA) está en el archivo `my_certificate_chain_file.pem`. Cuando el archivo haya terminado de cargarse, estará disponible con el nombre myServerCertificate. Los parámetros que comienzan con `file://` indican al comando que lea el contenido del archivo y lo use como valor del parámetro en lugar del nombre del archivo en sí.

```
aws iam upload-server-certificate \
    --server-certificate-name myServerCertificate \
    --certificate-body file://public_key_cert_file.pem \
    --private-key file://my_private_key.pem \
    --certificate-chain file://my_certificate_chain_file.pem
```

Salida:

```
{
    "ServerCertificateMetadata": {
        "Path": "/",
        "ServerCertificateName": "myServerCertificate",
        "ServerCertificateId": "ASCAEXAMPLE123EXAMPLE",
        "Arn": "arn:aws:iam::1234567989012:server-certificate/
myServerCertificate",
        "UploadDate": "2019-04-22T21:13:44+00:00",
        "Expiration": "2019-10-15T22:23:16+00:00"
    }
}
```

Para obtener más información, consulte Creación, carga y eliminación de certificados de servidor en la guía Uso de IAM.

- Para obtener información sobre la API, consulte UploadServerCertificate en la Referencia de comandos de la AWS CLI.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "path";

const client = new IAMClient({});

const certMessage = `Generate a certificate and key with the following command,
 or the equivalent for your system.

openssl req -x509 -newkey rsa:4096 -sha256 -days 3650 -nodes \
```

```
-keyout example.key -out example.crt -subj "/CN=example.com" \
-addext "subjectAltName=DNS:example.com,DNS:www.example.net,IP:10.0.0.1"
`;

const getCertAndKey = () => {
  try {
    const cert = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.crt"),
    );
    const key = readFileSync(
      path.join(dirnameFromMetaUrl(import.meta.url), "./example.key"),
    );
    return { cert, key };
  } catch (err) {
    if (err.code === "ENOENT") {
      throw new Error(
        `Certificate and/or private key not found. ${certMessage}`,
      );
    }

    throw err;
  }
};

/**
 *
 * @param {string} certificateName
 */
export const uploadServerCertificate = (certificateName) => {
  const { cert, key } = getCertAndKey();
  const command = new UploadServerCertificateCommand({
    ServerCertificateName: certificateName,
    CertificateBody: cert.toString(),
    PrivateKey: key.toString(),
  });

  return client.send(command);
};
```

- Para obtener detalles de la API, consulte UploadServerCertificate en la Referencia de la API de AWS SDK for JavaScript.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Situaciones de IAM con SDK de AWS

En los siguientes ejemplos de código se muestra cómo implementar situaciones comunes en IAM con SDK de AWS. Estas situaciones muestran cómo llevar a cabo tareas específicas llamando a varias funciones dentro de IAM. En cada escenario se incluye un enlace a GitHub, con instrucciones de configuración y ejecución del código.

Ejemplos

- Cree y gestione un servicio resiliente mediante un SDK de AWS
- Creación de un grupo de IAM y adición de un usuario a un grupo mediante un SDK de AWS
- Crear un usuario de IAM y asumir un rol con AWS STS con un SDK de AWS
- Creación de usuarios de IAM de solo lectura y lectura y escritura con un SDK de AWS
- Administrar claves de acceso de IAM con un SDK de AWS
- Administrar políticas de IAM con un SDK de AWS
- Administrar roles de IAM con un SDK de AWS
- Administrar la cuenta de IAM con un SDK de AWS
- Revertir una versión de la política de IAM con un SDK de AWS
- Trabajar con la API del creador de políticas de IAM mediante un SDK de AWS

Cree y gestione un servicio resiliente mediante un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un servicio web con equilibrio de carga que muestre recomendaciones de libros, películas y canciones. El ejemplo muestra cómo responde el servicio a los errores y cómo reestructurarlo para aumentar la resiliencia cuando se produzcan errores.

- Utilice un grupo de Amazon EC2 Auto Scaling para crear instancias de Amazon Elastic Compute Cloud (Amazon EC2) basadas en una plantilla de lanzamiento y para mantener el número de instancias dentro de un rango específico.
- Administre y distribuya las solicitudes HTTP con Elastic Load Balancing.

- Supervise el estado de las instancias de un grupo de escalado automático y reenvíe las solicitudes solo a las instancias en buen estado.
- Ejecute un servidor web Python en cada instancia de EC2 para administrar las solicitudes HTTP. El servidor web responde con recomendaciones y comprobaciones de estado.
- Simule un servicio de recomendaciones con una tabla de Amazon DynamoDB.
- Controle la respuesta del servidor web a las solicitudes y las comprobaciones de estado mediante la actualización de AWS Systems Manager parámetros.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute el escenario interactivo en un símbolo del sistema.

```
    static async Task Main(string[] args)
    {
        _configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally, load local settings.
            .Build();


        // Set up dependency injection for the AWS services.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonIdentityManagementService>()
```

```
                    .AddAWSService<IAmazonDynamoDB>()
                    .AddAWSService<IAmazonElasticLoadBalancingV2>()
                    .AddAWSService<IAmazonSimpleSystemsManagement>()
                    .AddAWSService<IAmazonAutoScaling>()
                    .AddAWSService<IAmazonEC2>()
                    .AddTransient<AutoScalerWrapper>()
                    .AddTransient<ElasticLoadBalancerWrapper>()
                    .AddTransient<SmParameterWrapper>()
                    .AddTransient<Recommendations>()
                    .AddSingleton<IConfiguration>(_configuration)
            )
            .Build();

        ServicesSetup(host);
        ResourcesSetup();

        try
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Welcome to the Resilient Architecture Example
Scenario.");
            Console.WriteLine(new string('-', 80));
            await Deploy(true);

            Console.WriteLine("Now let's begin the scenario.");
            Console.WriteLine(new string('-', 80));
            await Demo(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Finally, let's clean up our resources.");
            Console.WriteLine(new string('-', 80));

            await DestroyResources(true);

            Console.WriteLine(new string('-', 80));
            Console.WriteLine("Resilient Architecture Example Scenario is
complete.");
            Console.WriteLine(new string('-', 80));
        }
        catch (Exception ex)
        {
            Console.WriteLine(new string('-', 80));
            Console.WriteLine($"There was a problem running the scenario:
{ex.Message}");
```

```
            await DestroyResources(true);
            Console.WriteLine(new string('-', 80));
        }
    }

    /// <summary>
    /// Setup any common resources, also used for integration testing.
    /// </summary>
    public static void ResourcesSetup()
    {
        _httpClient = new HttpClient();
    }

    /// <summary>
    /// Populate the services for use within the console application.
    /// </summary>
    /// <param name="host">The services host.</param>
    private static void ServicesSetup(IHost host)
    {
        _elasticLoadBalancerWrapper =
host.Services.GetRequiredService<ElasticLoadBalancerWrapper>();
        _iamClient =
host.Services.GetRequiredService<IAmazonIdentityManagementService>();
        _recommendations = host.Services.GetRequiredService<Recommendations>();
        _autoScalerWrapper =
host.Services.GetRequiredService<AutoScalerWrapper>();
        _smParameterWrapper =
host.Services.GetRequiredService<SmParameterWrapper>();
    }

    /// <summary>
    /// Deploy necessary resources for the scenario.
    /// </summary>
    /// <param name="interactive">True to run as interactive.</param>
    /// <returns>True if successful.</returns>
    public static async Task<bool> Deploy(bool interactive)
    {
        var protocol = "HTTP";
        var port = 80;
        var sshPort = 22;

        Console.WriteLine(
            "\nFor this demo, we'll use the AWS SDK for .NET to create several
AWS resources\n" +
```

```
            "to set up a load-balanced web service endpoint and explore some ways
to make it resilient\n" +
            "against various kinds of failures.\n\n" +
            "Some of the resources create by this demo are:\n");

        Console.WriteLine(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations.");
        Console.WriteLine(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server.");
        Console.WriteLine(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones.");
        Console.WriteLine(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests.");
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to start deploying
resources.");
        if (interactive)
            Console.ReadLine();

        // Create and populate the DynamoDB table.
        var databaseTableName = _configuration["databaseName"];
        var recommendationsPath = Path.Join(_configuration["resourcePath"],
            "recommendations_objects.json");
        Console.WriteLine($"Creating and populating a DynamoDB table named
{databaseTableName}.");
        await _recommendations.CreateDatabaseWithName(databaseTableName);
        await _recommendations.PopulateDatabase(databaseTableName,
recommendationsPath);
        Console.WriteLine(new string('-', 80));

        // Create the EC2 Launch Template.

        Console.WriteLine(
            $"Creating an EC2 launch template that runs
'server_startup_script.sh' when an instance starts.\n"
            + "\nThis script starts a Python web server defined in the
`server.py` script. The web server\n"
            + "listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.\n"
```

```
            + "For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            + "run a web server, such as Apache, with least-privileged
credentials.");
        Console.WriteLine(
            "\nThe template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            + "permissions to access the DynamoDB recommendation table and
Systems Manager parameters\n"
            + "that control the flow of the demo.");

        var startupScriptPath = Path.Join(_configuration["resourcePath"],
            "server_startup_script.sh");
        var instancePolicyPath = Path.Join(_configuration["resourcePath"],
            "instance_policy.json");
        await _autoScalerWrapper.CreateTemplate(startupScriptPath,
instancePolicyPath);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            + "Availability Zone.\n");
        var zones = await _autoScalerWrapper.DescribeAvailabilityZones();
        await _autoScalerWrapper.CreateGroupOfSize(3,
_autoScalerWrapper.GroupName, zones);
        Console.WriteLine(new string('-', 80));

        Console.WriteLine(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
            + "HTTP requests. You can see these instances in the console or
continue with the demo.\n");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue.");
        if (interactive)
            Console.ReadLine();

        Console.WriteLine("Creating variables that control the flow of the
demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine(
```

```
            "\nCreating an Elastic Load Balancing target group and load balancer.
 The target group\n"
            + "defines how the load balancer connects to instances. The load
 balancer provides a\n"
            + "single endpoint where clients connect and dispatches requests to
 instances in the group.");

        var defaultVpc = await _autoScalerWrapper.GetDefaultVpc();
        var subnets = await
 _autoScalerWrapper.GetAllVpcSubnetsForZones(defaultVpc.VpcId, zones);
        var subnetIds = subnets.Select(s => s.SubnetId).ToList();
        var targetGroup = await
 _elasticLoadBalancerWrapper.CreateTargetGroupOnVpc(_elasticLoadBalancerWrapper.TargetGro
 protocol, port, defaultVpc.VpcId);

        await
 _elasticLoadBalancerWrapper.CreateLoadBalancerAndListener(_elasticLoadBalancerWrapper.Lo
 subnetIds, targetGroup);
        await
 _autoScalerWrapper.AttachLoadBalancerToGroup(_autoScalerWrapper.GroupName,
 targetGroup.TargetGroupArn);
        Console.WriteLine("\nVerifying access to the load balancer endpoint...");
        var endPoint = await
 _elasticLoadBalancerWrapper.GetEndpointForLoadBalancerByName(_elasticLoadBalancerWrapper
        var loadBalancerAccess = await
 _elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);

        if (!loadBalancerAccess)
        {
            Console.WriteLine("\nCouldn't connect to the load balancer, verifying
 that the port is open...");

            var ipString = await _httpClient.GetStringAsync("https://
checkip.amazonaws.com");
            ipString = ipString.Trim();

            var defaultSecurityGroup = await
 _autoScalerWrapper.GetDefaultSecurityGroupForVpc(defaultVpc);
            var portIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, port,
 ipString);
            var sshPortIsOpen =
 _autoScalerWrapper.VerifyInboundPortForGroup(defaultSecurityGroup, sshPort,
 ipString);
```

```
            if (!portIsOpen)
            {
                Console.WriteLine(
                    "\nFor this example to work, the default security group for
your default VPC must\n"
                    + "allows access from this computer. You can either add it
automatically from this\n"
                    + "example or add it yourself using the AWS Management
Console.\n");

                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound traffic from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, port,
ipString);
                }
            }

            if (!sshPortIsOpen)
            {
                if (!interactive || GetYesNoResponse(
                        "Do you want to add a rule to the security group to allow
inbound SSH traffic for debugging from your computer's IP address?"))
                {
                    await
_autoScalerWrapper.OpenInboundPort(defaultSecurityGroup.GroupId, sshPort,
ipString);
                }
            }
            loadBalancerAccess = await
_elasticLoadBalancerWrapper.VerifyLoadBalancerEndpoint(endPoint);
        }

        if (loadBalancerAccess)
        {
            Console.WriteLine("Your load balancer is ready. You can access it by
browsing to:");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        else
        {
```

```
            Console.WriteLine(
                "\nCouldn't get a successful response from the load balancer
endpoint. Troubleshoot by\n"
                + "manually verifying that your VPC and security group are
configured correctly and that\n"
                + "you can successfully make a GET request to the load balancer
endpoint:\n");
            Console.WriteLine($"\thttp://{endPoint}\n");
        }
        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Press Enter when you're ready to continue with the
demo.");
        if (interactive)
            Console.ReadLine();
        return true;
    }

    /// <summary>
    /// Demonstrate the steps of the scenario.
    /// </summary>
    /// <param name="interactive">True to run as an interactive scenario.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> Demo(bool interactive)
    {
        var ssmOnlyPolicy = Path.Join(_configuration["resourcePath"],
            "ssm_only_policy.json");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Resetting parameters to starting values for demo.");
        await _smParameterWrapper.Reset();

        Console.WriteLine("\nThis part of the demonstration shows how to toggle
different parts of the system\n" +
                          "to create situations where the web service fails, and
shows how using a resilient\n" +
                          "architecture can keep the web service running in spite
of these failures.");
        Console.WriteLine(new string('-', 88));
        Console.WriteLine("At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.");
        if (interactive)
            await DemoActionChoices();
```

```
        Console.WriteLine($"The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.\n" +
                          $"The table name is contained in a Systems Manager
parameter named '{_smParameterWrapper.TableParameter}'.\n" +
                          $"To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as\n" +
                          "healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Instead of failing when the recommendation service
fails, the web service can return a static response.");
        Console.WriteLine("While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.FailureResponseParameter,
"static");

        Console.WriteLine("\nNow, sending a GET request to the load balancer
endpoint returns a static response.");
        Console.WriteLine("The service still reports as healthy because health
checks are still shallow.");
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("Let's reinstate the recommendation service.\n");
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
_smParameterWrapper.TableName);
        Console.WriteLine(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n" +
            "access the DynamoDB recommendation table.\n"
        );
        await _autoScalerWrapper.CreateInstanceProfileWithName(
            _autoScalerWrapper.BadCredsPolicyName,
            _autoScalerWrapper.BadCredsRoleName,
```

```
            _autoScalerWrapper.BadCredsProfileName,
            ssmOnlyPolicy,
            new List<string> { "AmazonSSMManagedInstanceCore" }
        );
        var instances = await
_autoScalerWrapper.GetInstancesByGroupName(_autoScalerWrapper.GroupName);
        var badInstanceId = instances.First();
        var instanceProfile = await
_autoScalerWrapper.GetInstanceProfile(badInstanceId);
        Console.WriteLine(
            $"Replacing the profile for instance {badInstanceId} with a profile
that contains\n" +
            "bad credentials...\n"
        );
        await _autoScalerWrapper.ReplaceInstanceProfile(
            badInstanceId,
            _autoScalerWrapper.BadCredsProfileName,
            instanceProfile.AssociationId
        );
        Console.WriteLine(
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n" +
            "depending on which instance is selected by the load balancer.\n"
        );
        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nLet's implement a deep health check. For this demo,
a deep health check tests whether");
        Console.WriteLine("the web service can access the DynamoDB table that it
depends on for recommendations. Note that");
        Console.WriteLine("the deep health check is only for ELB routing and not
for Auto Scaling instance health.");
        Console.WriteLine("This kind of deep health check is not recommended for
Auto Scaling instance health, because it");
        Console.WriteLine("risks accidental termination of all instances in the
Auto Scaling group when a dependent service fails.");

        Console.WriteLine("\nBy implementing deep health checks, the load
balancer can detect when one of the instances is failing");
        Console.WriteLine("and take that instance out of rotation.");
```

```
        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.HealthCheckParameter,
"deep");

        Console.WriteLine($"\nNow, checking target health indicates that the
instance with bad credentials ({badInstanceId})");
        Console.WriteLine("is unhealthy. Note that it might take a minute or two
for the load balancer to detect the unhealthy");
        Console.WriteLine("instance. Sending a GET request to the load balancer
endpoint always returns a recommendation, because");
        Console.WriteLine("the load balancer takes unhealthy instances out of its
rotation.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nBecause the instances in this demo are controlled by
an auto scaler, the simplest way to fix an unhealthy");
        Console.WriteLine("instance is to terminate it and let the auto scaler
start a new instance to replace it.");

        await _autoScalerWrapper.TryTerminateInstanceById(badInstanceId);

        Console.WriteLine($"\nEven while the instance is terminating and the new
instance is starting, sending a GET");
        Console.WriteLine("request to the web service continues to get a
successful recommendation response because");
        Console.WriteLine("starts and reports as healthy, it is included in the
load balancing rotation.");
        Console.WriteLine("Note that terminating and replacing an instance
typically takes several minutes, during which time you");
        Console.WriteLine("can see the changing health check status until the new
instance is running and healthy.");

        if (interactive)
            await DemoActionChoices();

        Console.WriteLine("\nIf the recommendation service fails now, deep health
checks mean all instances report as unhealthy.");

        await
_smParameterWrapper.PutParameterByName(_smParameterWrapper.TableParameter,
"this-is-not-a-table");
```

```
        Console.WriteLine($"\nWhen all instances are unhealthy, the load balancer
continues to route requests even to");
        Console.WriteLine("unhealthy instances, allowing them to fail open and
return a static response rather than fail");
        Console.WriteLine("closed and report failure to the customer.");

        if (interactive)
            await DemoActionChoices();
        await _smParameterWrapper.Reset();

        Console.WriteLine(new string('-', 80));
        return true;
    }

    /// <summary>
    /// Clean up the resources from the scenario.
    /// </summary>
    /// <param name="interactive">True to ask the user for cleanup.</param>
    /// <returns>Async task.</returns>
    public static async Task<bool> DestroyResources(bool interactive)
    {
        Console.WriteLine(new string('-', 80));
        Console.WriteLine(
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n" +
            "that were created for this demo."
        );

        if (!interactive || GetYesNoResponse("Do you want to clean up all demo
resources? (y/n) "))
        {
            await
_elasticLoadBalancerWrapper.DeleteLoadBalancerByName(_elasticLoadBalancerWrapper.LoadBal
            await
_elasticLoadBalancerWrapper.DeleteTargetGroupByName(_elasticLoadBalancerWrapper.TargetGr
            await
_autoScalerWrapper.TerminateAndDeleteAutoScalingGroupWithName(_autoScalerWrapper.GroupNa
            await
_autoScalerWrapper.DeleteKeyPairByName(_autoScalerWrapper.KeyPairName);
            await
_autoScalerWrapper.DeleteTemplateByName(_autoScalerWrapper.LaunchTemplateName);
            await _autoScalerWrapper.DeleteInstanceProfile(
                _autoScalerWrapper.BadCredsProfileName,
                _autoScalerWrapper.BadCredsRoleName
```

```
            );
            await
 _recommendations.DestroyDatabaseByName(_recommendations.TableName);
        }
        else
        {
            Console.WriteLine(
                "Ok, we'll leave the resources intact.\n" +
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            );
        }

        Console.WriteLine(new string('-', 80));
        return true;
    }
```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```
/// <summary>
/// Encapsulates Amazon EC2 Auto Scaling and EC2 management methods.
/// </summary>
public class AutoScalerWrapper
{
    private readonly IAmazonAutoScaling _amazonAutoScaling;
    private readonly IAmazonEC2 _amazonEc2;
    private readonly IAmazonSimpleSystemsManagement _amazonSsm;
    private readonly IAmazonIdentityManagementService _amazonIam;

    private readonly string _instanceType = "";
    private readonly string _amiParam = "";
    private readonly string _launchTemplateName = "";
    private readonly string _groupName = "";
    private readonly string _instancePolicyName = "";
    private readonly string _instanceRoleName = "";
    private readonly string _instanceProfileName = "";
    private readonly string _badCredsProfileName = "";
    private readonly string _badCredsRoleName = "";
    private readonly string _badCredsPolicyName = "";
    private readonly string _keyPairName = "";

    public string GroupName => _groupName;
```

```
    public string KeyPairName => _keyPairName;
    public string LaunchTemplateName => _launchTemplateName;
    public string InstancePolicyName => _instancePolicyName;
    public string BadCredsProfileName => _badCredsProfileName;
    public string BadCredsRoleName => _badCredsRoleName;
    public string BadCredsPolicyName => _badCredsPolicyName;

    /// <summary>
    /// Constructor for the AutoScalerWrapper.
    /// </summary>
    /// <param name="amazonAutoScaling">The injected AutoScaling client.</param>
    /// <param name="amazonEc2">The injected EC2 client.</param>
    /// <param name="amazonIam">The injected IAM client.</param>
    /// <param name="amazonSsm">The injected SSM client.</param>
    public AutoScalerWrapper(
        IAmazonAutoScaling amazonAutoScaling,
        IAmazonEC2 amazonEc2,
        IAmazonSimpleSystemsManagement amazonSsm,
        IAmazonIdentityManagementService amazonIam,
        IConfiguration configuration)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        var prefix = configuration["resourcePrefix"];
        _instanceType = configuration["instanceType"];
        _amiParam = configuration["amiParam"];

        _launchTemplateName = prefix + "-template";
        _groupName = prefix + "-group";
        _instancePolicyName = prefix + "-pol";
        _instanceRoleName = prefix + "-role";
        _instanceProfileName = prefix + "-prof";
        _badCredsPolicyName = prefix + "-bc-pol";
        _badCredsRoleName = prefix + "-bc-role";
        _badCredsProfileName = prefix + "-bc-prof";
        _keyPairName = prefix + "-key-pair";
    }

    /// <summary>
    /// Create a policy, role, and profile that is associated with instances with
a specified name.
```

```
   /// An instance's associated profile defines a role that is assumed by the
   /// instance.The role has attached policies that specify the AWS permissions
granted to
   /// clients that run on the instance.
   /// </summary>
   /// <param name="policyName">Name to use for the policy.</param>
   /// <param name="roleName">Name to use for the role.</param>
   /// <param name="profileName">Name to use for the profile.</param>
   /// <param name="ssmOnlyPolicyFile">Path to a policy file for SSM.</param>
   /// <param name="awsManagedPolicies">AWS Managed policies to be attached to
the role.</param>
   /// <returns>The Arn of the profile.</returns>
   public async Task<string> CreateInstanceProfileWithName(
       string policyName,
       string roleName,
       string profileName,
       string ssmOnlyPolicyFile,
       List<string>? awsManagedPolicies = null)
   {

       var assumeRoleDoc = "{" +
                                "\"Version\": \"2012-10-17\"," +
                                "\"Statement\": [{" +
                                     "\"Effect\": \"Allow\"," +
                                     "\"Principal\": {" +
                                     "\"Service\": [" +
                                         "\"ec2.amazonaws.com\"" +
                                     "]" +
                                     "}," +
                                "\"Action\": \"sts:AssumeRole\"" +
                                "}]" +
                           "}";

       var policyDocument = await File.ReadAllTextAsync(ssmOnlyPolicyFile);

       var policyArn = "";

       try
       {
           var createPolicyResult = await _amazonIam.CreatePolicyAsync(
               new CreatePolicyRequest
               {
                   PolicyName = policyName,
                   PolicyDocument = policyDocument
```

```
                });
            policyArn = createPolicyResult.Policy.Arn;
        }
        catch (EntityAlreadyExistsException)
        {
            // The policy already exists, so we look it up to get the Arn.
            var policiesPaginator = _amazonIam.Paginators.ListPolicies(
                new ListPoliciesRequest()
                {
                    Scope = PolicyScopeType.Local
                });
            // Get the entire list using the paginator.
            await foreach (var policy in policiesPaginator.Policies)
            {
                if (policy.PolicyName.Equals(policyName))
                {
                    policyArn = policy.Arn;
                }
            }

            if (policyArn == null)
            {
                throw new InvalidOperationException("Policy not found");
            }
        }

        try
        {
            await _amazonIam.CreateRoleAsync(new CreateRoleRequest()
            {
                RoleName = roleName,
                AssumeRolePolicyDocument = assumeRoleDoc,
            });
            await _amazonIam.AttachRolePolicyAsync(new AttachRolePolicyRequest()
            {
                RoleName = roleName,
                PolicyArn = policyArn
            });
            if (awsManagedPolicies != null)
            {
                foreach (var awsPolicy in awsManagedPolicies)
                {
                    await _amazonIam.AttachRolePolicyAsync(new
AttachRolePolicyRequest()
```

```
                    {
                        PolicyArn = $"arn:aws:iam::aws:policy/{awsPolicy}",
                        RoleName = roleName
                    });
                }
            }
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Role already exists.");
        }

        string profileArn = "";
        try
        {
            var profileCreateResponse = await
_amazonIam.CreateInstanceProfileAsync(
                new CreateInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            // Allow time for the profile to be ready.
            profileArn = profileCreateResponse.InstanceProfile.Arn;
            Thread.Sleep(10000);
            await _amazonIam.AddRoleToInstanceProfileAsync(
                new AddRoleToInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });

        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine("Policy already exists.");
            var profileGetResponse = await _amazonIam.GetInstanceProfileAsync(
                new GetInstanceProfileRequest()
                {
                    InstanceProfileName = profileName
                });
            profileArn = profileGetResponse.InstanceProfile.Arn;
        }
        return profileArn;
    }
```

```
    /// <summary>
    /// Create a new key pair and save the file.
    /// </summary>
    /// <param name="newKeyPairName">The name of the new key pair.</param>
    /// <returns>Async task.</returns>
    public async Task CreateKeyPair(string newKeyPairName)
    {
        try
        {
            var keyResponse = await _amazonEc2.CreateKeyPairAsync(
                new CreateKeyPairRequest() { KeyName = newKeyPairName });
            await File.WriteAllTextAsync($"{newKeyPairName}.pem",
                keyResponse.KeyPair.KeyMaterial);
            Console.WriteLine($"Created key pair {newKeyPairName}.");
        }
        catch (AlreadyExistsException)
        {
            Console.WriteLine("Key pair already exists.");
        }
    }

    /// <summary>
    /// Delete the key pair and file by name.
    /// </summary>
    /// <param name="deleteKeyPairName">The key pair to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteKeyPairByName(string deleteKeyPairName)
    {
        try
        {
            await _amazonEc2.DeleteKeyPairAsync(
                new DeleteKeyPairRequest() { KeyName = deleteKeyPairName });
            File.Delete($"{deleteKeyPairName}.pem");
        }
        catch (FileNotFoundException)
        {
            Console.WriteLine($"Key pair {deleteKeyPairName} not found.");
        }
    }

    /// <summary>
    /// Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling.
```

```
   /// The launch template specifies a Bash script in its user data field that
runs after
   /// the instance is started. This script installs the Python packages and
starts a Python
   /// web server on the instance.
   /// </summary>
   /// <param name="startupScriptPath">The path to a Bash script file that is
run.</param>
   /// <param name="instancePolicyPath">The path to a permissions policy to
create and attach to the profile.</param>
   /// <returns>The template object.</returns>
   public async Task<Amazon.EC2.Model.LaunchTemplate> CreateTemplate(string
startupScriptPath, string instancePolicyPath)
   {
       await CreateKeyPair(_keyPairName);
       await CreateInstanceProfileWithName(_instancePolicyName,
_instanceRoleName, _instanceProfileName, instancePolicyPath);

       var startServerText = await File.ReadAllTextAsync(startupScriptPath);
       var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(startServerText);

       var amiLatest = await _amazonSsm.GetParameterAsync(
           new GetParameterRequest() { Name = _amiParam });
       var amiId = amiLatest.Parameter.Value;
       var launchTemplateResponse = await _amazonEc2.CreateLaunchTemplateAsync(
           new CreateLaunchTemplateRequest()
           {
               LaunchTemplateName = _launchTemplateName,
               LaunchTemplateData = new RequestLaunchTemplateData()
               {
                   InstanceType = _instanceType,
                   ImageId = amiId,
                   IamInstanceProfile =
                       new

LaunchTemplateIamInstanceProfileSpecificationRequest()
                       {
                           Name = _instanceProfileName
                       },
                   KeyName = _keyPairName,
                   UserData = System.Convert.ToBase64String(plainTextBytes)
               }
           });
       return launchTemplateResponse.LaunchTemplate;
```

```
    }

    /// <summary>
    /// Get a list of Availability Zones in the AWS Region of the Amazon EC2
 Client.
    /// </summary>
    /// <returns>A list of availability zones.</returns>
    public async Task<List<string>> DescribeAvailabilityZones()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <summary>
    /// Create an EC2 Auto Scaling group of a specified size and name.
    /// </summary>
    /// <param name="groupSize">The size for the group.</param>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="availabilityZones">The availability zones for the group.</
param>
    /// <returns>Async task.</returns>
    public async Task CreateGroupOfSize(int groupSize, string groupName,
 List<string> availabilityZones)
    {
        try
        {
            await _amazonAutoScaling.CreateAutoScalingGroupAsync(
                new CreateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    AvailabilityZones = availabilityZones,
                    LaunchTemplate =
                        new
 Amazon.AutoScaling.Model.LaunchTemplateSpecification()
                        {
                            LaunchTemplateName = _launchTemplateName,
                            Version = "$Default"
                        },
                    MaxSize = groupSize,
                    MinSize = groupSize
                });
```

```
            Console.WriteLine($"Created EC2 Auto Scaling group {groupName} with
size {groupSize}.");
        }
        catch (EntityAlreadyExistsException)
        {
            Console.WriteLine($"EC2 Auto Scaling group {groupName} already
exists.");
        }
    }

    /// <summary>
    /// Get the default VPC for the account.
    /// </summary>
    /// <returns>The default VPC object.</returns>
    public async Task<Vpc> GetDefaultVpc()
    {
        var vpcResponse = await _amazonEc2.DescribeVpcsAsync(
            new DescribeVpcsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("is-default", new List<string>() { "true" })
                }
            });
        return vpcResponse.Vpcs[0];
    }

    /// <summary>
    /// Get all the subnets for a Vpc in a set of availability zones.
    /// </summary>
    /// <param name="vpcId">The Id of the Vpc.</param>
    /// <param name="availabilityZones">The list of availability zones.</param>
    /// <returns>The collection of subnet objects.</returns>
    public async Task<List<Subnet>> GetAllVpcSubnetsForZones(string vpcId,
List<string> availabilityZones)
    {
        var subnets = new List<Subnet>();
        var subnetPaginator = _amazonEc2.Paginators.DescribeSubnets(
            new DescribeSubnetsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("vpc-id", new List<string>() { vpcId}),
                    new ("availability-zone", availabilityZones),
```

```
                    new ("default-for-az", new List<string>() { "true" })
                }
            });

        // Get the entire list using the paginator.
        await foreach (var subnet in subnetPaginator.Subnets)
        {
            subnets.Add(subnet);
        }

        return subnets;
    }

    /// <summary>
    /// Delete a launch template by name.
    /// </summary>
    /// <param name="templateName">The name of the template to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTemplateByName(string templateName)
    {
        try
        {
            await _amazonEc2.DeleteLaunchTemplateAsync(
                new DeleteLaunchTemplateRequest()
                {
                    LaunchTemplateName = templateName
                });
        }
        catch (AmazonClientException)
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <summary>
    /// Detaches a role from an instance profile, detaches policies from the
role,
    /// and deletes all the resources.
    /// </summary>
    /// <param name="profileName">The name of the profile to delete.</param>
    /// <param name="roleName">The name of the role to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteInstanceProfile(string profileName, string roleName)
    {
```

```
        try
        {
            await _amazonIam.RemoveRoleFromInstanceProfileAsync(
                new RemoveRoleFromInstanceProfileRequest()
                {
                    InstanceProfileName = profileName,
                    RoleName = roleName
                });
            await _amazonIam.DeleteInstanceProfileAsync(
                new DeleteInstanceProfileRequest() { InstanceProfileName =
profileName });
            var attachedPolicies = await
_amazonIam.ListAttachedRolePoliciesAsync(
                new ListAttachedRolePoliciesRequest() { RoleName = roleName });
            foreach (var policy in attachedPolicies.AttachedPolicies)
            {
                await _amazonIam.DetachRolePolicyAsync(
                    new DetachRolePolicyRequest()
                    {
                        RoleName = roleName,
                        PolicyArn = policy.PolicyArn
                    });
                // Delete the custom policies only.
                if (!policy.PolicyArn.StartsWith("arn:aws:iam::aws"))
                {
                    await _amazonIam.DeletePolicyAsync(
                        new Amazon.IdentityManagement.Model.DeletePolicyRequest()
                        {
                            PolicyArn = policy.PolicyArn
                        });
                }
            }

            await _amazonIam.DeleteRoleAsync(
                new DeleteRoleRequest() { RoleName = roleName });
        }
        catch (NoSuchEntityException)
        {
            Console.WriteLine($"Instance profile {profileName} does not exist.");
        }
    }

    /// <summary>
```

```
    /// Gets data about the instances in an EC2 Auto Scaling group by its group
name.
    /// </summary>
    /// <param name="group">The name of the auto scaling group.</param>
    /// <returns>A collection of instance Ids.</returns>
    public async Task<IEnumerable<string>> GetInstancesByGroupName(string group)
    {
        var instanceResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
            new DescribeAutoScalingGroupsRequest()
            {
                AutoScalingGroupNames = new List<string>() { group }
            });
        var instanceIds = instanceResponse.AutoScalingGroups.SelectMany(
            g => g.Instances.Select(i => i.InstanceId));
        return instanceIds;
    }

    /// <summary>
    /// Get the instance profile association data for an instance.
    /// </summary>
    /// <param name="instanceId">The Id of the instance.</param>
    /// <returns>Instance profile associations data.</returns>
    public async Task<IamInstanceProfileAssociation> GetInstanceProfile(string
instanceId)
    {
        var response = await
_amazonEc2.DescribeIamInstanceProfileAssociationsAsync(
            new DescribeIamInstanceProfileAssociationsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("instance-id", new List<string>() { instanceId })
                },
            });
        return response.IamInstanceProfileAssociations[0];
    }

    /// <summary>
    /// Replace the profile associated with a running instance. After the profile
is replaced, the instance
    /// is rebooted to ensure that it uses the new profile. When the instance is
ready, Systems Manager is
    /// used to restart the Python web server.
```

```
    /// </summary>
    /// <param name="instanceId">The Id of the instance to update.</param>
    /// <param name="credsProfileName">The name of the new profile to associate
with the specified instance.</param>
    /// <param name="associationId">The Id of the existing profile association
for the instance.</param>
    /// <returns>Async task.</returns>
    public async Task ReplaceInstanceProfile(string instanceId, string
credsProfileName, string associationId)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);

            var instancesPaginator =
_amazonSsm.Paginators.DescribeInstanceInformation(
                new DescribeInstanceInformationRequest());
            // Get the entire list using the paginator.
            await foreach (var instance in
instancesPaginator.InstanceInformationList)
            {
                instanceReady = instance.InstanceId == instanceId;
                if (instanceReady)
                {
                    break;
                }
            }
        }
        Console.WriteLine($"Sending restart command to instance {instanceId}");
```

```
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <summary>
    /// Try to terminate an instance by its Id.
    /// </summary>
    /// <param name="instanceId">The Id of the instance to terminate.</param>
    /// <returns>Async task.</returns>
    public async Task TryTerminateInstanceById(string instanceId)
    {
        var stopping = false;
        Console.WriteLine($"Stopping {instanceId}...");
        while (!stopping)
        {
            try
            {
                await
_amazonAutoScaling.TerminateInstanceInAutoScalingGroupAsync(
                    new TerminateInstanceInAutoScalingGroupRequest()
                    {
                        InstanceId = instanceId,
                        ShouldDecrementDesiredCapacity = false
                    });
                stopping = true;
            }
            catch (ScalingActivityInProgressException)
            {
                Console.WriteLine($"Scaling activity in progress for
{instanceId}. Waiting...");
                Thread.Sleep(10000);
            }
        }
    }
```

```
   /// <summary>
   /// Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
   /// waits and retries until the group is successfully deleted.
   /// </summary>
   /// <param name="groupName">The name of the group to try to delete.</param>
   /// <returns>Async task.</returns>
   public async Task TryDeleteGroupByName(string groupName)
   {
       var stopped = false;
       while (!stopped)
       {
           try
           {
               await _amazonAutoScaling.DeleteAutoScalingGroupAsync(
                   new DeleteAutoScalingGroupRequest()
                   {
                       AutoScalingGroupName = groupName
                   });
               stopped = true;
           }
           catch (Exception e)
               when ((e is ScalingActivityInProgressException)
                     || (e is Amazon.AutoScaling.Model.ResourceInUseException))
           {
               Console.WriteLine($"Some instances are still running.
Waiting...");
               Thread.Sleep(10000);
           }
       }
   }

   /// <summary>
   /// Terminate instances and delete the Auto Scaling group by name.
   /// </summary>
   /// <param name="groupName">The name of the group to delete.</param>
   /// <returns>Async task.</returns>
   public async Task TerminateAndDeleteAutoScalingGroupWithName(string
groupName)
   {
       var describeGroupsResponse = await
_amazonAutoScaling.DescribeAutoScalingGroupsAsync(
           new DescribeAutoScalingGroupsRequest()
```

```
            {
                AutoScalingGroupNames = new List<string>() { groupName }
            });
        if (describeGroupsResponse.AutoScalingGroups.Any())
        {
            // Update the size to 0.
            await _amazonAutoScaling.UpdateAutoScalingGroupAsync(
                new UpdateAutoScalingGroupRequest()
                {
                    AutoScalingGroupName = groupName,
                    MinSize = 0
                });
            var group = describeGroupsResponse.AutoScalingGroups[0];
            foreach (var instance in group.Instances)
            {
                await TryTerminateInstanceById(instance.InstanceId);
            }

            await TryDeleteGroupByName(groupName);
        }
        else
        {
            Console.WriteLine($"No groups found with name {groupName}.");
        }
    }


    /// <summary>
    /// Get the default security group for a specified Vpc.
    /// </summary>
    /// <param name="vpc">The Vpc to search.</param>
    /// <returns>The default security group.</returns>
    public async Task<SecurityGroup> GetDefaultSecurityGroupForVpc(Vpc vpc)
    {
        var groupResponse = await _amazonEc2.DescribeSecurityGroupsAsync(
            new DescribeSecurityGroupsRequest()
            {
                Filters = new List<Amazon.EC2.Model.Filter>()
                {
                    new ("group-name", new List<string>() { "default" }),
                    new ("vpc-id", new List<string>() { vpc.VpcId })
                }
            });
        return groupResponse.SecurityGroups[0];
```

```
    }

    /// <summary>
    /// Verify the default security group of a Vpc allows ingress from the
calling computer.
    /// This can be done by allowing ingress from this computer's IP address.
    /// In some situations, such as connecting from a corporate network, you must
instead specify
    /// a prefix list Id. You can also temporarily open the port to any IP
address while running this example.
    /// If you do, be sure to remove public access when you're done.
    /// </summary>
    /// <param name="vpc">The group to check.</param>
    /// <param name="port">The port to verify.</param>
    /// <param name="ipAddress">This computer's IP address.</param>
    /// <returns>True if the ip address is allowed on the group.</returns>
    public bool VerifyInboundPortForGroup(SecurityGroup group, int port, string
ipAddress)
    {
        var portIsOpen = false;
        foreach (var ipPermission in group.IpPermissions)
        {
            if (ipPermission.FromPort == port)
            {
                foreach (var ipRange in ipPermission.Ipv4Ranges)
                {
                    var cidr = ipRange.CidrIp;
                    if (cidr.StartsWith(ipAddress) || cidr == "0.0.0.0/0")
                    {
                        portIsOpen = true;
                    }
                }

                if (ipPermission.PrefixListIds.Any())
                {
                    portIsOpen = true;
                }

                if (!portIsOpen)
                {
                    Console.WriteLine("The inbound rule does not appear to be
open to either this computer's IP\n" +
                                      "address, to all IP addresses (0.0.0.0/0),
or to a prefix list ID.");
```

```
                }
                else
                {
                    break;
                }
            }
        }

        return portIsOpen;
    }

    /// <summary>
    /// Add an ingress rule to the specified security group that allows access on
the
    /// specified port from the specified IP address.
    /// </summary>
    /// <param name="groupId">The Id of the security group to modify.</param>
    /// <param name="port">The port to open.</param>
    /// <param name="ipAddress">The IP address to allow access.</param>
    /// <returns>Async task.</returns>
    public async Task OpenInboundPort(string groupId, int port, string ipAddress)
    {
        await _amazonEc2.AuthorizeSecurityGroupIngressAsync(
            new AuthorizeSecurityGroupIngressRequest()
            {
                GroupId = groupId,
                IpPermissions = new List<IpPermission>()
                {
                    new IpPermission()
                    {
                        FromPort = port,
                        ToPort = port,
                        IpProtocol = "tcp",
                        Ipv4Ranges = new List<IpRange>()
                        {
                            new IpRange() { CidrIp = $"{ipAddress}/32" }
                        }
                    }
                }
            });
    }

    /// <summary>
```

```
    /// Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
 Scaling group.
    /// The
    /// </summary>
    /// <param name="autoScalingGroupName">The name of the Auto Scaling group.</
param>
    /// <param name="targetGroupArn">The Arn for the target group.</param>
    /// <returns>Async task.</returns>
    public async Task AttachLoadBalancerToGroup(string autoScalingGroupName,
 string targetGroupArn)
    {
        await _amazonAutoScaling.AttachLoadBalancerTargetGroupsAsync(
            new AttachLoadBalancerTargetGroupsRequest()
            {
                AutoScalingGroupName = autoScalingGroupName,
                TargetGroupARNs = new List<string>() { targetGroupArn }
            });
    }
}
```

Cree una clase que resuma las acciones de Elastic Load Balancing.

```
/// <summary>
/// Encapsulates Elastic Load Balancer actions.
/// </summary>
public class ElasticLoadBalancerWrapper
{
    private readonly IAmazonElasticLoadBalancingV2 _amazonElasticLoadBalancingV2;
    private string? _endpoint = null;
    private readonly string _targetGroupName = "";
    private readonly string _loadBalancerName = "";
    HttpClient _httpClient = new();

    public string TargetGroupName => _targetGroupName;
    public string LoadBalancerName => _loadBalancerName;

    /// <summary>
    /// Constructor for the Elastic Load Balancer wrapper.
    /// </summary>
    /// <param name="amazonElasticLoadBalancingV2">The injected load balancing v2
 client.</param>
```

```
    /// <param name="configuration">The injected configuration.</param>
    public ElasticLoadBalancerWrapper(
        IAmazonElasticLoadBalancingV2 amazonElasticLoadBalancingV2,
        IConfiguration configuration)
    {
        _amazonElasticLoadBalancingV2 = amazonElasticLoadBalancingV2;
        var prefix = configuration["resourcePrefix"];
        _targetGroupName = prefix + "-tg";
        _loadBalancerName = prefix + "-lb";
    }

    /// <summary>
    /// Get the HTTP Endpoint of a load balancer by its name.
    /// </summary>
    /// <param name="loadBalancerName">The name of the load balancer.</param>
    /// <returns>The HTTP endpoint.</returns>
    public async Task<string> GetEndpointForLoadBalancerByName(string
loadBalancerName)
    {
        if (_endpoint == null)
        {
            var endpointResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { loadBalancerName }
                    });
            _endpoint = endpointResponse.LoadBalancers[0].DNSName;
        }

        return _endpoint;
    }

    /// <summary>
    /// Return the GET response for an endpoint as text.
    /// </summary>
    /// <param name="endpoint">The endpoint for the request.</param>
    /// <returns>The request response.</returns>
    public async Task<string> GetEndPointResponse(string endpoint)
    {
        var endpointResponse = await _httpClient.GetAsync($"http://{endpoint}");
        var textResponse = await endpointResponse.Content.ReadAsStringAsync();
        return textResponse!;
    }
```

```
    /// <summary>
    /// Get the target health for a group by name.
    /// </summary>
    /// <param name="groupName">The name of the group.</param>
    /// <returns>The collection of health descriptions.</returns>
    public async Task<List<TargetHealthDescription>>
CheckTargetHealthForGroup(string groupName)
    {
        List<TargetHealthDescription> result = null!;
        try
        {
            var groupResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                    new DescribeTargetGroupsRequest()
                    {
                        Names = new List<string>() { groupName }
                    });
            var healthResponse =
                await _amazonElasticLoadBalancingV2.DescribeTargetHealthAsync(
                    new DescribeTargetHealthRequest()
                    {
                        TargetGroupArn =
groupResponse.TargetGroups[0].TargetGroupArn
                    });
            ;
            result = healthResponse.TargetHealthDescriptions;
        }
        catch (TargetGroupNotFoundException)
        {
            Console.WriteLine($"Target group {groupName} not found.");
        }
        return result;
    }

    /// <summary>
    /// Create an Elastic Load Balancing target group. The target group specifies
how the load balancer forwards
    /// requests to instances in the group and how instance health is checked.
    ///
    /// To speed up this demo, the health check is configured with shortened
times and lower thresholds. In production,
    /// you might want to decrease the sensitivity of your health checks to avoid
unwanted failures.
```

```
    /// </summary>
    /// <param name="groupName">The name for the group.</param>
    /// <param name="protocol">The protocol, such as HTTP.</param>
    /// <param name="port">The port to use to forward requests, such as 80.</
param>
    /// <param name="vpcId">The Id of the Vpc in which the load balancer
 exists.</param>
    /// <returns>The new TargetGroup object.</returns>
    public async Task<TargetGroup> CreateTargetGroupOnVpc(string groupName,
 ProtocolEnum protocol, int port, string vpcId)
    {
        var createResponse = await
 _amazonElasticLoadBalancingV2.CreateTargetGroupAsync(
            new CreateTargetGroupRequest()
            {
                Name = groupName,
                Protocol = protocol,
                Port = port,
                HealthCheckPath = "/healthcheck",
                HealthCheckIntervalSeconds = 10,
                HealthCheckTimeoutSeconds = 5,
                HealthyThresholdCount = 2,
                UnhealthyThresholdCount = 2,
                VpcId = vpcId
            });
        var targetGroup = createResponse.TargetGroups[0];
        return targetGroup;
    }

    /// <summary>
    /// Create an Elastic Load Balancing load balancer that uses the specified
 subnets
    /// and forwards requests to the specified target group.
    /// </summary>
    /// <param name="name">The name for the new load balancer.</param>
    /// <param name="subnetIds">Subnets for the load balancer.</param>
    /// <param name="targetGroup">Target group for forwarded requests.</param>
    /// <returns>The new LoadBalancer object.</returns>
    public async Task<LoadBalancer> CreateLoadBalancerAndListener(string name,
 List<string> subnetIds, TargetGroup targetGroup)
    {
        var createLbResponse = await
 _amazonElasticLoadBalancingV2.CreateLoadBalancerAsync(
            new CreateLoadBalancerRequest()
```

```
            {
                Name = name,
                Subnets = subnetIds
            });
        var loadBalancerArn = createLbResponse.LoadBalancers[0].LoadBalancerArn;

        // Wait for load balancer to be available.
        var loadBalancerReady = false;
        while (!loadBalancerReady)
        {
            try
            {
                var describeResponse =
                    await
_amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                        new DescribeLoadBalancersRequest()
                        {
                            Names = new List<string>() { name }
                        });

                var loadBalancerState =
describeResponse.LoadBalancers[0].State.Code;

                loadBalancerReady = loadBalancerState ==
LoadBalancerStateEnum.Active;
            }
            catch (LoadBalancerNotFoundException)
            {
                loadBalancerReady = false;
            }
            Thread.Sleep(10000);
        }
        // Create the listener.
        await _amazonElasticLoadBalancingV2.CreateListenerAsync(
            new CreateListenerRequest()
            {
                LoadBalancerArn = loadBalancerArn,
                Protocol = targetGroup.Protocol,
                Port = targetGroup.Port,
                DefaultActions = new List<Action>()
                {
                    new Action()
                    {
                        Type = ActionTypeEnum.Forward,
```

```
                        TargetGroupArn = targetGroup.TargetGroupArn
                    }
                }
            });
        return createLbResponse.LoadBalancers[0];
    }

    /// <summary>
    /// Verify this computer can successfully send a GET request to the
    /// load balancer endpoint.
    /// </summary>
    /// <param name="endpoint">The endpoint to check.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> VerifyLoadBalancerEndpoint(string endpoint)
    {
        var success = false;
        var retries = 3;
        while (!success && retries > 0)
        {
            try
            {
                var endpointResponse = await _httpClient.GetAsync($"http://
{endpoint}");
                Console.WriteLine($"Response: {endpointResponse.StatusCode}.");

                if (endpointResponse.IsSuccessStatusCode)
                {
                    success = true;
                }
                else
                {
                    retries = 0;
                }
            }
            catch (HttpRequestException)
            {
                Console.WriteLine("Connection error, retrying...");
                retries--;
                Thread.Sleep(10000);
            }
        }

        return success;
    }
```

```
    /// <summary>
    /// Delete a load balancer by its specified name.
    /// </summary>
    /// <param name="name">The name of the load balancer to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteLoadBalancerByName(string name)
    {
        try
        {
            var describeLoadBalancerResponse =
                await _amazonElasticLoadBalancingV2.DescribeLoadBalancersAsync(
                    new DescribeLoadBalancersRequest()
                    {
                        Names = new List<string>() { name }
                    });
            var lbArn =
describeLoadBalancerResponse.LoadBalancers[0].LoadBalancerArn;
            await _amazonElasticLoadBalancingV2.DeleteLoadBalancerAsync(
                new DeleteLoadBalancerRequest()
                {
                    LoadBalancerArn = lbArn
                }
            );
        }
        catch (LoadBalancerNotFoundException)
        {
            Console.WriteLine($"Load balancer {name} not found.");
        }
    }

    /// <summary>
    /// Delete a TargetGroup by its specified name.
    /// </summary>
    /// <param name="groupName">Name of the group to delete.</param>
    /// <returns>Async task.</returns>
    public async Task DeleteTargetGroupByName(string groupName)
    {
        var done = false;
        while (!done)
        {
            try
            {
                var groupResponse =
```

```
                    await
 _amazonElasticLoadBalancingV2.DescribeTargetGroupsAsync(
                        new DescribeTargetGroupsRequest()
                        {
                            Names = new List<string>() { groupName }
                        });

                var targetArn = groupResponse.TargetGroups[0].TargetGroupArn;
                await _amazonElasticLoadBalancingV2.DeleteTargetGroupAsync(
                    new DeleteTargetGroupRequest() { TargetGroupArn =
 targetArn });
                Console.WriteLine($"Deleted load balancing target group
 {groupName}.");
                done = true;
            }
            catch (TargetGroupNotFoundException)
            {
                Console.WriteLine(
                    $"Target group {groupName} not found, could not delete.");
                done = true;
            }
            catch (ResourceInUseException)
            {
                Console.WriteLine("Target group not yet released, waiting...");
                Thread.Sleep(10000);
            }
        }
    }
}
```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```
/// <summary>
/// Encapsulates a DynamoDB table to use as a service that recommends books,
 movies, and songs.
/// </summary>
public class Recommendations
{
    private readonly IAmazonDynamoDB _amazonDynamoDb;
    private readonly DynamoDBContext _context;
    private readonly string _tableName;
```

```
    public string TableName => _tableName;

    /// <summary>
    /// Constructor for the Recommendations service.
    /// </summary>
    /// <param name="amazonDynamoDb">The injected DynamoDb client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public Recommendations(IAmazonDynamoDB amazonDynamoDb, IConfiguration
configuration)
    {
        _amazonDynamoDb = amazonDynamoDb;
        _context = new DynamoDBContext(_amazonDynamoDb);
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Create the DynamoDb table with a specified name.
    /// </summary>
    /// <param name="tableName">The name for the table.</param>
    /// <returns>True when ready.</returns>
    public async Task<bool> CreateDatabaseWithName(string tableName)
    {
        try
        {
            Console.Write($"Creating table {tableName}...");
            var createRequest = new CreateTableRequest()
            {
                TableName = tableName,
                AttributeDefinitions = new List<AttributeDefinition>()
                    {
                        new AttributeDefinition()
                        {
                            AttributeName = "MediaType",
                            AttributeType = ScalarAttributeType.S
                        },
                        new AttributeDefinition()
                        {
                            AttributeName = "ItemId",
                            AttributeType = ScalarAttributeType.N
                        }
                    },
                KeySchema = new List<KeySchemaElement>()
                    {
                        new KeySchemaElement()
```

```
                        {
                            AttributeName = "MediaType",
                            KeyType = KeyType.HASH
                        },
                        new KeySchemaElement()
                        {
                            AttributeName = "ItemId",
                            KeyType = KeyType.RANGE
                        }
                    },
                ProvisionedThroughput = new ProvisionedThroughput()
                {
                    ReadCapacityUnits = 5,
                    WriteCapacityUnits = 5
                }
            };
            await _amazonDynamoDb.CreateTableAsync(createRequest);

            // Wait until the table is ACTIVE and then report success.
            Console.Write("\nWaiting for table to become active...");

            var request = new DescribeTableRequest
            {
                TableName = tableName
            };

            TableStatus status;
            do
            {
                Thread.Sleep(2000);

                var describeTableResponse = await
_amazonDynamoDb.DescribeTableAsync(request);
                status = describeTableResponse.Table.TableStatus;

                Console.Write(".");
            }
            while (status != "ACTIVE");

            return status == TableStatus.ACTIVE;
        }
        catch (ResourceInUseException)
        {
            Console.WriteLine($"Table {tableName} already exists.");
```

```
            return false;
        }
    }

    /// <summary>
    /// Populate the database table with data from a specified path.
    /// </summary>
    /// <param name="databaseTableName">The name of the table.</param>
    /// <param name="recommendationsPath">The path of the recommendations data.</
param>
    /// <returns>Async task.</returns>
    public async Task PopulateDatabase(string databaseTableName, string
 recommendationsPath)
    {
        var recommendationsText = await
 File.ReadAllTextAsync(recommendationsPath);
        var records =

 JsonSerializer.Deserialize<RecommendationModel[]>(recommendationsText);
        var batchWrite = _context.CreateBatchWrite<RecommendationModel>();

        foreach (var record in records!)
        {
            batchWrite.AddPutItem(record);
        }

        await batchWrite.ExecuteAsync();
    }

    /// <summary>
    /// Delete the recommendation table by name.
    /// </summary>
    /// <param name="tableName">The name of the recommendation table.</param>
    /// <returns>Async task.</returns>
    public async Task DestroyDatabaseByName(string tableName)
    {
        try
        {
            await _amazonDynamoDb.DeleteTableAsync(
                new DeleteTableRequest() { TableName = tableName });
            Console.WriteLine($"Table {tableName} was deleted.");
        }
        catch (ResourceNotFoundException)
        {
```

```
            Console.WriteLine($"Table {tableName} not found");
        }
    }
}
```

Cree una clase que agrupe las acciones de Systems Manager.

```
/// <summary>
/// Encapsulates Systems Manager parameter operations. This example uses these
 parameters
/// to drive the demonstration of resilient architecture, such as failure of a
 dependency or
/// how the service responds to a health check.
/// </summary>
public class SmParameterWrapper
{
    private readonly IAmazonSimpleSystemsManagement
 _amazonSimpleSystemsManagement;

    private readonly string _tableParameter = "doc-example-resilient-
architecture-table";
    private readonly string _failureResponseParameter = "doc-example-resilient-
architecture-failure-response";
    private readonly string _healthCheckParameter = "doc-example-resilient-
architecture-health-check";
    private readonly string _tableName = "";

    public string TableParameter => _tableParameter;
    public string TableName => _tableName;
    public string HealthCheckParameter => _healthCheckParameter;
    public string FailureResponseParameter => _failureResponseParameter;

    /// <summary>
    /// Constructor for the SmParameterWrapper.
    /// </summary>
    /// <param name="amazonSimpleSystemsManagement">The injected Simple Systems
 Management client.</param>
    /// <param name="configuration">The injected configuration.</param>
    public SmParameterWrapper(IAmazonSimpleSystemsManagement
 amazonSimpleSystemsManagement, IConfiguration configuration)
    {
        _amazonSimpleSystemsManagement = amazonSimpleSystemsManagement;
```

```
        _tableName = configuration["databaseName"]!;
    }

    /// <summary>
    /// Reset the Systems Manager parameters to starting values for the demo.
    /// </summary>
    /// <returns>Async task.</returns>
    public async Task Reset()
    {
        await this.PutParameterByName(_tableParameter, _tableName);
        await this.PutParameterByName(_failureResponseParameter, "none");
        await this.PutParameterByName(_healthCheckParameter, "shallow");
    }

    /// <summary>
    /// Set the value of a named Systems Manager parameter.
    /// </summary>
    /// <param name="name">The name of the parameter.</param>
    /// <param name="value">The value to set.</param>
    /// <returns>Async task.</returns>
    public async Task PutParameterByName(string name, string value)
    {
        await _amazonSimpleSystemsManagement.PutParameterAsync(
            new PutParameterRequest() { Name = name, Value = value, Overwrite =
 true });
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup

- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute el escenario interactivo en un símbolo del sistema.

```
public class Main {

    public static final String fileName = "C:\\AWS\\resworkflow\
\recommendations.json"; // Modify file location.
    public static final String tableName = "doc-example-recommendation-service";
```

```
    public static final String startScript = "C:\\AWS\\resworkflow\
\server_startup_script.sh"; // Modify file location.
    public static final String policyFile = "C:\\AWS\\resworkflow\
\instance_policy.json"; // Modify file location.
    public static final String ssmJSON = "C:\\AWS\\resworkflow\
\ssm_only_policy.json"; // Modify file location.
    public static final String failureResponse = "doc-example-resilient-
architecture-failure-response";
    public static final String healthCheck = "doc-example-resilient-architecture-
health-check";
    public static final String templateName = "doc-example-resilience-template";
    public static final String roleName = "doc-example-resilience-role";
    public static final String policyName = "doc-example-resilience-pol";
    public static final String profileName = "doc-example-resilience-prof";

    public static final String badCredsProfileName = "doc-example-resilience-
prof-bc";

    public static final String targetGroupName = "doc-example-resilience-tg";
    public static final String autoScalingGroupName = "doc-example-resilience-
group";
    public static final String lbName = "doc-example-resilience-lb";
    public static final String protocol = "HTTP";
    public static final int port = 80;

    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws IOException,
 InterruptedException {
        Scanner in = new Scanner(System.in);
        Database database = new Database();
        AutoScaler autoScaler = new AutoScaler();
        LoadBalancer loadBalancer = new LoadBalancer();

        System.out.println(DASHES);
        System.out.println("Welcome to the demonstration of How to Build and
 Manage a Resilient Service!");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("A - SETUP THE RESOURCES");
        System.out.println("Press Enter when you're ready to start deploying
 resources.");
```

```
        in.nextLine();
        deploy(loadBalancer);
        System.out.println(DASHES);
        System.out.println(DASHES);
        System.out.println("B - DEMO THE RESILIENCE FUNCTIONALITY");
        System.out.println("Press Enter when you're ready.");
        in.nextLine();
        demo(loadBalancer);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("C - DELETE THE RESOURCES");
        System.out.println("""
                This concludes the demo of how to build and manage a resilient
service.
                To keep things tidy and to avoid unwanted charges on your
account, we can clean up all AWS resources
                that were created for this demo.
                """);

        System.out.println("\n Do you want to delete the resources (y/n)? ");
        String userInput = in.nextLine().trim().toLowerCase(); // Capture user
input

        if (userInput.equals("y")) {
            // Delete resources here
            deleteResources(loadBalancer, autoScaler, database);
            System.out.println("Resources deleted.");
        } else {
            System.out.println("""
                    Okay, we'll leave the resources intact.
                    Don't forget to delete them when you're done with them or you
might incur unexpected charges.
                    """);
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("The example has completed. ");
        System.out.println("\n Thanks for watching!");
        System.out.println(DASHES);
    }

    // Deletes the AWS resources used in this example.
```

```
   private static void deleteResources(LoadBalancer loadBalancer, AutoScaler
autoScaler, Database database)
           throws IOException, InterruptedException {
       loadBalancer.deleteLoadBalancer(lbName);
       System.out.println("*** Wait 30 secs for resource to be deleted");
       TimeUnit.SECONDS.sleep(30);
       loadBalancer.deleteTargetGroup(targetGroupName);
       autoScaler.deleteAutoScaleGroup(autoScalingGroupName);
       autoScaler.deleteRolesPolicies(policyName, roleName, profileName);
       autoScaler.deleteTemplate(templateName);
       database.deleteTable(tableName);
   }

   private static void deploy(LoadBalancer loadBalancer) throws
InterruptedException, IOException {
       Scanner in = new Scanner(System.in);
       System.out.println(
               """
                       For this demo, we'll use the AWS SDK for Java (v2) to
create several AWS resources
                       to set up a load-balanced web service endpoint and
explore some ways to make it resilient
                       against various kinds of failures.

                       Some of the resources create by this demo are:
                       \t* A DynamoDB table that the web service depends on to
provide book, movie, and song recommendations.
                       \t* An EC2 launch template that defines EC2 instances
that each contain a Python web server.
                       \t* An EC2 Auto Scaling group that manages EC2 instances
across several Availability Zones.
                       \t* An Elastic Load Balancing (ELB) load balancer that
targets the Auto Scaling group to distribute requests.
                       """);

       System.out.println("Press Enter when you're ready.");
       in.nextLine();
       System.out.println(DASHES);

       System.out.println(DASHES);
       System.out.println("Creating and populating a DynamoDB table named " +
tableName);
       Database database = new Database();
       database.createTable(tableName, fileName);
```

```
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.
                This script starts a Python web server defined in the `server.py`
script. The web server
                listens to HTTP requests on port 80 and responds to requests to
'/' and to '/healthcheck'.
                For demo purposes, this server is run as the root user. In
production, the best practice is to
                run a web server, such as Apache, with least-privileged
credentials.

                The template also defines an IAM policy that each instance uses
to assume a role that grants
                permissions to access the DynamoDB recommendation table and
Systems Manager parameters
                that control the flow of the demo.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        templateCreator.createTemplate(policyFile, policyName, profileName,
startScript, templateName, roleName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println(
                "Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different Availability Zone.");
        System.out.println("*** Wait 30 secs for the VPC to be created");
        TimeUnit.SECONDS.sleep(30);
        AutoScaler autoScaler = new AutoScaler();
        String[] zones = autoScaler.createGroup(3, templateName,
autoScalingGroupName);

        System.out.println("""
                At this point, you have EC2 instances created. Once each instance
starts, it listens for
                HTTP requests. You can see these instances in the console or
continue with the demo.
                Press Enter when you're ready to continue.
                """);
```

```
        in.nextLine();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Creating variables that control the flow of the
demo.");
        ParameterHelper paramHelper = new ParameterHelper();
        paramHelper.reset();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("""
                Creating an Elastic Load Balancing target group and load
balancer. The target group
                defines how the load balancer connects to instances. The load
balancer provides a
                single endpoint where clients connect and dispatches requests to
instances in the group.
                """);

        String vpcId = autoScaler.getDefaultVPC();
        List<Subnet> subnets = autoScaler.getSubnets(vpcId, zones);
        System.out.println("You have retrieved a list with " + subnets.size() + "
subnets");
        String targetGroupArn = loadBalancer.createTargetGroup(protocol, port,
vpcId, targetGroupName);
        String elbDnsName = loadBalancer.createLoadBalancer(subnets,
targetGroupArn, lbName, port, protocol);
        autoScaler.attachLoadBalancerTargetGroup(autoScalingGroupName,
targetGroupArn);
        System.out.println("Verifying access to the load balancer endpoint...");
        boolean wasSuccessul =
loadBalancer.verifyLoadBalancerEndpoint(elbDnsName);
        if (!wasSuccessul) {
            System.out.println("Couldn't connect to the load balancer, verifying
that the port is open...");
            CloseableHttpClient httpClient = HttpClients.createDefault();

            // Create an HTTP GET request to "http://checkip.amazonaws.com"
            HttpGet httpGet = new HttpGet("http://checkip.amazonaws.com");
            try {
                // Execute the request and get the response
                HttpResponse response = httpClient.execute(httpGet);
```

```
                // Read the response content.
                String ipAddress =
IOUtils.toString(response.getEntity().getContent(),
StandardCharsets.UTF_8).trim();

                // Print the public IP address.
                System.out.println("Public IP Address: " + ipAddress);
                GroupInfo groupInfo = autoScaler.verifyInboundPort(vpcId, port,
ipAddress);
                if (!groupInfo.isPortOpen()) {
                    System.out.println("""
                            For this example to work, the default security group
for your default VPC must
                            allow access from this computer. You can either add
it automatically from this
                            example or add it yourself using the AWS Management
Console.
                            """);

                    System.out.println(
                            "Do you want to add a rule to security group " +
groupInfo.getGroupName() + " to allow");
                    System.out.println("inbound traffic on port " + port + " from
your computer's IP address (y/n) ");
                    String ans = in.nextLine();
                    if ("y".equalsIgnoreCase(ans)) {
                        autoScaler.openInboundPort(groupInfo.getGroupName(),
String.valueOf(port), ipAddress);
                        System.out.println("Security group rule added.");
                    } else {
                        System.out.println("No security group rule added.");
                    }
                }

            } catch (AutoScalingException e) {
                e.printStackTrace();
            }
        } else if (wasSuccessul) {
            System.out.println("Your load balancer is ready. You can access it by
browsing to:");
            System.out.println("\t http://" + elbDnsName);
        } else {
```

```
            System.out.println("Couldn't get a successful response from the load
balancer endpoint. Troubleshoot by");
            System.out.println("manually verifying that your VPC and security
group are configured correctly and that");
            System.out.println("you can successfully make a GET request to the
load balancer.");
        }

        System.out.println("Press Enter when you're ready to continue with the
demo.");
        in.nextLine();
    }

    // A method that controls the demo part of the Java program.
    public static void demo(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        ParameterHelper paramHelper = new ParameterHelper();
        System.out.println("Read the ssm_only_policy.json file");
        String ssmOnlyPolicy = readFileAsString(ssmJSON);

        System.out.println("Resetting parameters to starting values for demo.");
        paramHelper.reset();

        System.out.println(
                """
                        This part of the demonstration shows how to toggle
different parts of the system
                        to create situations where the web service fails, and
shows how using a resilient
                        architecture can keep the web service running in spite
of these failures.

                        At the start, the load balancer endpoint returns
recommendations and reports that all targets are healthy.
                       """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        The web service running on the EC2 instances gets
recommendations by querying a DynamoDB table.
                        The table name is contained in a Systems Manager
parameter named self.param_helper.table.
```

```
                        To simulate a failure of the recommendation service,
let's set this parameter to name a non-existent table.
                        """);
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");

        System.out.println(
                """
                        \nNow, sending a GET request to the load balancer
endpoint returns a failure code. But, the service reports as
                        healthy to the load balancer because shallow health
checks don't check for failure of the recommendation service.
                        """);
        demoChoices(loadBalancer);

        System.out.println(
                """
                        Instead of failing when the recommendation service fails,
the web service can return a static response.
                        While this is not a perfect solution, it presents the
customer with a somewhat better experience than failure.
                        """);
        paramHelper.put(paramHelper.failureResponse, "static");

        System.out.println("""
                Now, sending a GET request to the load balancer endpoint returns
a static response.
                The service still reports as healthy because health checks are
still shallow.
                """);
        demoChoices(loadBalancer);

        System.out.println("Let's reinstate the recommendation service.");
        paramHelper.put(paramHelper.tableName, paramHelper.dyntable);

        System.out.println("""
                Let's also substitute bad credentials for one of the instances in
the target group so that it can't
                access the DynamoDB recommendation table. We will get an instance
id value.
                """);

        LaunchTemplateCreator templateCreator = new LaunchTemplateCreator();
        AutoScaler autoScaler = new AutoScaler();
```

```
        // Create a new instance profile based on badCredsProfileName.
        templateCreator.createInstanceProfile(policyFile, policyName,
badCredsProfileName, roleName);
        String badInstanceId = autoScaler.getBadInstance(autoScalingGroupName);
        System.out.println("The bad instance id values used for this demo is " +
badInstanceId);

        String profileAssociationId =
autoScaler.getInstanceProfile(badInstanceId);
        System.out.println("The association Id value is " +
profileAssociationId);
        System.out.println("Replacing the profile for instance " + badInstanceId
                + " with a profile that contains bad credentials");
        autoScaler.replaceInstanceProfile(badInstanceId, badCredsProfileName,
profileAssociationId);

        System.out.println(
                """
                        Now, sending a GET request to the load balancer endpoint
returns either a recommendation or a static response,
                        depending on which instance is selected by the load
balancer.
                        """);

        demoChoices(loadBalancer);

        System.out.println("""
                Let's implement a deep health check. For this demo, a deep health
check tests whether
                the web service can access the DynamoDB table that it depends on
for recommendations. Note that
                the deep health check is only for ELB routing and not for Auto
Scaling instance health.
                This kind of deep health check is not recommended for Auto
Scaling instance health, because it
                risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.
                """);

        System.out.println("""
                By implementing deep health checks, the load balancer can detect
when one of the instances is failing
                and take that instance out of rotation.
                """);
```

```
        paramHelper.put(paramHelper.healthCheck, "deep");

        System.out.println("""
                Now, checking target health indicates that the instance with bad
credentials
                is unhealthy. Note that it might take a minute or two for the
load balancer to detect the unhealthy
                instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because
                the load balancer takes unhealthy instances out of its rotation.
                """);

        demoChoices(loadBalancer);

        System.out.println(
                """
                        Because the instances in this demo are controlled by an
auto scaler, the simplest way to fix an unhealthy
                        instance is to terminate it and let the auto scaler start
a new instance to replace it.
                        """);
        autoScaler.terminateInstance(badInstanceId);

        System.out.println("""
                Even while the instance is terminating and the new instance is
starting, sending a GET
                request to the web service continues to get a successful
recommendation response because
                the load balancer routes requests to the healthy instances. After
the replacement instance
                starts and reports as healthy, it is included in the load
balancing rotation.
                Note that terminating and replacing an instance typically takes
several minutes, during which time you
                can see the changing health check status until the new instance
is running and healthy.
                """);

        demoChoices(loadBalancer);
        System.out.println(
                "If the recommendation service fails now, deep health checks mean
all instances report as unhealthy.");
        paramHelper.put(paramHelper.tableName, "this-is-not-a-table");
```

```
        demoChoices(loadBalancer);
        paramHelper.reset();
    }

    public static void demoChoices(LoadBalancer loadBalancer) throws IOException,
InterruptedException {
        String[] actions = {
                "Send a GET request to the load balancer endpoint.",
                "Check the health of load balancer targets.",
                "Go to the next part of the demo."
        };
        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("-".repeat(88));
            System.out.println("See the current state of the service by selecting
one of the following choices:");
            for (int i = 0; i < actions.length; i++) {
                System.out.println(i + ": " + actions[i]);
            }

            try {
                System.out.print("\nWhich action would you like to take? ");
                int choice = scanner.nextInt();
                System.out.println("-".repeat(88));

                switch (choice) {
                    case 0 -> {
                        System.out.println("Request:\n");
                        System.out.println("GET http://" +
loadBalancer.getEndpoint(lbName));
                        CloseableHttpClient httpClient =
HttpClients.createDefault();

                        // Create an HTTP GET request to the ELB.
                        HttpGet httpGet = new HttpGet("http://" +
loadBalancer.getEndpoint(lbName));

                        // Execute the request and get the response.
                        HttpResponse response = httpClient.execute(httpGet);
                        int statusCode =
response.getStatusLine().getStatusCode();
                        System.out.println("HTTP Status Code: " + statusCode);
```

```
                    // Display the JSON response
                    BufferedReader reader = new BufferedReader(
                            new
InputStreamReader(response.getEntity().getContent()));
                    StringBuilder jsonResponse = new StringBuilder();
                    String line;
                    while ((line = reader.readLine()) != null) {
                        jsonResponse.append(line);
                    }
                    reader.close();

                    // Print the formatted JSON response.
                    System.out.println("Full Response:\n");
                    System.out.println(jsonResponse.toString());

                    // Close the HTTP client.
                    httpClient.close();

                }
                case 1 -> {
                    System.out.println("\nChecking the health of load
balancer targets:\n");
                    List<TargetHealthDescription> health =
loadBalancer.checkTargetHealth(targetGroupName);
                    for (TargetHealthDescription target : health) {
                        System.out.printf("\tTarget %s on port %d is %s%n",
target.target().id(),
                                target.target().port(),
target.targetHealth().stateAsString());
                    }
                    System.out.println("""
                            Note that it can take a minute or two for the
health check to update
                            after changes are made.
                            """);
                }
                case 2 -> {
                    System.out.println("\nOkay, let's move on.");
                    System.out.println("-".repeat(88));
                    return; // Exit the method when choice is 2
                }
                default -> System.out.println("You must choose a value
between 0-2. Please select again.");
```

```
                }

            } catch (java.util.InputMismatchException e) {
                System.out.println("Invalid input. Please select again.");
                scanner.nextLine(); // Clear the input buffer.
            }
        }
    }

    public static String readFileAsString(String filePath) throws IOException {
        byte[] bytes = Files.readAllBytes(Paths.get(filePath));
        return new String(bytes);
    }
}
```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```
public class AutoScaler {

    private static Ec2Client ec2Client;
    private static AutoScalingClient autoScalingClient;
    private static IamClient iamClient;

    private static SsmClient ssmClient;

    private IamClient getIAMClient() {
        if (iamClient == null) {
            iamClient = IamClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return iamClient;
    }

    private SsmClient getSSMClient() {
        if (ssmClient == null) {
            ssmClient = SsmClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return ssmClient;
    }
```

```
   private Ec2Client getEc2Client() {
       if (ec2Client == null) {
           ec2Client = Ec2Client.builder()
                   .region(Region.US_EAST_1)
                   .build();
       }
       return ec2Client;
   }

   private AutoScalingClient getAutoScalingClient() {
       if (autoScalingClient == null) {
           autoScalingClient = AutoScalingClient.builder()
                   .region(Region.US_EAST_1)
                   .build();
       }
       return autoScalingClient;
   }

   /**
    * Terminates and instances in an EC2 Auto Scaling group. After an instance
is
    * terminated, it can no longer be accessed.
    */
   public void terminateInstance(String instanceId) {
       TerminateInstanceInAutoScalingGroupRequest terminateInstanceIRequest =
TerminateInstanceInAutoScalingGroupRequest
               .builder()
               .instanceId(instanceId)
               .shouldDecrementDesiredCapacity(false)
               .build();


getAutoScalingClient().terminateInstanceInAutoScalingGroup(terminateInstanceIRequest);
       System.out.format("Terminated instance %s.", instanceId);
   }

   /**
    * Replaces the profile associated with a running instance. After the profile
is
    * replaced, the instance is rebooted to ensure that it uses the new profile.
    * When
    * the instance is ready, Systems Manager is used to restart the Python web
    * server.
```

```
    */
   public void replaceInstanceProfile(String instanceId, String
newInstanceProfileName, String profileAssociationId)
           throws InterruptedException {
       // Create an IAM instance profile specification.
       software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
iamInstanceProfile =
software.amazon.awssdk.services.ec2.model.IamInstanceProfileSpecification
               .builder()
               .name(newInstanceProfileName) // Make sure
'newInstanceProfileName' is a valid IAM Instance Profile
                                             // name.
               .build();

       // Replace the IAM instance profile association for the EC2 instance.
       ReplaceIamInstanceProfileAssociationRequest replaceRequest =
ReplaceIamInstanceProfileAssociationRequest
               .builder()
               .iamInstanceProfile(iamInstanceProfile)
               .associationId(profileAssociationId) // Make sure
'profileAssociationId' is a valid association ID.
               .build();

       try {
           getEc2Client().replaceIamInstanceProfileAssociation(replaceRequest);
           // Handle the response as needed.
       } catch (Ec2Exception e) {
           // Handle exceptions, log, or report the error.
           System.err.println("Error: " + e.getMessage());
       }
       System.out.format("Replaced instance profile for association %s with
profile %s.", profileAssociationId,
               newInstanceProfileName);
       TimeUnit.SECONDS.sleep(15);
       boolean instReady = false;
       int tries = 0;

       // Reboot after 60 seconds
       while (!instReady) {
           if (tries % 6 == 0) {
               getEc2Client().rebootInstances(RebootInstancesRequest.builder()
                       .instanceIds(instanceId)
                       .build());
```

```
                System.out.println("Rebooting instance " + instanceId + " and
waiting for it to be ready.");
            }
            tries++;
            try {
                TimeUnit.SECONDS.sleep(10);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }

            DescribeInstanceInformationResponse informationResponse =
getSSMClient().describeInstanceInformation();
            List<InstanceInformation> instanceInformationList =
informationResponse.instanceInformationList();
            for (InstanceInformation info : instanceInformationList) {
                if (info.instanceId().equals(instanceId)) {
                    instReady = true;
                    break;
                }
            }
        }

        SendCommandRequest sendCommandRequest = SendCommandRequest.builder()
                .instanceIds(instanceId)
                .documentName("AWS-RunShellScript")
                .parameters(Collections.singletonMap("commands",
                        Collections.singletonList("cd / && sudo python3 server.py
80")))
                .build();

        getSSMClient().sendCommand(sendCommandRequest);
        System.out.println("Restarted the Python web server on instance " +
instanceId + ".");
    }

    public void openInboundPort(String secGroupId, String port, String ipAddress)
{
        AuthorizeSecurityGroupIngressRequest ingressRequest =
AuthorizeSecurityGroupIngressRequest.builder()
                .groupName(secGroupId)
                .cidrIp(ipAddress)
                .fromPort(Integer.parseInt(port))
                .build();
```

```
        getEc2Client().authorizeSecurityGroupIngress(ingressRequest);
        System.out.format("Authorized ingress to %s on port %s from %s.",
secGroupId, port, ipAddress);
    }

    /**
     * Detaches a role from an instance profile, detaches policies from the role,
     * and deletes all the resources.
     */
    public void deleteInstanceProfile(String roleName, String profileName) {
        try {
            software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
getInstanceProfileRequest =
software.amazon.awssdk.services.iam.model.GetInstanceProfileRequest
                    .builder()
                    .instanceProfileName(profileName)
                    .build();

            GetInstanceProfileResponse response =
getIAMClient().getInstanceProfile(getInstanceProfileRequest);
            String name = response.instanceProfile().instanceProfileName();
            System.out.println(name);

            RemoveRoleFromInstanceProfileRequest profileRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .roleName(roleName)
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(profileRequest);
            DeleteInstanceProfileRequest deleteInstanceProfileRequest =
DeleteInstanceProfileRequest.builder()
                    .instanceProfileName(profileName)
                    .build();

            getIAMClient().deleteInstanceProfile(deleteInstanceProfileRequest);
            System.out.println("Deleted instance profile " + profileName);

            DeleteRoleRequest deleteRoleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            // List attached role policies.
            ListAttachedRolePoliciesResponse rolesResponse = getIAMClient()
```

```
                    .listAttachedRolePolicies(role -> role.roleName(roleName));
            List<AttachedPolicy> attachedPolicies =
rolesResponse.attachedPolicies();
            for (AttachedPolicy attachedPolicy : attachedPolicies) {
                DetachRolePolicyRequest request =
DetachRolePolicyRequest.builder()
                        .roleName(roleName)
                        .policyArn(attachedPolicy.policyArn())
                        .build();

                getIAMClient().detachRolePolicy(request);
                System.out.println("Detached and deleted policy " +
attachedPolicy.policyName());
            }

            getIAMClient().deleteRole(deleteRoleRequest);
            System.out.println("Instance profile and role deleted.");

        } catch (IamException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public void deleteTemplate(String templateName) {
        getEc2Client().deleteLaunchTemplate(name ->
name.launchTemplateName(templateName));
        System.out.format(templateName + " was deleted.");
    }

    public void deleteAutoScaleGroup(String groupName) {
        DeleteAutoScalingGroupRequest deleteAutoScalingGroupRequest =
DeleteAutoScalingGroupRequest.builder()
                .autoScalingGroupName(groupName)
                .forceDelete(true)
                .build();


getAutoScalingClient().deleteAutoScalingGroup(deleteAutoScalingGroupRequest);
        System.out.println(groupName + " was deleted.");
    }

    /*
     * Verify the default security group of the specified VPC allows ingress from
```

```
    * this
    * computer. This can be done by allowing ingress from this computer's IP
    * address. In some situations, such as connecting from a corporate network,
you
    * must instead specify a prefix list ID. You can also temporarily open the
port
    * to
    * any IP address while running this example. If you do, be sure to remove
    * public
    * access when you're done.
    *
    */
   public GroupInfo verifyInboundPort(String VPC, int port, String ipAddress) {
       boolean portIsOpen = false;
       GroupInfo groupInfo = new GroupInfo();
       try {
           Filter filter = Filter.builder()
                   .name("group-name")
                   .values("default")
                   .build();

           Filter filter1 = Filter.builder()
                   .name("vpc-id")
                   .values(VPC)
                   .build();

           DescribeSecurityGroupsRequest securityGroupsRequest =
DescribeSecurityGroupsRequest.builder()
                   .filters(filter, filter1)
                   .build();

           DescribeSecurityGroupsResponse securityGroupsResponse =
getEc2Client()
                   .describeSecurityGroups(securityGroupsRequest);
           String securityGroup =
securityGroupsResponse.securityGroups().get(0).groupName();
           groupInfo.setGroupName(securityGroup);

           for (SecurityGroup secGroup :
securityGroupsResponse.securityGroups()) {
               System.out.println("Found security group: " +
secGroup.groupId());

               for (IpPermission ipPermission : secGroup.ipPermissions()) {
```

```
                    if (ipPermission.fromPort() == port) {
                        System.out.println("Found inbound rule: " +
ipPermission);
                        for (IpRange ipRange : ipPermission.ipRanges()) {
                            String cidrIp = ipRange.cidrIp();
                            if (cidrIp.startsWith(ipAddress) ||
cidrIp.equals("0.0.0.0/0")) {
                                System.out.println(cidrIp + " is applicable");
                                portIsOpen = true;
                            }
                        }

                        if (!ipPermission.prefixListIds().isEmpty()) {
                            System.out.println("Prefix lList is applicable");
                            portIsOpen = true;
                        }

                        if (!portIsOpen) {
                            System.out
                                    .println("The inbound rule does not appear to
be open to either this computer's IP,"
                                            + " all IP addresses (0.0.0.0/0), or
to a prefix list ID.");
                        } else {
                            break;
                        }
                    }
                }
            }

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }

        groupInfo.setPortOpen(portIsOpen);
        return groupInfo;
    }

    /*
     * Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
     * Scaling group.
     * The target group specifies how the load balancer forward requests to the
     * instances
     * in the group.
```

```
    */
   public void attachLoadBalancerTargetGroup(String asGroupName, String
targetGroupARN) {
       try {
           AttachLoadBalancerTargetGroupsRequest targetGroupsRequest =
AttachLoadBalancerTargetGroupsRequest.builder()
                   .autoScalingGroupName(asGroupName)
                   .targetGroupARNs(targetGroupARN)
                   .build();


getAutoScalingClient().attachLoadBalancerTargetGroups(targetGroupsRequest);
           System.out.println("Attached load balancer to " + asGroupName);

       } catch (AutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }

   // Creates an EC2 Auto Scaling group with the specified size.
   public String[] createGroup(int groupSize, String templateName, String
autoScalingGroupName) {

       // Get availability zones.

software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
zonesRequest =
software.amazon.awssdk.services.ec2.model.DescribeAvailabilityZonesRequest
               .builder()
               .build();

       DescribeAvailabilityZonesResponse zonesResponse =
getEc2Client().describeAvailabilityZones(zonesRequest);
       List<String> availabilityZoneNames =
zonesResponse.availabilityZones().stream()

.map(software.amazon.awssdk.services.ec2.model.AvailabilityZone::zoneName)
               .collect(Collectors.toList());

       String availabilityZones = String.join(",", availabilityZoneNames);
       LaunchTemplateSpecification specification =
LaunchTemplateSpecification.builder()
               .launchTemplateName(templateName)
```

```
                .version("$Default")
                .build();

        String[] zones = availabilityZones.split(",");
        CreateAutoScalingGroupRequest groupRequest =
CreateAutoScalingGroupRequest.builder()
                .launchTemplate(specification)
                .availabilityZones(zones)
                .maxSize(groupSize)
                .minSize(groupSize)
                .autoScalingGroupName(autoScalingGroupName)
                .build();

        try {
            getAutoScalingClient().createAutoScalingGroup(groupRequest);

        } catch (AutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        System.out.println("Created an EC2 Auto Scaling group named " +
autoScalingGroupName);
        return zones;
    }

    public String getDefaultVPC() {
        // Define the filter.
        Filter defaultFilter = Filter.builder()
                .name("is-default")
                .values("true")
                .build();

        software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest request =
software.amazon.awssdk.services.ec2.model.DescribeVpcsRequest
                .builder()
                .filters(defaultFilter)
                .build();

        DescribeVpcsResponse response = getEc2Client().describeVpcs(request);
        return response.vpcs().get(0).vpcId();
    }

    // Gets the default subnets in a VPC for a specified list of Availability
Zones.
```

```
    public List<Subnet> getSubnets(String vpcId, String[] availabilityZones) {
        List<Subnet> subnets = null;
        Filter vpcFilter = Filter.builder()
                .name("vpc-id")
                .values(vpcId)
                .build();

        Filter azFilter = Filter.builder()
                .name("availability-zone")
                .values(availabilityZones)
                .build();

        Filter defaultForAZ = Filter.builder()
                .name("default-for-az")
                .values("true")
                .build();

        DescribeSubnetsRequest request = DescribeSubnetsRequest.builder()
                .filters(vpcFilter, azFilter, defaultForAZ)
                .build();

        DescribeSubnetsResponse response =
getEc2Client().describeSubnets(request);
        subnets = response.subnets();
        return subnets;
    }

    // Gets data about the instances in the EC2 Auto Scaling group.
    public String getBadInstance(String groupName) {
        DescribeAutoScalingGroupsRequest request =
DescribeAutoScalingGroupsRequest.builder()
                .autoScalingGroupNames(groupName)
                .build();

        DescribeAutoScalingGroupsResponse response =
getAutoScalingClient().describeAutoScalingGroups(request);
        AutoScalingGroup autoScalingGroup = response.autoScalingGroups().get(0);
        List<String> instanceIds = autoScalingGroup.instances().stream()
                .map(instance -> instance.instanceId())
                .collect(Collectors.toList());

        String[] instanceIdArray = instanceIds.toArray(new String[0]);
        for (String instanceId : instanceIdArray) {
            System.out.println("Instance ID: " + instanceId);
```

```
            return instanceId;
        }
        return "";
    }

    // Gets data about the profile associated with an instance.
    public String getInstanceProfile(String instanceId) {
        Filter filter = Filter.builder()
                .name("instance-id")
                .values(instanceId)
                .build();

        DescribeIamInstanceProfileAssociationsRequest associationsRequest =
 DescribeIamInstanceProfileAssociationsRequest
                .builder()
                .filters(filter)
                .build();

        DescribeIamInstanceProfileAssociationsResponse response = getEc2Client()
                .describeIamInstanceProfileAssociations(associationsRequest);
        return response.iamInstanceProfileAssociations().get(0).associationId();
    }

    public void deleteRolesPolicies(String policyName, String roleName, String
 InstanceProfile) {
        ListPoliciesRequest listPoliciesRequest =
 ListPoliciesRequest.builder().build();
        ListPoliciesResponse listPoliciesResponse =
 getIAMClient().listPolicies(listPoliciesRequest);
        for (Policy policy : listPoliciesResponse.policies()) {
            if (policy.policyName().equals(policyName)) {
                // List the entities (users, groups, roles) that are attached to
 the policy.

 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
 listEntitiesRequest =
 software.amazon.awssdk.services.iam.model.ListEntitiesForPolicyRequest
                        .builder()
                        .policyArn(policy.arn())
                        .build();
                ListEntitiesForPolicyResponse listEntitiesResponse = iamClient
                        .listEntitiesForPolicy(listEntitiesRequest);
                if (!listEntitiesResponse.policyGroups().isEmpty() || !
listEntitiesResponse.policyUsers().isEmpty()
```

```
                        || !listEntitiesResponse.policyRoles().isEmpty()) {
                    // Detach the policy from any entities it is attached to.
                    DetachRolePolicyRequest detachPolicyRequest =
DetachRolePolicyRequest.builder()
                            .policyArn(policy.arn())
                            .roleName(roleName) // Specify the name of the IAM
role
                            .build();

                    getIAMClient().detachRolePolicy(detachPolicyRequest);
                    System.out.println("Policy detached from entities.");
                }

                // Now, you can delete the policy.
                DeletePolicyRequest deletePolicyRequest =
DeletePolicyRequest.builder()
                        .policyArn(policy.arn())
                        .build();

                getIAMClient().deletePolicy(deletePolicyRequest);
                System.out.println("Policy deleted successfully.");
                break;
            }
        }

        // List the roles associated with the instance profile
        ListInstanceProfilesForRoleRequest listRolesRequest =
ListInstanceProfilesForRoleRequest.builder()
                .roleName(roleName)
                .build();

        // Detach the roles from the instance profile
        ListInstanceProfilesForRoleResponse listRolesResponse =
iamClient.listInstanceProfilesForRole(listRolesRequest);
        for (software.amazon.awssdk.services.iam.model.InstanceProfile profile :
listRolesResponse.instanceProfiles()) {
            RemoveRoleFromInstanceProfileRequest removeRoleRequest =
RemoveRoleFromInstanceProfileRequest.builder()
                    .instanceProfileName(InstanceProfile)
                    .roleName(roleName) // Remove the extra dot here
                    .build();

            getIAMClient().removeRoleFromInstanceProfile(removeRoleRequest);
```

```
            System.out.println("Role " + roleName + " removed from instance
 profile " + InstanceProfile);
        }

        // Delete the instance profile after removing all roles
        DeleteInstanceProfileRequest deleteInstanceProfileRequest =
 DeleteInstanceProfileRequest.builder()
                .instanceProfileName(InstanceProfile)
                .build();

        getIAMClient().deleteInstanceProfile(r ->
 r.instanceProfileName(InstanceProfile));
        System.out.println(InstanceProfile + " Deleted");
        System.out.println("All roles and policies are deleted.");
    }
}
```

Cree una clase que resuma las acciones de Elastic Load Balancing.

```
public class LoadBalancer {
    public ElasticLoadBalancingV2Client elasticLoadBalancingV2Client;

    public ElasticLoadBalancingV2Client getLoadBalancerClient() {
        if (elasticLoadBalancingV2Client == null) {
            elasticLoadBalancingV2Client = ElasticLoadBalancingV2Client.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }

        return elasticLoadBalancingV2Client;
    }

    // Checks the health of the instances in the target group.
    public List<TargetHealthDescription> checkTargetHealth(String
 targetGroupName) {
        DescribeTargetGroupsRequest targetGroupsRequest =
 DescribeTargetGroupsRequest.builder()
                .names(targetGroupName)
                .build();

        DescribeTargetGroupsResponse tgResponse =
 getLoadBalancerClient().describeTargetGroups(targetGroupsRequest);
```

```
        DescribeTargetHealthRequest healthRequest =
DescribeTargetHealthRequest.builder()

.targetGroupArn(tgResponse.targetGroups().get(0).targetGroupArn())
                .build();

        DescribeTargetHealthResponse healthResponse =
getLoadBalancerClient().describeTargetHealth(healthRequest);
        return healthResponse.targetHealthDescriptions();
    }

    // Gets the HTTP endpoint of the load balancer.
    public String getEndpoint(String lbName) {
        DescribeLoadBalancersResponse res = getLoadBalancerClient()
                .describeLoadBalancers(describe -> describe.names(lbName));
        return res.loadBalancers().get(0).dnsName();
    }

    // Deletes a load balancer.
    public void deleteLoadBalancer(String lbName) {
        try {
            // Use a waiter to delete the Load Balancer.
            DescribeLoadBalancersResponse res = getLoadBalancerClient()
                    .describeLoadBalancers(describe -> describe.names(lbName));
            ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
            DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()

.loadBalancerArns(res.loadBalancers().get(0).loadBalancerArn())
                    .build();

            getLoadBalancerClient().deleteLoadBalancer(
                    builder ->
builder.loadBalancerArn(res.loadBalancers().get(0).loadBalancerArn()));
            WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                    .waitUntilLoadBalancersDeleted(request);
            waiterResponse.matched().response().ifPresent(System.out::println);

        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
```

```
        System.out.println(lbName + " was deleted.");
    }

    // Deletes the target group.
    public void deleteTargetGroup(String targetGroupName) {
        try {
            DescribeTargetGroupsResponse res = getLoadBalancerClient()
                    .describeTargetGroups(describe ->
describe.names(targetGroupName));
            getLoadBalancerClient()
                    .deleteTargetGroup(builder ->
builder.targetGroupArn(res.targetGroups().get(0).targetGroupArn()));
        } catch (ElasticLoadBalancingV2Exception e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
        System.out.println(targetGroupName + " was deleted.");
    }

    // Verify this computer can successfully send a GET request to the load
balancer
    // endpoint.
    public boolean verifyLoadBalancerEndpoint(String elbDnsName) throws
IOException, InterruptedException {
        boolean success = false;
        int retries = 3;
        CloseableHttpClient httpClient = HttpClients.createDefault();

        // Create an HTTP GET request to the ELB.
        HttpGet httpGet = new HttpGet("http://" + elbDnsName);
        try {
            while ((!success) && (retries > 0)) {
                // Execute the request and get the response.
                HttpResponse response = httpClient.execute(httpGet);
                int statusCode = response.getStatusLine().getStatusCode();
                System.out.println("HTTP Status Code: " + statusCode);
                if (statusCode == 200) {
                    success = true;
                } else {
                    retries--;
                    System.out.println("Got connection error from load balancer
endpoint, retrying...");
                    TimeUnit.SECONDS.sleep(15);
                }
            }
```

```
        } catch (org.apache.http.conn.HttpHostConnectException e) {
            System.out.println(e.getMessage());
        }

        System.out.println("Status.." + success);
        return success;
    }

    /*
     * Creates an Elastic Load Balancing target group. The target group specifies
     * how
     * the load balancer forward requests to instances in the group and how
instance
     * health is checked.
     */
    public String createTargetGroup(String protocol, int port, String vpcId,
String targetGroupName) {
        CreateTargetGroupRequest targetGroupRequest =
CreateTargetGroupRequest.builder()
                .healthCheckPath("/healthcheck")
                .healthCheckTimeoutSeconds(5)
                .port(port)
                .vpcId(vpcId)
                .name(targetGroupName)
                .protocol(protocol)
                .build();

        CreateTargetGroupResponse targetGroupResponse =
getLoadBalancerClient().createTargetGroup(targetGroupRequest);
        String targetGroupArn =
targetGroupResponse.targetGroups().get(0).targetGroupArn();
        String targetGroup =
targetGroupResponse.targetGroups().get(0).targetGroupName();
        System.out.println("The " + targetGroup + " was created with ARN" +
targetGroupArn);
        return targetGroupArn;
    }

    /*
     * Creates an Elastic Load Balancing load balancer that uses the specified
     * subnets
     * and forwards requests to the specified target group.
     */
```

```
   public String createLoadBalancer(List<Subnet> subnetIds, String
targetGroupARN, String lbName, int port,
           String protocol) {
       try {
           List<String> subnetIdStrings = subnetIds.stream()
                   .map(Subnet::subnetId)
                   .collect(Collectors.toList());

           CreateLoadBalancerRequest balancerRequest =
CreateLoadBalancerRequest.builder()
                   .subnets(subnetIdStrings)
                   .name(lbName)
                   .scheme("internet-facing")
                   .build();

           // Create and wait for the load balancer to become available.
           CreateLoadBalancerResponse lsResponse =
getLoadBalancerClient().createLoadBalancer(balancerRequest);
           String lbARN = lsResponse.loadBalancers().get(0).loadBalancerArn();

           ElasticLoadBalancingV2Waiter loadBalancerWaiter =
getLoadBalancerClient().waiter();
           DescribeLoadBalancersRequest request =
DescribeLoadBalancersRequest.builder()
                   .loadBalancerArns(lbARN)
                   .build();

           System.out.println("Waiting for Load Balancer " + lbName + " to
become available.");
           WaiterResponse<DescribeLoadBalancersResponse> waiterResponse =
loadBalancerWaiter
                   .waitUntilLoadBalancerAvailable(request);
           waiterResponse.matched().response().ifPresent(System.out::println);
           System.out.println("Load Balancer " + lbName + " is available.");

           // Get the DNS name (endpoint) of the load balancer.
           String lbDNSName = lsResponse.loadBalancers().get(0).dnsName();
           System.out.println("*** Load Balancer DNS Name: " + lbDNSName);

           // Create a listener for the load balance.
           Action action = Action.builder()
                   .targetGroupArn(targetGroupARN)
                   .type("forward")
                   .build();
```

```
            CreateListenerRequest listenerRequest =
 CreateListenerRequest.builder()

 .loadBalancerArn(lsResponse.loadBalancers().get(0).loadBalancerArn())
                    .defaultActions(action)
                    .port(port)
                    .protocol(protocol)
                    .defaultActions(action)
                    .build();

            getLoadBalancerClient().createListener(listenerRequest);
            System.out.println("Created listener to forward traffic from load
 balancer " + lbName + " to target group "
                    + targetGroupARN);

            // Return the load balancer DNS name.
            return lbDNSName;

        } catch (ElasticLoadBalancingV2Exception e) {
            e.printStackTrace();
        }
        return "";
    }
}
```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```
public class Database {

    private static DynamoDbClient dynamoDbClient;

    public static DynamoDbClient getDynamoDbClient() {
        if (dynamoDbClient == null) {
            dynamoDbClient = DynamoDbClient.builder()
                    .region(Region.US_EAST_1)
                    .build();
        }
        return dynamoDbClient;
    }

    // Checks to see if the Amazon DynamoDB table exists.
```

```
   private boolean doesTableExist(String tableName) {
       try {
           // Describe the table and catch any exceptions.
           DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
                   .tableName(tableName)
                   .build();

           getDynamoDbClient().describeTable(describeTableRequest);
           System.out.println("Table '" + tableName + "' exists.");
           return true;

       } catch (ResourceNotFoundException e) {
           System.out.println("Table '" + tableName + "' does not exist.");
       } catch (DynamoDbException e) {
           System.err.println("Error checking table existence: " +
e.getMessage());
       }
       return false;
   }

   /*
    * Creates a DynamoDB table to use a recommendation service. The table has a
    * hash key named 'MediaType' that defines the type of media recommended,
such
    * as
    * Book or Movie, and a range key named 'ItemId' that, combined with the
    * MediaType,
    * forms a unique identifier for the recommended item.
    */
   public void createTable(String tableName, String fileName) throws IOException
{
       // First check to see if the table exists.
       boolean doesExist = doesTableExist(tableName);
       if (!doesExist) {
           DynamoDbWaiter dbWaiter = getDynamoDbClient().waiter();
           CreateTableRequest createTableRequest = CreateTableRequest.builder()
                   .tableName(tableName)
                   .attributeDefinitions(
                           AttributeDefinition.builder()
                                   .attributeName("MediaType")
                                   .attributeType(ScalarAttributeType.S)
                                   .build(),
                           AttributeDefinition.builder()
```

```
                                .attributeName("ItemId")
                                .attributeType(ScalarAttributeType.N)
                                .build())
                .keySchema(
                        KeySchemaElement.builder()
                                .attributeName("MediaType")
                                .keyType(KeyType.HASH)
                                .build(),
                        KeySchemaElement.builder()
                                .attributeName("ItemId")
                                .keyType(KeyType.RANGE)
                                .build())
                .provisionedThroughput(
                        ProvisionedThroughput.builder()
                                .readCapacityUnits(5L)
                                .writeCapacityUnits(5L)
                                .build())
                .build();

            getDynamoDbClient().createTable(createTableRequest);
            System.out.println("Creating table " + tableName + "...");

            // Wait until the Amazon DynamoDB table is created.
            DescribeTableRequest tableRequest = DescribeTableRequest.builder()
                    .tableName(tableName)
                    .build();

            WaiterResponse<DescribeTableResponse> waiterResponse =
dbWaiter.waitUntilTableExists(tableRequest);
            waiterResponse.matched().response().ifPresent(System.out::println);
            System.out.println("Table " + tableName + " created.");

            // Add records to the table.
            populateTable(fileName, tableName);
        }
    }

    public void deleteTable(String tableName) {
        getDynamoDbClient().deleteTable(table -> table.tableName(tableName));
        System.out.println("Table " + tableName + " deleted.");
    }

    // Populates the table with data located in a JSON file using the DynamoDB
    // enhanced client.
```

```
    public void populateTable(String fileName, String tableName) throws
 IOException {
        DynamoDbEnhancedClient enhancedClient = DynamoDbEnhancedClient.builder()
                .dynamoDbClient(getDynamoDbClient())
                .build();
        ObjectMapper objectMapper = new ObjectMapper();
        File jsonFile = new File(fileName);
        JsonNode rootNode = objectMapper.readTree(jsonFile);

        DynamoDbTable<Recommendation> mappedTable =
 enhancedClient.table(tableName,
                TableSchema.fromBean(Recommendation.class));
        for (JsonNode currentNode : rootNode) {
            String mediaType = currentNode.path("MediaType").path("S").asText();
            int itemId = currentNode.path("ItemId").path("N").asInt();
            String title = currentNode.path("Title").path("S").asText();
            String creator = currentNode.path("Creator").path("S").asText();

            // Create a Recommendation object and set its properties.
            Recommendation rec = new Recommendation();
            rec.setMediaType(mediaType);
            rec.setItemId(itemId);
            rec.setTitle(title);
            rec.setCreator(creator);

            // Put the item into the DynamoDB table.
            mappedTable.putItem(rec); // Add the Recommendation to the list.
        }
        System.out.println("Added all records to the " + tableName);
    }
}
```

Cree una clase que agrupe las acciones de Systems Manager.

```
public class ParameterHelper {

    String tableName = "doc-example-resilient-architecture-table";
    String dyntable = "doc-example-recommendation-service";
    String failureResponse = "doc-example-resilient-architecture-failure-
response";
    String healthCheck = "doc-example-resilient-architecture-health-check";
```

```
    public void reset() {
        put(dyntable, tableName);
        put(failureResponse, "none");
        put(healthCheck, "shallow");
    }

    public void put(String name, String value) {
        SsmClient ssmClient = SsmClient.builder()
                .region(Region.US_EAST_1)
                .build();

        PutParameterRequest parameterRequest = PutParameterRequest.builder()
                .name(name)
                .value(value)
                .overwrite(true)
                .type("String")
                .build();

        ssmClient.putParameter(parameterRequest);
        System.out.printf("Setting demo parameter %s to '%s'.", name, value);
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup

- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute el escenario interactivo en un símbolo del sistema.

```
#!/usr/bin/env node
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0


import {
  Scenario,
  parseScenarioArgs,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
```

```
/**
 * The workflow steps are split into three stages:
 *   - deploy
 *   - demo
 *   - destroy
 *
 * Each of these stages has a corresponding file prefixed with steps-*.
 */
import { deploySteps } from "./steps-deploy.js";
import { demoSteps } from "./steps-demo.js";
import { destroySteps } from "./steps-destroy.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Deploys all resources necessary for the workflow.
  deploy: new Scenario("Resilient Workflow - Deploy", deploySteps, context),
  // Demonstrates how a fragile web service can be made more resilient.
  demo: new Scenario("Resilient Workflow - Demo", demoSteps, context),
  // Destroys the resources created for the workflow.
  destroy: new Scenario("Resilient Workflow - Destroy", destroySteps, context),
};

// Call function if run directly
import { fileURLToPath } from "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

Cree los pasos para implementar todos los recursos.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
// SPDX-License-Identifier: Apache-2.0
import { join } from "node:path";
import { readFileSync, writeFileSync } from "node:fs";
import axios from "axios";

import {
  BatchWriteItemCommand,
  CreateTableCommand,
  DynamoDBClient,
  waitUntilTableExists,
} from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  CreateKeyPairCommand,
  CreateLaunchTemplateCommand,
  DescribeAvailabilityZonesCommand,
  DescribeVpcsCommand,
  DescribeSubnetsCommand,
  DescribeSecurityGroupsCommand,
  AuthorizeSecurityGroupIngressCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  AttachRolePolicyCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import { SSMClient, GetParameterCommand } from "@aws-sdk/client-ssm";
import {
  CreateAutoScalingGroupCommand,
  AutoScalingClient,
  AttachLoadBalancerTargetGroupsCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  CreateListenerCommand,
  CreateLoadBalancerCommand,
  CreateTargetGroupCommand,
  ElasticLoadBalancingV2Client,
  waitUntilLoadBalancerAvailable,
} from "@aws-sdk/client-elastic-load-balancing-v2";
```

```
import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH, ROOT } from "./constants.js";
import { initParamsSteps } from "./steps-reset-params.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const deploySteps = [
  new ScenarioOutput("introduction", MESSAGES.introduction, { header: true }),
  new ScenarioInput("confirmDeployment", MESSAGES.confirmDeployment, {
    type: "confirm",
  }),
  new ScenarioAction(
    "handleConfirmDeployment",
    (c) => c.confirmDeployment === false && process.exit(),
  ),
  new ScenarioOutput(
    "creatingTable",
    MESSAGES.creatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("createTable", async () => {
    const client = new DynamoDBClient({});
    await client.send(
      new CreateTableCommand({
        TableName: NAMES.tableName,
        ProvisionedThroughput: {
          ReadCapacityUnits: 5,
          WriteCapacityUnits: 5,
        },
        AttributeDefinitions: [
          {
            AttributeName: "MediaType",
            AttributeType: "S",
          },
          {
            AttributeName: "ItemId",
            AttributeType: "N",
          },
```

```
        ],
        KeySchema: [
          {
            AttributeName: "MediaType",
            KeyType: "HASH",
          },
          {
            AttributeName: "ItemId",
            KeyType: "RANGE",
          },
        ],
      }),
    );
    await waitUntilTableExists({ client }, { TableName: NAMES.tableName });
  }),
  new ScenarioOutput(
    "createdTable",
    MESSAGES.createdTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "populatingTable",
    MESSAGES.populatingTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioAction("populateTable", () => {
    const client = new DynamoDBClient({});
    /**
     * @type {{ default: import("@aws-sdk/client-dynamodb").PutRequest['Item']
[] }}
     */
    const recommendations = JSON.parse(
      readFileSync(join(RESOURCES_PATH, "recommendations.json")),
    );

    return client.send(
      new BatchWriteItemCommand({
        RequestItems: {
          [NAMES.tableName]: recommendations.map((item) => ({
            PutRequest: { Item: item },
          })),
        },
      }),
    );
  }),
  new ScenarioOutput(
```

```
    "populatedTable",
    MESSAGES.populatedTable.replace("${TABLE_NAME}", NAMES.tableName),
  ),
  new ScenarioOutput(
    "creatingKeyPair",
    MESSAGES.creatingKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioAction("createKeyPair", async () => {
    const client = new EC2Client({});
    const { KeyMaterial } = await client.send(
      new CreateKeyPairCommand({
        KeyName: NAMES.keyPairName,
      }),
    );

    writeFileSync(`${NAMES.keyPairName}.pem`, KeyMaterial, { mode: 0o600 });
  }),
  new ScenarioOutput(
    "createdKeyPair",
    MESSAGES.createdKeyPair.replace("${KEY_PAIR_NAME}", NAMES.keyPairName),
  ),
  new ScenarioOutput(
    "creatingInstancePolicy",
    MESSAGES.creatingInstancePolicy.replace(
      "${INSTANCE_POLICY_NAME}",
      NAMES.instancePolicyName,
    ),
  ),
  new ScenarioAction("createInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const {
      Policy: { Arn },
    } = await client.send(
      new CreatePolicyCommand({
        PolicyName: NAMES.instancePolicyName,
        PolicyDocument: readFileSync(
          join(RESOURCES_PATH, "instance_policy.json"),
        ),
      }),
    );
    state.instancePolicyArn = Arn;
  }),
  new ScenarioOutput("createdInstancePolicy", (state) =>
    MESSAGES.createdInstancePolicy
```

```
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_POLICY_ARN}", state.instancePolicyArn),
  ),
  new ScenarioOutput(
    "creatingInstanceRole",
    MESSAGES.creatingInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioAction("createInstanceRole", () => {
    const client = new IAMClient({});
    return client.send(
      new CreateRoleCommand({
        RoleName: NAMES.instanceRoleName,
        AssumeRolePolicyDocument: readFileSync(
          join(ROOT, "assume-role-policy.json"),
        ),
      }),
    );
  }),
  new ScenarioOutput(
    "createdInstanceRole",
    MESSAGES.createdInstanceRole.replace(
      "${INSTANCE_ROLE_NAME}",
      NAMES.instanceRoleName,
    ),
  ),
  new ScenarioOutput(
    "attachingPolicyToRole",
    MESSAGES.attachingPolicyToRole
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName)
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName),
  ),
  new ScenarioAction("attachPolicyToRole", async (state) => {
    const client = new IAMClient({});
    await client.send(
      new AttachRolePolicyCommand({
        RoleName: NAMES.instanceRoleName,
        PolicyArn: state.instancePolicyArn,
      }),
    );
  }),
  new ScenarioOutput(
```

```
    "attachedPolicyToRole",
    MESSAGES.attachedPolicyToRole
      .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioOutput(
    "creatingInstanceProfile",
    MESSAGES.creatingInstanceProfile.replace(
      "${INSTANCE_PROFILE_NAME}",
      NAMES.instanceProfileName,
    ),
  ),
  new ScenarioAction("createInstanceProfile", async (state) => {
    const client = new IAMClient({});
    const {
      InstanceProfile: { Arn },
    } = await client.send(
      new CreateInstanceProfileCommand({
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
    state.instanceProfileArn = Arn;

    await waitUntilInstanceProfileExists(
      { client },
      { InstanceProfileName: NAMES.instanceProfileName },
    );
  }),
  new ScenarioOutput("createdInstanceProfile", (state) =>
    MESSAGES.createdInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_PROFILE_ARN}", state.instanceProfileArn),
  ),
  new ScenarioOutput(
    "addingRoleToInstanceProfile",
    MESSAGES.addingRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  new ScenarioAction("addRoleToInstanceProfile", () => {
    const client = new IAMClient({});
    return client.send(
      new AddRoleToInstanceProfileCommand({
        RoleName: NAMES.instanceRoleName,
```

```
        InstanceProfileName: NAMES.instanceProfileName,
      }),
    );
  }),
  new ScenarioOutput(
    "addedRoleToInstanceProfile",
    MESSAGES.addedRoleToInstanceProfile
      .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
      .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName),
  ),
  ...initParamsSteps,
  new ScenarioOutput("creatingLaunchTemplate", MESSAGES.creatingLaunchTemplate),
  new ScenarioAction("createLaunchTemplate", async () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    const ssmClient = new SSMClient({});
    const { Parameter } = await ssmClient.send(
      new GetParameterCommand({
        Name: "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
      }),
    );
    const ec2Client = new EC2Client({});
    await ec2Client.send(
      new CreateLaunchTemplateCommand({
        LaunchTemplateName: NAMES.launchTemplateName,
        LaunchTemplateData: {
          InstanceType: "t3.micro",
          ImageId: Parameter.Value,
          IamInstanceProfile: { Name: NAMES.instanceProfileName },
          UserData: readFileSync(
            join(RESOURCES_PATH, "server_startup_script.sh"),
          ).toString("base64"),
          KeyName: NAMES.keyPairName,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  new ScenarioOutput(
    "createdLaunchTemplate",
    MESSAGES.createdLaunchTemplate.replace(
      "${LAUNCH_TEMPLATE_NAME}",
      NAMES.launchTemplateName,
    ),
  ),
```

```
  new ScenarioOutput(
    "creatingAutoScalingGroup",
    MESSAGES.creatingAutoScalingGroup.replace(
      "${AUTO_SCALING_GROUP_NAME}",
      NAMES.autoScalingGroupName,
    ),
  ),
  new ScenarioAction("createAutoScalingGroup", async (state) => {
    const ec2Client = new EC2Client({});
    const { AvailabilityZones } = await ec2Client.send(
      new DescribeAvailabilityZonesCommand({}),
    );
    state.availabilityZoneNames = AvailabilityZones.map((az) => az.ZoneName);
    const autoScalingClient = new AutoScalingClient({});
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new CreateAutoScalingGroupCommand({
          AvailabilityZones: state.availabilityZoneNames,
          AutoScalingGroupName: NAMES.autoScalingGroupName,
          LaunchTemplate: {
            LaunchTemplateName: NAMES.launchTemplateName,
            Version: "$Default",
          },
          MinSize: 3,
          MaxSize: 3,
        }),
      ),
    );
  }),
  new ScenarioOutput(
    "createdAutoScalingGroup",
    /**
     * @param {{ availabilityZoneNames: string[] }} state
     */
    (state) =>
      MESSAGES.createdAutoScalingGroup
        .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName)
        .replace(
          "${AVAILABILITY_ZONE_NAMES}",
          state.availabilityZoneNames.join(", "),
        ),
  ),
  new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
    type: "confirm",
```

```
  }),
  new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
  new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
  new ScenarioAction("getVpc", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
    const client = new EC2Client({});
    const { Vpcs } = await client.send(
      new DescribeVpcsCommand({
        Filters: [{ Name: "is-default", Values: ["true"] }],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
    state.defaultVpc = Vpcs[0].VpcId;
  }),
  new ScenarioOutput("gotVpc", (state) =>
    MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
  ),
  new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
  new ScenarioAction("getSubnets", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
    const client = new EC2Client({});
    const { Subnets } = await client.send(
      new DescribeSubnetsCommand({
        Filters: [
          { Name: "vpc-id", Values: [state.defaultVpc] },
          { Name: "availability-zone", Values: state.availabilityZoneNames },
          { Name: "default-for-az", Values: ["true"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    state.subnets = Subnets.map((subnet) => subnet.SubnetId);
  }),
  new ScenarioOutput(
    "gotSubnets",
    /**
     * @param {{ subnets: string[] }} state
     */
    (state) =>
      MESSAGES.gotSubnets.replace("${SUBNETS}", state.subnets.join(", ")),
  ),
  new ScenarioOutput(
    "creatingLoadBalancerTargetGroup",
    MESSAGES.creatingLoadBalancerTargetGroup.replace(
```

```
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioAction("createLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    const { TargetGroups } = await client.send(
      new CreateTargetGroupCommand({
        Name: NAMES.loadBalancerTargetGroupName,
        Protocol: "HTTP",
        Port: 80,
        HealthCheckPath: "/healthcheck",
        HealthCheckIntervalSeconds: 10,
        HealthCheckTimeoutSeconds: 5,
        HealthyThresholdCount: 2,
        UnhealthyThresholdCount: 2,
        VpcId: state.defaultVpc,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    const targetGroup = TargetGroups[0];
    state.targetGroupArn = targetGroup.TargetGroupArn;
    state.targetGroupProtocol = targetGroup.Protocol;
    state.targetGroupPort = targetGroup.Port;
  }),
  new ScenarioOutput(
    "createdLoadBalancerTargetGroup",
    MESSAGES.createdLoadBalancerTargetGroup.replace(
      "${TARGET_GROUP_NAME}",
      NAMES.loadBalancerTargetGroupName,
    ),
  ),
  new ScenarioOutput(
    "creatingLoadBalancer",
    MESSAGES.creatingLoadBalancer.replace("${LB_NAME}", NAMES.loadBalancerName),
  ),
  new ScenarioAction("createLoadBalancer", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    const client = new ElasticLoadBalancingV2Client({});
    const { LoadBalancers } = await client.send(
      new CreateLoadBalancerCommand({
        Name: NAMES.loadBalancerName,
        Subnets: state.subnets,
```

```
      }),
    );
    state.loadBalancerDns = LoadBalancers[0].DNSName;
    state.loadBalancerArn = LoadBalancers[0].LoadBalancerArn;
    await waitUntilLoadBalancerAvailable(
      { client },
      { Names: [NAMES.loadBalancerName] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  new ScenarioOutput("createdLoadBalancer", (state) =>
    MESSAGES.createdLoadBalancer
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioOutput(
    "creatingListener",
    MESSAGES.creatingLoadBalancerListener
      .replace("${LB_NAME}", NAMES.loadBalancerName)
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName),
  ),
  new ScenarioAction("createListener", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    const client = new ElasticLoadBalancingV2Client({});
    const { Listeners } = await client.send(
      new CreateListenerCommand({
        LoadBalancerArn: state.loadBalancerArn,
        Protocol: state.targetGroupProtocol,
        Port: state.targetGroupPort,
        DefaultActions: [
          { Type: "forward", TargetGroupArn: state.targetGroupArn },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    const listener = Listeners[0];
    state.loadBalancerListenerArn = listener.ListenerArn;
  }),
  new ScenarioOutput("createdListener", (state) =>
    MESSAGES.createdLoadBalancerListener.replace(
      "${LB_LISTENER_ARN}",
      state.loadBalancerListenerArn,
    ),
  ),
```

```
  new ScenarioOutput(
    "attachingLoadBalancerTargetGroup",
    MESSAGES.attachingLoadBalancerTargetGroup
      .replace("${TARGET_GROUP_NAME}", NAMES.loadBalancerTargetGroupName)
      .replace("${AUTO_SCALING_GROUP_NAME}", NAMES.autoScalingGroupName),
  ),
  new ScenarioAction("attachLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  new ScenarioOutput(
    "attachedLoadBalancerTargetGroup",
    MESSAGES.attachedLoadBalancerTargetGroup,
  ),
  new ScenarioOutput("verifyingInboundPort", MESSAGES.verifyingInboundPort),
  new ScenarioAction(
    "verifyInboundPort",
    /**
     *
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup}} state
     */
    async (state) => {
      const client = new EC2Client({});
      const { SecurityGroups } = await client.send(
        new DescribeSecurityGroupsCommand({
          Filters: [{ Name: "group-name", Values: ["default"] }],
        }),
      );
      if (!SecurityGroups) {
        state.verifyInboundPortError = new Error(MESSAGES.noSecurityGroups);
      }
      state.defaultSecurityGroup = SecurityGroups[0];

      /**
       * @type {string}
       */
```

```
      const ipResponse = (await axios.get("http://checkip.amazonaws.com")).data;
      state.myIp = ipResponse.trim();
      const myIpRules = state.defaultSecurityGroup.IpPermissions.filter(
        ({ IpRanges }) =>
          IpRanges.some(
            ({ CidrIp }) =>
              CidrIp.startsWith(state.myIp) || CidrIp === "0.0.0.0/0",
          ),
      )
        .filter(({ IpProtocol }) => IpProtocol === "tcp")
        .filter(({ FromPort }) => FromPort === 80);

      state.myIpRules = myIpRules;
    },
  ),
  new ScenarioOutput(
    "verifiedInboundPort",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return MESSAGES.foundIpRules.replace(
          "${IP_RULES}",
          JSON.stringify(state.myIpRules, null, 2),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        return MESSAGES.noIpRules;
      }
    },
    { type: "confirm" },
```

```
  ),
  new ScenarioAction(
    "addInboundRule",
    /**
     * @param {{ defaultSecurityGroup: import('@aws-sdk/client-
ec2').SecurityGroup }} state
     */
    async (state) => {
      if (!state.shouldAddInboundRule) {
        return;
      }

      const client = new EC2Client({});
      await client.send(
        new AuthorizeSecurityGroupIngressCommand({
          GroupId: state.defaultSecurityGroup.GroupId,
          CidrIp: `${state.myIp}/32`,
          FromPort: 80,
          ToPort: 80,
          IpProtocol: "tcp",
        }),
      );
    },
  ),
  new ScenarioOutput("addedInboundRule", (state) => {
    if (state.shouldAddInboundRule) {
      return MESSAGES.addedInboundRule.replace("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${DNS_NAME}", state.loadBalancerDns),
  ),
  new ScenarioAction("verifyEndpoint", async (state) => {
    try {
      const response = await retry({ intervalInMs: 2000, maxRetries: 30 }, () =>
        axios.get(`http://${state.loadBalancerDns}`),
      );
      state.endpointResponse = JSON.stringify(response.data, null, 2);
    } catch (e) {
      state.verifyEndpointError = e;
    }
  }),
```

```
  new ScenarioOutput("verifiedEndpoint", (state) => {
    if (state.verifyEndpointError) {
      console.error(state.verifyEndpointError);
    } else {
      return MESSAGES.verifiedEndpoint.replace(
        "${ENDPOINT_RESPONSE}",
        state.endpointResponse,
      );
    }
  }),
];
```

Cree los pasos para ejecutar la demostración.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { readFileSync } from "node:fs";
import { join } from "node:path";

import axios from "axios";

import {
  DescribeTargetGroupsCommand,
  DescribeTargetHealthCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";
import {
  DescribeInstanceInformationCommand,
  PutParameterCommand,
  SSMClient,
  SendCommandCommand,
} from "@aws-sdk/client-ssm";
import {
  IAMClient,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  CreateInstanceProfileCommand,
  AddRoleToInstanceProfileCommand,
  waitUntilInstanceProfileExists,
} from "@aws-sdk/client-iam";
import {
```

```
  AutoScalingClient,
  DescribeAutoScalingGroupsCommand,
  TerminateInstanceInAutoScalingGroupCommand,
} from "@aws-sdk/client-auto-scaling";
import {
  DescribeIamInstanceProfileAssociationsCommand,
  EC2Client,
  RebootInstancesCommand,
  ReplaceIamInstanceProfileAssociationCommand,
} from "@aws-sdk/client-ec2";

import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES, RESOURCES_PATH } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

const getRecommendation = new ScenarioAction(
  "getRecommendation",
  async (state) => {
    const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
    if (loadBalancer) {
      state.loadBalancerDnsName = loadBalancer.DNSName;
      try {
        state.recommendation = (
          await axios.get(`http://${state.loadBalancerDnsName}`)
        ).data;
      } catch (e) {
        state.recommendation = e instanceof Error ? e.message : e;
      }
    } else {
      throw new Error(MESSAGES.demoFindLoadBalancerError);
    }
  },
);

const getRecommendationResult = new ScenarioOutput(
  "getRecommendationResult",
  (state) =>
    `Recommendation:\n${JSON.stringify(state.recommendation, null, 2)}`,
```

```
  { preformatted: true },
);

const getHealthCheck = new ScenarioAction("getHealthCheck", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  state.targetHealthDescriptions = TargetHealthDescriptions;
});

const getHealthCheckResult = new ScenarioOutput(
  "getHealthCheckResult",
  /**
   * @param {{ targetHealthDescriptions: import('@aws-sdk/client-elastic-load-
balancing-v2').TargetHealthDescription[]}} state
   */
  (state) => {
    const status = state.targetHealthDescriptions
      .map((th) => `${th.Target.Id}: ${th.TargetHealth.State}`)
      .join("\n");
    return `Health check:\n${status}`;
  },
  { preformatted: true },
);

const loadBalancerLoop = new ScenarioAction(
  "loadBalancerLoop",
  getRecommendation.action,
  {
    whileConfig: {
      inputEquals: true,
```

```
      input: new ScenarioInput(
        "loadBalancerCheck",
        MESSAGES.demoLoadBalancerCheck,
        {
          type: "confirm",
        },
      ),
      output: getRecommendationResult,
    },
  },
);

const healthCheckLoop = new ScenarioAction(
  "healthCheckLoop",
  getHealthCheck.action,
  {
    whileConfig: {
      inputEquals: true,
      input: new ScenarioInput("healthCheck", MESSAGES.demoHealthCheck, {
        type: "confirm",
      }),
      output: getHealthCheckResult,
    },
  },
);

const statusSteps = [
  getRecommendation,
  getRecommendationResult,
  getHealthCheck,
  getHealthCheckResult,
];

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const demoSteps = [
  new ScenarioOutput("header", MESSAGES.demoHeader, { header: true }),
  new ScenarioOutput("sanityCheck", MESSAGES.demoSanityCheck),
  ...statusSteps,
  new ScenarioInput(
    "brokenDependencyConfirmation",
    MESSAGES.demoBrokenDependencyConfirmation,
    { type: "confirm" },
```

```
  ),
  new ScenarioAction("brokenDependency", async (state) => {
    if (!state.brokenDependencyConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      state.badTableName = `fake-table-${Date.now()}`;
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmTableNameKey,
          Value: state.badTableName,
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
  new ScenarioOutput("testBrokenDependency", (state) =>
    MESSAGES.demoTestBrokenDependency.replace(
      "${TABLE_NAME}",
      state.badTableName,
    ),
  ),
  ...statusSteps,
  new ScenarioInput(
    "staticResponseConfirmation",
    MESSAGES.demoStaticResponseConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("staticResponse", async (state) => {
    if (!state.staticResponseConfirmation) {
      process.exit();
    } else {
      const client = new SSMClient({});
      await client.send(
        new PutParameterCommand({
          Name: NAMES.ssmFailureResponseKey,
          Value: "static",
          Overwrite: true,
          Type: "String",
        }),
      );
    }
  }),
```

```
  new ScenarioOutput("testStaticResponse", MESSAGES.demoTestStaticResponse),
  ...statusSteps,
  new ScenarioInput(
    "badCredentialsConfirmation",
    MESSAGES.demoBadCredentialsConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("badCredentialsExit", (state) => {
    if (!state.badCredentialsConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("fixDynamoDBName", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioAction(
    "badCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-auto-
scaling').Instance }} state
     */
    async (state) => {
      await createSsmOnlyInstanceProfile();
      const autoScalingClient = new AutoScalingClient({});
      const { AutoScalingGroups } = await autoScalingClient.send(
        new DescribeAutoScalingGroupsCommand({
          AutoScalingGroupNames: [NAMES.autoScalingGroupName],
        }),
      );
      state.targetInstance = AutoScalingGroups[0].Instances[0];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
```

```
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]

      await ec2Client.send(
        new RebootInstancesCommand({
          InstanceIds: [state.targetInstance.InstanceId],
        }),
      );

      const ssmClient = new SSMClient({});
      await retry({ intervalInMs: 20000, maxRetries: 15 }, async () => {
        const { InstanceInformationList } = await ssmClient.send(
          new DescribeInstanceInformationCommand({}),
        );

        const instance = InstanceInformationList.find(
          (info) => info.InstanceId === state.targetInstance.InstanceId,
        );

        if (!instance) {
          throw new Error("Instance not found.");
        }
      });

      await ssmClient.send(
        new SendCommandCommand({
```

```
          InstanceIds: [state.targetInstance.InstanceId],
          DocumentName: "AWS-RunShellScript",
          Parameters: { commands: ["cd / && sudo python3 server.py 80"] },
        }),
      );
    },
  ),
  new ScenarioOutput(
    "testBadCredentials",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation}} state
     */
    (state) =>
      MESSAGES.demoTestBadCredentials.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
  ),
  loadBalancerLoop,
  new ScenarioInput(
    "deepHealthCheckConfirmation",
    MESSAGES.demoDeepHealthCheckConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("deepHealthCheckExit", (state) => {
    if (!state.deepHealthCheckConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("deepHealthCheck", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmHealthCheckKey,
        Value: "deep",
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testDeepHealthCheck", MESSAGES.demoTestDeepHealthCheck),
  healthCheckLoop,
  loadBalancerLoop,
```

```
  new ScenarioInput(
    "killInstanceConfirmation",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    (state) =>
      MESSAGES.demoKillInstanceConfirmation.replace(
        "${INSTANCE_ID}",
        state.targetInstance.InstanceId,
      ),
    { type: "confirm" },
  ),
  new ScenarioAction("killInstanceExit", (state) => {
    if (!state.killInstanceConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction(
    "killInstance",
    /**
     * @param {{ targetInstance: import('@aws-sdk/client-
ssm').InstanceInformation }} state
     */
    async (state) => {
      const client = new AutoScalingClient({});
      await client.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: state.targetInstance.InstanceId,
          ShouldDecrementDesiredCapacity: false,
        }),
      );
    },
  ),
  new ScenarioOutput("testKillInstance", MESSAGES.demoTestKillInstance),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput("failOpenConfirmation", MESSAGES.demoFailOpenConfirmation, {
    type: "confirm",
  }),
  new ScenarioAction("failOpenExit", (state) => {
    if (!state.failOpenConfirmation) {
      process.exit();
    }
```

```
  }),
  new ScenarioAction("failOpen", () => {
    const client = new SSMClient({});
    return client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: `fake-table-${Date.now()}`,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testFailOpen", MESSAGES.demoFailOpenTest),
  healthCheckLoop,
  loadBalancerLoop,
  new ScenarioInput(
    "resetTableConfirmation",
    MESSAGES.demoResetTableConfirmation,
    { type: "confirm" },
  ),
  new ScenarioAction("resetTableExit", (state) => {
    if (!state.resetTableConfirmation) {
      process.exit();
    }
  }),
  new ScenarioAction("resetTable", async () => {
    const client = new SSMClient({});
    await client.send(
      new PutParameterCommand({
        Name: NAMES.ssmTableNameKey,
        Value: NAMES.tableName,
        Overwrite: true,
        Type: "String",
      }),
    );
  }),
  new ScenarioOutput("testResetTable", MESSAGES.demoTestResetTable),
  healthCheckLoop,
  loadBalancerLoop,
];

async function createSsmOnlyInstanceProfile() {
  const iamClient = new IAMClient({});
  const { Policy } = await iamClient.send(
```

```
    new CreatePolicyCommand({
      PolicyName: NAMES.ssmOnlyPolicyName,
      PolicyDocument: readFileSync(
        join(RESOURCES_PATH, "ssm_only_policy.json"),
      ),
    }),
  );
  await iamClient.send(
    new CreateRoleCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      AssumeRolePolicyDocument: JSON.stringify({
        Version: "2012-10-17",
        Statement: [
          {
            Effect: "Allow",
            Principal: { Service: "ec2.amazonaws.com" },
            Action: "sts:AssumeRole",
          },
        ],
      }),
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: Policy.Arn,
    }),
  );
  await iamClient.send(
    new AttachRolePolicyCommand({
      RoleName: NAMES.ssmOnlyRoleName,
      PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
```

```
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Cree los pasos para destruir todos los recursos.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} from "@aws-sdk/client-ec2";
import {
  IAMClient,
  DeleteInstanceProfileCommand,
  RemoveRoleFromInstanceProfileCommand,
  DeletePolicyCommand,
  DeleteRoleCommand,
  DetachRolePolicyCommand,
  paginateListPolicies,
} from "@aws-sdk/client-iam";
import {
  AutoScalingClient,
  DeleteAutoScalingGroupCommand,
  TerminateInstanceInAutoScalingGroupCommand,
  UpdateAutoScalingGroupCommand,
  paginateDescribeAutoScalingGroups,
} from "@aws-sdk/client-auto-scaling";
import {
  DeleteLoadBalancerCommand,
  DeleteTargetGroupCommand,
```

```
  DescribeTargetGroupsCommand,
  ElasticLoadBalancingV2Client,
} from "@aws-sdk/client-elastic-load-balancing-v2";

import {
  ScenarioOutput,
  ScenarioInput,
  ScenarioAction,
} from "@aws-doc-sdk-examples/lib/scenario/index.js";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

import { MESSAGES, NAMES } from "./constants.js";
import { findLoadBalancer } from "./shared.js";

/**
 * @type {import('@aws-doc-sdk-examples/lib/scenario.js').Step[]}
 */
export const destroySteps = [
  new ScenarioInput("destroy", MESSAGES.destroy, { type: "confirm" }),
  new ScenarioAction(
    "abort",
    (state) => state.destroy === false && process.exit(),
  ),
  new ScenarioAction("deleteTable", async (c) => {
    try {
      const client = new DynamoDBClient({});
      await client.send(new DeleteTableCommand({ TableName: NAMES.tableName }));
    } catch (e) {
      c.deleteTableError = e;
    }
  }),
  new ScenarioOutput("deleteTableResult", (state) => {
    if (state.deleteTableError) {
      console.error(state.deleteTableError);
      return MESSAGES.deleteTableError.replace(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${TABLE_NAME}", NAMES.tableName);
    }
  }),
  new ScenarioAction("deleteKeyPair", async (state) => {
    try {
```

```
      const client = new EC2Client({});
      await client.send(
        new DeleteKeyPairCommand({ KeyName: NAMES.keyPairName }),
      );
      unlinkSync(`${NAMES.keyPairName}.pem`);
    } catch (e) {
      state.deleteKeyPairError = e;
    }
  }),
  new ScenarioOutput("deleteKeyPairResult", (state) => {
    if (state.deleteKeyPairError) {
      console.error(state.deleteKeyPairError);
      return MESSAGES.deleteKeyPairError.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    } else {
      return MESSAGES.deletedKeyPair.replace(
        "${KEY_PAIR_NAME}",
        NAMES.keyPairName,
      );
    }
  }),
  new ScenarioAction("detachPolicyFromRole", async (state) => {
    try {
      const client = new IAMClient({});
      const policy = await findPolicy(NAMES.instancePolicyName);

      if (!policy) {
        state.detachPolicyFromRoleError = new Error(
          `Policy ${NAMES.instancePolicyName} not found.`,
        );
      } else {
        await client.send(
          new DetachRolePolicyCommand({
            RoleName: NAMES.instanceRoleName,
            PolicyArn: policy.Arn,
          }),
        );
      }
    } catch (e) {
      state.detachPolicyFromRoleError = e;
    }
  }),
```

```
  new ScenarioOutput("detachedPolicyFromRole", (state) => {
    if (state.detachPolicyFromRoleError) {
      console.error(state.detachPolicyFromRoleError);
      return MESSAGES.detachPolicyFromRoleError
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${INSTANCE_POLICY_NAME}", NAMES.instancePolicyName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstancePolicy", async (state) => {
    const client = new IAMClient({});
    const policy = await findPolicy(NAMES.instancePolicyName);

    if (!policy) {
      state.deletePolicyError = new Error(
        `Policy ${NAMES.instancePolicyName} not found.`,
      );
    } else {
      return client.send(
        new DeletePolicyCommand({
          PolicyArn: policy.Arn,
        }),
      );
    }
  }),
  new ScenarioOutput("deletePolicyResult", (state) => {
    if (state.deletePolicyError) {
      console.error(state.deletePolicyError);
      return MESSAGES.deletePolicyError.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    }
  }),
  new ScenarioAction("removeRoleFromInstanceProfile", async (state) => {
    try {
```

```
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          RoleName: NAMES.instanceRoleName,
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
    } catch (e) {
      state.removeRoleFromInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("removeRoleFromInstanceProfileResult", (state) => {
    if (state.removeRoleFromInstanceProfile) {
      console.error(state.removeRoleFromInstanceProfileError);
      return MESSAGES.removeRoleFromInstanceProfileError
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.removedRoleFromInstanceProfile
        .replace("${INSTANCE_PROFILE_NAME}", NAMES.instanceProfileName)
        .replace("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    }
  }),
  new ScenarioAction("deleteInstanceRole", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new DeleteRoleCommand({
          RoleName: NAMES.instanceRoleName,
        }),
      );
    } catch (e) {
      state.deleteInstanceRoleError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceRoleResult", (state) => {
    if (state.deleteInstanceRoleError) {
      console.error(state.deleteInstanceRoleError);
      return MESSAGES.deleteInstanceRoleError.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
```

```
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteAutoScalingGroup", async (state) => {
    try {
      await terminateGroupInstances(NAMES.autoScalingGroupName);
      await retry({ intervalInMs: 60000, maxRetries: 60 }, async () => {
        await deleteAutoScalingGroup(NAMES.autoScalingGroupName);
      });
    } catch (e) {
      state.deleteAutoScalingGroupError = e;
    }
  }),
  new ScenarioOutput("deleteAutoScalingGroupResult", (state) => {
```

```
    if (state.deleteAutoScalingGroupError) {
      console.error(state.deleteAutoScalingGroupError);
      return MESSAGES.deleteAutoScalingGroupError.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
```

```
      const client = new ElasticLoadBalancingV2Client({});
      const loadBalancer = await findLoadBalancer(NAMES.loadBalancerName);
      await client.send(
        new DeleteLoadBalancerCommand({
          LoadBalancerArn: loadBalancer.LoadBalancerArn,
        }),
      );
      await retry({ intervalInMs: 1000, maxRetries: 60 }, async () => {
        const lb = await findLoadBalancer(NAMES.loadBalancerName);
        if (lb) {
          throw new Error("Load balancer still exists.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    const client = new ElasticLoadBalancingV2Client({});
    try {
      const { TargetGroups } = await client.send(
        new DescribeTargetGroupsCommand({
          Names: [NAMES.loadBalancerTargetGroupName],
        }),
      );

      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        client.send(
```

```
          new DeleteTargetGroupCommand({
            TargetGroupArn: TargetGroups[0].TargetGroupArn,
          }),
        ),
      );
    } catch (e) {
      state.deleteLoadBalancerTargetGroupError = e;
    }
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    } else {
      return MESSAGES.deletedLoadBalancerTargetGroup.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    }
  }),
  new ScenarioAction("detachSsmOnlyRoleFromProfile", async (state) => {
    try {
      const client = new IAMClient({});
      await client.send(
        new RemoveRoleFromInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyRoleFromProfileError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyRoleFromProfileResult", (state) => {
    if (state.detachSsmOnlyRoleFromProfileError) {
      console.error(state.detachSsmOnlyRoleFromProfileError);
      return MESSAGES.detachSsmOnlyRoleFromProfileError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    } else {
```

```
      return MESSAGES.detachedSsmOnlyRoleFromProfile
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${PROFILE_NAME}", NAMES.ssmOnlyInstanceProfileName);
    }
  }),
  new ScenarioAction("detachSsmOnlyCustomRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.detachSsmOnlyCustomRolePolicyError = e;
    }
  }),
  new ScenarioOutput("detachSsmOnlyCustomRolePolicyResult", (state) => {
    if (state.detachSsmOnlyCustomRolePolicyError) {
      console.error(state.detachSsmOnlyCustomRolePolicyError);
      return MESSAGES.detachSsmOnlyCustomRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    } else {
      return MESSAGES.detachedSsmOnlyCustomRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", NAMES.ssmOnlyPolicyName);
    }
  }),
  new ScenarioAction("detachSsmOnlyAWSRolePolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DetachRolePolicyCommand({
          RoleName: NAMES.ssmOnlyRoleName,
          PolicyArn: "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
        }),
      );
    } catch (e) {
      state.detachSsmOnlyAWSRolePolicyError = e;
    }
  }),
```

```
  new ScenarioOutput("detachSsmOnlyAWSRolePolicyResult", (state) => {
    if (state.detachSsmOnlyAWSRolePolicyError) {
      console.error(state.detachSsmOnlyAWSRolePolicyError);
      return MESSAGES.detachSsmOnlyAWSRolePolicyError
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    } else {
      return MESSAGES.detachedSsmOnlyAWSRolePolicy
        .replace("${ROLE_NAME}", NAMES.ssmOnlyRoleName)
        .replace("${POLICY_NAME}", "AmazonSSMManagedInstanceCore");
    }
  }),
  new ScenarioAction("deleteSsmOnlyInstanceProfile", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyInstanceProfileResult", (state) => {
    if (state.deleteSsmOnlyInstanceProfileError) {
      console.error(state.deleteSsmOnlyInstanceProfileError);
      return MESSAGES.deleteSsmOnlyInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyPolicy", async (state) => {
    try {
      const iamClient = new IAMClient({});
      const ssmOnlyPolicy = await findPolicy(NAMES.ssmOnlyPolicyName);
      await iamClient.send(
        new DeletePolicyCommand({
```

```
          PolicyArn: ssmOnlyPolicy.Arn,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyPolicyError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyPolicyResult", (state) => {
    if (state.deleteSsmOnlyPolicyError) {
      console.error(state.deleteSsmOnlyPolicyError);
      return MESSAGES.deleteSsmOnlyPolicyError.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    }
  }),
  new ScenarioAction("deleteSsmOnlyRole", async (state) => {
    try {
      const iamClient = new IAMClient({});
      await iamClient.send(
        new DeleteRoleCommand({
          RoleName: NAMES.ssmOnlyRoleName,
        }),
      );
    } catch (e) {
      state.deleteSsmOnlyRoleError = e;
    }
  }),
  new ScenarioOutput("deleteSsmOnlyRoleResult", (state) => {
    if (state.deleteSsmOnlyRoleError) {
      console.error(state.deleteSsmOnlyRoleError);
      return MESSAGES.deleteSsmOnlyRoleError.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
```

```
      );
    }
  }),
];

/**
 * @param {string} policyName
 */
async function findPolicy(policyName) {
  const client = new IAMClient({});
  const paginatedPolicies = paginateListPolicies({ client }, {});
  for await (const page of paginatedPolicies) {
    const policy = page.Policies.find((p) => p.PolicyName === policyName);
    if (policy) {
      return policy;
    }
  }
}

/**
 * @param {string} groupName
 */
async function deleteAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  try {
    await client.send(
      new DeleteAutoScalingGroupCommand({
        AutoScalingGroupName: groupName,
      }),
    );
  } catch (err) {
    if (!(err instanceof Error)) {
      throw err;
    } else {
      console.log(err.name);
      throw err;
    }
  }
}

/**
 * @param {string} groupName
 */
async function terminateGroupInstances(groupName) {
```

```
  const autoScalingClient = new AutoScalingClient({});
  const group = await findAutoScalingGroup(groupName);
  await autoScalingClient.send(
    new UpdateAutoScalingGroupCommand({
      AutoScalingGroupName: group.AutoScalingGroupName,
      MinSize: 0,
    }),
  );
  for (const i of group.Instances) {
    await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
      autoScalingClient.send(
        new TerminateInstanceInAutoScalingGroupCommand({
          InstanceId: i.InstanceId,
          ShouldDecrementDesiredCapacity: true,
        }),
      ),
    );
  }
}

async function findAutoScalingGroup(groupName) {
  const client = new AutoScalingClient({});
  const paginatedGroups = paginateDescribeAutoScalingGroups({ client }, {});
  for await (const page of paginatedGroups) {
    const group = page.AutoScalingGroups.find(
      (g) => g.AutoScalingGroupName === groupName,
    );
    if (group) {
      return group;
    }
  }
  throw new Error(`Auto scaling group ${groupName} not found.`);
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for JavaScript.
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate

- CreateListener
- CreateLoadBalancer
- CreateTargetGroup
- DeleteAutoScalingGroup
- DeleteInstanceProfile
- DeleteLaunchTemplate
- DeleteLoadBalancer
- DeleteTargetGroup
- DescribeAutoScalingGroups
- DescribeAvailabilityZones
- DescribeIamInstanceProfileAssociations
- DescribeInstances
- DescribeLoadBalancers
- DescribeSubnets
- DescribeTargetGroups
- DescribeTargetHealth
- DescribeVpcs
- RebootInstances
- ReplaceIamInstanceProfileAssociation
- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute el escenario interactivo en un símbolo del sistema.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        self.resource_path = resource_path
        self.recommendation = recommendation
        self.autoscaler = autoscaler
        self.loadbalancer = loadbalancer
        self.param_helper = param_helper
        self.protocol = "HTTP"
        self.port = 80
        self.ssh_port = 22

    def deploy(self):
        recommendations_path = f"{self.resource_path}/recommendations.json"
        startup_script = f"{self.resource_path}/server_startup_script.sh"
        instance_policy = f"{self.resource_path}/instance_policy.json"

        print(
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) to create
 several AWS resources\n"
            "to set up a load-balanced web service endpoint and explore some ways
 to make it resilient\n"
            "against various kinds of failures.\n\n"
            "Some of the resources create by this demo are:\n"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
 book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
 contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
 several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
 Auto Scaling group to distribute requests."
        )
```

```
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "At this point, you have EC2 instances created. Once each instance
starts, it listens for\n"
```

```
            "HTTP requests. You can see these instances in the console or
continue with the demo."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\nCreating an Elastic Load Balancing target group and load balancer.
The target group\n"
            "defines how the load balancer connects to instances. The load
balancer provides a\n"
            "single endpoint where clients connect and dispatches requests to
instances in the group.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            sec_group, port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.port, current_ip_address
            )
            sec_group, ssh_port_is_open = self.autoscaler.verify_inbound_port(
                vpc, self.ssh_port, current_ip_address
            )
            if not port_is_open:
                print(
```

```
                    "For this example to work, the default security group for
 your default VPC must\n"
                    "allows access from this computer. You can either add it
 automatically from this\n"
                    "example or add it yourself using the AWS Management Console.
\n"
                )
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound traffic on port {self.port} from your computer's IP
 address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.port, current_ip_address
                    )
            if not ssh_port_is_open:
                if q.ask(
                    f"Do you want to add a rule to security group
 {sec_group['GroupId']} to allow\n"
                    f"inbound SSH traffic on port {self.ssh_port} for debugging
 from your computer's IP address of {current_ip_address}? (y/n) ",
                    q.is_yesno,
                ):
                    self.autoscaler.open_inbound_port(
                        sec_group["GroupId"], self.ssh_port, current_ip_address
                    )
            lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn't get a successful response from the load balancer
 endpoint. Troubleshoot by\n"
                "manually verifying that your VPC and security group are
 configured correctly and that\n"
                "you can successfully make a GET request to the load balancer
 endpoint:\n"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
```

```
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        actions = [
            "Send a GET request to the load balancer endpoint.",
            "Check the health of load balancer targets.",
            "Go to the next part of the demo.",
        ]
        choice = 0
        while choice != 2:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
```

```
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\nThis part of the demonstration shows how to toggle different parts
of the system\n"
            "to create situations where the web service fails, and shows how
using a resilient\n"
            "architecture can keep the web service running in spite of these
failures."
        )
        print("-" * 88)

        print(
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nNow, sending a GET request to the load balancer endpoint returns a
failure code. But, the service reports as\n"
            "healthy to the load balancer because shallow health checks don't
check for failure of the recommendation service."
        )
        self.demo_choices()

        print(
```

```
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\nLet's also substitute bad credentials for one of the instances in
the target group so that it can't\n"
            "access the DynamoDB recommendation table.\n"
        )
        self.autoscaler.create_instance_profile(
            ssm_only_policy,
            self.autoscaler.bad_creds_policy_name,
            self.autoscaler.bad_creds_role_name,
            self.autoscaler.bad_creds_profile_name,
            ["AmazonSSMManagedInstanceCore"],
        )
        instances = self.autoscaler.get_instances()
        bad_instance_id = instances[0]
        instance_profile = self.autoscaler.get_instance_profile(bad_instance_id)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
```

```
            "Now, sending a GET request to the load balancer endpoint returns
either a recommendation or a static response,\n"
            "depending on which instance is selected by the load balancer.\n"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
```

```
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
            "unhealthy instances, allowing them to fail open and return a static
response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
service.\n"
            "To keep things tidy and to avoid unwanted charges on your account,
we can clean up all AWS resources\n"
            "that were created for this demo."
        )
        if q.ask("Do you want to clean up all demo resources? (y/n) ",
q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            self.autoscaler.delete_key_pair()
```

```
            self.autoscaler.delete_template()
            self.autoscaler.delete_instance_profile(
                self.autoscaler.bad_creds_profile_name,
                self.autoscaler.bad_creds_role_name,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, we'll leave the resources intact.\n"
                "Don't forget to delete them when you're done with them or you
 might incur unexpected charges."
            )


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "--action",
        required=True,
        choices=["all", "deploy", "demo", "destroy"],
        help="The action to take for the demo. When 'all' is specified, resources
 are\n"
        "deployed, the demo is run, and resources are destroyed.",
    )
    parser.add_argument(
        "--resource_path",
        default="../../../workflows/resilient_service/resources",
        help="The path to resource files used by this example, such as IAM
 policies and\n"
        "instance scripts.",
    )
    args = parser.parse_args()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
```

```
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    )
    actions = [args.action] if args.action != "all" else ["deploy", "demo",
 "destroy"]
    for action in actions:
        if action == "deploy":
            runner.deploy()
        elif action == "demo":
            runner.demo()
        elif action == "destroy":
            runner.destroy()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    main()
```

Cree una clase que agrupe las acciones de escalado automático y Amazon EC2.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
```

```
    ):
        """
        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        :param inst_type: The type of EC2 instance to create, such as t3.micro.
        :param ami_param: The Systems Manager parameter used to look up the AMI
that is
                          created.
        :param autoscaling_client: A Boto3 EC2 Auto Scaling client.
        :param ec2_client: A Boto3 EC2 client.
        :param ssm_client: A Boto3 Systems Manager client.
        :param iam_client: A Boto3 IAM client.
        """
        self.inst_type = inst_type
        self.ami_param = ami_param
        self.autoscaling_client = autoscaling_client
        self.ec2_client = ec2_client
        self.ssm_client = ssm_client
        self.iam_client = iam_client
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        self.bad_creds_policy_name = f"{resource_prefix}-bc-pol"
        self.bad_creds_role_name = f"{resource_prefix}-bc-role"
        self.bad_creds_profile_name = f"{resource_prefix}-bc-prof"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
```

```
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        Creates a policy, role, and profile that is associated with instances
created by
        this class. An instance's associated profile defines a role that is
assumed by the
        instance. The role has attached policies that specify the AWS permissions
granted to
        clients that run on the instance.

        :param policy_file: The name of a JSON file that contains the policy
definition to
                            create and attach to the role.
        :param policy_name: The name to give the created policy.
        :param role_name: The name to give the created role.
        :param profile_name: The name to the created profile.
        :param aws_managed_policies: Additional AWS-managed policies that are
attached to
                                     the role, such as
AmazonSSMManagedInstanceCore to grant
                                     use of Systems Manager to send commands to
the instance.
        :return: The ARN of the profile that is created.
        """
        assume_role_doc = {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {"Service": "ec2.amazonaws.com"},
                    "Action": "sts:AssumeRole",
                }
            ],
```

```
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            self.iam_client.attach_role_policy(RoleName=role_name,
PolicyArn=policy_arn)
            for aws_policy in aws_managed_policies:
                self.iam_client.attach_role_policy(
                    RoleName=role_name,
                    PolicyArn=f"arn:aws:iam::aws:policy/{aws_policy}",
                )
            log.info("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        try:
```

```
            profile_response = self.iam_client.create_instance_profile(
                InstanceProfileName=profile_name
            )
            waiter = self.iam_client.get_waiter("instance_profile_exists")
            waiter.wait(InstanceProfileName=profile_name)
            time.sleep(10)  # wait a little longer
            profile_arn = profile_response["InstanceProfile"]["Arn"]
            self.iam_client.add_role_to_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )
            log.info("Created profile %s and added role %s.", profile_name,
role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                prof_response = self.iam_client.get_instance_profile(
                    InstanceProfileName=profile_name
                )
                profile_arn = prof_response["InstanceProfile"]["Arn"]
                log.info(
                    "Instance profile %s already exists, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        Gets data about the profile associated with an instance.

        :param instance_id: The ID of the instance to look up.
        :return: The profile data.
        """
        try:
            response =
self.ec2_client.describe_iam_instance_profile_associations(
                Filters=[{"Name": "instance-id", "Values": [instance_id]}]
            )
        except ClientError as err:
```

```
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        Replaces the profile associated with a running instance. After the
profile is
        replaced, the instance is rebooted to ensure that it uses the new
profile. When
        the instance is ready, Systems Manager is used to restart the Python web
server.

        :param instance_id: The ID of the instance to update.
        :param new_instance_profile_name: The name of the new profile to
associate with
                                          the specified instance.
        :param profile_association_id: The ID of the existing profile association
for the
                                       instance.
        """
        try:
            self.ec2_client.replace_iam_instance_profile_association(
                IamInstanceProfile={"Name": new_instance_profile_name},
                AssociationId=profile_association_id,
            )
            log.info(
                "Replaced instance profile for association %s with profile %s.",
                profile_association_id,
                new_instance_profile_name,
            )
            time.sleep(5)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
```

```
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )


    def delete_instance_profile(self, profile_name, role_name):
        """
        Detaches a role from an instance profile, detaches policies from the
role,
        and deletes all the resources.

        :param profile_name: The name of the profile to delete.
        :param role_name: The name of the role to delete.
        """
        try:
            self.iam_client.remove_role_from_instance_profile(
                InstanceProfileName=profile_name, RoleName=role_name
            )

self.iam_client.delete_instance_profile(InstanceProfileName=profile_name)
            log.info("Deleted instance profile %s.", profile_name)
            attached_policies = self.iam_client.list_attached_role_policies(
                RoleName=role_name
            )
            for pol in attached_policies["AttachedPolicies"]:
```

```
                self.iam_client.detach_role_policy(
                    RoleName=role_name, PolicyArn=pol["PolicyArn"]
                )
                if not pol["PolicyArn"].startswith("arn:aws:iam::aws"):
                    self.iam_client.delete_policy(PolicyArn=pol["PolicyArn"])
                log.info("Detached and deleted policy %s.", pol["PolicyName"])
            self.iam_client.delete_role(RoleName=role_name)
            log.info("Deleted role %s.", role_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )


    def create_key_pair(self, key_pair_name):
        """
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        try:
            response = self.ec2_client.create_key_pair(KeyName=key_pair_name)
            with open(f"{key_pair_name}.pem", "w") as file:
                file.write(response["KeyMaterial"])
            chmod(f"{key_pair_name}.pem", 0o600)
            log.info("Created key pair %s.", key_pair_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
```

```
        """
        try:
            self.ec2_client.delete_key_pair(KeyName=self.key_pair_name)
            remove(f"{self.key_pair_name}.pem")
            log.info("Deleted key pair %s.", self.key_pair_name)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )


    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        Creates an Amazon EC2 launch template to use with Amazon EC2 Auto
Scaling. The
        launch template specifies a Bash script in its user data field that runs
after
        the instance is started. This script installs Python packages and starts
a
        Python web server on the instance.

        :param server_startup_script_file: The path to a Bash script file that is
run
                                           when an instance starts.
        :param instance_policy_file: The path to a file that defines a
permissions policy
                                     to create and attach to the instance
profile.
        :return: Information about the newly created template.
        """
        template = {}
        try:
```

```
            self.create_key_pair(self.key_pair_name)
            self.create_instance_profile(
                instance_policy_file,
                self.instance_policy_name,
                self.instance_role_name,
                self.instance_profile_name,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            lt_response = self.ec2_client.create_launch_template(
                LaunchTemplateName=self.launch_template_name,
                LaunchTemplateData={
                    "InstanceType": self.inst_type,
                    "ImageId": ami_id,
                    "IamInstanceProfile": {"Name": self.instance_profile_name},
                    "UserData": base64.b64encode(
                        start_server_script.encode(encoding="utf-8")
                    ).decode(encoding="utf-8"),
                    "KeyName": self.key_pair_name,
                },
            )
            template = lt_response["LaunchTemplate"]
            log.info(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {err}."
                )
```

```
        return template


    def delete_template(self):
        """
        Deletes a launch template.
        """
        try:
            self.ec2_client.delete_launch_template(
                LaunchTemplateName=self.launch_template_name
            )
            self.delete_instance_profile(
                self.instance_profile_name, self.instance_role_name
            )
            log.info("Launch template %s deleted.", self.launch_template_name)
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.NotFoundException"
            ):
                log.info(
                    "Launch template %s does not exist, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        Gets a list of Availability Zones in the AWS Region of the Amazon EC2
client.

        :return: The list of Availability Zones for the client Region.
        """
        try:
            response = self.ec2_client.describe_availability_zones()
            zones = [zone["ZoneName"] for zone in response["AvailabilityZones"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
```

```
            return zones


    def create_group(self, group_size):
        """
        Creates an EC2 Auto Scaling group with the specified size.

        :param group_size: The number of instances to set for the minimum and
maximum in
                           the group.
        :return: The list of Availability Zones specified for the group.
        """
        zones = []
        try:
            zones = self.get_availability_zones()
            self.autoscaling_client.create_auto_scaling_group(
                AutoScalingGroupName=self.group_name,
                AvailabilityZones=zones,
                LaunchTemplate={
                    "LaunchTemplateName": self.launch_template_name,
                    "Version": "$Default",
                },
                MinSize=group_size,
                MaxSize=group_size,
            )
            log.info(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones
```

```
    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        try:
            as_response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            instance_ids = [
                i["InstanceId"]
                for i in as_response["AutoScalingGroups"][0]["Instances"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids


    def terminate_instance(self, instance_id):
        """
        Terminates and instances in an EC2 Auto Scaling group. After an instance
is
        terminated, it can no longer be accessed.

        :param instance_id: The ID of the instance to terminate.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        Attaches an Elastic Load Balancing (ELB) target group to this EC2 Auto
Scaling group.
```

```
        The target group specifies how the load balancer forward requests to the
instances
        in the group.

        :param lb_target_group: Data about the ELB target group to attach.
        """
        try:
            self.autoscaling_client.attach_load_balancer_target_groups(
                AutoScalingGroupName=self.group_name,
                TargetGroupARNs=[lb_target_group["TargetGroupArn"]],
            )
            log.info(
                "Attached load balancer target group %s to auto scaling group
%s.",
                lb_target_group["TargetGroupName"],
                self.group_name,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )


    def _try_terminate_instance(self, inst_id):
        stopping = False
        log.info(f"Stopping {inst_id}.")
        while not stopping:
            try:
                self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                    InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
                )
                stopping = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                    log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                    time.sleep(10)
                else:
                    raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

    def _try_delete_group(self):
```

```
        """
        Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
        the function waits and retries until the group is successfully deleted.
        """
        stopped = False
        while not stopped:
            try:
                self.autoscaling_client.delete_auto_scaling_group(
                    AutoScalingGroupName=self.group_name
                )
                stopped = True
                log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
            except ClientError as err:
                if (
                    err.response["Error"]["Code"] == "ResourceInUse"
                    or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
                ):
                    log.info(
                        "Some instances are still running. Waiting for them to
stop..."
                    )
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
```

```
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        Gets the default VPC for the account.

        :return: Data about the default VPC.
        """
        try:
            response = self.ec2_client.describe_vpcs(
                Filters=[{"Name": "is-default", "Values": ["true"]}]
            )
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        Verify the default security group of the specified VPC allows ingress
 from this
        computer. This can be done by allowing ingress from this computer's IP
        address. In some situations, such as connecting from a corporate network,
 you
        must instead specify a prefix list ID. You can also temporarily open the
 port to
        any IP address while running this example. If you do, be sure to remove
 public
        access when you're done.

        :param vpc: The VPC used by this example.
        :param port: The port to verify.
```

```
        :param ip_address: This computer's IP address.
        :return: The default security group of the specific VPC, and a value that
indicates
                 whether the specified port is open.
        """
        try:
            response = self.ec2_client.describe_security_groups(
                Filters=[
                    {"Name": "group-name", "Values": ["default"]},
                    {"Name": "vpc-id", "Values": [vpc["VpcId"]]},
                ]
            )
            sec_group = response["SecurityGroups"][0]
            port_is_open = False
            log.info("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", ip_perm)
                    for ip_range in ip_perm["IpRanges"]:
                        cidr = ip_range.get("CidrIp", "")
                        if cidr.startswith(ip_address) or cidr == "0.0.0.0/0":
                            port_is_open = True
                    if ip_perm["PrefixListIds"]:
                        port_is_open = True
                    if not port_is_open:
                        log.info(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
```

```
        """
        Add an ingress rule to the specified security group that allows access on
the
        specified port from the specified IP address.

        :param sec_group_id: The ID of the security group to modify.
        :param port: The port to open.
        :param ip_address: The IP address that is granted access.
        """
        try:
            self.ec2_client.authorize_security_group_ingress(
                GroupId=sec_group_id,
                CidrIp=f"{ip_address}/32",
                FromPort=port,
                ToPort=port,
                IpProtocol="tcp",
            )
            log.info(
                "Authorized ingress to %s on port %s from %s.",
                sec_group_id,
                port,
                ip_address,
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        Gets the default subnets in a VPC for a specified list of Availability
Zones.

        :param vpc_id: The ID of the VPC to look up.
        :param zones: The list of Availability Zones to look up.
        :return: The list of subnets found.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
```

```
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

Cree una clase que resuma las acciones de Elastic Load Balancing.

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
 this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)
```

```
    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        Creates an Elastic Load Balancing target group. The target group
specifies how
        the load balancer forward requests to instances in the group and how
instance
        health is checked.

        To speed up this demo, the health check is configured with shortened
times and
        lower thresholds. In production, you might want to decrease the
sensitivity of
        your health checks to avoid unwanted failures.

        :param protocol: The protocol to use to forward requests, such as 'HTTP'.
        :param port: The port to use to forward requests, such as 80.
        :param vpc_id: The ID of the VPC in which the load balancer exists.
        :return: Data about the newly created target group.
        """
        try:
            response = self.elb_client.create_target_group(
                Name=self.target_group_name,
                Protocol=protocol,
                Port=port,
```

```
                HealthCheckPath="/healthcheck",
                HealthCheckIntervalSeconds=10,
                HealthCheckTimeoutSeconds=5,
                HealthyThresholdCount=2,
                UnhealthyThresholdCount=2,
                VpcId=vpc_id,
            )
            target_group = response["TargetGroups"][0]
            log.info("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
```

```
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param subnet_ids: A list of subnets to associate with the load balancer.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created load balancer.
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
```

```
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )
```

```
   def verify_load_balancer_endpoint(self):
       """
       Verify this computer can successfully send a GET request to the load
balancer endpoint.
       """
       success = False
       retries = 3
       while not success and retries > 0:
           try:
               lb_response = requests.get(f"http://{self.endpoint()}")
               log.info(
                   "Got response %s from load balancer endpoint.",
                   lb_response.status_code,
               )
               if lb_response.status_code == 200:
                   success = True
               else:
                   retries = 0
           except requests.exceptions.ConnectionError:
               log.info(
                   "Got connection error from load balancer endpoint,
retrying..."
               )
               retries -= 1
               time.sleep(10)
       return success

   def check_target_health(self):
       """
       Checks the health of the instances in the target group.

       :return: The health status of the target group.
       """
       try:
           tg_response = self.elb_client.describe_target_groups(
               Names=[self.target_group_name]
           )
           health_response = self.elb_client.describe_target_health(
               TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
           )
       except ClientError as err:
           raise LoadBalancerError(
```

```
                f"Couldn't check health of {self.target_group_name} targets:
 {err}"
            )
        else:
            return health_response["TargetHealthDescriptions"]
```

Cree una clase que utilice DynamoDB para simular un servicio de recomendaciones.

```
class RecommendationService:
    """
    Encapsulates a DynamoDB table to use as a service that recommends books,
 movies,
    and songs.
    """

    def __init__(self, table_name, dynamodb_client):
        """
        :param table_name: The name of the DynamoDB recommendations table.
        :param dynamodb_client: A Boto3 DynamoDB client.
        """
        self.table_name = table_name
        self.dynamodb_client = dynamodb_client

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        Creates a DynamoDB table to use a recommendation service. The table has a
        hash key named 'MediaType' that defines the type of media recommended,
 such as
```

```
        Book or Movie, and a range key named 'ItemId' that, combined with the
MediaType,
        forms a unique identifier for the recommended item.

        :return: Data about the newly created table.
        """
        try:
            response = self.dynamodb_client.create_table(
                TableName=self.table_name,
                AttributeDefinitions=[
                    {"AttributeName": "MediaType", "AttributeType": "S"},
                    {"AttributeName": "ItemId", "AttributeType": "N"},
                ],
                KeySchema=[
                    {"AttributeName": "MediaType", "KeyType": "HASH"},
                    {"AttributeName": "ItemId", "KeyType": "RANGE"},
                ],
                ProvisionedThroughput={"ReadCapacityUnits": 5,
"WriteCapacityUnits": 5},
            )
            log.info("Creating table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s created.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceInUseException":
                log.info("Table %s exists, nothing to be do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        try:
            with open(data_file) as data:
                items = json.load(data)
            batch = [{"PutRequest": {"Item": item}} for item in items]
```

```
            self.dynamodb_client.batch_write_item(RequestItems={self.table_name:
 batch})
            log.info(
                "Populated table %s with items from %s.", self.table_name,
 data_file
            )
        except ClientError as err:
            raise RecommendationServiceError(
                self.table_name, f"Couldn't populate table from {data_file}:
 {err}"
            )

    def destroy(self):
        """
        Deletes the recommendations table.
        """
        try:
            self.dynamodb_client.delete_table(TableName=self.table_name)
            log.info("Deleting table %s...", self.table_name)
            waiter = self.dynamodb_client.get_waiter("table_not_exists")
            waiter.wait(TableName=self.table_name)
            log.info("Table %s deleted.", self.table_name)
        except ClientError as err:
            if err.response["Error"]["Code"] == "ResourceNotFoundException":
                log.info("Table %s does not exist, nothing to do.",
 self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when deleting table: {err}."
                )
```

Cree una clase que agrupe las acciones de Systems Manager.

```
class ParameterHelper:
    """
    Encapsulates Systems Manager parameters. This example uses these parameters
 to drive
    the demonstration of resilient architecture, such as failure of a dependency
 or
    how the service responds to a health check.
```

```
    """

    table = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :param table_name: The name of the DynamoDB table that is used as a
recommendation
                           service.
        :param ssm_client: A Boto3 Systems Manager client.
        """
        self.ssm_client = ssm_client
        self.table_name = table_name

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        Resets the Systems Manager parameters to starting values for the demo.
        These are the name of the DynamoDB recommendation table, no response when
a
        dependency fails, and shallow health checks.
        """
        self.put(self.table, self.table_name)
        self.put(self.failure_response, "none")
        self.put(self.health_check, "shallow")

    def put(self, name, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
```

```
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {err}"
            )
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - AttachLoadBalancerTargetGroups
  - CreateAutoScalingGroup
  - CreateInstanceProfile
  - CreateLaunchTemplate
  - CreateListener
  - CreateLoadBalancer
  - CreateTargetGroup
  - DeleteAutoScalingGroup
  - DeleteInstanceProfile
  - DeleteLaunchTemplate
  - DeleteLoadBalancer
  - DeleteTargetGroup
  - DescribeAutoScalingGroups
  - DescribeAvailabilityZones
  - DescribeIamInstanceProfileAssociations
  - DescribeInstances
  - DescribeLoadBalancers
  - DescribeSubnets
  - DescribeTargetGroups
  - DescribeTargetHealth
  - DescribeVpcs
  - RebootInstances
  - ReplaceIamInstanceProfileAssociation

- TerminateInstanceInAutoScalingGroup
- UpdateAutoScalingGroup

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de un grupo de IAM y adición de un usuario a un grupo mediante un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Cree un grupo y concédale todos los permisos de acceso a Amazon S3.
- Cree un nuevo usuario sin permisos para acceder a Amazon S3.
- Agregue el usuario al grupo, muestre que ahora tiene permisos para Amazon S3 y, a continuación, limpie los recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;
```

```
namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
   public async Task<bool> AttachRolePolicyAsync(string policyArn, string
roleName)
   {
       var response = await _IAMService.AttachRolePolicyAsync(new
AttachRolePolicyRequest
       {
           PolicyArn = policyArn,
           RoleName = roleName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Create an IAM access key for a user.
   /// </summary>
   /// <param name="userName">The username for which to create the IAM access
   /// key.</param>
   /// <returns>The AccessKey.</returns>
   public async Task<AccessKey> CreateAccessKeyAsync(string userName)
   {
       var response = await _IAMService.CreateAccessKeyAsync(new
CreateAccessKeyRequest
       {
           UserName = userName,
       });

       return response.AccessKey;

   }


   /// <summary>
   /// Create an IAM group.
   /// </summary>
   /// <param name="groupName">The name to give the IAM group.</param>
   /// <returns>The IAM group that was created.</returns>
   public async Task<Group> CreateGroupAsync(string groupName)
   {
       var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
       return response.Group;
   }
```

```
    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
        return response.Role.Arn;
    }
```

```
    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
 userName)
    {
```

```
        var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
        {
            AccessKeyId = accessKeyId,
            UserName = userName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
        });
```

```
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
roleName)
```

```
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
```

```
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
```

```
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }


    /// <summary>
    /// List IAM role policies.
    /// </summary>
```

```
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
```

```
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
```

```
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
accessKeyId });
```

```
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}


using Microsoft.Extensions.Configuration;

namespace IAMGroups;

public class IAMGroups
{
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"]," +
            " \"Effect\" : \"Allow\"," +
            " \"Resource\" : \"*\"" +
        "}]" +
    "}";


    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
```

```
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMGroups>();

        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var groupUserName = configuration["GroupUserName"];
        var groupName = configuration["GroupName"];
        var groupPolicyName = configuration["GroupPolicyName"];
        var groupBucketName = configuration["GroupBucketName"];

        var wrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayGroupsOverview();
        uiWrapper.PressEnter();

        // Create an IAM group.
        uiWrapper.DisplayTitle("Create IAM group");
        Console.WriteLine("Let's begin by creating a new IAM group.");
        var group = await wrapper.CreateGroupAsync(groupName);

        // Add an inline IAM policy to the group.
        uiWrapper.DisplayTitle("Add policy to group");
        Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");

        await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);
```

```
        uiWrapper.PressEnter();

        // Now create a new user.
        uiWrapper.DisplayTitle("Create an IAM user");
        Console.WriteLine("Now let's create a new IAM user.");
        var groupUser = await wrapper.CreateUserAsync(groupUserName);

        // Add the new user to the group.
        uiWrapper.DisplayTitle("Add the user to the group");
        Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
        await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

        Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
        uiWrapper.PressEnter();

        Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

        // Create access and secret keys for the user.
        var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
        Console.WriteLine("Key created.");
        uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

        uiWrapper.DisplayTitle("List buckets");
        Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

        var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
        var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client, stsClient);

        var buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
```

```
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        // Show that the user also has write access to Amazon S3 by creating
        // a new bucket.
        uiWrapper.DisplayTitle("Create a bucket");
        Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
        var success = await s3Wrapper.PutBucketAsync(groupBucketName);

        if (success)
        {
            Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
        }

        uiWrapper.PressEnter();

        Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");

        buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("First delete the bucket we created.");
        await s3Wrapper.DeleteBucketAsync(groupBucketName);

        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
{groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);
```

```
        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);

        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
 group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
```

```
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
 { BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
```

```
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
{ BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
```

```
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
 (IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
 full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
 creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
```

```
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
```

```
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET.
  - AddUserToGroup
  - AttachRolePolicy
  - CreateAccessKey
  - CreateGroup
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeleteGroup
  - DeleteGroupPolicy
  - DeleteUser
  - PutGroupPolicy
  - RemoveUserFromGroup

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear un usuario de IAM y asumir un rol con AWS STS con un SDK de AWS

Los siguientes ejemplos de código muestran cómo crear un usuario y asumir un rol.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Crear un usuario que no tenga permisos.
- Crear un rol que conceda permiso para enumerar los buckets de Amazon S3 para la cuenta.
- Agregar una política para que el usuario asuma el rol.
- Asuma el rol y enumere los buckets de S3 con credenciales temporales. A continuación, limpie los recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
global using Amazon.IdentityManagement;
global using Amazon.S3;
global using Amazon.SecurityToken;
global using IAMActions;
global using IamScenariosCommon;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
```

```
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;


namespace IAMActions;

public class IAMWrapper
{
    private readonly IAmazonIdentityManagementService _IAMService;

    /// <summary>
    /// Constructor for the IAMWrapper class.
    /// </summary>
    /// <param name="IAMService">An IAM client object.</param>
    public IAMWrapper(IAmazonIdentityManagementService IAMService)
    {
        _IAMService = IAMService;
    }

    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Attach an IAM policy to a role.
    /// </summary>
    /// <param name="policyArn">The policy to attach.</param>
```

```
    /// <param name="roleName">The role that the policy will be attached to.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AttachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.AttachRolePolicyAsync(new
 AttachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an IAM access key for a user.
    /// </summary>
    /// <param name="userName">The username for which to create the IAM access
    /// key.</param>
    /// <returns>The AccessKey.</returns>
    public async Task<AccessKey> CreateAccessKeyAsync(string userName)
    {
        var response = await _IAMService.CreateAccessKeyAsync(new
 CreateAccessKeyRequest
        {
            UserName = userName,
        });

        return response.AccessKey;

    }


    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
```

```
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <summary>
    /// Create an IAM policy.
    /// </summary>
    /// <param name="policyName">The name to give the new IAM policy.</param>
    /// <param name="policyDocument">The policy document for the new policy.</
param>
    /// <returns>The new IAM policy object.</returns>
    public async Task<ManagedPolicy> CreatePolicyAsync(string policyName, string
 policyDocument)
    {
        var response = await _IAMService.CreatePolicyAsync(new
 CreatePolicyRequest
        {
            PolicyDocument = policyDocument,
            PolicyName = policyName,
        });

        return response.Policy;
    }


    /// <summary>
    /// Create a new IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="rolePolicyDocument">The name of the IAM policy document
    /// for the new role.</param>
    /// <returns>The Amazon Resource Name (ARN) of the role.</returns>
    public async Task<string> CreateRoleAsync(string roleName, string
 rolePolicyDocument)
    {
        var request = new CreateRoleRequest
        {
            RoleName = roleName,
            AssumeRolePolicyDocument = rolePolicyDocument,
        };

        var response = await _IAMService.CreateRoleAsync(request);
```

```
        return response.Role.Arn;
    }


    /// <summary>
    /// Create an IAM service-linked role.
    /// </summary>
    /// <param name="serviceName">The name of the AWS Service.</param>
    /// <param name="description">A description of the IAM service-linked role.</
param>
    /// <returns>The IAM role that was created.</returns>
    public async Task<Role> CreateServiceLinkedRoleAsync(string serviceName,
 string description)
    {
        var request = new CreateServiceLinkedRoleRequest
        {
            AWSServiceName = serviceName,
            Description = description
        };

        var response = await _IAMService.CreateServiceLinkedRoleAsync(request);
        return response.Role;
    }


    /// <summary>
    /// Create an IAM user.
    /// </summary>
    /// <param name="userName">The username for the new IAM user.</param>
    /// <returns>The IAM user that was created.</returns>
    public async Task<User> CreateUserAsync(string userName)
    {
        var response = await _IAMService.CreateUserAsync(new CreateUserRequest
 { UserName = userName });
        return response.User;
    }


    /// <summary>
    /// Delete an IAM user's access key.
    /// </summary>
    /// <param name="accessKeyId">The Id for the IAM access key.</param>
    /// <param name="userName">The username of the user that owns the IAM
    /// access key.</param>
```

```
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteAccessKeyAsync(string accessKeyId, string
userName)
   {
       var response = await _IAMService.DeleteAccessKeyAsync(new
DeleteAccessKeyRequest
       {
           AccessKeyId = accessKeyId,
           UserName = userName,
       });

       return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
   }


   /// <summary>
   /// Delete an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group to delete.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteGroupAsync(string groupName)
   {
       var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
       return response.HttpStatusCode == HttpStatusCode.OK;
   }


   /// <summary>
   /// Delete an IAM policy associated with an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group associated with the
   /// policy.</param>
   /// <param name="policyName">The name of the policy to delete.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
   {
       var request = new DeleteGroupPolicyRequest()
       {
           GroupName = groupName,
           PolicyName = policyName,
       };
```

```
        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the policy to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeletePolicyAsync(string policyArn)
    {
        var response = await _IAMService.DeletePolicyAsync(new
 DeletePolicyRequest { PolicyArn = policyArn });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRoleAsync(string roleName)
    {
        var response = await _IAMService.DeleteRoleAsync(new DeleteRoleRequest
 { RoleName = roleName });
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM role policy.
    /// </summary>
    /// <param name="roleName">The name of the IAM role.</param>
    /// <param name="policyName">The name of the IAM role policy to delete.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteRolePolicyAsync(string roleName, string
 policyName)
    {
        var response = await _IAMService.DeleteRolePolicyAsync(new
 DeleteRolePolicyRequest
```

```
        {
            PolicyName = policyName,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user.
    /// </summary>
    /// <param name="userName">The username of the IAM user to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserAsync(string userName)
    {
        var response = await _IAMService.DeleteUserAsync(new DeleteUserRequest
{ UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM user policy.
    /// </summary>
    /// <param name="policyName">The name of the IAM policy to delete.</param>
    /// <param name="userName">The username of the IAM user.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteUserPolicyAsync(string policyName, string
userName)
    {
        var response = await _IAMService.DeleteUserPolicyAsync(new
DeleteUserPolicyRequest { PolicyName = policyName, UserName = userName });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Detach an IAM policy from an IAM role.
    /// </summary>
    /// <param name="policyArn">The Amazon Resource Name (ARN) of the IAM
policy.</param>
```

```
    /// <param name="roleName">The name of the IAM role.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DetachRolePolicyAsync(string policyArn, string
 roleName)
    {
        var response = await _IAMService.DetachRolePolicyAsync(new
 DetachRolePolicyRequest
        {
            PolicyArn = policyArn,
            RoleName = roleName,
        });

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Gets the IAM password policy for an AWS account.
    /// </summary>
    /// <returns>The PasswordPolicy for the AWS account.</returns>
    public async Task<PasswordPolicy> GetAccountPasswordPolicyAsync()
    {
        var response = await _IAMService.GetAccountPasswordPolicyAsync(new
 GetAccountPasswordPolicyRequest());
        return response.PasswordPolicy;
    }


    /// <summary>
    /// Get information about an IAM policy.
    /// </summary>
    /// <param name="policyArn">The IAM policy to retrieve information for.</
param>
    /// <returns>The IAM policy.</returns>
    public async Task<ManagedPolicy> GetPolicyAsync(string policyArn)
    {

        var response = await _IAMService.GetPolicyAsync(new GetPolicyRequest
 { PolicyArn = policyArn });
        return response.Policy;
    }


    /// <summary>
```

```
    /// Get information about an IAM role.
    /// </summary>
    /// <param name="roleName">The name of the IAM role to retrieve information
    /// for.</param>
    /// <returns>The IAM role that was retrieved.</returns>
    public async Task<Role> GetRoleAsync(string roleName)
    {
        var response = await _IAMService.GetRoleAsync(new GetRoleRequest
        {
            RoleName = roleName,
        });

        return response.Role;
    }


    /// <summary>
    /// Get information about an IAM user.
    /// </summary>
    /// <param name="userName">The username of the user.</param>
    /// <returns>An IAM user object.</returns>
    public async Task<User> GetUserAsync(string userName)
    {
        var response = await _IAMService.GetUserAsync(new GetUserRequest
{ UserName = userName });
        return response.User;
    }


    /// <summary>
    /// List the IAM role policies that are attached to an IAM role.
    /// </summary>
    /// <param name="roleName">The IAM role to list IAM policies for.</param>
    /// <returns>A list of the IAM policies attached to the IAM role.</returns>
    public async Task<List<AttachedPolicyType>>
ListAttachedRolePoliciesAsync(string roleName)
    {
        var attachedPolicies = new List<AttachedPolicyType>();
        var attachedRolePoliciesPaginator =
_IAMService.Paginators.ListAttachedRolePolicies(new
ListAttachedRolePoliciesRequest { RoleName = roleName });

        await foreach (var response in attachedRolePoliciesPaginator.Responses)
        {
```

```
            attachedPolicies.AddRange(response.AttachedPolicies);
        }

        return attachedPolicies;
    }


    /// <summary>
    /// List IAM groups.
    /// </summary>
    /// <returns>A list of IAM groups.</returns>
    public async Task<List<Group>> ListGroupsAsync()
    {
        var groupsPaginator = _IAMService.Paginators.ListGroups(new
ListGroupsRequest());
        var groups = new List<Group>();

        await foreach (var response in groupsPaginator.Responses)
        {
            groups.AddRange(response.Groups);
        }

        return groups;
    }


    /// <summary>
    /// List IAM policies.
    /// </summary>
    /// <returns>A list of the IAM policies.</returns>
    public async Task<List<ManagedPolicy>> ListPoliciesAsync()
    {
        var listPoliciesPaginator = _IAMService.Paginators.ListPolicies(new
ListPoliciesRequest());
        var policies = new List<ManagedPolicy>();

        await foreach (var response in listPoliciesPaginator.Responses)
        {
            policies.AddRange(response.Policies);
        }

        return policies;
    }
```

```
    /// <summary>
    /// List IAM role policies.
    /// </summary>
    /// <param name="roleName">The IAM role for which to list IAM policies.</
param>
    /// <returns>A list of IAM policy names.</returns>
    public async Task<List<string>> ListRolePoliciesAsync(string roleName)
    {
        var listRolePoliciesPaginator =
 _IAMService.Paginators.ListRolePolicies(new ListRolePoliciesRequest { RoleName =
 roleName });
        var policyNames = new List<string>();

        await foreach (var response in listRolePoliciesPaginator.Responses)
        {
            policyNames.AddRange(response.PolicyNames);
        }

        return policyNames;
    }


    /// <summary>
    /// List IAM roles.
    /// </summary>
    /// <returns>A list of IAM roles.</returns>
    public async Task<List<Role>> ListRolesAsync()
    {
        var listRolesPaginator = _IAMService.Paginators.ListRoles(new
 ListRolesRequest());
        var roles = new List<Role>();

        await foreach (var response in listRolesPaginator.Responses)
        {
            roles.AddRange(response.Roles);
        }

        return roles;
    }


    /// <summary>
    /// List SAML authentication providers.
```

```
    /// </summary>
    /// <returns>A list of SAML providers.</returns>
    public async Task<List<SAMLProviderListEntry>> ListSAMLProvidersAsync()
    {
        var response = await _IAMService.ListSAMLProvidersAsync(new
ListSAMLProvidersRequest());
        return response.SAMLProviderList;
    }


    /// <summary>
    /// List IAM users.
    /// </summary>
    /// <returns>A list of IAM users.</returns>
    public async Task<List<User>> ListUsersAsync()
    {
        var listUsersPaginator = _IAMService.Paginators.ListUsers(new
ListUsersRequest());
        var users = new List<User>();

        await foreach (var response in listUsersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
```

```
        };

        var response = await
 _IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
 policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Update the inline policy document embedded in a role.
    /// </summary>
    /// <param name="policyName">The name of the policy to embed.</param>
    /// <param name="roleName">The name of the role to update.</param>
    /// <param name="policyDocument">The policy document that defines the role.</
param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutRolePolicyAsync(string policyName, string
 roleName, string policyDocument)
    {
        var request = new PutRolePolicyRequest
        {
```

```
            PolicyName = policyName,
            RoleName = roleName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutRolePolicyAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM user.
    /// </summary>
    /// <param name="userName">The name of the IAM user.</param>
    /// <param name="policyName">The name of the IAM policy.</param>
    /// <param name="policyDocument">The policy document defining the IAM
policy.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> PutUserPolicyAsync(string userName, string
policyName, string policyDocument)
    {
        var request = new PutUserPolicyRequest
        {
            UserName = userName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutUserPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }

    /// <summary>
    /// Wait for a new access key to be ready to use.
    /// </summary>
    /// <param name="accessKeyId">The Id of the access key.</param>
    /// <returns>A boolean value indicating the success of the action.</returns>
    public async Task<bool> WaitUntilAccessKeyIsReady(string accessKeyId)
    {
        var keyReady = false;

        do
        {
            try
```

```
            {
                var response = await _IAMService.GetAccessKeyLastUsedAsync(
                    new GetAccessKeyLastUsedRequest { AccessKeyId =
 accessKeyId });
                if (response.UserName is not null)
                {
                    keyReady = true;
                }
            }
            catch (NoSuchEntityException)
            {
                keyReady = false;
            }
        } while (!keyReady);

        return keyReady;
    }
}



using Microsoft.Extensions.Configuration;

namespace IAMBasics;

public class IAMBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for the AWS service.
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
            services.AddAWSService<IAmazonIdentityManagementService>()
            .AddTransient<IAMWrapper>()
            .AddTransient<UIWrapper>()
            )
```

```
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<IAMBasics>();


        IConfiguration configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load test settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        // Values needed for user, role, and policies.
        string userName = configuration["UserName"]!;
        string s3PolicyName = configuration["S3PolicyName"]!;
        string roleName = configuration["RoleName"]!;


        var iamWrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayBasicsOverview();
        uiWrapper.PressEnter();

        // First create a user. By default, the new user has
        // no permissions.
        uiWrapper.DisplayTitle("Create User");
        Console.WriteLine($"Creating a new user with user name: {userName}.");
        var user = await iamWrapper.CreateUserAsync(userName);
        var userArn = user.Arn;

        Console.WriteLine($"Successfully created user: {userName} with ARN:
{userArn}.");
        uiWrapper.WaitABit(15, "Now let's wait for the user to be ready for
use.");

        // Define a role policy document that allows the new user
        // to assume the role.
        string assumeRolePolicyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          "\"Statement\": [{" +
              "\"Effect\": \"Allow\"," +
              "\"Principal\": {" +
```

```
            $" \"AWS\": \"{userArn}\"" +
            "}," +
            "\"Action\": \"sts:AssumeRole\"" +
        "}]" +
      "}";

      // Permissions to list all buckets.
      string policyDocument = "{" +
          "\"Version\": \"2012-10-17\"," +
          " \"Statement\" : [{" +
              " \"Action\" : [\"s3:ListAllMyBuckets\"]," +
              " \"Effect\" : \"Allow\"," +
              " \"Resource\" : \"*\"" +
          "}]" +
      "}";

      // Create an AccessKey for the user.
      uiWrapper.DisplayTitle("Create access key");
      Console.WriteLine("Now let's create an access key for the new user.");
      var accessKey = await iamWrapper.CreateAccessKeyAsync(userName);

      var accessKeyId = accessKey.AccessKeyId;
      var secretAccessKey = accessKey.SecretAccessKey;

      Console.WriteLine($"We have created the access key with Access key id:
{accessKeyId}.");

      Console.WriteLine("Now let's wait until the IAM access key is ready to
use.");
      var keyReady = await iamWrapper.WaitUntilAccessKeyIsReady(accessKeyId);

      // Now try listing the Amazon Simple Storage Service (Amazon S3)
      // buckets. This should fail at this point because the user doesn't
      // have permissions to perform this task.
      uiWrapper.DisplayTitle("Try to display Amazon S3 buckets");
      Console.WriteLine("Now let's try to display a list of the user's Amazon
S3 buckets.");
      var s3Client1 = new AmazonS3Client(accessKeyId, secretAccessKey);
      var stsClient1 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

      var s3Wrapper = new S3Wrapper(s3Client1, stsClient1);
      var buckets = await s3Wrapper.ListMyBucketsAsync();
```

```
        Console.WriteLine(buckets is null
            ? "As expected, the call to list the buckets has returned a null
list."
            : "Something went wrong. This shouldn't have worked.");

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create IAM role");
        Console.WriteLine($"Creating the role: {roleName}");

        // Creating an IAM role to allow listing the S3 buckets. A role name
        // is not case sensitive and must be unique to the account for which it
        // is created.
        var roleArn = await iamWrapper.CreateRoleAsync(roleName,
assumeRolePolicyDocument);

        uiWrapper.PressEnter();

        // Create a policy with permissions to list S3 buckets.
        uiWrapper.DisplayTitle("Create IAM policy");
        Console.WriteLine($"Creating the policy: {s3PolicyName}");
        Console.WriteLine("with permissions to list the Amazon S3 buckets for the
account.");
        var policy = await iamWrapper.CreatePolicyAsync(s3PolicyName,
policyDocument);

        // Wait 15 seconds for the IAM policy to be available.
        uiWrapper.WaitABit(15, "Waiting for the policy to be available.");

        // Attach the policy to the role you created earlier.
        uiWrapper.DisplayTitle("Attach new IAM policy");
        Console.WriteLine("Now let's attach the policy to the role.");
        await iamWrapper.AttachRolePolicyAsync(policy.Arn, roleName);

        // Wait 15 seconds for the role to be updated.
        Console.WriteLine();
        uiWrapper.WaitABit(15, "Waiting for the policy to be attached.");

        // Use the AWS Security Token Service (AWS STS) to have the user
        // assume the role we created.
        var stsClient2 = new AmazonSecurityTokenServiceClient(accessKeyId,
secretAccessKey);

        // Wait for the new credentials to become valid.
```

```
        uiWrapper.WaitABit(10, "Waiting for the credentials to be valid.");

        var assumedRoleCredentials = await
s3Wrapper.AssumeS3RoleAsync("temporary-session", roleArn);

        // Try again to list the buckets using the client created with
        // the new user's credentials. This time, it should work.
        var s3Client2 = new AmazonS3Client(assumedRoleCredentials);

        s3Wrapper.UpdateClients(s3Client2, stsClient2);

        buckets = await s3Wrapper.ListMyBucketsAsync();

        uiWrapper.DisplayTitle("List Amazon S3 buckets");
        Console.WriteLine("This time we should have buckets to list.");
        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName} created:
{bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        // Now clean up all the resources used in the example.
        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("Thank you for watching. The IAM Basics demo is
complete.");
        Console.WriteLine("Please wait while we clean up the resources we
created.");

        await iamWrapper.DetachRolePolicyAsync(policy.Arn, roleName);

        await iamWrapper.DeletePolicyAsync(policy.Arn);

        await iamWrapper.DeleteRoleAsync(roleName);

        await iamWrapper.DeleteAccessKeyAsync(accessKeyId, userName);

        await iamWrapper.DeleteUserAsync(userName);

        uiWrapper.PressEnter();
```

```
        Console.WriteLine("All done cleaning up our resources. Thank you for your
 patience.");
    }
}


namespace IamScenariosCommon;

using System.Net;

/// <summary>
/// A class to perform Amazon Simple Storage Service (Amazon S3) actions for
/// the IAM Basics scenario.
/// </summary>
public class S3Wrapper
{
    private IAmazonS3 _s3Service;
    private IAmazonSecurityTokenService _stsService;

    /// <summary>
    /// Constructor for the S3Wrapper class.
    /// </summary>
    /// <param name="s3Service">An Amazon S3 client object.</param>
    /// <param name="stsService">An AWS Security Token Service (AWS STS)
    /// client object.</param>
    public S3Wrapper(IAmazonS3 s3Service, IAmazonSecurityTokenService stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }

    /// <summary>
    /// Assumes an AWS Identity and Access Management (IAM) role that allows
    /// Amazon S3 access for the current session.
    /// </summary>
    /// <param name="roleSession">A string representing the current session.</
param>
    /// <param name="roleToAssume">The name of the IAM role to assume.</param>
    /// <returns>Credentials for the newly assumed IAM role.</returns>
    public async Task<Credentials> AssumeS3RoleAsync(string roleSession, string
 roleToAssume)
    {
        // Create the request to use with the AssumeRoleAsync call.
```

```
        var request = new AssumeRoleRequest()
        {
            RoleSessionName = roleSession,
            RoleArn = roleToAssume,
        };

        var response = await _stsService.AssumeRoleAsync(request);

        return response.Credentials;
    }


    /// <summary>
    /// Delete an S3 bucket.
    /// </summary>
    /// <param name="bucketName">Name of the S3 bucket to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteBucketAsync(string bucketName)
    {
        var result = await _s3Service.DeleteBucketAsync(new DeleteBucketRequest
{ BucketName = bucketName });
        return result.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// List the buckets that are owned by the user's account.
    /// </summary>
    /// <returns>Async Task.</returns>
    public async Task<List<S3Bucket>?> ListMyBucketsAsync()
    {
        try
        {
            // Get the list of buckets accessible by the new user.
            var response = await _s3Service.ListBucketsAsync();

            return response.Buckets;
        }
        catch (AmazonS3Exception ex)
        {
            // Something else went wrong. Display the error message.
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }
```

```
    /// <summary>
    /// Create a new S3 bucket.
    /// </summary>
    /// <param name="bucketName">The name for the new bucket.</param>
    /// <returns>A Boolean value indicating whether the action completed
    /// successfully.</returns>
    public async Task<bool> PutBucketAsync(string bucketName)
    {
        var response = await _s3Service.PutBucketAsync(new PutBucketRequest
 { BucketName = bucketName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

    /// <summary>
    /// Update the client objects with new client objects. This is available
    /// because the scenario uses the methods of this class without and then
    /// with the proper permissions to list S3 buckets.
    /// </summary>
    /// <param name="s3Service">The Amazon S3 client object.</param>
    /// <param name="stsService">The AWS STS client object.</param>
    public void UpdateClients(IAmazonS3 s3Service, IAmazonSecurityTokenService
 stsService)
    {
        _s3Service = s3Service;
        _stsService = stsService;
    }
}


namespace IamScenariosCommon;

public class UIWrapper
{
    public readonly string SepBar = new('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the IAM Groups scenario.
    /// </summary>
    public void DisplayGroupsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to the IAM Groups Demo");
```

```
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates an Amazon Identity and Access Management
(IAM) group.");
        Console.WriteLine("\t2. Adds an IAM policy to the IAM group giving it
full access to Amazon S3.");
        Console.WriteLine("\t3. Creates a new IAM user.");
        Console.WriteLine("\t4. Creates an IAM access key for the user.");
        Console.WriteLine("\t5. Adds the user to the IAM group.");
        Console.WriteLine("\t6. Lists the buckets on the account.");
        Console.WriteLine("\t7. Proves that the user has full Amazon S3 access by
creating a bucket.");
        Console.WriteLine("\t8. List the buckets again to show the new bucket.");
        Console.WriteLine("\t9. Cleans up all the resources created.");
    }

    /// <summary>
    /// Show information about the IAM Basics scenario.
    /// </summary>
    public void DisplayBasicsOverview()
    {
        Console.Clear();

        DisplayTitle("Welcome to IAM Basics");
        Console.WriteLine("This example application does the following:");
        Console.WriteLine("\t1. Creates a user with no permissions.");
        Console.WriteLine("\t2. Creates a role and policy that grant
s3:ListAllMyBuckets permission.");
        Console.WriteLine("\t3. Grants the user permission to assume the role.");
        Console.WriteLine("\t4. Creates an S3 client object as the user and tries
to list buckets (this will fail).");
        Console.WriteLine("\t5. Gets temporary credentials by assuming the
role.");
        Console.WriteLine("\t6. Creates a new S3 client object with the temporary
credentials and lists the buckets (this will succeed).");
        Console.WriteLine("\t7. Deletes all the resources.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPress <Enter> to continue. ");
        _ = Console.ReadLine();
```

```
        Console.WriteLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to be centered.</param>
    /// <returns>The padded string.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title, and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }

    /// <summary>
    /// Display a countdown and wait for a number of seconds.
    /// </summary>
    /// <param name="numSeconds">The number of seconds to wait.</param>
    public void WaitABit(int numSeconds, string msg)
    {
        Console.WriteLine(msg);

        // Wait for the requested number of seconds.
        for (int i = numSeconds; i > 0; i--)
        {
            System.Threading.Thread.Sleep(1000);
            Console.Write($"{i}...");
        }

        PressEnter();
    }
```

```
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for .NET.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iam_create_user_assume_role
#
# Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
#
#     "IAM access" permissions are needed to run this code.
```

```
#     "STS assume role" permissions are needed to run this code. (Note: It might
 be necessary to
#           create a custom policy).
#
# Returns:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function iam_create_user_assume_role() {
  {
    if [ "$IAM_OPERATIONS_SOURCED" != "True" ]; then

      source ./iam_operations.sh
    fi
  }

  echo_repeat "*" 88
  echo "Welcome to the IAM create user and assume role demo."
  echo
  echo "This demo will create an IAM user, create an IAM role, and apply the role
 to the user."
  echo_repeat "*" 88
  echo

  echo -n "Enter a name for a new IAM user: "
  get_input
  user_name=$get_input_result

  local user_arn
  user_arn=$(iam_create_user -u "$user_name")

  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created demo IAM user named $user_name"
  else
    errecho "$user_arn"
    errecho "The user failed to create. This demo will exit."
    return 1
  fi

  local access_key_response
  access_key_response=$(iam_create_user_access_key -u "$user_name")
  # shellcheck disable=SC2181
  if [[ ${?} != 0 ]]; then
```

```
    errecho "The access key failed to create. This demo will exit."
    clean_up "$user_name"
    return 1
  fi

  IFS=$'\t ' read -r -a access_key_values <<<"$access_key_response"
  local key_name=${access_key_values[0]}
  local key_secret=${access_key_values[1]}

  echo "Created access key named $key_name"

  echo "Wait 10 seconds for the user to be ready."
  sleep 10
  echo_repeat "*" 88
  echo

  local iam_role_name
  iam_role_name=$(generate_random_name "test-role")
  echo "Creating a role named $iam_role_name with user $user_name as the
principal."

  local assume_role_policy_document="{
    \"Version\": \"2012-10-17\",
    \"Statement\": [{
        \"Effect\": \"Allow\",
        \"Principal\": {\"AWS\": \"$user_arn\"},
        \"Action\": \"sts:AssumeRole\"
        }]
    }"

  local role_arn
  role_arn=$(iam_create_role -n "$iam_role_name" -p
"$assume_role_policy_document")

  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created IAM role named $iam_role_name"
  else
    errecho "The role failed to create. This demo will exit."
    clean_up "$user_name" "$key_name"
    return 1
  fi

  local policy_name
```

```
  policy_name=$(generate_random_name "test-policy")
  local policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]}"

  local policy_arn
  policy_arn=$(iam_create_policy -n "$policy_name" -p "$policy_document")
  # shellcheck disable=SC2181
  if [[ ${?} == 0 ]]; then
    echo "Created  IAM policy named $policy_name"
  else
    errecho "The policy failed to create."
    clean_up "$user_name" "$key_name" "$iam_role_name"
    return 1
  fi

  if (iam_attach_role_policy -n "$iam_role_name" -p "$policy_arn"); then
    echo "Attached policy $policy_arn to role $iam_role_name"
  else
    errecho "The policy failed to attach."
    clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
    return 1
  fi

  local assume_role_policy_document="{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"$role_arn\"}]}"

  local assume_role_policy_name
  assume_role_policy_name=$(generate_random_name "test-assume-role-")

  # shellcheck disable=SC2181
  local assume_role_policy_arn
  assume_role_policy_arn=$(iam_create_policy -n "$assume_role_policy_name" -p
"$assume_role_policy_document")
  # shellcheck disable=SC2181
  if [ ${?} == 0 ]; then
    echo "Created  IAM policy named $assume_role_policy_name for sts assume role"
```

```
 else
   errecho "The policy failed to create."
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn"
   return 1
 fi

 echo "Wait 10 seconds to give AWS time to propagate these new resources and
connections."
 sleep 10
 echo_repeat "*" 88
 echo

 echo "Try to list buckets without the new user assuming the role."
 echo_repeat "*" 88
 echo

 # Set the environment variables for the created user.
 # bashsupport disable=BP2001
 export AWS_ACCESS_KEY_ID=$key_name
 # bashsupport disable=BP2001
 export AWS_SECRET_ACCESS_KEY=$key_secret

 local buckets
 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. This should not have
happened."
 else
   errecho "Because the role with permissions has not been assumed, listing
buckets failed."
 fi

 echo
 echo_repeat "*" 88
 echo "Now assume the role $iam_role_name and list the buckets."
 echo_repeat "*" 88
 echo

 local credentials
```

```
 credentials=$(sts_assume_role -r "$role_arn" -n "AssumeRoleDemoSession")
 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   echo "Assumed role $iam_role_name"
 else
   errecho "Failed to assume role."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 IFS=$'\t ' read -r -a credentials <<<"$credentials"

 export AWS_ACCESS_KEY_ID=${credentials[0]}
 export AWS_SECRET_ACCESS_KEY=${credentials[1]}
 # bashsupport disable=BP2001
 export AWS_SESSION_TOKEN=${credentials[2]}

 buckets=$(s3_list_buckets)

 # shellcheck disable=SC2181
 if [ ${?} == 0 ]; then
   local bucket_count
   bucket_count=$(echo "$buckets" | wc -w | xargs)
   echo "There are $bucket_count buckets in the account. Listing buckets
succeeded because of "
   echo "the assumed role."
 else
   errecho "Failed to list buckets. This should not happen."
   export AWS_ACCESS_KEY_ID=""
   export AWS_SECRET_ACCESS_KEY=""
   export AWS_SESSION_TOKEN=""
   clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn"
"$policy_arn" "$assume_role_policy_arn"
   return 1
 fi

 local result=0
 export AWS_ACCESS_KEY_ID=""
 export AWS_SECRET_ACCESS_KEY=""
```

```
  echo
  echo_repeat "*" 88
  echo "The created resources will now be deleted."
  echo_repeat "*" 88
  echo

  clean_up "$user_name" "$key_name" "$iam_role_name" "$policy_arn" "$policy_arn"
 "$assume_role_policy_arn"

  # shellcheck disable=SC2181
  if [[ ${?} -ne 0 ]]; then
    result=1
  fi

  return $result
}
```

Las funciones de IAM que se usan en este escenario.

```
###############################################################################
# function iam_user_exists
#
# This function checks to see if the specified AWS Identity and Access Management
 (IAM) user already exists.
#
# Parameters:
#       $1 - The name of the IAM user to check.
#
# Returns:
#       0 - If the user already exists.
#       1 - If the user doesn't exist.
###############################################################################
function iam_user_exists() {
  local user_name
  user_name=$1

  # Check whether the IAM user already exists.
  # We suppress all output - we're interested only in the return code.

  local errors
  errors=$(aws iam get-user \
    --user-name "$user_name" 2>&1 >/dev/null)
```

```
  local error_code=${?}

  if [[ $error_code -eq 0 ]]; then
    return 0 # 0 in Bash script means true.
  else
    if [[ $errors != *"error"*"(NoSuchEntity)"* ]]; then
      aws_cli_error_log $error_code
      errecho "Error calling iam get-user $errors"
    fi

    return 1 # 1 in Bash script means false.
  fi
}

###############################################################################
# function iam_create_user
#
# This function creates the specified IAM user, unless
# it already exists.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       The ARN of the user.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user"
    echo "Creates an WS Identity and Access Management (IAM) user. You must
 supply a username:"
    echo "  -u user_name    The name of the user. It must be unique within the
 account."
    echo ""
  }
```

```
  # Retrieve the calling parameters.
  while getopts "u:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    User name:   $user_name"
  iecho ""

  # If the user already exists, we don't want to try to create it.
  if (iam_user_exists "$user_name"); then
    errecho "ERROR: A user with that name already exists in the account."
    return 1
  fi

  response=$(aws iam create-user --user-name "$user_name" \
    --output text \
    --query 'User.Arn')

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-user operation failed.$response"
    return 1
  fi
```

```
  echo "$response"

  return 0
}

###############################################################################
# function iam_create_user_access_key
#
# This function creates an IAM access key for the specified user.
#
# Parameters:
#       -u user_name -- The name of the IAM user.
#       [-f file_name] -- The optional file name for the access key output.
#
# Returns:
#       [access_key_id access_key_secret]
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_user_access_key() {
  local user_name file_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) key pair."
    echo "  -u user_name   The name of the IAM user."
    echo "  [-f file_name]   Optional file name for the access key output."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:f:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      f) file_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  response=$(aws iam create-access-key \
    --user-name "$user_name" \
    --output text)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-access-key operation failed.$response"
    return 1
  fi

  if [[ -n "$file_name" ]]; then
    echo "$response" >"$file_name"
  fi

  local key_id key_secret
  # shellcheck disable=SC2086
  key_id=$(echo $response | cut -f 2 -d ' ')
  # shellcheck disable=SC2086
  key_secret=$(echo $response | cut -f 4 -d ' ')

  echo "$key_id $key_secret"

  return 0
}

###############################################################################
# function iam_create_role
#
```

```
# This function creates an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_json -- The assume role policy document.
#
# Returns:
#       The ARN of the role.
#     And:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_role() {
  local role_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_user_access_key"
    echo "Creates an AWS Identity and Access Management (IAM) role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_json -- The assume role policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
```

```
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-role \
    --role-name "$role_name" \
    --assume-role-policy-document "$policy_document" \
    --output text \
    --query Role.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_create_policy
#
# This function creates an IAM policy.
#
# Parameters:
#       -n policy_name -- The name of the IAM policy.
#       -p policy_json -- The policy document.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_create_policy() {
```

```
  local policy_name policy_document response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_create_policy"
    echo "Creates an AWS Identity and Access Management (IAM) policy."
    echo "  -n policy_name   The name of the IAM policy."
    echo "  -p policy_json -- The policy document."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) policy_name="${OPTARG}" ;;
      p) policy_document="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_name" ]]; then
    errecho "ERROR: You must provide a policy name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_document" ]]; then
    errecho "ERROR: You must provide a policy document with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam create-policy \
    --policy-name "$policy_name" \
```

```
    --policy-document "$policy_document" \
    --output text \
    --query Policy.Arn)

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"
}

###############################################################################
# function iam_attach_role_policy
#
# This function attaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_attach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_attach_role_policy"
    echo "Attaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
```

```
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam attach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports attach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

```
########################################################################################
# function iam_detach_role_policy
#
# This function detaches an IAM policy to a tole.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#       -p policy_ARN -- The IAM policy document ARN..
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
########################################################################################
function iam_detach_role_policy() {
  local role_name policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_detach_role_policy"
    echo "Detaches an AWS Identity and Access Management (IAM) policy to an IAM
 role."
    echo "  -n role_name   The name of the IAM role."
    echo "  -p policy_ARN -- The IAM policy document ARN."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:p:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      p) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
```

```
  export OPTIND=1

  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy ARN with the -p parameter."
    usage
    return 1
  fi

  response=$(aws iam detach-role-policy \
    --role-name "$role_name" \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports detach-role-policy operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}

###############################################################################
# function iam_delete_policy
#
# This function deletes an IAM policy.
#
# Parameters:
#       -n policy_arn -- The name of the IAM policy arn.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_policy() {
```

```
  local policy_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_policy"
    echo "Deletes an WS Identity and Access Management (IAM) policy"
    echo "  -n policy_arn -- The name of the IAM policy arn."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) policy_arn="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$policy_arn" ]]; then
    errecho "ERROR: You must provide a policy arn with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Policy arn:  $policy_arn"
  iecho ""

  response=$(aws iam delete-policy \
    --policy-arn "$policy_arn")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
```

```
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-policy operation failed.\n$response"
    return 1
  fi

  iecho "delete-policy response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_role
#
# This function deletes an IAM role.
#
# Parameters:
#       -n role_name -- The name of the IAM role.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_role() {
  local role_name response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_role"
    echo "Deletes an WS Identity and Access Management (IAM) role"
    echo "  -n role_name -- The name of the IAM role."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "n:h" option; do
    case "${option}" in
      n) role_name="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
```

```
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  echo "role_name:$role_name"
  if [[ -z "$role_name" ]]; then
    errecho "ERROR: You must provide a role name with the -n parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Role name:  $role_name"
  iecho ""

  response=$(aws iam delete-role \
    --role-name "$role_name")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-role operation failed.\n$response"
    return 1
  fi

  iecho "delete-role response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_access_key
#
# This function deletes an IAM access key for the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user.
#       -k access_key -- The access key to delete.
```

```
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_access_key() {
  local user_name access_key response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function iam_delete_access_key"
    echo "Deletes an WS Identity and Access Management (IAM) access key for the
 specified IAM user"
    echo "  -u user_name    The name of the user."
    echo "  -k access_key   The access key to delete."
    echo ""
  }

  # Retrieve the calling parameters.
  while getopts "u:k:h" option; do
    case "${option}" in
      u) user_name="${OPTARG}" ;;
      k) access_key="${OPTARG}" ;;
      h)
        usage
        return 0
        ;;
      \?)
        echo "Invalid parameter"
        usage
        return 1
        ;;
    esac
  done
  export OPTIND=1

  if [[ -z "$user_name" ]]; then
    errecho "ERROR: You must provide a username with the -u parameter."
    usage
    return 1
  fi

  if [[ -z "$access_key" ]]; then
```

```
    errecho "ERROR: You must provide an access key with the -k parameter."
    usage
    return 1
  fi

  iecho "Parameters:\n"
  iecho "    Username:   $user_name"
  iecho "    Access key:   $access_key"
  iecho ""

  response=$(aws iam delete-access-key \
    --user-name "$user_name" \
    --access-key-id "$access_key")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-access-key operation failed.\n$response"
    return 1
  fi

  iecho "delete-access-key response:$response"
  iecho

  return 0
}

###############################################################################
# function iam_delete_user
#
# This function deletes the specified IAM user.
#
# Parameters:
#       -u user_name  -- The name of the user to create.
#
# Returns:
#       0 - If successful.
#       1 - If it fails.
###############################################################################
function iam_delete_user() {
  local user_name response
  local option OPTARG # Required to use getopts command in a function.
```

```
 # bashsupport disable=BP5008
 function usage() {
   echo "function iam_delete_user"
   echo "Deletes an WS Identity and Access Management (IAM) user. You must
supply a username:"
   echo "  -u user_name    The name of the user."
   echo ""
 }

 # Retrieve the calling parameters.
 while getopts "u:h" option; do
   case "${option}" in
     u) user_name="${OPTARG}" ;;
     h)
       usage
       return 0
       ;;
     \?)
       echo "Invalid parameter"
       usage
       return 1
       ;;
   esac
 done
 export OPTIND=1

 if [[ -z "$user_name" ]]; then
   errecho "ERROR: You must provide a username with the -u parameter."
   usage
   return 1
 fi

 iecho "Parameters:\n"
 iecho "    User name:   $user_name"
 iecho ""

 # If the user does not exist, we don't want to try to delete it.
 if (! iam_user_exists "$user_name"); then
   errecho "ERROR: A user with that name does not exist in the account."
   return 1
 fi

 response=$(aws iam delete-user \
   --user-name "$user_name")
```

```
  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports delete-user operation failed.$response"
    return 1
  fi

  iecho "delete-user response:$response"
  iecho

  return 0
}
```

- Para obtener información de la API, consulte los siguientes temas en la Referencia de comandos de AWS CLI.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace AwsDoc {
    namespace IAM {

        //! Cleanup by deleting created entities.
        /*!
          \sa DeleteCreatedEntities
          \param client: IAM client.
          \param role: IAM role.
          \param user: IAM user.
          \param policy: IAM policy.
        */
        static bool DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                          const Aws::IAM::Model::Role &role,
                                          const Aws::IAM::Model::User &user,
                                          const Aws::IAM::Model::Policy &policy);
    }

    static const int LIST_BUCKETS_WAIT_SEC = 20;

    static const char ALLOCATION_TAG[] = "example_code";
}

//! Scenario to create an IAM user, create an IAM role, and apply the role to the
 user.
// "IAM access" permissions are needed to run this code.
// "STS assume role" permissions are needed to run this code. (Note: It might be
 necessary to
//    create a custom policy).
/*!
  \sa iamCreateUserAssumeRoleScenario
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
```

```
*/
bool AwsDoc::IAM::iamCreateUserAssumeRoleScenario(
        const Aws::Client::ClientConfiguration &clientConfig) {

    Aws::IAM::IAMClient client(clientConfig);
    Aws::IAM::Model::User user;
    Aws::IAM::Model::Role role;
    Aws::IAM::Model::Policy policy;

    // 1. Create a user.
    {
        Aws::IAM::Model::CreateUserRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String userName = "iam-demo-user-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetUserName(userName);

        Aws::IAM::Model::CreateUserOutcome outcome = client.CreateUser(request);
        if (!outcome.IsSuccess()) {
            std::cout << "Error creating IAM user " << userName << ":" <<
                      outcome.GetError().GetMessage() << std::endl;
            return false;
        }
        else {
            std::cout << "Successfully created IAM user " << userName <<
 std::endl;
        }

        user = outcome.GetResult().GetUser();
    }

    // 2. Create a role.
    {
        // Get the IAM user for the current client in order to access its ARN.
        Aws::String iamUserArn;
        {
            Aws::IAM::Model::GetUserRequest request;
            Aws::IAM::Model::GetUserOutcome outcome = client.GetUser(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error getting Iam user. " <<
                          outcome.GetError().GetMessage() << std::endl;

                DeleteCreatedEntities(client, role, user, policy);
                return false;
```

```
            }
            else {
                std::cout << "Successfully retrieved Iam user "
                          << outcome.GetResult().GetUser().GetUserName()
                          << std::endl;
            }

            iamUserArn = outcome.GetResult().GetUser().GetArn();
        }

        Aws::IAM::Model::CreateRoleRequest request;

        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleName = "iam-demo-role-" +
                               Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleName(roleName);

        // Build policy document for role.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");

        Aws::Utils::Document jsonPrincipal;
        jsonPrincipal.WithString("AWS", iamUserArn);
        jsonStatement.WithObject("Principal", jsonPrincipal);
        jsonStatement.WithString("Action", "sts:AssumeRole");
        jsonStatement.WithObject("Condition", Aws::Utils::Document());

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Setting policy for role\n   "
                  << policyDocument.View().WriteCompact() << std::endl;

        // Set role policy document as JSON string.

    request.SetAssumeRolePolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreateRoleOutcome outcome = client.CreateRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating role. " <<
```

```
                    outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a role with name " << roleName
                      << std::endl;
        }

        role = outcome.GetResult().GetRole();
    }

    // 3. Create an IAM policy.
    {
        Aws::IAM::Model::CreatePolicyRequest request;
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String policyName = "iam-demo-policy-" +
                                 Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetPolicyName(policyName);

        // Build IAM policy document.
        Aws::Utils::Document jsonStatement;
        jsonStatement.WithString("Effect", "Allow");
        jsonStatement.WithString("Action", "s3:ListAllMyBuckets");
        jsonStatement.WithString("Resource", "arn:aws:s3:::*");

        Aws::Utils::Document policyDocument;
        policyDocument.WithString("Version", "2012-10-17");

        Aws::Utils::Array<Aws::Utils::Document> statements(1);
        statements[0] = jsonStatement;
        policyDocument.WithArray("Statement", statements);

        std::cout << "Creating a policy.\n   " <<
policyDocument.View().WriteCompact()
                  << std::endl;

        // Set IAM policy document as JSON string.
        request.SetPolicyDocument(policyDocument.View().WriteCompact());

        Aws::IAM::Model::CreatePolicyOutcome outcome =
client.CreatePolicy(request);
        if (!outcome.IsSuccess()) {
```

```
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully created a policy with name, " <<
policyName <<
                      "." << std::endl;
        }

        policy = outcome.GetResult().GetPolicy();
    }

    // 4. Assume the new role using the AWS Security Token Service (STS).
    Aws::STS::Model::Credentials credentials;
    {
        Aws::STS::STSClient stsClient(clientConfig);

        Aws::STS::Model::AssumeRoleRequest request;
        request.SetRoleArn(role.GetArn());
        Aws::String uuid = Aws::Utils::UUID::RandomUUID();
        Aws::String roleSessionName = "iam-demo-role-session-" +

Aws::Utils::StringUtils::ToLower(uuid.c_str());
        request.SetRoleSessionName(roleSessionName);

        Aws::STS::Model::AssumeRoleOutcome assumeRoleOutcome;

        // Repeatedly call AssumeRole, because there is often a delay
        // before the role is available to be assumed.
        // Repeat at most 20 times when access is denied.
        int count = 0;
        while (true) {
            assumeRoleOutcome = stsClient.AssumeRole(request);
            if (!assumeRoleOutcome.IsSuccess()) {
                if (count > 20 ||
                    assumeRoleOutcome.GetError().GetErrorType() !=
                    Aws::STS::STSErrors::ACCESS_DENIED) {
                    std::cerr << "Error assuming role after 20 tries. " <<
                              assumeRoleOutcome.GetError().GetMessage() <<
std::endl;
```

```
                    DeleteCreatedEntities(client, role, user, policy);
                    return false;
                }
                std::this_thread::sleep_for(std::chrono::seconds(1));
            }
            else {
                std::cout << "Successfully assumed the role after " << count
                          << " seconds." << std::endl;
                break;
            }
            count++;
        }

        credentials = assumeRoleOutcome.GetResult().GetCredentials();
    }


    // 5. List objects in the bucket (This should fail).
    {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if (listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
std::endl;
            }
            else {
                std::cout
                        << "Access to list buckets denied because privileges have
not been applied."
                        << std::endl;
            }
        }
        else {
            std::cerr
```

```
                    << "Successfully retrieved bucket lists when this should not
happen."
                    << std::endl;
        }
    }

    // 6. Attach the policy to the role.
    {
        Aws::IAM::Model::AttachRolePolicyRequest request;
        request.SetRoleName(role.GetRoleName());
        request.WithPolicyArn(policy.GetArn());

        Aws::IAM::Model::AttachRolePolicyOutcome outcome =
client.AttachRolePolicy(
                request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error creating policy. " <<
                      outcome.GetError().GetMessage() << std::endl;

            DeleteCreatedEntities(client, role, user, policy);
            return false;
        }
        else {
            std::cout << "Successfully attached the policy with name, "
                      << policy.GetPolicyName() <<
                      ", to the role, " << role.GetRoleName() << "." <<
std::endl;
        }
    }

    int count = 0;
    // 7. List objects in the bucket (this should succeed).
    // Repeatedly call ListBuckets, because there is often a delay
    // before the policy with ListBucket permissions has been applied to the
role.
    // Repeat at most LIST_BUCKETS_WAIT_SEC times when access is denied.
    while (true) {
        Aws::S3::S3Client s3Client(
                Aws::Auth::AWSCredentials(credentials.GetAccessKeyId(),
                                          credentials.GetSecretAccessKey(),
                                          credentials.GetSessionToken()),
                Aws::MakeShared<Aws::S3::S3EndpointProvider>(ALLOCATION_TAG),
                clientConfig);
```

```
        Aws::S3::Model::ListBucketsOutcome listBucketsOutcome =
 s3Client.ListBuckets();
        if (!listBucketsOutcome.IsSuccess()) {
            if ((count > LIST_BUCKETS_WAIT_SEC) ||
                listBucketsOutcome.GetError().GetErrorType() !=
                Aws::S3::S3Errors::ACCESS_DENIED) {
                std::cerr << "Could not lists buckets after " <<
 LIST_BUCKETS_WAIT_SEC << " seconds. " <<
                          listBucketsOutcome.GetError().GetMessage() <<
 std::endl;
                DeleteCreatedEntities(client, role, user, policy);
                return false;
            }

            std::this_thread::sleep_for(std::chrono::seconds(1));
        }
        else {

            std::cout << "Successfully retrieved bucket lists after " << count
                      << " seconds." << std::endl;
            break;
        }
        count++;
    }

    // 8. Delete all the created resources.
    return DeleteCreatedEntities(client, role, user, policy);
}

bool AwsDoc::IAM::DeleteCreatedEntities(const Aws::IAM::IAMClient &client,
                                        const Aws::IAM::Model::Role &role,
                                        const Aws::IAM::Model::User &user,
                                        const Aws::IAM::Model::Policy &policy) {
    bool result = true;
    if (policy.ArnHasBeenSet()) {
        // Detach the policy from the role.
        {
            Aws::IAM::Model::DetachRolePolicyRequest request;
            request.SetPolicyArn(policy.GetArn());
            request.SetRoleName(role.GetRoleName());

            Aws::IAM::Model::DetachRolePolicyOutcome outcome =
 client.DetachRolePolicy(
                    request);
```

```
            if (!outcome.IsSuccess()) {
                std::cerr << "Error Detaching policy from roles. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully detached the policy with arn "
                          << policy.GetArn()
                          << " from role " << role.GetRoleName() << "." <<
std::endl;
            }
        }

        // Delete the policy.
        {
            Aws::IAM::Model::DeletePolicyRequest request;
            request.WithPolicyArn(policy.GetArn());

            Aws::IAM::Model::DeletePolicyOutcome outcome =
client.DeletePolicy(request);
            if (!outcome.IsSuccess()) {
                std::cerr << "Error deleting policy. " <<
                          outcome.GetError().GetMessage() << std::endl;
                result = false;
            }
            else {
                std::cout << "Successfully deleted the policy with arn "
                          << policy.GetArn() << std::endl;
            }
        }

    }

    if (role.RoleIdHasBeenSet()) {
        // Delete the role.
        Aws::IAM::Model::DeleteRoleRequest request;
        request.SetRoleName(role.GetRoleName());

        Aws::IAM::Model::DeleteRoleOutcome outcome = client.DeleteRole(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting role. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

```
        else {
            std::cout << "Successfully deleted the role with name "
                      << role.GetRoleName() << std::endl;
        }
    }

    if (user.ArnHasBeenSet()) {
        // Delete the user.
        Aws::IAM::Model::DeleteUserRequest request;
        request.WithUserName(user.GetUserName());

        Aws::IAM::Model::DeleteUserOutcome outcome = client.DeleteUser(request);
        if (!outcome.IsSuccess()) {
            std::cerr << "Error deleting user. " <<
                      outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
        else {
            std::cout << "Successfully deleted the user with name "
                      << user.GetUserName() << std::endl;
        }
    }

    return result;
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for C++.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy

- DetachRolePolicy
- PutUserPolicy

Go

SDK para Go V2

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Ejecute un escenario interactivo en un símbolo del sistema.

```
// AssumeRoleScenario shows you how to use the AWS Identity and Access Management
 (IAM)
// service to perform the following actions:
//
// 1. Create a user who has no permissions.
// 2. Create a role that grants permission to list Amazon Simple Storage Service
//    (Amazon S3) buckets for the account.
// 3. Add a policy to let the user assume the role.
// 4. Try and fail to list buckets without permissions.
// 5. Assume the role and list S3 buckets using temporary credentials.
// 6. Delete the policy, role, and user.
type AssumeRoleScenario struct {
 sdkConfig aws.Config
 accountWrapper actions.AccountWrapper
 policyWrapper actions.PolicyWrapper
 roleWrapper actions.RoleWrapper
 userWrapper actions.UserWrapper
 questioner demotools.IQuestioner
 helper IScenarioHelper
 isTestRun bool
}

// NewAssumeRoleScenario constructs an AssumeRoleScenario instance from a
 configuration.
```

```
// It uses the specified config to get an IAM client and create wrappers for the
 actions
// used in the scenario.
func NewAssumeRoleScenario(sdkConfig aws.Config, questioner
 demotools.IQuestioner,
  helper IScenarioHelper) AssumeRoleScenario {
 iamClient := iam.NewFromConfig(sdkConfig)
 return AssumeRoleScenario{
  sdkConfig:   sdkConfig,
  accountWrapper: actions.AccountWrapper{IamClient: iamClient},
  policyWrapper:  actions.PolicyWrapper{IamClient: iamClient},
  roleWrapper:    actions.RoleWrapper{IamClient: iamClient},
  userWrapper:    actions.UserWrapper{IamClient: iamClient},
  questioner:     questioner,
  helper:         helper,
 }
}

// addTestOptions appends the API options specified in the original configuration
 to
// another configuration. This is used to attach the middleware stubber to
 clients
// that are constructed during the scenario, which is needed for unit testing.
func (scenario AssumeRoleScenario) addTestOptions(scenarioConfig *aws.Config) {
 if scenario.isTestRun {
  scenarioConfig.APIOptions = append(scenarioConfig.APIOptions,
 scenario.sdkConfig.APIOptions...)
 }
}

// Run runs the interactive scenario.
func (scenario AssumeRoleScenario) Run() {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong with the demo.\n")
   log.Println(r)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Println("Welcome to the AWS Identity and Access Management (IAM) assume role
 demo.")
 log.Println(strings.Repeat("-", 88))
```

```
 user := scenario.CreateUser()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(user, role)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}

// CreateUser creates a new IAM user. This user has no permissions.
func (scenario AssumeRoleScenario) CreateUser() *types.User {
 log.Println("Let's create an example user with no permissions.")
 userName := scenario.questioner.Ask("Enter a name for the example user:",
 demotools.NotEmpty{})
 user, err := scenario.userWrapper.GetUser(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(userName)
  if err != nil {
   panic(err)
  }
  log.Printf("Created user %v.\n", *user.UserName)
 } else {
  log.Printf("User %v already exists.\n", *user.UserName)
 }
 log.Println(strings.Repeat("-", 88))
 return user
}

// CreateAccessKey creates an access key for the user.
func (scenario AssumeRoleScenario) CreateAccessKey(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*user.UserName)
 if err != nil {
  panic(err)
 }
 log.Printf("Created access key %v for your user.", *accessKey.AccessKeyId)
 log.Println("Waiting a few seconds for your user to be ready...")
 scenario.helper.Pause(10)
```

```
 log.Println(strings.Repeat("-", 88))
 return accessKey
}

// CreateRoleAndPolicies creates a policy that grants permission to list S3
 buckets for
// the current account and attaches the policy to a newly created role. It also
 adds an
// inline policy to the specified user that grants the user permission to assume
 the role.
func (scenario AssumeRoleScenario) CreateRoleAndPolicies(user *types.User)
 *types.Role {
 log.Println("Let's create a role and policy that grant permission to list S3
 buckets.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 listBucketsRole, err :=
 scenario.roleWrapper.CreateRole(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  scenario.helper.GetName(), []string{"s3:ListAllMyBuckets"}, "arn:aws:s3:::*")
 if err != nil {panic(err)}
 log.Printf("Created policy %v.\n", *listBucketsPolicy.PolicyName)
 err = scenario.roleWrapper.AttachRolePolicy(*listBucketsPolicy.Arn,
 *listBucketsRole.RoleName)
 if err != nil {panic(err)}
 log.Printf("Attached policy %v to role %v.\n", *listBucketsPolicy.PolicyName,
  *listBucketsRole.RoleName)
 err = scenario.userWrapper.CreateUserPolicy(*user.UserName,
 scenario.helper.GetName(),
  []string{"sts:AssumeRole"}, *listBucketsRole.Arn)
 if err != nil {panic(err)}
 log.Printf("Created an inline policy for user %v that lets the user assume the
 role.\n",
  *user.UserName)
 log.Println("Let's give AWS a few seconds to propagate these new resources and
 connections...")
 scenario.helper.Pause(10)
 log.Println(strings.Repeat("-", 88))
 return listBucketsRole
}

// ListBucketsWithoutPermissions creates an Amazon S3 client from the user's
 access key
```

```
// credentials and tries to list buckets for the account. Because the user does
 not have
// permission to perform this action, the action fails.
func (scenario AssumeRoleScenario) ListBucketsWithoutPermissions(accessKey
 *types.AccessKey) *aws.Config {
  log.Println("Let's try to list buckets without permissions. This should return
 an AccessDenied error.")
  scenario.questioner.Ask("Press Enter when you're ready.")
  noPermsConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *accessKey.AccessKeyId, *accessKey.SecretAccessKey, ""),
 ))
  if err != nil {panic(err)}

  // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&noPermsConfig)

  s3Client := s3.NewFromConfig(noPermsConfig)
  _, err = s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
  if err != nil {
   // The SDK for Go does not model the AccessDenied error, so check ErrorCode
 directly.
  var ae smithy.APIError
  if errors.As(err, &ae) {
   switch ae.ErrorCode() {
   case "AccessDenied":
    log.Println("Got AccessDenied error, which is the expected result because\n"
 +
     "the ListBuckets call was made without permissions.")
   default:
    log.Println("Expected AccessDenied, got something else.")
    panic(err)
   }
  }
  } else {
   log.Println("Expected AccessDenied error when calling ListBuckets without
 permissions,\n" +
    "but the call succeeded. Continuing the example anyway...")
 }
 log.Println(strings.Repeat("-", 88))
  return &noPermsConfig
}
```

```
// ListBucketsWithAssumedRole performs the following actions:
//
// 1. Creates an AWS Security Token Service (AWS STS) client from the config
 created from
//   the user's access key credentials.
// 2. Gets temporary credentials by assuming the role that grants permission to
 list the
//    buckets.
// 3. Creates an Amazon S3 client from the temporary credentials.
// 4. Lists buckets for the account. Because the temporary credentials are
 generated by
//    assuming the role that grants permission, the action succeeds.
func (scenario AssumeRoleScenario) ListBucketsWithAssumedRole(noPermsConfig
 *aws.Config, role *types.Role) {
 log.Println("Let's assume the role that grants permission to list buckets and
 try again.")
 scenario.questioner.Ask("Press Enter when you're ready.")
 stsClient := sts.NewFromConfig(*noPermsConfig)
 tempCredentials, err := stsClient.AssumeRole(context.TODO(),
 &sts.AssumeRoleInput{
  RoleArn:           role.Arn,
  RoleSessionName:   aws.String("AssumeRoleExampleSession"),
  DurationSeconds:   aws.Int32(900),
 })
 if err != nil {
  log.Printf("Couldn't assume role %v.\n", *role.RoleName)
  panic(err)
 }
 log.Printf("Assumed role %v, got temporary credentials.\n", *role.RoleName)
 assumeRoleConfig, err := config.LoadDefaultConfig(context.TODO(),
  config.WithCredentialsProvider(credentials.NewStaticCredentialsProvider(
   *tempCredentials.Credentials.AccessKeyId,
   *tempCredentials.Credentials.SecretAccessKey,
   *tempCredentials.Credentials.SessionToken),
  ),
 )
 if err != nil {panic(err)}

 // Add test options if this is a test run. This is needed only for testing
 purposes.
 scenario.addTestOptions(&assumeRoleConfig)

 s3Client := s3.NewFromConfig(assumeRoleConfig)
 result, err := s3Client.ListBuckets(context.TODO(), &s3.ListBucketsInput{})
```

```
 if err != nil {
  log.Println("Couldn't list buckets with assumed role credentials.")
  panic(err)
 }
 log.Println("Successfully called ListBuckets with assumed role credentials, \n"
 +
  "here are some of them:")
 for i := 0; i < len(result.Buckets) && i < 5; i++ {
  log.Printf("\t%v\n", *result.Buckets[i].Name)
 }
 log.Println(strings.Repeat("-", 88))
}

// Cleanup deletes all resources created for the scenario.
func (scenario AssumeRoleScenario) Cleanup(user *types.User, role *types.Role) {
 if scenario.questioner.AskBool(
  "Do you want to delete the resources created for this example? (y/n)", "y",
 ) {
   policies, err := scenario.roleWrapper.ListAttachedRolePolicies(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*policy.PolicyArn)
    if err != nil {panic(err)}
    log.Printf("Detached policy %v from role %v and deleted the policy.\n",
     *policy.PolicyName, *role.RoleName)
   }
   err = scenario.roleWrapper.DeleteRole(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)

   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*user.UserName, userPol)
    if err != nil {panic(err)}
    log.Printf("Deleted policy %v from user %v.\n", userPol, *user.UserName)
   }
   keys, err := scenario.userWrapper.ListAccessKeys(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*user.UserName, *key.AccessKeyId)
```

```
    if err != nil {panic(err)}
    log.Printf("Deleted access key %v from user %v.\n", *key.AccessKeyId,
 *user.UserName)
   }
   err = scenario.userWrapper.DeleteUser(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Defina una estructura que incluya las acciones de la cuenta.

```
// AccountWrapper encapsulates AWS Identity and Access Management (IAM) account
 actions
// used in the examples.
// It contains an IAM service client that is used to perform account actions.
type AccountWrapper struct {
 IamClient *iam.Client
}



// GetAccountPasswordPolicy gets the account password policy for the current
 account.
// If no policy has been set, a NoSuchEntityException is error is returned.
func (wrapper AccountWrapper) GetAccountPasswordPolicy() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &iam.GetAccountPasswordPolicyInput{})
 if err != nil {
  log.Printf("Couldn't get account password policy. Here's why: %v\n", err)
 } else {
  pwPolicy = result.PasswordPolicy
 }
 return pwPolicy, err
}
```

```
// ListSAMLProviders gets the SAML providers for the account.
func (wrapper AccountWrapper) ListSAMLProviders() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &iam.ListSAMLProvidersInput{})
 if err != nil {
  log.Printf("Couldn't list SAML providers. Here's why: %v\n", err)
 } else {
  providers = result.SAMLProviderList
 }
 return providers, err
}
```

Defina una estructura que incluya las acciones de la política.

```
// PolicyDocument defines a policy document as a Go struct that can be serialized
// to JSON.
type PolicyDocument struct {
 Version string
 Statement []PolicyStatement
}

// PolicyStatement defines a statement in a policy document.
type PolicyStatement struct {
 Effect string
 Action []string
 Principal map[string]string `json:",omitempty"`
 Resource *string `json:",omitempty"`
}



// PolicyWrapper encapsulates AWS Identity and Access Management (IAM) policy
 actions
// used in the examples.
// It contains an IAM service client that is used to perform policy actions.
type PolicyWrapper struct {
```

```
 IamClient *iam.Client
}



// ListPolicies gets up to maxPolicies policies.
func (wrapper PolicyWrapper) ListPolicies(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &iam.ListPoliciesInput{
  MaxItems: aws.Int32(maxPolicies),
 })
 if err != nil {
  log.Printf("Couldn't list policies. Here's why: %v\n", err)
 } else {
  policies = result.Policies
 }
 return policies, err
}



// CreatePolicy creates a policy that grants a list of actions to the specified
 resource.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper PolicyWrapper) CreatePolicy(policyName string, actions []string,
  resourceArn string) (*types.Policy, error) {
 var policy *types.Policy
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(resourceArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n",
 resourceArn, err)
  return nil, err
```

```
 }
 result, err := wrapper.IamClient.CreatePolicy(context.TODO(),
 &iam.CreatePolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
 })
 if err != nil {
  log.Printf("Couldn't create policy %v. Here's why: %v\n", policyName, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// GetPolicy gets data about a policy.
func (wrapper PolicyWrapper) GetPolicy(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &iam.GetPolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't get policy %v. Here's why: %v\n", policyArn, err)
 } else {
  policy = result.Policy
 }
 return policy, err
}



// DeletePolicy deletes a policy.
func (wrapper PolicyWrapper) DeletePolicy(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &iam.DeletePolicyInput{
  PolicyArn: aws.String(policyArn),
 })
 if err != nil {
  log.Printf("Couldn't delete policy %v. Here's why: %v\n", policyArn, err)
 }
 return err
}
```

Defina una estructura que incluya las acciones de rol.

```
// RoleWrapper encapsulates AWS Identity and Access Management (IAM) role actions
// used in the examples.
// It contains an IAM service client that is used to perform role actions.
type RoleWrapper struct {
 IamClient *iam.Client
}



// ListRoles gets up to maxRoles roles.
func (wrapper RoleWrapper) ListRoles(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &iam.ListRolesInput{MaxItems: aws.Int32(maxRoles)},
 )
 if err != nil {
  log.Printf("Couldn't list roles. Here's why: %v\n", err)
 } else {
  roles = result.Roles
 }
 return roles, err
}



// CreateRole creates a role that trusts a specified user. The trusted user can
 assume
// the role to acquire its permissions.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper RoleWrapper) CreateRole(roleName string, trustedUserArn string)
 (*types.Role, error) {
 var role *types.Role
 trustPolicy := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
```

```
    Principal: map[string]string{"AWS": trustedUserArn},
    Action: []string{"sts:AssumeRole"},
   }},
  }
  policyBytes, err := json.Marshal(trustPolicy)
  if err != nil {
   log.Printf("Couldn't create trust policy for %v. Here's why: %v\n",
  trustedUserArn, err)
   return nil, err
  }
  result, err := wrapper.IamClient.CreateRole(context.TODO(),
  &iam.CreateRoleInput{
   AssumeRolePolicyDocument: aws.String(string(policyBytes)),
   RoleName:                 aws.String(roleName),
  })
  if err != nil {
   log.Printf("Couldn't create role %v. Here's why: %v\n", roleName, err)
  } else {
   role = result.Role
  }
  return role, err
}



// GetRole gets data about a role.
func (wrapper RoleWrapper) GetRole(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &iam.GetRoleInput{RoleName: aws.String(roleName)})
 if err != nil {
  log.Printf("Couldn't get role %v. Here's why: %v\n", roleName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// CreateServiceLinkedRole creates a service-linked role that is owned by the
 specified service.
func (wrapper RoleWrapper) CreateServiceLinkedRole(serviceName string,
 description string) (*types.Role, error) {
```

```
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &iam.CreateServiceLinkedRoleInput{
  AWSServiceName: aws.String(serviceName),
  Description:    aws.String(description),
 })
 if err != nil {
  log.Printf("Couldn't create service-linked role %v. Here's why: %v\n",
 serviceName, err)
 } else {
  role = result.Role
 }
 return role, err
}



// DeleteServiceLinkedRole deletes a service-linked role.
func (wrapper RoleWrapper) DeleteServiceLinkedRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &iam.DeleteServiceLinkedRoleInput{
  RoleName: aws.String(roleName)},
 )
 if err != nil {
  log.Printf("Couldn't delete service-linked role %v. Here's why: %v\n",
 roleName, err)
 }
 return err
}



// AttachRolePolicy attaches a policy to a role.
func (wrapper RoleWrapper) AttachRolePolicy(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &iam.AttachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't attach policy %v to role %v. Here's why: %v\n", policyArn,
 roleName, err)
 }
```

```
 return err
}



// ListAttachedRolePolicies lists the policies that are attached to the specified
 role.
func (wrapper RoleWrapper) ListAttachedRolePolicies(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &iam.ListAttachedRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list attached policies for role %v. Here's why: %v\n",
 roleName, err)
 } else {
  policies = result.AttachedPolicies
 }
 return policies, err
}



// DetachRolePolicy detaches a policy from a role.
func (wrapper RoleWrapper) DetachRolePolicy(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &iam.DetachRolePolicyInput{
  PolicyArn: aws.String(policyArn),
  RoleName:  aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't detach policy from role %v. Here's why: %v\n", roleName,
 err)
 }
 return err
}



// ListRolePolicies lists the inline policies for a role.
func (wrapper RoleWrapper) ListRolePolicies(roleName string) ([]string, error) {
```

```
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &iam.ListRolePoliciesInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for role %v. Here's why: %v\n", roleName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteRole deletes a role. All attached policies must be detached before a
// role can be deleted.
func (wrapper RoleWrapper) DeleteRole(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &iam.DeleteRoleInput{
  RoleName: aws.String(roleName),
 })
 if err != nil {
  log.Printf("Couldn't delete role %v. Here's why: %v\n", roleName, err)
 }
 return err
}
```

Defina una estructura que incluya las acciones del usuario.

```


// UserWrapper encapsulates user actions used in the examples.
// It contains an IAM service client that is used to perform user actions.
type UserWrapper struct {
 IamClient *iam.Client
}



// ListUsers gets up to maxUsers number of users.
```

```
func (wrapper UserWrapper) ListUsers(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &iam.ListUsersInput{
  MaxItems: aws.Int32(maxUsers),
 })
 if err != nil {
  log.Printf("Couldn't list users. Here's why: %v\n", err)
 } else {
  users = result.Users
 }
 return users, err
}



// GetUser gets data about a user.
func (wrapper UserWrapper) GetUser(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &iam.GetUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  var apiError smithy.APIError
  if errors.As(err, &apiError) {
   switch apiError.(type) {
   case *types.NoSuchEntityException:
    log.Printf("User %v does not exist.\n", userName)
    err = nil
   default:
    log.Printf("Couldn't get user %v. Here's why: %v\n", userName, err)
   }
  }
 } else {
  user = result.User
 }
 return user, err
}



// CreateUser creates a new user with the specified name.
func (wrapper UserWrapper) CreateUser(userName string) (*types.User, error) {
 var user *types.User
```

```
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &iam.CreateUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
 } else {
  user = result.User
 }
 return user, err
}



// CreateUserPolicy adds an inline policy to a user. This example creates a
 policy that
// grants a list of actions on a specified role.
// PolicyDocument shows how to work with a policy document as a data structure
 and
// serialize it to JSON by using Go's JSON marshaler.
func (wrapper UserWrapper) CreateUserPolicy(userName string, policyName string,
 actions []string,
  roleArn string) error {
 policyDoc := PolicyDocument{
  Version:   "2012-10-17",
  Statement: []PolicyStatement{{
   Effect: "Allow",
   Action: actions,
   Resource: aws.String(roleArn),
  }},
 }
 policyBytes, err := json.Marshal(policyDoc)
 if err != nil {
  log.Printf("Couldn't create policy document for %v. Here's why: %v\n", roleArn,
 err)
  return err
 }
 _, err = wrapper.IamClient.PutUserPolicy(context.TODO(),
 &iam.PutUserPolicyInput{
  PolicyDocument: aws.String(string(policyBytes)),
  PolicyName:     aws.String(policyName),
  UserName:       aws.String(userName),
 })
 if err != nil {
```

```
  log.Printf("Couldn't create policy for user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// ListUserPolicies lists the inline policies for the specified user.
func (wrapper UserWrapper) ListUserPolicies(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &iam.ListUserPoliciesInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list policies for user %v. Here's why: %v\n", userName,
 err)
 } else {
  policies = result.PolicyNames
 }
 return policies, err
}



// DeleteUserPolicy deletes an inline policy from a user.
func (wrapper UserWrapper) DeleteUserPolicy(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &iam.DeleteUserPolicyInput{
  PolicyName: aws.String(policyName),
  UserName:   aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete policy from user %v. Here's why: %v\n", userName,
 err)
 }
 return err
}



// DeleteUser deletes a user.
```

```
func (wrapper UserWrapper) DeleteUser(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &iam.DeleteUserInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete user %v. Here's why: %v\n", userName, err)
 }
 return err
}



// CreateAccessKeyPair creates an access key for a user. The returned access key
 contains
// the ID and secret credentials needed to use the key.
func (wrapper UserWrapper) CreateAccessKeyPair(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &iam.CreateAccessKeyInput{
  UserName: aws.String(userName)})
 if err != nil {
  log.Printf("Couldn't create access key pair for user %v. Here's why: %v\n",
 userName, err)
 } else {
  key = result.AccessKey
 }
 return key, err
}



// DeleteAccessKey deletes an access key from a user.
func (wrapper UserWrapper) DeleteAccessKey(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &iam.DeleteAccessKeyInput{
  AccessKeyId: aws.String(keyId),
  UserName:    aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't delete access key %v. Here's why: %v\n", keyId, err)
 }
 return err
}
```

```
// ListAccessKeys lists the access keys for the specified user.
func (wrapper UserWrapper) ListAccessKeys(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &iam.ListAccessKeysInput{
  UserName: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't list access keys for user %v. Here's why: %v\n", userName,
 err)
 } else {
  keys = result.AccessKeyMetadata
 }
 return keys, err
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Go.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del usuario de IAM.

```
/*
  To run this Java V2 code example, set up your development environment,
 including your credentials.

  For information, see this documentation topic:

  https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html

  This example performs these operations:

  1. Creates a user that has no permissions.
  2. Creates a role and policy that grants Amazon S3 permissions.
  3. Creates a role.
  4. Grants the user permissions.
  5. Gets temporary credentials by assuming the role.  Creates an Amazon S3
 Service client object with the temporary credentials.
  6. Deletes the resources.
 */

public class IAMScenario {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");
    public static final String PolicyDocument = "{" +
            "  \"Version\": \"2012-10-17\"," +
            "  \"Statement\": [" +
            "    {" +
            "        \"Effect\": \"Allow\"," +
            "        \"Action\": [" +
            "            \"s3:*\"" +
            "       ]," +
```

```
            "       \"Resource\": \"*\"" +
            "    }" +
            "   ]" +
            "}";

   public static String userArn;

   public static void main(String[] args) throws Exception {

       final String usage = """

               Usage:
                   <username> <policyName> <roleName> <roleSessionName>
<bucketName>\s

               Where:
                   username - The name of the IAM user to create.\s
                   policyName - The name of the policy to create.\s
                   roleName - The name of the role to create.\s
                   roleSessionName - The name of the session required for the
assumeRole operation.\s
                   bucketName - The name of the Amazon S3 bucket from which
objects are read.\s
               """;

       if (args.length != 5) {
           System.out.println(usage);
           System.exit(1);
       }

       String userName = args[0];
       String policyName = args[1];
       String roleName = args[2];
       String roleSessionName = args[3];
       String bucketName = args[4];

       Region region = Region.AWS_GLOBAL;
       IamClient iam = IamClient.builder()
               .region(region)
               .build();

       System.out.println(DASHES);
       System.out.println("Welcome to the AWS IAM example scenario.");
       System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println(" 1. Create the IAM user.");
        User createUser = createIAMUser(iam, userName);

        System.out.println(DASHES);
        userArn = createUser.arn();

        AccessKey myKey = createIAMAccessKey(iam, userName);
        String accessKey = myKey.accessKeyId();
        String secretKey = myKey.secretAccessKey();
        String assumeRolePolicyDocument = "{" +
                "\"Version\": \"2012-10-17\"," +
                "\"Statement\": [{" +
                "\"Effect\": \"Allow\"," +
                "\"Principal\": {" +
                " \"AWS\": \"" + userArn + "\"" +
                "}," +
                "\"Action\": \"sts:AssumeRole\"" +
                "}]" +
                "}";

        System.out.println(assumeRolePolicyDocument);
        System.out.println(userName + " was successfully created.");
        System.out.println(DASHES);
        System.out.println("2. Creates a policy.");
        String polArn = createIAMPolicy(iam, policyName);
        System.out.println("The policy " + polArn + " was successfully
created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Creates a role.");
        TimeUnit.SECONDS.sleep(30);
        String roleArn = createIAMRole(iam, roleName, assumeRolePolicyDocument);
        System.out.println(roleArn + " was successfully created.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Grants the user permissions.");
        attachIAMRolePolicy(iam, roleName, polArn);
        System.out.println(DASHES);

        System.out.println(DASHES);
```

```
        System.out.println("*** Wait for 30 secs so the resource is available");
        TimeUnit.SECONDS.sleep(30);
        System.out.println("5. Gets temporary credentials by assuming the
role.");
        System.out.println("Perform an Amazon S3 Service operation using the
temporary credentials.");
        assumeRole(roleArn, roleSessionName, bucketName, accessKey, secretKey);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6 Getting ready to delete the AWS resources");
        deleteKey(iam, userName, accessKey);
        deleteRole(iam, roleName, polArn);
        deleteIAMUser(iam, userName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("This IAM Scenario has successfully completed");
        System.out.println(DASHES);
    }

    public static AccessKey createIAMAccessKey(IamClient iam, String user) {
        try {
            CreateAccessKeyRequest request = CreateAccessKeyRequest.builder()
                    .userName(user)
                    .build();

            CreateAccessKeyResponse response = iam.createAccessKey(request);
            return response.accessKey();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static User createIAMUser(IamClient iam, String username) {
        try {
            // Create an IamWaiter object
            IamWaiter iamWaiter = iam.waiter();
            CreateUserRequest request = CreateUserRequest.builder()
                    .userName(username)
                    .build();
```

```
            // Wait until the user is created.
            CreateUserResponse response = iam.createUser(request);
            GetUserRequest userRequest = GetUserRequest.builder()
                    .userName(response.user().userName())
                    .build();

            WaiterResponse<GetUserResponse> waitUntilUserExists =
iamWaiter.waitUntilUserExists(userRequest);

waitUntilUserExists.matched().response().ifPresent(System.out::println);
            return response.user();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return null;
    }

    public static String createIAMRole(IamClient iam, String rolename, String
json) {

        try {
            CreateRoleRequest request = CreateRoleRequest.builder()
                    .roleName(rolename)
                    .assumeRolePolicyDocument(json)
                    .description("Created using the AWS SDK for Java")
                    .build();

            CreateRoleResponse response = iam.createRole(request);
            System.out.println("The ARN of the role is " +
response.role().arn());
            return response.role().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static String createIAMPolicy(IamClient iam, String policyName) {
        try {
```

```
            // Create an IamWaiter object.
            IamWaiter iamWaiter = iam.waiter();
            CreatePolicyRequest request = CreatePolicyRequest.builder()
                    .policyName(policyName)
                    .policyDocument(PolicyDocument).build();

            CreatePolicyResponse response = iam.createPolicy(request);
            GetPolicyRequest polRequest = GetPolicyRequest.builder()
                    .policyArn(response.policy().arn())
                    .build();

            WaiterResponse<GetPolicyResponse> waitUntilPolicyExists =
iamWaiter.waitUntilPolicyExists(polRequest);

waitUntilPolicyExists.matched().response().ifPresent(System.out::println);
            return response.policy().arn();

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }

    public static void attachIAMRolePolicy(IamClient iam, String roleName, String
policyArn) {
        try {
            ListAttachedRolePoliciesRequest request =
ListAttachedRolePoliciesRequest.builder()
                    .roleName(roleName)
                    .build();

            ListAttachedRolePoliciesResponse response =
iam.listAttachedRolePolicies(request);
            List<AttachedPolicy> attachedPolicies = response.attachedPolicies();
            String polArn;
            for (AttachedPolicy policy : attachedPolicies) {
                polArn = policy.policyArn();
                if (polArn.compareTo(policyArn) == 0) {
                    System.out.println(roleName + " policy is already attached to
this role.");
                    return;
                }
            }
```

```
            AttachRolePolicyRequest attachRequest =
AttachRolePolicyRequest.builder()
                    .roleName(roleName)
                    .policyArn(policyArn)
                    .build();

            iam.attachRolePolicy(attachRequest);
            System.out.println("Successfully attached policy " + policyArn + " to
role " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    // Invoke an Amazon S3 operation using the Assumed Role.
    public static void assumeRole(String roleArn, String roleSessionName, String
bucketName, String keyVal,
            String keySecret) {

        // Use the creds of the new IAM user that was created in this code
example.
        AwsBasicCredentials credentials = AwsBasicCredentials.create(keyVal,
keySecret);
        StsClient stsClient = StsClient.builder()
                .region(Region.US_EAST_1)

.credentialsProvider(StaticCredentialsProvider.create(credentials))
                .build();

        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();
            String key = myCreds.accessKeyId();
            String secKey = myCreds.secretAccessKey();
            String secToken = myCreds.sessionToken();
```

```
            // List all objects in an Amazon S3 bucket using the temp creds
retrieved by
            // invoking assumeRole.
            Region region = Region.US_EAST_1;
            S3Client s3 = S3Client.builder()
                    .credentialsProvider(

StaticCredentialsProvider.create(AwsSessionCredentials.create(key, secKey,
secToken)))
                    .region(region)
                    .build();

            System.out.println("Created a S3Client using temp credentials.");
            System.out.println("Listing objects in " + bucketName);
            ListObjectsRequest listObjects = ListObjectsRequest.builder()
                    .bucket(bucketName)
                    .build();

            ListObjectsResponse res = s3.listObjects(listObjects);
            List<S3Object> objects = res.contents();
            for (S3Object myValue : objects) {
                System.out.println("The name of the key is " + myValue.key());
                System.out.println("The owner is " + myValue.owner());
            }

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void deleteRole(IamClient iam, String roleName, String polArn)
{

        try {
            // First the policy needs to be detached.
            DetachRolePolicyRequest rolePolicyRequest =
DetachRolePolicyRequest.builder()
                    .policyArn(polArn)
                    .roleName(roleName)
                    .build();

            iam.detachRolePolicy(rolePolicyRequest);
```

```
            // Delete the policy.
            DeletePolicyRequest request = DeletePolicyRequest.builder()
                    .policyArn(polArn)
                    .build();

            iam.deletePolicy(request);
            System.out.println("*** Successfully deleted " + polArn);

            // Delete the role.
            DeleteRoleRequest roleRequest = DeleteRoleRequest.builder()
                    .roleName(roleName)
                    .build();

            iam.deleteRole(roleRequest);
            System.out.println("*** Successfully deleted " + roleName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteKey(IamClient iam, String username, String
accessKey) {
        try {
            DeleteAccessKeyRequest request = DeleteAccessKeyRequest.builder()
                    .accessKeyId(accessKey)
                    .userName(username)
                    .build();

            iam.deleteAccessKey(request);
            System.out.println("Successfully deleted access key " + accessKey +
                    " from user " + username);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteIAMUser(IamClient iam, String userName) {
        try {
            DeleteUserRequest request = DeleteUserRequest.builder()
                    .userName(userName)
```

```
                    .build();

            iam.deleteUser(request);
            System.out.println("*** Successfully deleted " + userName);

        } catch (IamException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
import {
  CreateUserCommand,
  CreateAccessKeyCommand,
  CreatePolicyCommand,
  CreateRoleCommand,
  AttachRolePolicyCommand,
  DeleteAccessKeyCommand,
  DeleteUserCommand,
  DeleteRoleCommand,
  DeletePolicyCommand,
  DetachRolePolicyCommand,
  IAMClient,
} from "@aws-sdk/client-iam";
import { ListBucketsCommand, S3Client } from "@aws-sdk/client-s3";
import { AssumeRoleCommand, STSClient } from "@aws-sdk/client-sts";
import { retry } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
  const { User } = await iamClient.send(
    new CreateUserCommand({ UserName: userName }),
  );

  if (!User) {
    throw new Error("User not created");
  }

  // Create an access key. This key is used to authenticate the new user to
  // Amazon Simple Storage Service (Amazon S3) and AWS Security Token Service
 (AWS STS).
  // It's not best practice to use access keys. For more information, see
 https://aws.amazon.com/iam/resources/best-practices/.
  const createAccessKeyResponse = await iamClient.send(
```

```
    new CreateAccessKeyCommand({ UserName: userName }),
  );

  if (
    !createAccessKeyResponse.AccessKey?.AccessKeyId ||
    !createAccessKeyResponse.AccessKey?.SecretAccessKey
  ) {
    throw new Error("Access key not created");
  }

  const {
    AccessKey: { AccessKeyId, SecretAccessKey },
  } = createAccessKeyResponse;

  let s3Client = new S3Client({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the list buckets operation until it succeeds. InvalidAccessKeyId is
  // thrown while the user and access keys are still stabilizing.
  await retry({ intervalInMs: 1000, maxRetries: 300 }, async () => {
    try {
      return await listBuckets(s3Client);
    } catch (err) {
      if (err instanceof Error && err.name === "InvalidAccessKeyId") {
        throw err;
      }
    }
  });

  // Retry the create role operation until it succeeds. A MalformedPolicyDocument
error
  // is thrown while the user and access keys are still stabilizing.
  const { Role } = await retry(
    {
      intervalInMs: 2000,
      maxRetries: 60,
    },
    () =>
      iamClient.send(
        new CreateRoleCommand({
```

```
        AssumeRolePolicyDocument: JSON.stringify({
          Version: "2012-10-17",
          Statement: [
            {
              Effect: "Allow",
              Principal: {
                // Allow the previously created user to assume this role.
                AWS: User.Arn,
              },
              Action: "sts:AssumeRole",
            },
          ],
        }),
        RoleName: roleName,
      }),
    ),
);

if (!Role) {
  throw new Error("Role not created");
}

// Create a policy that allows the user to list S3 buckets.
const { Policy: listBucketPolicy } = await iamClient.send(
  new CreatePolicyCommand({
    PolicyDocument: JSON.stringify({
      Version: "2012-10-17",
      Statement: [
        {
          Effect: "Allow",
          Action: ["s3:ListAllMyBuckets"],
          Resource: "*",
        },
      ],
    }),
    PolicyName: policyName,
  }),
);

if (!listBucketPolicy) {
  throw new Error("Policy not created");
}

// Attach the policy granting the 's3:ListAllMyBuckets' action to the role.
```

```
  await iamClient.send(
    new AttachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  // Assume the role.
  const stsClient = new STSClient({
    credentials: {
      accessKeyId: AccessKeyId,
      secretAccessKey: SecretAccessKey,
    },
  });

  // Retry the assume role operation until it succeeds.
  const { Credentials } = await retry(
    { intervalInMs: 2000, maxRetries: 60 },
    () =>
      stsClient.send(
        new AssumeRoleCommand({
          RoleArn: Role.Arn,
          RoleSessionName: `iamBasicScenarioSession-${Math.floor(
            Math.random() * 1000000,
          )}`,
          DurationSeconds: 900,
        }),
      ),
  );

  if (!Credentials?.AccessKeyId || !Credentials?.SecretAccessKey) {
    throw new Error("Credentials not created");
  }

  s3Client = new S3Client({
    credentials: {
      accessKeyId: Credentials.AccessKeyId,
      secretAccessKey: Credentials.SecretAccessKey,
      sessionToken: Credentials.SessionToken,
    },
  });

  // List the S3 buckets again.
  // Retry the list buckets operation until it succeeds. AccessDenied might
```

```
  // be thrown while the role policy is still stabilizing.
  await retry({ intervalInMs: 2000, maxRetries: 60 }, () =>
    listBuckets(s3Client),
  );

  // Clean up.
  await iamClient.send(
    new DetachRolePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeletePolicyCommand({
      PolicyArn: listBucketPolicy.Arn,
    }),
  );

  await iamClient.send(
    new DeleteRoleCommand({
      RoleName: Role.RoleName,
    }),
  );

  await iamClient.send(
    new DeleteAccessKeyCommand({
      UserName: userName,
      AccessKeyId,
    }),
  );

  await iamClient.send(
    new DeleteUserCommand({
      UserName: userName,
    }),
  );
};

/**
 *
 * @param {S3Client} s3Client
 */
const listBuckets = async (s3Client) => {
```

```
  const { Buckets } = await s3Client.send(new ListBucketsCommand({}));

  if (!Buckets) {
    throw new Error("Buckets not listed");
  }

  console.log(Buckets.map((bucket) => bucket.Name).join("\n"));
};
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for JavaScript.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK para Kotlin

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del usuario de IAM.

```
suspend fun main(args: Array<String>) {

    val usage = """
    Usage:
        <username> <policyName> <roleName> <roleSessionName> <fileLocation>
 <bucketName>

    Where:
        username - The name of the IAM user to create.
        policyName - The name of the policy to create.
        roleName - The name of the role to create.
        roleSessionName - The name of the session required for the assumeRole
 operation.
        fileLocation - The file location to the JSON required to create the role
 (see Readme).
        bucketName - The name of the Amazon S3 bucket from which objects are
 read.
    """

    if (args.size != 6) {
        println(usage)
        exitProcess(1)
    }

    val userName = args[0]
    val policyName = args[1]
    val roleName = args[2]
    val roleSessionName = args[3]
    val fileLocation = args[4]
    val bucketName = args[5]

    createUser(userName)
    println("$userName was successfully created.")

    val polArn = createPolicy(policyName)
    println("The policy $polArn was successfully created.")

    val roleArn = createRole(roleName, fileLocation)
    println("$roleArn was successfully created.")
    attachRolePolicy(roleName, polArn)

    println("*** Wait for 1 MIN so the resource is available.")
```

```
    delay(60000)
    assumeGivenRole(roleArn, roleSessionName, bucketName)

    println("*** Getting ready to delete the AWS resources.")
    deleteRole(roleName, polArn)
    deleteUser(userName)
    println("This IAM Scenario has successfully completed.")
}

suspend fun createUser(usernameVal: String?): String? {

    val request = CreateUserRequest {
        userName = usernameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}

suspend fun createPolicy(policyNameVal: String?): String {

    val policyDocumentValue: String = "{" +
        "  \"Version\": \"2012-10-17\"," +
        "  \"Statement\": [" +
        "    {" +
        "        \"Effect\": \"Allow\"," +
        "        \"Action\": [" +
        "            \"s3:*\"" +
        "       ]," +
        "       \"Resource\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentValue
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
```

```
    }
}

suspend fun createRole(rolenameVal: String?, fileLocation: String?): String? {

    val jsonObject = fileLocation?.let { readJsonSimpleDemo(it) } as JSONObject

    val request = CreateRoleRequest {
        roleName = rolenameVal
        assumeRolePolicyDocument = jsonObject.toJSONString()
        description = "Created using the AWS SDK for Kotlin"
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createRole(request)
        return response.role?.arn
    }
}

suspend fun attachRolePolicy(roleNameVal: String, policyArnVal: String) {

    val request = ListAttachedRolePoliciesRequest {
        roleName = roleNameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAttachedRolePolicies(request)
        val attachedPolicies = response.attachedPolicies

        // Ensure that the policy is not attached to this role.
        val checkStatus: Int
        if (attachedPolicies != null) {
            checkStatus = checkMyList(attachedPolicies, policyArnVal)
            if (checkStatus == -1)
                return
        }

        val policyRequest = AttachRolePolicyRequest {
            roleName = roleNameVal
            policyArn = policyArnVal
        }
        iamClient.attachRolePolicy(policyRequest)
        println("Successfully attached policy $policyArnVal to role
 $roleNameVal")
```

```
    }
}

fun checkMyList(attachedPolicies: List<AttachedPolicy>, policyArnVal: String):
 Int {

    for (policy in attachedPolicies) {
        val polArn = policy.policyArn.toString()

        if (polArn.compareTo(policyArnVal) == 0) {
            println("The policy is already attached to this role.")
            return -1
        }
    }
    return 0
}

suspend fun assumeGivenRole(roleArnVal: String?, roleSessionNameVal: String?,
 bucketName: String) {

    val stsClient = StsClient {
        region = "us-east-1"
    }

    val roleRequest = AssumeRoleRequest {
        roleArn = roleArnVal
        roleSessionName = roleSessionNameVal
    }

    val roleResponse = stsClient.assumeRole(roleRequest)
    val myCreds = roleResponse.credentials
    val key = myCreds?.accessKeyId
    val secKey = myCreds?.secretAccessKey
    val secToken = myCreds?.sessionToken

    val staticCredentials = StaticCredentialsProvider {
        accessKeyId = key
        secretAccessKey = secKey
        sessionToken = secToken
    }

    // List all objects in an Amazon S3 bucket using the temp creds.
    val s3 = S3Client {
        credentialsProvider = staticCredentials
```

```
        region = "us-east-1"
    }

    println("Created a S3Client using temp credentials.")
    println("Listing objects in $bucketName")

    val listObjects = ListObjectsRequest {
        bucket = bucketName
    }

    val response = s3.listObjects(listObjects)
    response.contents?.forEach { myObject ->
        println("The name of the key is ${myObject.key}")
        println("The owner is ${myObject.owner}")
    }
}

suspend fun deleteRole(roleNameVal: String, polArn: String) {

    val iam = IamClient { region = "AWS_GLOBAL" }

    // First the policy needs to be detached.
    val rolePolicyRequest = DetachRolePolicyRequest {
        policyArn = polArn
        roleName = roleNameVal
    }

    iam.detachRolePolicy(rolePolicyRequest)

    // Delete the policy.
    val request = DeletePolicyRequest {
        policyArn = polArn
    }

    iam.deletePolicy(request)
    println("*** Successfully deleted $polArn")

    // Delete the role.
    val roleRequest = DeleteRoleRequest {
        roleName = roleNameVal
    }

    iam.deleteRole(roleRequest)
    println("*** Successfully deleted $roleNameVal")
```

```
}

suspend fun deleteUser(userNameVal: String) {
    val iam = IamClient { region = "AWS_GLOBAL" }
    val request = DeleteUserRequest {
        userName = userNameVal
    }

    iam.deleteUser(request)
    println("*** Successfully deleted $userNameVal")
}

@Throws(java.lang.Exception::class)
fun readJsonSimpleDemo(filename: String): Any? {
    val reader = FileReader(filename)
    val jsonParser = JSONParser()
    return jsonParser.parse(reader)
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Kotlin.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

PHP

SDK para PHP

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
namespace Iam\Basics;

require 'vendor/autoload.php';

use Aws\Credentials\Credentials;
use Aws\S3\Exception\S3Exception;
use Aws\S3\S3Client;
use Aws\Sts\StsClient;
use Iam\IAMService;

echo("\n");
echo("--------------------------------------\n");
print("Welcome to the IAM getting started demo using PHP!\n");
echo("--------------------------------------\n");

$uuid = uniqid();
$service = new IAMService();

$user = $service->createUser("iam_demo_user_$uuid");
echo "Created user with the arn: {$user['Arn']}\n";

$key = $service->createAccessKey($user['UserName']);
$assumeRolePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{$user['Arn']}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }";
$assumeRoleRole = $service->createRole("iam_demo_role_$uuid",
 $assumeRolePolicyDocument);
```

```
echo "Created role: {$assumeRoleRole['RoleName']}\n";

$listAllBucketsPolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
}";
$listAllBucketsPolicy = $service->createPolicy("iam_demo_policy_$uuid",
 $listAllBucketsPolicyDocument);
echo "Created policy: {$listAllBucketsPolicy['PolicyName']}\n";

$service->attachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);

$inlinePolicyDocument = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{$assumeRoleRole['Arn']}\"}]
}";
$inlinePolicy = $service->createUserPolicy("iam_demo_inline_policy_$uuid",
 $inlinePolicyDocument, $user['UserName']);
//First, fail to list the buckets with the user
$credentials = new Credentials($key['AccessKeyId'], $key['SecretAccessKey']);
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
try {
    $s3Client->listBuckets([
    ]);
    echo "this should not run";
} catch (S3Exception $exception) {
    echo "successfully failed!\n";
}

$stsClient = new StsClient(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $credentials]);
sleep(10);
$assumedRole = $stsClient->assumeRole([
    'RoleArn' => $assumeRoleRole['Arn'],
    'RoleSessionName' => "DemoAssumeRoleSession_$uuid",
]);
```

```
$assumedCredentials = [
    'key' => $assumedRole['Credentials']['AccessKeyId'],
    'secret' => $assumedRole['Credentials']['SecretAccessKey'],
    'token' => $assumedRole['Credentials']['SessionToken'],
];
$s3Client = new S3Client(['region' => 'us-west-2', 'version' => 'latest',
 'credentials' => $assumedCredentials]);
try {
    $s3Client->listBuckets([]);
    echo "this should now run!\n";
} catch (S3Exception $exception) {
    echo "this should now not fail\n";
}

$service->detachRolePolicy($assumeRoleRole['RoleName'],
 $listAllBucketsPolicy['Arn']);
$deletePolicy = $service->deletePolicy($listAllBucketsPolicy['Arn']);
echo "Delete policy: {$listAllBucketsPolicy['PolicyName']}\n";
$deletedRole = $service->deleteRole($assumeRoleRole['Arn']);
echo "Deleted role: {$assumeRoleRole['RoleName']}\n";
$deletedKey = $service->deleteAccessKey($key['AccessKeyId'], $user['UserName']);
$deletedUser = $service->deleteUser($user['UserName']);
echo "Delete user: {$user['UserName']}\n";
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for PHP.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy

- PutUserPolicy

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
import json
import sys
import time
from uuid import uuid4

import boto3
from botocore.exceptions import ClientError


def progress_bar(seconds):
    """Shows a simple progress bar in the command window."""
    for _ in range(seconds):
        time.sleep(1)
        print(".", end="")
        sys.stdout.flush()
    print()


def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates an access key pair for the user.
    Creates a role with a policy that lets the user assume the role.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
```

```
    Creates an inline policy for the user that lets the user assume the role.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, and role.
    """
    try:
        user = iam_resource.create_user(UserName=f"demo-user-{uuid4()}")
        print(f"Created user {user.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a user for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user_key = user.create_access_key_pair()
        print(f"Created access key pair for user.")
    except ClientError as error:
        print(
            f"Couldn't create access keys for user {user.name}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    try:
        role = iam_resource.create_role(
            RoleName=f"demo-role-{uuid4()}",
            AssumeRolePolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {"AWS": user.arn},
                            "Action": "sts:AssumeRole",
                        }
```

```
                    ],
                }
            ),
        )
        print(f"Created role {role.name}.")
    except ClientError as error:
        print(
            f"Couldn't create a role for the demo. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        policy = iam_resource.create_policy(
            PolicyName=f"demo-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "s3:ListAllMyBuckets",
                            "Resource": "arn:aws:s3:::*",
                        }
                    ],
                }
            ),
        )
        role.attach_policy(PolicyArn=policy.arn)
        print(f"Created policy {policy.policy_name} and attached it to the
role.")
    except ClientError as error:
        print(
            f"Couldn't create a policy and attach it to role {role.name}. Here's
why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        user.create_policy(
            PolicyName=f"demo-user-policy-{uuid4()}",
            PolicyDocument=json.dumps(
                {
```

```
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Action": "sts:AssumeRole",
                            "Resource": role.arn,
                        }
                    ],
                }
            ),
        )
        print(
            f"Created an inline policy for {user.name} that lets the user assume
"
            f"the role."
        )
    except ClientError as error:
        print(
            f"Couldn't create an inline policy for user {user.name}. Here's why:
"
            f"{error.response['Error']['Message']}"
        )
        raise

    print("Give AWS time to propagate these new resources and connections.",
end="")
    progress_bar(10)

    return user, user_key, role


def show_access_denied_without_role(user_key):
    """
    Shows that listing buckets without first assuming the role is not allowed.

    :param user_key: The key of the user created during setup. This user does not
                     have permission to list buckets in the account.
    """
    print(f"Try to list buckets without first assuming the role.")
    s3_denied_resource = boto3.resource(
        "s3", aws_access_key_id=user_key.id,
aws_secret_access_key=user_key.secret
    )
    try:
```

```
        for bucket in s3_denied_resource.buckets.all():
            print(bucket.name)
        raise RuntimeError("Expected to get AccessDenied error when listing
 buckets!")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Attempt to list buckets with no permissions: AccessDenied.")
        else:
            raise


def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the
 account.
    Uses the temporary credentials from the role to list the buckets that are
 owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the
 role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary
 credentials.
```

```
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise




def teardown(user, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    try:
        for attached in role.attached_policies.all():
            policy_name = attached.policy_name
            role.detach_policy(PolicyArn=attached.arn)
            attached.delete()
            print(f"Detached and deleted {policy_name}.")
        role.delete()
        print(f"Deleted {role.name}.")
    except ClientError as error:
        print(
            "Couldn't detach policy, delete policy, or delete role. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    try:
        for user_pol in user.policies.all():
```

```
            user_pol.delete()
            print("Deleted inline user policy.")
        for key in user.access_keys.all():
            key.delete()
            print("Deleted user's access key.")
        user.delete()
        print(f"Deleted {user.name}.")
    except ClientError as error:
        print(
            "Couldn't delete user policy or delete user. Here's why: "
            f"{error.response['Error']['Message']}"
        )


def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the IAM create user and assume role demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user = None
    role = None
    try:
        user, user_key, role = setup(iam_resource)
        print(f"Created {user.name} and {role.name}.")
        show_access_denied_without_role(user_key)
        list_buckets_from_assumed_role(user_key, role.arn,
 "AssumeRoleDemoSession")
    except Exception:
        print("Something went wrong!")
    finally:
        if user is not None and role is not None:
            teardown(user, role)
        print("Thanks for watching!")


if __name__ == "__main__":
    usage_demo()
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - AttachRolePolicy

- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM y un rol que conceda permiso para enumerar los buckets de Amazon S3. El usuario solo tiene derechos para asumir el rol. Después de asumir el rol, use las credenciales temporales para enumerar los buckets de la cuenta.

```
# Wraps the scenario actions.
class ScenarioCreateUserAssumeRole
  attr_reader :iam_client

  # @param [Aws::IAM::Client] iam_client: The AWS IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Waits for the specified number of seconds.
```

```
  #
  # @param duration [Integer] The number of seconds to wait.
  def wait(duration)
    puts("Give AWS time to propagate resources...")
    sleep(duration)
  end

  # Creates a user.
  #
  # @param user_name [String] The name to give the user.
  # @return [Aws::IAM::User] The newly created user.
  def create_user(user_name)
    user = @iam_client.create_user(user_name: user_name).user
    @logger.info("Created demo user named #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Tried and failed to create demo user.")
    @logger.info("\t#{e.code}: #{e.message}")
    @logger.info("\nCan't continue the demo without a user!")
    raise
  else
    user
  end

  # Creates an access key for a user.
  #
  # @param user [Aws::IAM::User] The user that owns the key.
  # @return [Aws::IAM::AccessKeyPair] The newly created access key.
  def create_access_key_pair(user)
    user_key = @iam_client.create_access_key(user_name:
user.user_name).access_key
    @logger.info("Created accesskey pair for user #{user.user_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create access keys for user #{user.user_name}.")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
role.
```

```
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Principal: {'AWS': user.arn},
                   Action: "sts:AssumeRole"
                 }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account,
and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                   Effect: "Allow",
                   Action: "s3:ListAllMyBuckets",
                   Resource: "arn:aws:s3:::*"
                 }]
    }.to_json
    policy = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document
    ).policy
    @iam_client.attach_role_policy(
```

```
      role_name: role.role_name,
      policy_arn: policy.arn
    )
    @logger.info("Created policy #{policy.policy_name} and attached it to role
#{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role
#{role.role_name}. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an inline policy for a user that lets the user assume a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param user [Aws::IAM::User] The user that owns the policy.
  # @param role [Aws::IAM::Role] The role that can be assumed.
  # @return [Aws::IAM::UserPolicy] The newly created policy.
  def create_user_policy(policy_name, user, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
                    Effect: "Allow",
                    Action: "sts:AssumeRole",
                    Resource: role.arn
                  }]
    }.to_json
    @iam_client.put_user_policy(
      user_name: user.user_name,
      policy_name: policy_name,
      policy_document: policy_document
    )
    puts("Created an inline policy for #{user.user_name} that lets the user
assume role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

  # Creates an Amazon S3 resource with specified credentials. This is separated
into a
  # factory function so that it can be mocked for unit testing.
```

```
 #
 # @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
 def create_s3_resource(credentials)
   Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
 end

 # Lists the S3 buckets for the account, using the specified Amazon S3 resource.
 # Because the resource uses credentials with limited access, it may not be able
to
 # list the S3 buckets.
 #
 # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
 def list_buckets(s3_resource)
   count = 10
   s3_resource.buckets.each do |bucket|
     @logger.info "\t#{bucket.name}"
     count -= 1
     break if count.zero?
   end
 rescue Aws::Errors::ServiceError => e
   if e.code == "AccessDenied"
     puts("Attempt to list buckets with no permissions: AccessDenied.")
   else
     @logger.info("Couldn't list buckets for the account. Here's why: ")
     @logger.info("\t#{e.code}: #{e.message}")
     raise
   end
 end

 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
```

```
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end

 # Deletes a role. If the role has policies attached, they are detached and
 # deleted before the role is deleted.
 #
 # @param role_name [String] The name of the role to delete.
 def delete_role(role_name)
   @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
     @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
     @iam_client.delete_policy(policy_arn: policy.policy_arn)
     @logger.info("Detached and deleted policy #{policy.policy_name}.")
   end
   @iam_client.delete_role({ role_name: role_name })
   @logger.info("Role deleted: #{role_name}.")
 rescue Aws::Errors::ServiceError => e
   @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
   @logger.info("\t#{e.code}: #{e.message}")
   raise
 end

 # Deletes a user. If the user has inline policies or access keys, they are
deleted
 # before the user is deleted.
 #
 # @param user [Aws::IAM::User] The user to delete.
 def delete_user(user_name)
   user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
```

```
    user.each do |key|
      @iam_client.delete_access_key({ access_key_id: key.access_key_id,
 user_name: user_name })
      @logger.info("Deleted access key #{key.access_key_id} for user
 '#{user_name}'.")
    end

    @iam_client.delete_user(user_name: user_name)
    @logger.info("Deleted user '#{user_name}'.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting user '#{user_name}': #{e.message}")
  end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-
#{Random.uuid}", role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user,
 role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
 user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
 user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
  scenario.delete_role(role.role_name)
  puts("Deleting user '#{user.user_name}', policies, and keys.")
  scenario.delete_user(user.user_name)
  puts("Thanks for watching!")
```

```
  puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
 $PROGRAM_NAME == __FILE__
```

- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Ruby.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
use aws_config::meta::region::RegionProviderChain;
use aws_sdk_iam::Error as iamError;
use aws_sdk_iam::{config::Credentials as iamCredentials, config::Region, Client
 as iamClient};
use aws_sdk_s3::Client as s3Client;
use aws_sdk_sts::Client as stsClient;
use tokio::time::{sleep, Duration};
use uuid::Uuid;

#[tokio::main]
async fn main() -> Result<(), iamError> {
    let (client, uuid, list_all_buckets_policy_document, inline_policy_document)
 =
        initialize_variables().await;

    if let Err(e) = run_iam_operations(
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
    .await
    {
        println!("{:?}", e);
    };

    Ok(())
}

async fn initialize_variables() -> (iamClient, String, String, String) {
    let region_provider = RegionProviderChain::first_try(Region::new("us-
west-2"));

    let shared_config =
 aws_config::from_env().region(region_provider).load().await;
    let client = iamClient::new(&shared_config);
    let uuid = Uuid::new_v4().to_string();

    let list_all_buckets_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"s3:ListAllMyBuckets\",
                    \"Resource\": \"arn:aws:s3:::*\"}]
```

```
    }"
    .to_string();
    let inline_policy_document = "{
                \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Action\": \"sts:AssumeRole\",
                    \"Resource\": \"{}\"}]
    }"
    .to_string();

    (
        client,
        uuid,
        list_all_buckets_policy_document,
        inline_policy_document,
    )
}

async fn run_iam_operations(
    client: iamClient,
    uuid: String,
    list_all_buckets_policy_document: String,
    inline_policy_document: String,
) -> Result<(), iamError> {
    let user = iam_service::create_user(&client, &format!("{}{}",
 "iam_demo_user_", uuid)).await?;
    println!("Created the user with the name: {}", user.user_name());
    let key = iam_service::create_access_key(&client, user.user_name()).await?;

    let assume_role_policy_document = "{
        \"Version\": \"2012-10-17\",
                \"Statement\": [{
                    \"Effect\": \"Allow\",
                    \"Principal\": {\"AWS\": \"{}\"},
                    \"Action\": \"sts:AssumeRole\"
                }]
            }"
    .to_string()
    .replace("{}", user.arn());

    let assume_role_role = iam_service::create_role(
        &client,
        &format!("{}{}", "iam_demo_role_", uuid),
```

```
        &assume_role_policy_document,
    )
    .await?;
    println!("Created the role with the ARN: {}", assume_role_role.arn());

    let list_all_buckets_policy = iam_service::create_policy(
        &client,
        &format!("{}{}", "iam_demo_policy_", uuid),
        &list_all_buckets_policy_document,
    )
    .await?;
    println!(
        "Created policy: {}",
        list_all_buckets_policy.policy_name.as_ref().unwrap()
    );

    let attach_role_policy_result =
        iam_service::attach_role_policy(&client, &assume_role_role,
&list_all_buckets_policy)
            .await?;
    println!(
        "Attached the policy to the role: {:?}",
        attach_role_policy_result
    );

    let inline_policy_name = format!("{}{}", "iam_demo_inline_policy_", uuid);
    let inline_policy_document = inline_policy_document.replace("{}",
assume_role_role.arn());
    iam_service::create_user_policy(&client, &user, &inline_policy_name,
&inline_policy_document)
        .await?;
    println!("Created inline policy.");

    //First, fail to list the buckets with the user.
    let creds = iamCredentials::from_keys(key.access_key_id(),
key.secret_access_key(), None);
    let fail_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    println!("Fail config: {:?}", fail_config);
    let fail_client: s3Client = s3Client::new(&fail_config);
    match fail_client.list_buckets().send().await {
        Ok(e) => {
```

```
            println!("This should not run. {:?}", e);
        }
        Err(e) => {
            println!("Successfully failed with error: {:?}", e)
        }
    }

    let sts_config = aws_config::from_env()
        .credentials_provider(creds.clone())
        .load()
        .await;
    let sts_client: stsClient = stsClient::new(&sts_config);
    sleep(Duration::from_secs(10)).await;
    let assumed_role = sts_client
        .assume_role()
        .role_arn(assume_role_role.arn())
        .role_session_name(&format!("{}{}", "iam_demo_assumerole_session_",
uuid))
        .send()
        .await;
    println!("Assumed role: {:?}", assumed_role);
    sleep(Duration::from_secs(10)).await;

    let assumed_credentials = iamCredentials::from_keys(
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .access_key_id(),
        assumed_role
            .as_ref()
            .unwrap()
            .credentials
            .as_ref()
            .unwrap()
            .secret_access_key(),
        Some(
            assumed_role
                .as_ref()
                .unwrap()
                .credentials
                .as_ref()
```

```
                .unwrap()
                .session_token
                .clone(),
        ),
    );

    let succeed_config = aws_config::from_env()
        .credentials_provider(assumed_credentials)
        .load()
        .await;
    println!("succeed config: {:?}", succeed_config);
    let succeed_client: s3Client = s3Client::new(&succeed_config);
    sleep(Duration::from_secs(10)).await;
    match succeed_client.list_buckets().send().await {
        Ok(_) => {
            println!("This should now run successfully.")
        }
        Err(e) => {
            println!("This should not run. {:?}", e);
            panic!()
        }
    }

    //Clean up.
    iam_service::detach_role_policy(
        &client,
        assume_role_role.role_name(),
        list_all_buckets_policy.arn().unwrap_or_default(),
    )
    .await?;
    iam_service::delete_policy(&client, list_all_buckets_policy).await?;
    iam_service::delete_role(&client, &assume_role_role).await?;
    println!("Deleted role {}", assume_role_role.role_name());
    iam_service::delete_access_key(&client, &user, &key).await?;
    println!("Deleted key for {}", key.user_name());
    iam_service::delete_user_policy(&client, &user, &inline_policy_name).await?;
    println!("Deleted inline user policy: {}", inline_policy_name);
    iam_service::delete_user(&client, &user).await?;
    println!("Deleted user {}", user.user_name());

    Ok(())
}
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Creación de usuarios de IAM de solo lectura y lectura y escritura con un SDK de AWS

El siguiente ejemplo de código muestra cómo crear usuarios y asociar políticas a ellos.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Crear dos usuarios de IAM.
- Asocie una política para que un usuario pueda obtener objetos y ponerlos en un bucket de Amazon S3.
- Asocie una política que permita al segundo usuario obtener objetos del bucket.
- Obtenga diferentes permisos para el bucket en función de las credenciales del usuario.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del usuario de IAM.

```
import logging
import time

import boto3
from botocore.exceptions import ClientError

import access_key_wrapper
import policy_wrapper

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_user(user_name):
    """
    Creates a user. By default, a user has no permissions or access keys.

    :param user_name: The name of the user.
    :return: The newly created user.
    """
    try:
        user = iam.create_user(UserName=user_name)
        logger.info("Created user %s.", user.name)
    except ClientError:
        logger.exception("Couldn't create user %s.", user_name)
        raise
    else:
        return user



def update_user(user_name, new_user_name):
```

```
    """
    Updates a user's name.

    :param user_name: The current name of the user to update.
    :param new_user_name: The new name to assign to the user.
    :return: The updated user.
    """
    try:
        user = iam.User(user_name)
        user.update(NewUserName=new_user_name)
        logger.info("Renamed %s to %s.", user_name, new_user_name)
    except ClientError:
        logger.exception("Couldn't update name for user %s.", user_name)
        raise
    return user



def list_users():
    """
    Lists the users in the current account.

    :return: The list of users.
    """
    try:
        users = list(iam.users.all())
        logger.info("Got %s users.", len(users))
    except ClientError:
        logger.exception("Couldn't get users.")
        raise
    else:
        return users



def delete_user(user_name):
    """
    Deletes a user. Before a user can be deleted, all associated resources,
    such as access keys and policies, must be deleted or detached.

    :param user_name: The name of the user.
    """
    try:
        iam.User(user_name).delete()
```

```
        logger.info("Deleted user %s.", user_name)
    except ClientError:
        logger.exception("Couldn't delete user %s.", user_name)
        raise



def attach_policy(user_name, policy_arn):
    """
    Attaches a policy to a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to user %s.", policy_arn,
 user_name)
        raise



def detach_policy(user_name, policy_arn):
    """
    Detaches a policy from a user.

    :param user_name: The name of the user.
    :param policy_arn: The Amazon Resource Name (ARN) of the policy.
    """
    try:
        iam.User(user_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from user %s.", policy_arn, user_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from user %s.", policy_arn, user_name
        )
        raise
```

Cree funciones que encapsulen las acciones de la política de IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
```

```
        raise
    else:
        return policy



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Cree funciones que encapsulen las acciones de la clave de acceso de IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
```

```
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilice las funciones del contenedor para crear usuarios con diferentes políticas y utilizar sus credenciales para acceder a un bucket de Amazon S3.

```
def usage_demo():
    """
    Shows how to manage users, keys, and policies.
    This demonstration creates two users: one user who can put and get objects in
 an
    Amazon S3 bucket, and another user who can only get objects from the bucket.
    The demo then shows how the users can perform only the actions they are
 permitted
    to perform.
```

```
    """
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management user demo.")
    print("-" * 88)
    print(
        "Users can have policies and roles attached to grant them specific "
        "permissions."
    )
    s3 = boto3.resource("s3")
    bucket = s3.create_bucket(
        Bucket=f"demo-iam-bucket-{time.time_ns()}",
        CreateBucketConfiguration={
            "LocationConstraint": s3.meta.client.meta.region_name
        },
    )
    print(f"Created an Amazon S3 bucket named {bucket.name}.")
    user_read_writer = create_user("demo-iam-read-writer")
    user_reader = create_user("demo-iam-reader")
    print(f"Created two IAM users: {user_read_writer.name} and
{user_reader.name}")
    update_user(user_read_writer.name, "demo-iam-creator")
    update_user(user_reader.name, "demo-iam-getter")
    users = list_users()
    user_read_writer = next(
        user for user in users if user.user_id == user_read_writer.user_id
    )
    user_reader = next(user for user in users if user.user_id ==
user_reader.user_id)
    print(
        f"Changed the names of the users to {user_read_writer.name} "
        f"and {user_reader.name}."
    )

    read_write_policy = policy_wrapper.create_policy(
        "demo-iam-read-write-policy",
        "Grants rights to create and get an object in the demo bucket.",
        ["s3:PutObject", "s3:GetObject"],
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(
        f"Created policy {read_write_policy.policy_name} with ARN:
{read_write_policy.arn}"
    )
```

```
    print(read_write_policy.description)
    read_policy = policy_wrapper.create_policy(
        "demo-iam-read-policy",
        "Grants rights to get an object from the demo bucket.",
        "s3:GetObject",
        f"arn:aws:s3:::{bucket.name}/*",
    )
    print(f"Created policy {read_policy.policy_name} with ARN:
{read_policy.arn}")
    print(read_policy.description)
    attach_policy(user_read_writer.name, read_write_policy.arn)
    print(f"Attached {read_write_policy.policy_name} to
{user_read_writer.name}.")
    attach_policy(user_reader.name, read_policy.arn)
    print(f"Attached {read_policy.policy_name} to {user_reader.name}.")

    user_read_writer_key = access_key_wrapper.create_key(user_read_writer.name)
    print(f"Created access key pair for {user_read_writer.name}.")
    user_reader_key = access_key_wrapper.create_key(user_reader.name)
    print(f"Created access key pair for {user_reader.name}.")

    s3_read_writer_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_read_writer_key.id,
        aws_secret_access_key=user_read_writer_key.secret,
    )
    demo_object_key = f"object-{time.time_ns()}"
    demo_object = None
    while demo_object is None:
        try:
            demo_object = s3_read_writer_resource.Bucket(bucket.name).put_object(
                Key=demo_object_key, Body=b"AWS IAM demo object content!"
            )
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise
    print(
        f"Put {demo_object_key} into {bucket.name} using "
        f"{user_read_writer.name}'s credentials."
    )
```

```
    read_writer_object = s3_read_writer_resource.Bucket(bucket.name).Object(
        demo_object_key
    )
    read_writer_content = read_writer_object.get()["Body"].read()
    print(f"Got object {read_writer_object.key} using read-writer user's
credentials.")
    print(f"Object content: {read_writer_content}")

    s3_reader_resource = boto3.resource(
        "s3",
        aws_access_key_id=user_reader_key.id,
        aws_secret_access_key=user_reader_key.secret,
    )
    demo_content = None
    while demo_content is None:
        try:
            demo_object =
s3_reader_resource.Bucket(bucket.name).Object(demo_object_key)
            demo_content = demo_object.get()["Body"].read()
            print(f"Got object {demo_object.key} using reader user's
credentials.")
            print(f"Object content: {demo_content}")
        except ClientError as error:
            if error.response["Error"]["Code"] == "InvalidAccessKeyId":
                print("Access key not yet available. Waiting...")
                time.sleep(1)
            else:
                raise

    try:
        demo_object.delete()
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("-" * 88)
            print(
                "Tried to delete the object using the reader user's credentials.
"
                "Got expected AccessDenied error because the reader is not "
                "allowed to delete objects."
            )
            print("-" * 88)

    access_key_wrapper.delete_key(user_reader.name, user_reader_key.id)
    detach_policy(user_reader.name, read_policy.arn)
```

```
    policy_wrapper.delete_policy(read_policy.arn)
    delete_user(user_reader.name)
    print(f"Deleted keys, detached and deleted policy, and deleted
{user_reader.name}.")

    access_key_wrapper.delete_key(user_read_writer.name, user_read_writer_key.id)
    detach_policy(user_read_writer.name, read_write_policy.arn)
    policy_wrapper.delete_policy(read_write_policy.arn)
    delete_user(user_read_writer.name)
    print(
        f"Deleted keys, detached and deleted policy, and deleted
{user_read_writer.name}."
    )

    bucket.objects.delete()
    bucket.delete()
    print(f"Emptied and deleted {bucket.name}.")
    print("Thanks for watching!")
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - AttachUserPolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteUser
  - DetachUserPolicy
  - ListUsers
  - UpdateUser

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Administrar claves de acceso de IAM con un SDK de AWS

El siguiente ejemplo de código muestra cómo administrar claves de acceso.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Crear y enumerar claves de acceso.
- Saber cuándo y cómo se utilizó por última vez una clave de acceso.
- Actualizar y eliminar las claves de acceso.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones de la clave de acceso de IAM.

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

iam = boto3.resource("iam")

def list_keys(user_name):
    """
    Lists the keys owned by the specified user.

    :param user_name: The name of the user.
```

```
    :return: The list of keys owned by the user.
    """
    try:
        keys = list(iam.User(user_name).access_keys.all())
        logger.info("Got %s access keys for %s.", len(keys), user_name)
    except ClientError:
        logger.exception("Couldn't get access keys for %s.", user_name)
        raise
    else:
        return keys



def create_key(user_name):
    """
    Creates an access key for the specified user. Each user can have a
    maximum of two keys.

    :param user_name: The name of the user.
    :return: The created access key.
    """
    try:
        key_pair = iam.User(user_name).create_access_key_pair()
        logger.info(
            "Created access key pair for %s. Key ID is %s.",
            key_pair.user_name,
            key_pair.id,
        )
    except ClientError:
        logger.exception("Couldn't create access key pair for %s.", user_name)
        raise
    else:
        return key_pair



def get_last_use(key_id):
    """
    Gets information about when and how a key was last used.

    :param key_id: The ID of the key to look up.
    :return: Information about the key's last use.
    """
    try:
```

```
        response = iam.meta.client.get_access_key_last_used(AccessKeyId=key_id)
        last_used_date = response["AccessKeyLastUsed"].get("LastUsedDate", None)
        last_service = response["AccessKeyLastUsed"].get("ServiceName", None)
        logger.info(
            "Key %s was last used by %s on %s to access %s.",
            key_id,
            response["UserName"],
            last_used_date,
            last_service,
        )
    except ClientError:
        logger.exception("Couldn't get last use of key %s.", key_id)
        raise
    else:
        return response



def update_key(user_name, key_id, activate):
    """
    Updates the status of a key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to update.
    :param activate: When True, the key is activated. Otherwise, the key is
 deactivated.
    """

    try:
        key = iam.User(user_name).AccessKey(key_id)
        if activate:
            key.activate()
        else:
            key.deactivate()
        logger.info("%s key %s.", "Activated" if activate else "Deactivated",
 key_id)
    except ClientError:
        logger.exception(
            "Couldn't %s key %s.", "Activate" if activate else "Deactivate",
 key_id
        )
        raise
```

```
def delete_key(user_name, key_id):
    """
    Deletes a user's access key.

    :param user_name: The user that owns the key.
    :param key_id: The ID of the key to delete.
    """

    try:
        key = iam.AccessKey(user_name, key_id)
        key.delete()
        logger.info("Deleted access key %s for %s.", key.id, key.user_name)
    except ClientError:
        logger.exception("Couldn't delete key %s for %s", key_id, user_name)
        raise
```

Utilice las funciones del contenedor para llevar a cabo acciones de clave de acceso para el usuario actual.

```
def usage_demo():
    """Shows how to create and manage access keys."""

    def print_keys():
        """Gets and prints the current keys for a user."""
        current_keys = list_keys(current_user_name)
        print("The current user's keys are now:")
        print(*[f"{key.id}: {key.status}" for key in current_keys], sep="\n")

    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management access key demo.")
    print("-" * 88)
    current_user_name = iam.CurrentUser().user_name
    print(
        f"This demo creates an access key for the current user "
        f"({current_user_name}), manipulates the key in a few ways, and then "
        f"deletes it."
    )
    all_keys = list_keys(current_user_name)
```

```
    if len(all_keys) == 2:
        print(
            "The current user already has the maximum of 2 access keys. To run "
            "this demo, either delete one of the access keys or use a user "
            "that has only 1 access key."
        )
    else:
        new_key = create_key(current_user_name)
        print(f"Created a new key with id {new_key.id} and secret
{new_key.secret}.")
        print_keys()
        existing_key = next(key for key in all_keys if key != new_key)
        last_use = get_last_use(existing_key.id)["AccessKeyLastUsed"]
        print(
            f"Key {all_keys[0].id} was last used to access
{last_use['ServiceName']} "
            f"on {last_use['LastUsedDate']}"
        )
        update_key(current_user_name, new_key.id, False)
        print(f"Key {new_key.id} is now deactivated.")
        print_keys()
        delete_key(current_user_name, new_key.id)
        print_keys()
        print("Thanks for watching!")
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - CreateAccessKey
  - DeleteAccessKey
  - GetAccessKeyLastUsed
  - ListAccessKeys
  - UpdateAccessKey

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Administrar políticas de IAM con un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Crear y enumerar políticas.
- Crear y obtener versiones de políticas.
- Revertir una política a una versión anterior.
- Eliminar políticas.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones de la política de IAM.

```
import json
import logging
import operator
import pprint
import time

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_policy(name, description, actions, resource_arn):
    """
    Creates a policy that contains a single statement.

    :param name: The name of the policy to create.
    :param description: The description of the policy.
    :param actions: The actions allowed by the policy. These typically take the
                    form of service:action, such as s3:PutObject.
```

```
    :param resource_arn: The Amazon Resource Name (ARN) of the resource this
 policy
                         applies to. This ARN can contain wildcards, such as
                         'arn:aws:s3:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-bucket'.
    :return: The newly created policy.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.create_policy(
            PolicyName=name,
            Description=description,
            PolicyDocument=json.dumps(policy_doc),
        )
        logger.info("Created policy %s.", policy.arn)
    except ClientError:
        logger.exception("Couldn't create policy %s.", name)
        raise
    else:
        return policy



def list_policies(scope):
    """
    Lists the policies in the current account.

    :param scope: Limits the kinds of policies that are returned. For example,
                  'Local' specifies that only locally managed policies are
 returned.
    :return: The list of policies.
    """
    try:
        policies = list(iam.policies.filter(Scope=scope))
        logger.info("Got %s policies in scope '%s'.", len(policies), scope)
    except ClientError:
        logger.exception("Couldn't get policies for scope '%s'.", scope)
        raise
    else:
```

```
        return policies



def create_policy_version(policy_arn, actions, resource_arn, set_as_default):
    """
    Creates a policy version. Policies can have up to five versions. The default
    version is the one that is used for all resources that reference the policy.

    :param policy_arn: The ARN of the policy.
    :param actions: The actions to allow in the policy version.
    :param resource_arn: The ARN of the resource this policy version applies to.
    :param set_as_default: When True, this policy version is set as the default
                           version for the policy. Otherwise, the default
                           is not changed.
    :return: The newly created policy version.
    """
    policy_doc = {
        "Version": "2012-10-17",
        "Statement": [{"Effect": "Allow", "Action": actions, "Resource":
 resource_arn}],
    }
    try:
        policy = iam.Policy(policy_arn)
        policy_version = policy.create_version(
            PolicyDocument=json.dumps(policy_doc), SetAsDefault=set_as_default
        )
        logger.info(
            "Created policy version %s for policy %s.",
            policy_version.version_id,
            policy_version.arn,
        )
    except ClientError:
        logger.exception("Couldn't create a policy version for %s.", policy_arn)
        raise
    else:
        return policy_version



def get_default_policy_statement(policy_arn):
    """
    Gets the statement of the default version of the specified policy.
```

```
    :param policy_arn: The ARN of the policy to look up.
    :return: The statement of the default policy version.
    """
    try:
        policy = iam.Policy(policy_arn)
        # To get an attribute of a policy, the SDK first calls get_policy.
        policy_doc = policy.default_version.document
        policy_statement = policy_doc.get("Statement", None)
        logger.info("Got default policy doc for %s.", policy.policy_name)
        logger.info(policy_doc)
    except ClientError:
        logger.exception("Couldn't get default policy statement for %s.",
 policy_arn)
        raise
    else:
        return policy_statement



def rollback_policy_version(policy_arn):
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
```

```
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
 rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
 default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
 so "
                "nothing to roll back to.",
                default_version.version_id,
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version



def delete_policy(policy_arn):
    """
    Deletes a policy.

    :param policy_arn: The ARN of the policy to delete.
    """
    try:
        iam.Policy(policy_arn).delete()
        logger.info("Deleted policy %s.", policy_arn)
    except ClientError:
        logger.exception("Couldn't delete policy %s.", policy_arn)
        raise
```

Utilice las funciones del contenedor para crear políticas, actualizar versiones y obtener información sobre ellas.

```
def usage_demo():
    """Shows how to use the policy functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management policy demo.")
    print("-" * 88)
    print(
        "Policies let you define sets of permissions that can be attached to "
        "other IAM resources, like users and roles."
    )
    bucket_arn = f"arn:aws:s3:::made-up-bucket-name"
    policy = create_policy(
        "demo-iam-policy",
        "Policy for IAM demonstration.",
        ["s3:ListObjects"],
        bucket_arn,
    )
    print(f"Created policy {policy.policy_name}.")
    policies = list_policies("Local")
    print(f"Your account has {len(policies)} managed policies:")
    print(*[pol.policy_name for pol in policies], sep=", ")
    time.sleep(1)
    policy_version = create_policy_version(
        policy.arn, ["s3:PutObject"], bucket_arn, True
    )
    print(
        f"Added policy version {policy_version.version_id} to policy "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
    print(f"The default policy statement for {policy.policy_name} is:")
    pprint.pprint(default_statement)
    rollback_version = rollback_policy_version(policy.arn)
    print(
        f"Rolled back to version {rollback_version.version_id} for "
        f"{policy.policy_name}."
    )
    default_statement = get_default_policy_statement(policy.arn)
```

```
    print(f"The default policy statement for {policy.policy_name} is now:")
    pprint.pprint(default_statement)
    delete_policy(policy.arn)
    print(f"Deleted policy {policy.policy_name}.")
    print("Thanks for watching!")
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - CreatePolicy
  - CreatePolicyVersion
  - DeletePolicy
  - DeletePolicyVersion
  - GetPolicyVersion
  - ListPolicies
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Administrar roles de IAM con un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Crear un rol de IAM.
- Adjuntar y separar políticas de los roles.
- Eliminar un rol.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones del rol de IAM.

```
import json
import logging
import pprint

import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def create_role(role_name, allowed_services):
    """
    Creates a role that lets a list of specified services assume the role.

    :param role_name: The name of the role.
    :param allowed_services: The services that can assume the role.
    :return: The newly created role.
    """
    trust_policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {"Service": service},
                "Action": "sts:AssumeRole",
            }
            for service in allowed_services
        ],
    }

    try:
```

```
        role = iam.create_role(
            RoleName=role_name, AssumeRolePolicyDocument=json.dumps(trust_policy)
        )
        logger.info("Created role %s.", role.name)
    except ClientError:
        logger.exception("Couldn't create role %s.", role_name)
        raise
    else:
        return role



def attach_policy(role_name, policy_arn):
    """
    Attaches a policy to a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).attach_policy(PolicyArn=policy_arn)
        logger.info("Attached policy %s to role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception("Couldn't attach policy %s to role %s.", policy_arn,
 role_name)
        raise



def detach_policy(role_name, policy_arn):
    """
    Detaches a policy from a role.

    :param role_name: The name of the role. **Note** this is the name, not the
 ARN.
    :param policy_arn: The ARN of the policy.
    """
    try:
        iam.Role(role_name).detach_policy(PolicyArn=policy_arn)
        logger.info("Detached policy %s from role %s.", policy_arn, role_name)
    except ClientError:
        logger.exception(
            "Couldn't detach policy %s from role %s.", policy_arn, role_name
```

```
        )
        raise



def delete_role(role_name):
    """
    Deletes a role.

    :param role_name: The name of the role to delete.
    """
    try:
        iam.Role(role_name).delete()
        logger.info("Deleted role %s.", role_name)
    except ClientError:
        logger.exception("Couldn't delete role %s.", role_name)
        raise
```

Utilice las funciones del contenedor para crear un rol, y luego asociar y desasociar una política.

```
def usage_demo():
    """Shows how to use the role functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management role demo.")
    print("-" * 88)
    print(
        "Roles let you define sets of permissions and can be assumed by "
        "other entities, like users and services."
    )
    print("The first 10 roles currently in your account are:")
    roles = list_roles(10)
    print(f"The inline policies for role {roles[0].name} are:")
    list_policies(roles[0].name)
    role = create_role(
        "demo-iam-role", ["lambda.amazonaws.com",
 "batchoperations.s3.amazonaws.com"]
    )
    print(f"Created role {role.name}, with trust policy:")
```

```
    pprint.pprint(role.assume_role_policy_document)
    policy_arn = "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess"
    attach_policy(role.name, policy_arn)
    print(f"Attached policy {policy_arn} to {role.name}.")
    print(f"Policies attached to role {role.name} are:")
    list_attached_policies(role.name)
    detach_policy(role.name, policy_arn)
    print(f"Detached policy {policy_arn} from {role.name}.")
    delete_role(role.name)
    print(f"Deleted {role.name}.")
    print("Thanks for watching!")
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - AttachRolePolicy
  - CreateRole
  - DeleteRole
  - DetachRolePolicy

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Administrar la cuenta de IAM con un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Obtenga y actualice el alias de la cuenta.
- Genere un informe de los usuarios y las credenciales.
- Obtener un resumen del uso de la cuenta.
- Obtenga información sobre todos los usuarios, grupos, roles y las políticas en su cuenta, incluidas sus relaciones entre sí.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree funciones que encapsulen las acciones de la cuenta de IAM.

```
import logging
import pprint
import sys
import time
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
iam = boto3.resource("iam")

def list_aliases():
    """
    Gets the list of aliases for the current account. An account has at most one
 alias.

    :return: The list of aliases for the account.
    """
    try:
        response = iam.meta.client.list_account_aliases()
        aliases = response["AccountAliases"]
        if len(aliases) > 0:
            logger.info("Got aliases for your account: %s.", ",".join(aliases))
        else:
            logger.info("Got no aliases for your account.")
    except ClientError:
        logger.exception("Couldn't list aliases for your account.")
        raise
    else:
        return response["AccountAliases"]
```

```
def create_alias(alias):
    """
    Creates an alias for the current account. The alias can be used in place of
 the
    account ID in the sign-in URL. An account can have only one alias. When a new
    alias is created, it replaces any existing alias.

    :param alias: The alias to assign to the account.
    """

    try:
        iam.create_account_alias(AccountAlias=alias)
        logger.info("Created an alias '%s' for your account.", alias)
    except ClientError:
        logger.exception("Couldn't create alias '%s' for your account.", alias)
        raise



def delete_alias(alias):
    """
    Removes the alias from the current account.

    :param alias: The alias to remove.
    """
    try:
        iam.meta.client.delete_account_alias(AccountAlias=alias)
        logger.info("Removed alias '%s' from your account.", alias)
    except ClientError:
        logger.exception("Couldn't remove alias '%s' from your account.", alias)
        raise



def generate_credential_report():
    """
    Starts generation of a credentials report about the current account. After
    calling this function to generate the report, call get_credential_report
    to get the latest report. A new report can be generated a minimum of four
 hours
    after the last one was generated.
    """
    try:
```

```
        response = iam.meta.client.generate_credential_report()
        logger.info(
            "Generating credentials report for your account. " "Current state is
 %s.",
            response["State"],
        )
    except ClientError:
        logger.exception("Couldn't generate a credentials report for your
 account.")
        raise
    else:
        return response



def get_credential_report():
    """
    Gets the most recently generated credentials report about the current
 account.

    :return: The credentials report.
    """
    try:
        response = iam.meta.client.get_credential_report()
        logger.debug(response["Content"])
    except ClientError:
        logger.exception("Couldn't get credentials report.")
        raise
    else:
        return response["Content"]



def get_summary():
    """
    Gets a summary of account usage.

    :return: The summary of account usage.
    """
    try:
        summary = iam.AccountSummary()
        logger.debug(summary.summary_map)
    except ClientError:
        logger.exception("Couldn't get a summary for your account.")
```

```
        raise
    else:
        return summary.summary_map



def get_authorization_details(response_filter):
    """
    Gets an authorization detail report for the current account.

    :param response_filter: A list of resource types to include in the report,
 such
                            as users or roles. When not specified, all resources
                            are included.
    :return: The authorization detail report.
    """
    try:
        account_details = iam.meta.client.get_account_authorization_details(
            Filter=response_filter
        )
        logger.debug(account_details)
    except ClientError:
        logger.exception("Couldn't get details for your account.")
        raise
    else:
        return account_details
```

Llame a las funciones del contenedor para cambiar el alias de la cuenta y obtener informes sobre la cuenta.

```
def usage_demo():
    """Shows how to use the account functions."""
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")
    print("-" * 88)
    print("Welcome to the AWS Identity and Account Management account demo.")
    print("-" * 88)
    print(
        "Setting an account alias lets you use the alias in your sign-in URL "
        "instead of your account number."
    )
```

```
    old_aliases = list_aliases()
    if len(old_aliases) > 0:
        print(f"Your account currently uses '{old_aliases[0]}' as its alias.")
    else:
        print("Your account currently has no alias.")
    for index in range(1, 3):
        new_alias = f"alias-{index}-{time.time_ns()}"
        print(f"Setting your account alias to {new_alias}")
        create_alias(new_alias)
    current_aliases = list_aliases()
    print(f"Your account alias is now {current_aliases}.")
    delete_alias(current_aliases[0])
    print(f"Your account now has no alias.")
    if len(old_aliases) > 0:
        print(f"Restoring your original alias back to {old_aliases[0]}...")
        create_alias(old_aliases[0])

    print("-" * 88)
    print("You can get various reports about your account.")
    print("Let's generate a credentials report...")
    report_state = None
    while report_state != "COMPLETE":
        cred_report_response = generate_credential_report()
        old_report_state = report_state
        report_state = cred_report_response["State"]
        if report_state != old_report_state:
            print(report_state, sep="")
        else:
            print(".", sep="")
        sys.stdout.flush()
        time.sleep(1)
    print()
    cred_report = get_credential_report()
    col_count = 3
    print(f"Got credentials report. Showing only the first {col_count} columns.")
    cred_lines = [
        line.split(",")[:col_count] for line in
 cred_report.decode("utf-8").split("\n")
    ]
    col_width = max([len(item) for line in cred_lines for item in line]) + 2
    for line in cred_report.decode("utf-8").split("\n"):
        print(
            "".join(element.ljust(col_width) for element in line.split(",")
[:col_count])
```

```
        )

    print("-" * 88)
    print("Let's get an account summary.")
    summary = get_summary()
    print("Here's your summary:")
    pprint.pprint(summary)

    print("-" * 88)
    print("Let's get authorization details!")
    details = get_authorization_details([])
    see_details = input("These are pretty long, do you want to see them (y/n)? ")
    if see_details.lower() == "y":
        pprint.pprint(details)

    print("-" * 88)
    pw_policy_created = None
    see_pw_policy = input("Want to see the password policy for the account (y/n)?
")
    if see_pw_policy.lower() == "y":
        while True:
            if print_password_policy():
                break
            else:
                answer = input(
                    "Do you want to create a default password policy (y/n)? "
                )
                if answer.lower() == "y":
                    pw_policy_created = iam.create_account_password_policy()
                else:
                    break
    if pw_policy_created is not None:
        answer = input("Do you want to delete the password policy (y/n)? ")
        if answer.lower() == "y":
            pw_policy_created.delete()
            print("Password policy deleted.")

    print("The SAML providers for your account are:")
    list_saml_providers(10)

    print("-" * 88)
    print("Thanks for watching.")
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - CreateAccountAlias
  - DeleteAccountAlias
  - GenerateCredentialReport
  - GetAccountAuthorizationDetails
  - GetAccountSummary
  - GetCredentialReport
  - ListAccountAliases

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Revertir una versión de la política de IAM con un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Obtener la lista de versiones de la política en orden por fecha.
- Buscar la versión predeterminada de la política.
- Hacer que la versión anterior de la política sea la predeterminada.
- Eliminar la versión anterior predeterminada.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
def rollback_policy_version(policy_arn):
```

```
    """
    Rolls back to the previous default policy, if it exists.

    1. Gets the list of policy versions in order by date.
    2. Finds the default.
    3. Makes the previous policy the default.
    4. Deletes the old default version.

    :param policy_arn: The ARN of the policy to roll back.
    :return: The default version of the policy after the rollback.
    """
    try:
        policy_versions = sorted(
            iam.Policy(policy_arn).versions.all(),
            key=operator.attrgetter("create_date"),
        )
        logger.info("Got %s versions for %s.", len(policy_versions), policy_arn)
    except ClientError:
        logger.exception("Couldn't get versions for %s.", policy_arn)
        raise

    default_version = None
    rollback_version = None
    try:
        while default_version is None:
            ver = policy_versions.pop()
            if ver.is_default_version:
                default_version = ver
        rollback_version = policy_versions.pop()
        rollback_version.set_as_default()
        logger.info("Set %s as the default version.",
rollback_version.version_id)
        default_version.delete()
        logger.info("Deleted original default version %s.",
default_version.version_id)
    except IndexError:
        if default_version is None:
            logger.warning("No default version found for %s.", policy_arn)
        elif rollback_version is None:
            logger.warning(
                "Default version %s found for %s, but no previous version exists,
so "
                "nothing to roll back to.",
                default_version.version_id,
```

```
                policy_arn,
            )
    except ClientError:
        logger.exception("Couldn't roll back version for %s.", policy_arn)
        raise
    else:
        return rollback_version
```

- Para obtener información sobre la API, consulte los siguientes temas en la Referencia de la API del SDK de AWS para Python (Boto3).
  - DeletePolicyVersion
  - ListPolicyVersions
  - SetDefaultPolicyVersion

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Trabajar con la API del creador de políticas de IAM mediante un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Cree políticas de IAM mediante la API orientada a objetos.
- Utilice la API del creador de políticas de IAM con el servicio de IAM.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

En los ejemplos se utilizan las siguientes importaciones.

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import software.amazon.awssdk.policybuilder.iam.IamConditionOperator;
import software.amazon.awssdk.policybuilder.iam.IamEffect;
import software.amazon.awssdk.policybuilder.iam.IamPolicy;
import software.amazon.awssdk.policybuilder.iam.IamPolicyWriter;
import software.amazon.awssdk.policybuilder.iam.IamPrincipal;
import software.amazon.awssdk.policybuilder.iam.IamPrincipalType;
import software.amazon.awssdk.policybuilder.iam.IamResource;
import software.amazon.awssdk.policybuilder.iam.IamStatement;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.iam.IamClient;
import software.amazon.awssdk.services.iam.model.GetPolicyResponse;
import software.amazon.awssdk.services.iam.model.GetPolicyVersionResponse;
import software.amazon.awssdk.services.sts.StsClient;

import java.net.URLDecoder;
import java.nio.charset.StandardCharsets;
import java.util.Arrays;
import java.util.List;
```

Cree una política basada en el tiempo.

```
        public String timeBasedPolicyExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")
                                                .addResource(IamResource.ALL)
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_GREATER_THAN)

 .key("aws:CurrentTime")

 .value("2020-04-01T00:00:00Z"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.DATE_LESS_THAN)

 .key("aws:CurrentTime")
```

```
 .value("2020-06-30T23:59:59Z")))
                                .build();

                // Use an IamPolicyWriter to write out the JSON string to a more
 readable
                // format.
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true)
                                .build());
        }
```

Cree una política con varias condiciones.

```
        public String multipleConditionsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:GetItem")

 .addAction("dynamodb:BatchGetItem")
                                                .addAction("dynamodb:Query")
                                                .addAction("dynamodb:PutItem")
                                                .addAction("dynamodb:UpdateItem")
                                                .addAction("dynamodb:DeleteItem")

 .addAction("dynamodb:BatchWriteItem")

 .addResource("arn:aws:dynamodb:*:*:table/table-name")

 .addConditions(IamConditionOperator.STRING_EQUALS

 .addPrefix("ForAllValues:"),

 "dynamodb:Attributes",
                                                                        List.of("column-
name1", "column-name2", "column-name3"))
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS

 .addSuffix("IfExists"))
```

```
.key("dynamodb:Select")

.value("SPECIFIC_ATTRIBUTES")))
                                .build();

                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Utilice las entidades principales en una política.

```
        public String specifyPrincipalsExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.DENY)
                                                .addAction("s3:*")
                                                .addPrincipal(IamPrincipal.ALL)

.addResource("arn:aws:s3:::BUCKETNAME/*")

.addResource("arn:aws:s3:::BUCKETNAME")
                                                .addCondition(b1 -> b1

.operator(IamConditionOperator.ARN_NOT_EQUALS)

.key("aws:PrincipalArn")

.value("arn:aws:iam::444455556666:user/user-name")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Permitir el acceso entre cuentas de.

```
        public String allowCrossAccountAccessExample() {
                IamPolicy policy = IamPolicy.builder()
                                .addStatement(b -> b
                                                .effect(IamEffect.ALLOW)
```

```
 .addPrincipal(IamPrincipalType.AWS, "111122223333")
                                                .addAction("s3:PutObject")
                                                .addResource("arn:aws:s3:::DOC-
EXAMPLE-BUCKET/*")
                                                .addCondition(b1 -> b1

 .operator(IamConditionOperator.STRING_EQUALS)
                                                                    .key("s3:x-amz-
acl")
                                                                    .value("bucket-
owner-full-control")))
                                .build();
                return policy.toJson(IamPolicyWriter.builder()
                                .prettyPrint(true).build());
        }
```

Cree y cargue una `IamPolicy`.

```
        public String createAndUploadPolicyExample(IamClient iam, String
 accountID, String policyName) {
                // Build the policy.
                IamPolicy policy = IamPolicy.builder() // 'version' defaults to
 "2012-10-17".
                                .addStatement(IamStatement.builder()
                                                .effect(IamEffect.ALLOW)
                                                .addAction("dynamodb:PutItem")

 .addResource("arn:aws:dynamodb:us-east-1:" + accountID
                                                                    + ":table/
exampleTableName")
                                                .build())
                                .build();
                // Upload the policy.
                iam.createPolicy(r ->
 r.policyName(policyName).policyDocument(policy.toJson()));
                return
 policy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

Descargue y trabaje con una `IamPolicy`.

```
       public String createNewBasedOnExistingPolicyExample(IamClient iam, String
accountID, String policyName,
                       String newPolicyName) {

               String policyArn = "arn:aws:iam::" + accountID + ":policy/" +
policyName;
               GetPolicyResponse getPolicyResponse = iam.getPolicy(r ->
r.policyArn(policyArn));

               String policyVersion =
getPolicyResponse.policy().defaultVersionId();
               GetPolicyVersionResponse getPolicyVersionResponse = iam
                               .getPolicyVersion(r ->
r.policyArn(policyArn).versionId(policyVersion));

               // Create an IamPolicy instance from the JSON string returned
from IAM.
               String decodedPolicy =
URLDecoder.decode(getPolicyVersionResponse.policyVersion().document(),
                               StandardCharsets.UTF_8);
               IamPolicy policy = IamPolicy.fromJson(decodedPolicy);

               /*
                * All IamPolicy components are immutable, so use the copy method
that creates a
                * new instance that
                * can be altered in the same method call.
                *
                * Add the ability to get an item from DynamoDB as an additional
action.
                */
               IamStatement newStatement = policy.statements().get(0).copy(s ->
s.addAction("dynamodb:GetItem"));

               // Create a new statement that replaces the original statement.
               IamPolicy newPolicy = policy.copy(p ->
p.statements(Arrays.asList(newStatement)));

               // Upload the new policy. IAM now has both policies.
               iam.createPolicy(r -> r.policyName(newPolicyName)
                               .policyDocument(newPolicy.toJson()));
```

```
                return
newPolicy.toJson(IamPolicyWriter.builder().prettyPrint(true).build());
        }
```

- Para obtener información, consulte la Guía para desarrolladores de AWS SDK for Java 2.x.
- Para obtener información sobre la API, consulte los siguientes temas en la referencia de la API de AWS SDK for Java 2.x.
  - CreatePolicy
  - GetPolicy
  - GetPolicyVersion

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código de AWS STS con SDK de AWS

Los siguientes ejemplos de código muestran cómo utilizar AWS STS con un kit de desarrollo de software (SDK) de AWS.

Las acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Mientras las acciones muestran cómo llamar a las funciones de servicio individuales, es posible ver las acciones en contexto en los escenarios relacionados y en los ejemplos entre servicios.

Los escenarios son ejemplos de código que muestran cómo llevar a cabo una tarea específica llamando a varias funciones dentro del mismo servicio.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Ejemplos de código

- Acciones de AWS STS con SDK de AWS
  - Asumir un rol con AWS STS con un SDK de AWS
  - Obtener un token de sesión con AWS STS con un SDK de AWS
- Situaciones de AWS STS con SDK de AWS

- Asumir un rol de IAM que requiera un token MFA con AWS STS con un SDK de AWS
- Crear una URL con AWS STS para usuarios federados que utilizan un SDK de AWS
- Obtener un token de sesión que requiera un token MFA con AWS STS con un SDK de AWS

Acciones de AWS STS con SDK de AWS

Los siguientes ejemplos de código muestran cómo llevar a cabo acciones individuales de AWS STS con SDK de AWS. Estos fragmentos llaman a la API de AWS STS y son fragmentos de código de programas más grandes que deben ejecutarse en contexto. En cada ejemplo se incluye un enlace a GitHub, con instrucciones de configuración y ejecución del código.

Los ejemplos siguientes incluyen solo las acciones que se utilizan con mayor frecuencia. Para ver una lista completa, consulte la Referencia de la API de AWS Security Token Service (AWS STS).

Ejemplos

- Asumir un rol con AWS STS con un SDK de AWS
- Obtener un token de sesión con AWS STS con un SDK de AWS

Asumir un rol con AWS STS con un SDK de AWS

Los siguientes ejemplos de código muestran cómo asumir un rol con AWS STS.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en su contexto en los siguientes ejemplos de código:

- Asumir un rol de IAM que requiera un token MFA
- Crear una URL para usuarios federados

.NET

AWS SDK for .NET

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
using System;
using System.Threading.Tasks;
using Amazon;
using Amazon.SecurityToken;
using Amazon.SecurityToken.Model;

namespace AssumeRoleExample
{
    class AssumeRole
    {
        /// <summary>
        /// This example shows how to use the AWS Security Token
        /// Service (AWS STS) to assume an IAM role.
        ///
        /// NOTE: It is important that the role that will be assumed has a
        /// trust relationship with the account that will assume the role.
        ///
        /// Before you run the example, you need to create the role you want to
        /// assume and have it trust the IAM account that will assume that role.
        ///
        /// See https://docs.aws.amazon.com/IAM/latest/UserGuide/
id_roles_create.html
        /// for help in working with roles.
        /// </summary>

        private static readonly RegionEndpoint REGION = RegionEndpoint.USWest2;

        static async Task Main()
        {
            // Create the SecurityToken client and then display the identity of
 the
            // default user.
            var roleArnToAssume = "arn:aws:iam::123456789012:role/
testAssumeRole";

            var client = new
 Amazon.SecurityToken.AmazonSecurityTokenServiceClient(REGION);

            // Get and display the information about the identity of the default
 user.
            var callerIdRequest = new GetCallerIdentityRequest();
            var caller = await client.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"Original Caller: {caller.Arn}");
```

```
            // Create the request to use with the AssumeRoleAsync call.
            var assumeRoleReq = new AssumeRoleRequest()
            {
                DurationSeconds = 1600,
                RoleSessionName = "Session1",
                RoleArn = roleArnToAssume
            };

            var assumeRoleRes = await client.AssumeRoleAsync(assumeRoleReq);

            // Now create a new client based on the credentials of the caller
 assuming the role.
            var client2 = new AmazonSecurityTokenServiceClient(credentials:
 assumeRoleRes.Credentials);

            // Get and display information about the caller that has assumed the
 defined role.
            var caller2 = await client2.GetCallerIdentityAsync(callerIdRequest);
            Console.WriteLine($"AssumedRole Caller: {caller2.Arn}");
        }
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for .NET.

Bash

AWS CLI con script Bash

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
###############################################################################
# function iecho
#
```

```
# This function enables the script to display the specified text only if
# the global variable $VERBOSE is set to true.
###############################################################################
function iecho() {
  if [[ $VERBOSE == true ]]; then
    echo "$@"
  fi
}

###############################################################################
# function errecho
#
# This function outputs everything sent to it to STDERR (standard error output).
###############################################################################
function errecho() {
  printf "%s\n" "$*" 1>&2
}

###############################################################################
# function sts_assume_role
#
# This function assumes a role in the AWS account and returns the temporary
#  credentials.
#
# Parameters:
#       -n role_session_name -- The name of the session.
#       -r role_arn -- The ARN of the role to assume.
#
# Returns:
#       [access_key_id, secret_access_key, session_token]
#     And:
#       0 - If successful.
#       1 - If an error occurred.
###############################################################################
function sts_assume_role() {
  local role_session_name role_arn response
  local option OPTARG # Required to use getopts command in a function.

  # bashsupport disable=BP5008
  function usage() {
    echo "function sts_assume_role"
    echo "Assumes a role in the AWS account and returns the temporary
 credentials:"
    echo "  -n role_session_name -- The name of the session."
```

```
    echo "  -r role_arn -- The ARN of the role to assume."
    echo ""
  }

  while getopts n:r:h option; do
    case "${option}" in
      n) role_session_name=${OPTARG} ;;
      r) role_arn=${OPTARG} ;;
      h)
        usage
        return 0
        ;;
      \?)
        ech o"Invalid parameter"
        usage
        return 1
        ;;
    esac
  done

  response=$(aws sts assume-role \
    --role-session-name "$role_session_name" \
    --role-arn "$role_arn" \
    --output text \
    --query "Credentials.[AccessKeyId, SecretAccessKey, SessionToken]")

  local error_code=${?}

  if [[ $error_code -ne 0 ]]; then
    aws_cli_error_log $error_code
    errecho "ERROR: AWS reports create-role operation failed.\n$response"
    return 1
  fi

  echo "$response"

  return 0
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de comandos de la AWS CLI.

C++

SDK para C++

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
bool AwsDoc::STS::assumeRole(const Aws::String &roleArn,
                             const Aws::String &roleSessionName,
                             const Aws::String &externalId,
                             Aws::Auth::AWSCredentials &credentials,
                             const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::STS::STSClient sts(clientConfig);
    Aws::STS::Model::AssumeRoleRequest sts_req;

    sts_req.SetRoleArn(roleArn);
    sts_req.SetRoleSessionName(roleSessionName);
    sts_req.SetExternalId(externalId);

    const Aws::STS::Model::AssumeRoleOutcome outcome = sts.AssumeRole(sts_req);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error assuming IAM role. " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Credentials successfully retrieved." << std::endl;
        const Aws::STS::Model::AssumeRoleResult result = outcome.GetResult();
        const Aws::STS::Model::Credentials &temp_credentials =
 result.GetCredentials();

        // Store temporary credentials in return argument.
        // Note: The credentials object returned by assumeRole differs
        // from the AWSCredentials object used in most situations.
        credentials.SetAWSAccessKeyId(temp_credentials.GetAccessKeyId());
        credentials.SetAWSSecretKey(temp_credentials.GetSecretAccessKey());
        credentials.SetSessionToken(temp_credentials.GetSessionToken());
    }
```

```
    return outcome.IsSuccess();
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for C++.

CLI

AWS CLI

Cómo asumir un rol

El siguiente comando `assume-role` recupera un conjunto de credenciales a corto plazo para el rol de IAM `s3-access-example`.

```
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/xaccounts3access \
    --role-session-name s3-access-example
```

Salida:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:s3-access-example",
        "Arn": "arn:aws:sts::123456789012:assumed-role/xaccounts3access/s3-
access-example"
    },
    "Credentials": {
        "SecretAccessKey": "9drTJvcXLB89EXAMPLELB8923FB892xMFI",
        "SessionToken": "AQoXdzELDDY//////////
wEaoAK1wvxJY12r2IrDFT2IvAzTCn3zHoZ7YNtpiQLF0MqZye/
qwjzP2iEXAMPLEbw/m3hsj8VBTkPORGvr9jM5sgP+w9IZWZnU+LWhmg
+a5fDi2oTGUYcdg9uexQ4mtCHIHfi4citgqZTgco40Yqr4lIlo4V2b2Dyauk0eYFNebHtYlFVgAUj
+7Indz3LU0aTWk1WKIjHmmMCIoTkyYp/k7kUG7moeEYKSitwQIi6Gjn+nyzM
+PtoA3685ixzv0R7i5rjQi0YE0lf1oeie3bDiNHncmzosRM6SFiPzSvp6h/32xQuZsjcypmwsPSDtTPYcs0+YN/8B
IcrxSpnWEXAMPLEXSDFTAQAM6Dl9zR0tXoybnlrZIwMLlMi1Kcgo5OytwU=",
        "Expiration": "2016-03-15T00:05:07Z",
        "AccessKeyId": "ASIAJEXAMPLEXEG2JICEA"
    }
}
```

El resultado del comando contiene una clave de acceso, una clave secreta y un token de sesión que puede utilizar para autenticarse con AWS.

Para el uso de la CLI de AWS, puede configurar un perfil con nombre asociado a un rol. Cuando utilice el perfil, la CLI de AWS llamará a assume-role y administrará las credenciales por usted. Para obtener más información, consulte Uso de un rol de IAM en la CLI de AWS en la Guía del usuario de la CLI de AWS.

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de comandos de la AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.sts.StsClient;
import software.amazon.awssdk.services.sts.model.AssumeRoleRequest;
import software.amazon.awssdk.services.sts.model.StsException;
import software.amazon.awssdk.services.sts.model.AssumeRoleResponse;
import software.amazon.awssdk.services.sts.model.Credentials;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * To make this code example work, create a Role that you want to assume.
 * Then define a Trust Relationship in the AWS Console. You can use this as an
 * example:
 *
 * {
 * "Version": "2012-10-17",
 * "Statement": [
```

```
 * {
 * "Effect": "Allow",
 * "Principal": {
 * "AWS": "<Specify the ARN of your IAM user you are using in this code
 * example>"
 * },
 * "Action": "sts:AssumeRole"
 * }
 * ]
 * }
 *
 * For more information, see "Editing the Trust Relationship for an Existing
 * Role" in the AWS Directory Service guide.
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class AssumeRole {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <roleArn> <roleSessionName>\s

                Where:
                    roleArn - The Amazon Resource Name (ARN) of the role to
 assume (for example, rn:aws:iam::000008047983:role/s3role).\s
                    roleSessionName - An identifier for the assumed role session
 (for example, mysession).\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String roleArn = args[0];
        String roleSessionName = args[1];
        Region region = Region.US_EAST_1;
        StsClient stsClient = StsClient.builder()
```

```
                .region(region)
                .build();

        assumeGivenRole(stsClient, roleArn, roleSessionName);
        stsClient.close();
    }

    public static void assumeGivenRole(StsClient stsClient, String roleArn,
 String roleSessionName) {
        try {
            AssumeRoleRequest roleRequest = AssumeRoleRequest.builder()
                    .roleArn(roleArn)
                    .roleSessionName(roleSessionName)
                    .build();

            AssumeRoleResponse roleResponse = stsClient.assumeRole(roleRequest);
            Credentials myCreds = roleResponse.credentials();

            // Display the time when the temp creds expire.
            Instant exTime = myCreds.expiration();
            String tokenInfo = myCreds.sessionToken();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(exTime);
            System.out.println("The token " + tokenInfo + "  expires on " +
 exTime);

        } catch (StsException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for Java 2.x.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree el cliente.

```
import { STSClient } from "@aws-sdk/client-sts";
// Set the AWS Region.
const REGION = "us-east-1";
// Create an AWS STS service client object.
export const client = new STSClient({ region: REGION });
```

Asuma un rol de IAM.

```
import { AssumeRoleCommand } from "@aws-sdk/client-sts";

import { client } from "../libs/client.js";

export const main = async () => {
  try {
    // Returns a set of temporary security credentials that you can use to
    // access Amazon Web Services resources that you might not normally
    // have access to.
    const command = new AssumeRoleCommand({
      // The Amazon Resource Name (ARN) of the role to assume.
      RoleArn: "ROLE_ARN",
      // An identifier for the assumed role session.
      RoleSessionName: "session1",
      // The duration, in seconds, of the role session. The value specified
      // can range from 900 seconds (15 minutes) up to the maximum session
      // duration set for the role.
      DurationSeconds: 900,
    });
    const response = await client.send(command);
    console.log(response);
```

```
  } catch (err) {
    console.error(err);
  }
};
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for JavaScript.

SDK para JavaScript (v2)

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
// Load the AWS SDK for Node.js
const AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

var roleToAssume = {
  RoleArn: "arn:aws:iam::123456789012:role/RoleName",
  RoleSessionName: "session1",
  DurationSeconds: 900,
};
var roleCreds;

// Create the STS service object
var sts = new AWS.STS({ apiVersion: "2011-06-15" });

//Assume Role
sts.assumeRole(roleToAssume, function (err, data) {
  if (err) console.log(err, err.stack);
  else {
    roleCreds = {
      accessKeyId: data.Credentials.AccessKeyId,
      secretAccessKey: data.Credentials.SecretAccessKey,
      sessionToken: data.Credentials.SessionToken,
    };
    stsGetCallerIdentity(roleCreds);
  }
```

```
});

//Get Arn of current identity
function stsGetCallerIdentity(creds) {
  var stsParams = { credentials: creds };
  // Create STS service object
  var sts = new AWS.STS(stsParams);

  sts.getCallerIdentity({}, function (err, data) {
    if (err) {
      console.log(err, err.stack);
    } else {
      console.log(data.Arn);
    }
  });
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for JavaScript.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Asuma un rol de IAM que requiera un token MFA y utilice credenciales temporales para enumerar los buckets de Amazon S3 para la cuenta.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.
```

```
   The assumed role must grant permission to list the buckets in the other
account.

   :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                           grants access to list the other account's buckets.
   :param session_name: The name of the STS session.
   :param mfa_serial_number: The serial number of the MFA device. For a virtual
MFA
                             device, this is an ARN.
   :param mfa_totp: A time-based, one-time password issued by the MFA device.
   :param sts_client: A Boto3 STS instance that has permission to assume the
role.
   """
   response = sts_client.assume_role(
       RoleArn=assume_role_arn,
       RoleSessionName=session_name,
       SerialNumber=mfa_serial_number,
       TokenCode=mfa_totp,
   )
   temp_credentials = response["Credentials"]
   print(f"Assumed role {assume_role_arn} and got temporary credentials.")

   s3_resource = boto3.resource(
       "s3",
       aws_access_key_id=temp_credentials["AccessKeyId"],
       aws_secret_access_key=temp_credentials["SecretAccessKey"],
       aws_session_token=temp_credentials["SessionToken"],
   )

   print(f"Listing buckets for the assumed role's account:")
   for bucket in s3_resource.buckets.all():
       print(bucket.name)
```

- Para obtener detalles sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
 # Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
 # This is separated into a factory function so that it can be mocked for unit
testing.
 #
 # @param key_id [String] The ID of the access key used by the STS client.
 # @param key_secret [String] The secret part of the access key used by the STS
client.
 def create_sts_client(key_id, key_secret)
   Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
 end

 # Gets temporary credentials that can be used to assume a role.
 #
 # @param role_arn [String] The ARN of the role that is assumed when these
credentials
 #                          are used.
 # @param sts_client [AWS::STS::Client] An AWS STS client.
 # @return [Aws::AssumeRoleCredentials] The credentials that can be used to
assume the role.
 def assume_role(role_arn, sts_client)
   credentials = Aws::AssumeRoleCredentials.new(
     client: sts_client,
     role_arn: role_arn,
     role_session_name: "create-use-assume-role-scenario"
   )
   @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
   credentials
 end
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API de AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Hay más información en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
async fn assume_role(config: &SdkConfig, role_name: String, session_name:
 Option<String>) {
    let provider = aws_config::sts::AssumeRoleProvider::builder(role_name)
        .session_name(session_name.unwrap_or("rust_sdk_example_session".into()))
        .configure(config)
        .build()
        .await;

    let local_config = aws_config::from_env()
        .credentials_provider(provider)
        .load()
        .await;
    let client = Client::new(&local_config);
    let req = client.get_caller_identity();
    let resp = req.send().await;
    match resp {
        Ok(e) => {
            println!("UserID :               {}",
 e.user_id().unwrap_or_default());
            println!("Account:               {}",
 e.account().unwrap_or_default());
            println!("Arn    :               {}", e.arn().unwrap_or_default());
        }
        Err(e) => println!("{:?}", e),
    }
}
```

- Para obtener información sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Rust.

Swift

SDK para Swift

> **Note**
>
> Esto es documentación preliminar para un SDK en versión preliminar. Está sujeta a cambios.

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

```
public func assumeRole(role: IAMClientTypes.Role, sessionName: String)
                async throws -> STSClientTypes.Credentials {
    let input = AssumeRoleInput(
        roleArn: role.arn,
        roleSessionName: sessionName
    )
    do {
        let output = try await stsClient.assumeRole(input: input)

        guard let credentials = output.credentials else {
            throw ServiceHandlerError.authError
        }

        return credentials
    } catch {
        throw error
    }
}
```

- Para obtener información acerca de la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Swift.

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un token de sesión con AWS STS con un SDK de AWS

En los siguientes ejemplos de código, se muestra cómo obtener un token de sesión con AWS STS y utilizarlo para hacer una acción de servicio que requiere un token MFA.

Los ejemplos de acciones son extractos de código de programas más grandes y deben ejecutarse en contexto. Puede ver esta acción en contexto en el siguiente ejemplo de código:

- Obtener un token de sesión que requiera un token MFA

CLI

AWS CLI

Cómo obtener un conjunto de credenciales a corto plazo para una identidad de IAM

El siguiente comando `get-session-token` recupera un conjunto de credenciales a corto plazo para la identidad de IAM que realiza la llamada. Las credenciales resultantes se pueden utilizar para las solicitudes donde la política requiere la autenticación multifactor (MFA). Las credenciales caducan 15 minutos después de haberse generado.

```
aws sts get-session-token \
    --duration-seconds 900 \
    --serial-number "YourMFADeviceSerialNumber" \
    --token-code 123456
```

Salida:

```
{
    "Credentials": {
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
```

```
        "SessionToken": "AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
        "Expiration": "2020-05-19T18:06:10+00:00"
    }
}
```

Para obtener más información, consulte Solicitud de credenciales de seguridad temporales en la Guía del usuario de IAM de AWS.

- Para obtener información sobre la API, consulte GetSessionToken en la Referencia de comandos de la AWS CLI.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Obtenga un token de sesión pasando un token MFA y utilícelo para enumerar los buckets de Amazon S3 de la cuenta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.

    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
```

```
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

- Para obtener detalles sobre la API, consulte GetSessionToken en la Referencia de la API del SDK de AWS para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Situaciones de AWS STS con SDK de AWS

En los siguientes ejemplos de código se muestra cómo implementar situaciones comunes en AWS STS con SDK de AWS. Estas situaciones muestran cómo llevar a cabo tareas específicas llamando a varias funciones dentro de AWS STS. En cada escenario se incluye un enlace a GitHub, con instrucciones de configuración y ejecución del código.

Ejemplos

- Asumir un rol de IAM que requiera un token MFA con AWS STS con un SDK de AWS
- Crear una URL con AWS STS para usuarios federados que utilizan un SDK de AWS
- Obtener un token de sesión que requiera un token MFA con AWS STS con un SDK de AWS

Asumir un rol de IAM que requiera un token MFA con AWS STS con un SDK de AWS

En el siguiente ejemplo de código se muestra cómo asumir un rol que requiere un token de MFA.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Cree un rol de IAM que otorgue permiso para enumerar los buckets de Amazon S3.
- Cree un usuario de IAM que tenga permiso para asumir el rol solo cuando se proporcionen las credenciales de MFA.
- Registre un dispositivo MFA para el usuario.
- Asuma el rol y enumere los buckets de S3 con credenciales temporales.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM, registre un dispositivo MFA y cree un rol que conceda permiso para enumerar buckets de S3. El usuario solo tiene derechos para asumir el rol.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual MFA device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
```

```
    Creates a role with a policy that lets the user assume the role and requires
MFA.
    Creates a policy that allows listing Amazon S3 buckets.
    Attaches the policy to the role.
    Creates an inline policy for the user that lets the user assume the role.

    For demonstration purposes, the user is created in the same account as the
role,
    but in practice the user would likely be from another account.

    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, roles, and
policies
                         in the account.
    :return: The newly created user, user key, virtual MFA device, and role.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
```

```
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": user.arn},
                        "Action": "sts:AssumeRole",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
True}},
                    }
                ],
            }
        ),
    )
    print(f"Created role {role.name} that requires MFA.")

    policy = iam_resource.create_policy(
        PolicyName=unique_name("policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                    }
                ],
            }
        ),
    )
```

```
    role.attach_policy(PolicyArn=policy.arn)
    print(f"Created policy {policy.policy_name} and attached it to the role.")

    user.create_policy(
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "sts:AssumeRole",
                        "Resource": role.arn,
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user assume "
        f"the role."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device, role
```

Muestre que no se permite asumir el rol sin un token MFA.

```
def try_to_assume_role_without_mfa(assume_role_arn, session_name, sts_client):
    """
    Shows that attempting to assume the role without sending MFA credentials
 results
    in an AccessDenied error.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role to assume.
    :param session_name: The name of the STS session.
```

```
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    print(f"Trying to assume the role without sending MFA credentials...")
    try:
        sts_client.assume_role(RoleArn=assume_role_arn,
 RoleSessionName=session_name)
        raise RuntimeError("Expected AccessDenied error.")
    except ClientError as error:
        if error.response["Error"]["Code"] == "AccessDenied":
            print("Got AccessDenied.")
        else:
            raise
```

Asuma el rol que otorga permiso para enumerar los buckets de S3, pasando el token MFA requerido, y muestre que los buckets se pueden enumerar.

```
def list_buckets_from_assumed_role_with_mfa(
    assume_role_arn, session_name, mfa_serial_number, mfa_totp, sts_client
):
    """
    Assumes a role from another account and uses the temporary credentials from
    that role to list the Amazon S3 buckets that are owned by the other account.
    Requires an MFA device serial number and token.

    The assumed role must grant permission to list the buckets in the other
 account.

    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an ARN.
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn,
```

```
        RoleSessionName=session_name,
        SerialNumber=mfa_serial_number,
        TokenCode=mfa_totp,
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Listing buckets for the assumed role's account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Elimine los recursos creados para la demostración.

```
def teardown(user, virtual_mfa_device, role):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        policy_name = attached.policy_name
        role.detach_policy(PolicyArn=attached.arn)
        attached.delete()
        print(f"Detached and deleted {policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
        print("Deleted user's access key.")
```

```
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Ejecute este escenario mediante las funciones previamente definidas.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device, role = setup(iam_resource)
    print(f"Created {user.name} and {role.name}.")
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try_to_assume_role_without_mfa(role.arn, "demo-sts-session", sts_client)
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_from_assumed_role_with_mfa(
            role.arn,
            "demo-sts-session",
            virtual_mfa_device.serial_number,
            mfa_totp,
            sts_client,
        )
    finally:
        teardown(user, virtual_mfa_device, role)
        print("Thanks for watching!")
```

- Para obtener detalles sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Crear una URL con AWS STS para usuarios federados que utilizan un SDK de AWS

En el siguiente ejemplo de código, se muestra cómo:

- Cree un rol de IAM que conceda acceso de solo lectura a los recursos de Amazon S3 de la cuenta actual.
- Obtenga un token de seguridad del punto de conexión de federación de AWS.
- Cree una URL que pueda utilizarse para acceder a la consola con credenciales federadas.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un rol que otorgue acceso de solo lectura a los recursos de Amazon S3 de la cuenta actual.

```
def setup(iam_resource):
    """
    Creates a role that can be assumed by the current user.
    Attaches a policy that allows only Amazon S3 read-only access.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
 instance
                         that has the permission to create a role.
    :return: The newly created role.
    """
```

```
    role = iam_resource.create_role(
        RoleName=unique_name("role"),
        AssumeRolePolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {"AWS": iam_resource.CurrentUser().arn},
                        "Action": "sts:AssumeRole",
                    }
                ],
            }
        ),
    )
    role.attach_policy(PolicyArn="arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess")
    print(f"Created role {role.name}.")

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return role
```

Obtenga un token de seguridad del punto de conexión de federación de AWS y cree una URL que pueda utilizarse para acceder a la consola con credenciales federadas.

```
def construct_federated_url(assume_role_arn, session_name, issuer, sts_client):
    """
    Constructs a URL that gives federated users direct access to the AWS
 Management
    Console.

    1. Acquires temporary credentials from AWS Security Token Service (AWS STS)
 that
       can be used to assume a role with limited permissions.
    2. Uses the temporary credentials to request a sign-in token from the
       AWS federation endpoint.
```

```
    3. Builds a URL that can be used in a browser to navigate to the AWS
federation
       endpoint, includes the sign-in token for authentication, and redirects to
       the AWS Management Console with permissions defined by the role that was
       specified in step 1.

    :param assume_role_arn: The role that specifies the permissions that are
granted.
                            The current user must have permission to assume the
role.
    :param session_name: The name for the STS session.
    :param issuer: The organization that issues the URL.
    :param sts_client: A Boto3 STS instance that can assume the role.
    :return: The federated URL.
    """
    response = sts_client.assume_role(
        RoleArn=assume_role_arn, RoleSessionName=session_name
    )
    temp_credentials = response["Credentials"]
    print(f"Assumed role {assume_role_arn} and got temporary credentials.")

    session_data = {
        "sessionId": temp_credentials["AccessKeyId"],
        "sessionKey": temp_credentials["SecretAccessKey"],
        "sessionToken": temp_credentials["SessionToken"],
    }
    aws_federated_signin_endpoint = "https://signin.aws.amazon.com/federation"

    # Make a request to the AWS federation endpoint to get a sign-in token.
    # The requests.get function URL-encodes the parameters and builds the query
string
    # before making the request.
    response = requests.get(
        aws_federated_signin_endpoint,
        params={
            "Action": "getSigninToken",
            "SessionDuration": str(datetime.timedelta(hours=12).seconds),
            "Session": json.dumps(session_data),
        },
    )
    signin_token = json.loads(response.text)
    print(f"Got a sign-in token from the AWS sign-in federation endpoint.")
```

```
    # Make a federated URL that can be used to sign into the AWS Management
 Console.
    query_string = urllib.parse.urlencode(
        {
            "Action": "login",
            "Issuer": issuer,
            "Destination": "https://console.aws.amazon.com/",
            "SigninToken": signin_token["SigninToken"],
        }
    )
    federated_url = f"{aws_federated_signin_endpoint}?{query_string}"
    return federated_url
```

Elimine los recursos creados para la demostración.

```
def teardown(role):
    """
    Removes all resources created during setup.

    :param role: The demo role.
    """
    for attached in role.attached_policies.all():
        role.detach_policy(PolicyArn=attached.arn)
        print(f"Detached {attached.policy_name}.")
    role.delete()
    print(f"Deleted {role.name}.")
```

Ejecute este escenario mediante las funciones previamente definidas.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(f"Welcome to the AWS Security Token Service federated URL demo.")
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    role = setup(iam_resource)
    sts_client = boto3.client("sts")
```

```
    try:
        federated_url = construct_federated_url(
            role.arn, "AssumeRoleDemoSession", "example.org", sts_client
        )
        print(
            "Constructed a federated URL that can be used to connect to the "
            "AWS Management Console with role-defined permissions:"
        )
        print("-" * 88)
        print(federated_url)
        print("-" * 88)
        _ = input(
            "Copy and paste the above URL into a browser to open the AWS "
            "Management Console with limited permissions. When done, press "
            "Enter to clean up and complete this demo."
        )
    finally:
        teardown(role)
        print("Thanks for watching!")
```

- Para obtener detalles sobre la API, consulte AssumeRole en la Referencia de la API del SDK de AWS para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Obtener un token de sesión que requiera un token MFA con AWS STS con un SDK de AWS

En el siguiente ejemplo de código se muestra cómo obtener un token de sesión que requiere un token de MFA.

> **Warning**
>
> Para evitar riesgos de seguridad, no utilice a los usuarios de IAM para la autenticación cuando desarrolle software especialmente diseñado o trabaje con datos reales. En cambio, utilice la federación con un proveedor de identidades como AWS IAM Identity Center.

- Cree un rol de IAM que otorgue permiso para enumerar los buckets de Amazon S3.
- Cree un usuario de IAM que tenga permiso para asumir el rol solo cuando se proporcionen las credenciales de MFA.
- Registre un dispositivo MFA para el usuario.
- Proporcione credenciales MFA para obtener un token de sesión y utilice credenciales temporales para enumerar los buckets de S3.

Python

SDK para Python (Boto3)

> **Note**
>
> Hay más en GitHub. Busque el ejemplo completo y aprenda a configurar y ejecutar en el Repositorio de ejemplos de código de AWS.

Cree un usuario de IAM, registre un dispositivo MFA y cree un rol que otorgue permiso para que el usuario enumere los buckets de S3 solo cuando se utilicen las credenciales de MFA.

```
def setup(iam_resource):
    """
    Creates a new user with no permissions.
    Creates a new virtual multi-factor authentication (MFA) device.
    Displays the QR code to seed the device.
    Asks for two codes from the MFA device.
    Registers the MFA device for the user.
    Creates an access key pair for the user.
    Creates an inline policy for the user that lets the user list Amazon S3
 buckets,
    but only when MFA credentials are used.
```

```
    Any MFA device that can scan a QR code will work with this demonstration.
    Common choices are mobile apps like LastPass Authenticator,
    Microsoft Authenticator, or Google Authenticator.

    :param iam_resource: A Boto3 AWS Identity and Access Management (IAM)
resource
                         that has permissions to create users, MFA devices, and
                         policies in the account.
    :return: The newly created user, user key, and virtual MFA device.
    """
    user = iam_resource.create_user(UserName=unique_name("user"))
    print(f"Created user {user.name}.")

    virtual_mfa_device = iam_resource.create_virtual_mfa_device(
        VirtualMFADeviceName=unique_name("mfa")
    )
    print(f"Created virtual MFA device {virtual_mfa_device.serial_number}")

    print(
        f"Showing the QR code for the device. Scan this in the MFA app of your "
        f"choice."
    )
    with open("qr.png", "wb") as qr_file:
        qr_file.write(virtual_mfa_device.qr_code_png)
    webbrowser.open(qr_file.name)

    print(f"Enter two consecutive code from your MFA device.")
    mfa_code_1 = input("Enter the first code: ")
    mfa_code_2 = input("Enter the second code: ")
    user.enable_mfa(
        SerialNumber=virtual_mfa_device.serial_number,
        AuthenticationCode1=mfa_code_1,
        AuthenticationCode2=mfa_code_2,
    )
    os.remove(qr_file.name)
    print(f"MFA device is registered with the user.")

    user_key = user.create_access_key_pair()
    print(f"Created access key pair for user.")

    print(f"Wait for user to be ready.", end="")
    progress_bar(10)

    user.create_policy(
```

```
        PolicyName=unique_name("user-policy"),
        PolicyDocument=json.dumps(
            {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Action": "s3:ListAllMyBuckets",
                        "Resource": "arn:aws:s3:::*",
                        "Condition": {"Bool": {"aws:MultiFactorAuthPresent":
 True}},
                    }
                ],
            }
        ),
    )
    print(
        f"Created an inline policy for {user.name} that lets the user list
 buckets, "
        f"but only when MFA credentials are present."
    )

    print("Give AWS time to propagate these new resources and connections.",
 end="")
    progress_bar(10)

    return user, user_key, virtual_mfa_device
```

Obtenga credenciales de sesión temporales al pasar un token MFA y utilice las credenciales para enumerar los buckets de S3 para la cuenta.

```
def list_buckets_with_session_token_with_mfa(mfa_serial_number, mfa_totp,
 sts_client):
    """
    Gets a session token with MFA credentials and uses the temporary session
    credentials to list Amazon S3 buckets.

    Requires an MFA device serial number and token.
```

```
    :param mfa_serial_number: The serial number of the MFA device. For a virtual
 MFA
                              device, this is an Amazon Resource Name (ARN).
    :param mfa_totp: A time-based, one-time password issued by the MFA device.
    :param sts_client: A Boto3 STS instance that has permission to assume the
 role.
    """
    if mfa_serial_number is not None:
        response = sts_client.get_session_token(
            SerialNumber=mfa_serial_number, TokenCode=mfa_totp
        )
    else:
        response = sts_client.get_session_token()
    temp_credentials = response["Credentials"]

    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )

    print(f"Buckets for the account:")
    for bucket in s3_resource.buckets.all():
        print(bucket.name)
```

Elimine los recursos creados para la demostración.

```
def teardown(user, virtual_mfa_device):
    """
    Removes all resources created during setup.

    :param user: The demo user.
    :param role: The demo MFA device.
    """
    for user_pol in user.policies.all():
        user_pol.delete()
        print("Deleted inline user policy.")
    for key in user.access_keys.all():
        key.delete()
```

```
        print("Deleted user's access key.")
    for mfa in user.mfa_devices.all():
        mfa.disassociate()
    virtual_mfa_device.delete()
    user.delete()
    print(f"Deleted {user.name}.")
```

Ejecute este escenario mediante las funciones previamente definidas.

```
def usage_demo():
    """Drives the demonstration."""
    print("-" * 88)
    print(
        f"Welcome to the AWS Security Token Service assume role demo, "
        f"starring multi-factor authentication (MFA)!"
    )
    print("-" * 88)
    iam_resource = boto3.resource("iam")
    user, user_key, virtual_mfa_device = setup(iam_resource)
    try:
        sts_client = boto3.client(
            "sts", aws_access_key_id=user_key.id,
 aws_secret_access_key=user_key.secret
        )
        try:
            print("Listing buckets without specifying MFA credentials.")
            list_buckets_with_session_token_with_mfa(None, None, sts_client)
        except ClientError as error:
            if error.response["Error"]["Code"] == "AccessDenied":
                print("Got expected AccessDenied error.")
        mfa_totp = input("Enter the code from your registered MFA device: ")
        list_buckets_with_session_token_with_mfa(
            virtual_mfa_device.serial_number, mfa_totp, sts_client
        )
    finally:
        teardown(user, virtual_mfa_device)
        print("Thanks for watching!")
```

- Para obtener detalles sobre la API, consulte GetSessionToken en la Referencia de la API del SDK de AWS para Python (Boto3).

Para obtener una lista completa de las guías para desarrolladores del SDK de AWS y ejemplos de código, consulte Uso de IAM con un SDK de AWS. En este tema también se incluye información sobre cómo comenzar a utilizar el SDK y detalles sobre sus versiones anteriores.

Seguridad en IAM y AWS STS

La seguridad en la nube de AWS es la mayor prioridad. Como cliente de AWS, se beneficiará de una arquitectura de red y de centros de datos diseñados para satisfacer los requisitos de seguridad de las organizaciones más exigentes.

La seguridad es una responsabilidad compartida entre AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta los servicios de AWS en la nube de AWS. AWS también proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWSProgramas de conformidad de . Para obtener información sobre los programas de conformidad que se aplican a AWS Identity and Access Management (IAM), consulte Servicios de AWS en el ámbito del programa de conformidad.
- Seguridad en la nube: su responsabilidad viene determinada por el servicio de AWS que utilice. También es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos vigentes.

Esta documentación le ayuda a comprender cómo puede aplicar el modelo de responsabilidad compartida cuando se utiliza AWS Identity and Access Management (IAM) y AWS Security Token Service (AWS STS). En los siguientes temas, se le mostrará cómo configurar IAM y AWS STS para satisfacer sus objetivos de seguridad y conformidad. También puede aprender a utilizar otros servicios de AWS que lo ayuden a monitorear y proteger los recursos de IAM.

Contenido

- Políticas administradas de AWS para el Analizador de acceso de AWS Identity and Access Management

Credenciales de seguridad de AWS

Al interactuar con AWS, debe especificar las credenciales de seguridad de AWS con el fin de demostrar quién es usted y si tiene permiso para acceder a los recursos que solicita. AWS utiliza las credenciales de seguridad para autenticar y autorizar sus solicitudes.

Por ejemplo, si desea descargar un archivo protegido de un bucket de Amazon Simple Storage Service (Amazon S3), sus credenciales deben permitir ese tipo de acceso. Si sus credenciales no muestran que está autorizado para descargar el archivo, AWS le denegará la solicitud. Sin embargo, sus credenciales de seguridad de AWS no son necesarias para descargar un archivo de un bucket de Amazon S3 que se comparte públicamente.

Hay diferentes tipos de usuarios en AWS. Todos los usuarios de AWS tienen credenciales de seguridad. Están el propietario de la cuenta (usuario raíz), usuarios en AWS IAM Identity Center, los usuarios federados y los usuarios de IAM.

Los usuarios tienen credenciales de seguridad temporales o a largo plazo. El usuario raíz, el usuario de IAM y las claves de acceso tienen credenciales de seguridad a largo plazo que no caducan. Para proteger las credenciales a largo plazo, utilice procesos a fin de administrar claves de acceso, cambiar contraseñas y habilitar la MFA.

Los roles de IAM, usuarios en AWS IAM Identity Center y los usuarios federados poseen credenciales de seguridad temporales. Las credenciales de seguridad temporales caducan después de un periodo definido o cuando el usuario finaliza la sesión. Las credenciales temporales funcionan prácticamente igual que las credenciales a largo plazo, con las siguientes diferencias:

- Las credenciales de seguridad temporales son a corto plazo, tal como su nombre indica. Se pueden configurar para durar entre unos cuantos minutos y varias horas. Cuando las credenciales caduquen, AWS dejará de reconocerlas o permitirá todo tipo de acceso a las solicitudes realizadas desde API que las utilicen.
- Las credenciales de seguridad temporales no se guardan con el usuario, sino que se generan de forma dinámica y se proporcionan al usuario cuando se solicitan. Cuando las credenciales de seguridad temporales caducan (o incluso antes), el usuario puede solicitar nuevas credenciales, siempre y cuando el usuario que las solicite tenga permiso para hacerlo.

Como resultado, las credenciales temporales tienen las siguientes ventajas con respecto a las credenciales a largo plazo:

- No tiene que distribuir ni incrustar credenciales de seguridad de AWS a largo plazo con una aplicación.
- Puede proporcionar acceso a sus recursos de AWS a usuarios, sin necesidad de definir una identidad de AWS para ellos. Las credenciales temporales son la base de los roles y las identidades federadas.
- Las credenciales de seguridad temporales tienen un ciclo de vida limitado, por lo que no tiene que actualizarlas ni revocarlas de forma explícita cuando ya no las necesite. Cuando las credenciales de seguridad temporales caducan, ya no se pueden volver a utilizar. Puede especificar el tiempo de validez de las credenciales, hasta un límite máximo.

Consideraciones de seguridad

Recomendamos que tenga en cuenta la siguiente información al determinar las disposiciones de seguridad para su Cuenta de AWS:

- Al crear una Cuenta de AWS, creamos el usuario raíz de la cuenta. Las credenciales del usuario raíz (propietario de la cuenta) permiten el acceso completo a todos los recursos de la cuenta. La primera tarea que debe realizar con el usuario raíz es conceder a otro usuario permisos administrativos para su Cuenta de AWS a fin de minimizar el uso del usuario raíz.
- No puede utilizar las políticas de IAM para denegar al usuario raíz el acceso a los recursos de forma explícita. Solo puede usar una política de control de servicios (SCP) de AWS Organizations para limitar los permisos del usuario raíz.
- Si olvida o pierde la contraseña de usuario raíz, debe tener acceso a la dirección de correo electrónico asociada a su cuenta para poder restablecerla.
- Si pierde las claves de acceso de usuario raíz, debe poder iniciar sesión en su cuenta como usuario raíz para crear otras nuevas.
- No utilice el usuario raíz para las tareas cotidianas. Utilícelo para realizar las tareas que solo puede realizar el usuario raíz. Para obtener la lista completa de tareas que requieren que inicie sesión como usuario raíz, consulte Tareas que requieren credenciales de usuario raíz.
- Las credenciales de seguridad son específicas de la cuenta. Si tiene acceso a varias Cuentas de AWS, tiene credenciales independientes para cada cuenta.

- Las políticas determinan qué acciones puede realizar un usuario, un rol o un miembro de un grupo de usuarios, en qué recursos de AWS y en qué condiciones. Con las políticas, puede controlar de forma segura el acceso a los recursos y Servicios de AWS de su Cuenta de AWS. Si debe modificar o revocar los permisos en respuesta a un evento de seguridad, elimine o modifique las políticas en lugar de realizar cambios directamente en la identidad.
- Asegúrese de guardar las credenciales de inicio de sesión de su usuario de IAM de Acceso de emergencia y cualquier clave de acceso que haya creado para el acceso programático en una ubicación segura. Si pierde sus claves de acceso, debe iniciar sesión en su cuenta para crear otras nuevas.
- Le recomendamos que utilice las credenciales temporales proporcionadas por los roles de IAM y los usuarios federados en lugar de las credenciales a largo plazo proporcionadas los usuarios de IAM y las claves de acceso.

Identidad federada

Las identidades federadas son usuarios con identidades externas a los que se les otorgan credenciales de AWS temporales que pueden usar para acceder a recursos seguros de la Cuenta de AWS. Las identidades externas pueden proceder de un almacén de identidades corporativas (por ejemplo, LDAP o Windows Active Directory) o de un tercero (como Login with Amazon, Facebook o Google). Las identidades federadas no inician sesión con la AWS Management Console o el portal de acceso de AWS.

Para permitir que las identidades federadas inicien sesión en AWS, debe crear una URL personalizada que incluya `https://signin.aws.amazon.com/federation`. Para obtener más información, consulte Permitir el acceso del agente de identidades personalizadas a la consola de AWS.

Para obtener más información acerca de las identidades federadas, consulte Federación y proveedores de identidades.

Multi-Factor authentication (MFA)

La autenticación multifactor (MFA) proporciona un nivel de seguridad adicional para los usuarios que pueden acceder a su Cuenta de AWS. Para aumentar la seguridad, le recomendamos que exija la MFA en las credenciales de Usuario raíz de la cuenta de AWS y todos los usuarios de IAM. Para obtener más información, consulte Uso de autenticación multifactor (MFA) en AWS.

Al activar la MFA e iniciar sesión en su Cuenta de AWS, se le solicitarán las credenciales de inicio de sesión y una respuesta generada por un dispositivo de MFA, como un código, un toque o un escaneo biométrico. Al agregar MFA, la configuración y los recursos de su Cuenta de AWS están más seguros.

De forma predeterminada, la MFA no está activada. Puede habilitar y administrar dispositivos de MFA para el Usuario raíz de la cuenta de AWS en la página Credenciales de seguridad o en el panel IAM de la AWS Management Console. Para obtener más información sobre la activación de MFA para usuarios de IAM, consulte Habilitación de dispositivos MFA para usuarios en AWS.

Para obtener más información sobre el inicio de sesión con autenticación multifactor (MFA), consulte Uso de dispositivos MFA con la página de inicio de sesión de IAM.

Acceso programático

Usted proporciona sus claves de acceso de AWS para realizar llamadas a AWS mediante programación o para utilizar la AWS Command Line Interface o AWS Tools for PowerShell. Cuando sea posible, le recomendamos utilizar claves de acceso a corto plazo.

Cuando crea una clave de acceso a largo plazo, crea el ID de clave de acceso (por ejemplo, `AKIAIOSFODNN7EXAMPLE`) y la clave de acceso secreta (por ejemplo, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) como un conjunto. La clave de acceso secreta solo está disponible para descarga en el momento en que se crea. Si no descarga su clave de acceso secreta o la pierde, tendrá que crear una nueva.

En muchos casos, no se requieren claves de acceso a largo plazo que nunca caducan (como cuando crea claves de acceso para un usuario de IAM). En su lugar, puede crear roles de IAM y generar credenciales de seguridad temporales. Las credenciales de seguridad temporales incluyen un ID de clave de acceso y una clave de acceso secreta, pero también incluyen un token de seguridad que indica cuándo caducan las credenciales. Después de caducar, ya no son válidas.

Los ID de clave de acceso que comienzan por `AKIA` son claves de acceso a largo plazo para un usuario de IAM o un usuario raíz de la Cuenta de AWS. Los ID de clave de acceso que comienzan por `ASIA` son claves de acceso de credenciales temporales que se crean mediante operaciones de AWS STS.

Los usuarios necesitan acceso programático si desean interactuar con AWS fuera de la AWS Management Console. La forma de conceder el acceso programático depende del tipo de usuario que acceda a AWS:

Para conceder acceso programático a los usuarios, elija una de las siguientes opciones.

¿Qué usuario necesita acceso programático?	Para	B
Identidad del personal (Usuarios administrados en el IAM Identity Center)	Utilice credenciales a largo plazo para firmar las solicitud es programáticas a la AWS CLI, los SDK de AWS o las API de AWS.	Siga las instrucciones de la interfaz que desea utilizar: • Para elloAWS CLI, consulte Configuración delAWS CLI para su usoAWS IAM Identity Center en la Guía delAWS Command Line Interface usuario. • Para verAWS los SDK, las herramientas y lasAWS API, consulte la autenticación de IAM Identity Center en la Guía de referencia deAWS SDK y herramientas.
IAM	Utilice credenciales a largo plazo para firmar las solicitud es programáticas a la AWS CLI, los SDK de AWS o las API de AWS.	Siguiendo las instrucciones de Uso de credenciales temporales con recursos de AWS de la Guía del usuario de IAM.
IAM	(No recomendado) Utilice credenciales a largo plazo para firmar las solicitud es programáticas a las API de AWS CLI o AWS (directam ente o mediante los AWS SDK).	Siga las instrucciones de la interfaz que desea utilizar: • Para elloAWS CLI, consulte Autenticación mediante credenciales de usuario de IAM en la Guía delAWS Command Line Interface usuario. • Para verAWS los SDK y las herramientas, consulta

¿Qué usuario necesita acceso programático?	Para	B
		Autenticar mediante credenciales a largo plazo en la Guía de referencia deAWS SDK y herramien tas. • Consulte AWSAdmini stración de claves de acceso para usuarios de IAM en la Guía del usuario de IAM.

Alternativas para claves de acceso a largo plazo

Para muchos casos de uso comunes, existen alternativas a las claves de acceso a largo plazo. Para mejorar la seguridad de su cuenta, tenga en cuenta lo siguiente.

- No inserte claves de acceso a largo plazo ni claves de acceso secretas en el código de la aplicación ni en un repositorio de códigos: en cambio, utilice AWS Secrets Manager u otra solución de administración de secretos para no tener que codificar las claves en texto sin formato. La aplicación o el cliente pueden entonces recuperar los secretos cuando sea necesario. Para obtener más información, consulte ¿Qué es AWS Secrets Manager? en la Guía del usuario de AWS Secrets Manager.

- Utilice los roles de IAM para generar credenciales de seguridad temporales siempre que sea posible: utilice siempre mecanismos para emitir credenciales de seguridad temporales cuando sea posible, en lugar de claves de acceso a largo plazo. Las credenciales de seguridad temporales son más seguras porque no se almacenan con el usuario, sino que se generan de forma dinámica y se proporcionan al usuario cuando se solicitan. Las credenciales de seguridad temporales tienen una duración limitada, por lo que no es necesario administrarlas ni actualizarlas. Los mecanismos que proporcionan claves de acceso temporales incluyen los roles de IAM o la autenticación de un usuario del Centro de identidades de IAM. Para máquinas que se ejecuten fuera de AWS, puede utilizar AWS Identity and Access Management Roles Anywhere.

- Utilice alternativas a las claves de acceso a largo plazo para la AWS Command Line Interface (AWS CLI) o **`aws-shell`**: las alternativas incluyen lo siguiente.
  - AWS CloudShell es un intérprete de comandos previamente autenticado y basado en el navegador que puede lanzar directamente desde la AWS Management Console. Puede ejecutar comandos de la AWS CLI para los Servicios de AWS con su intérprete de comandos de preferencia (Bash, PowerShell o Z shell). Al hacerlo, no necesita descargar o instalar herramientas de línea de comandos. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell.
  - Integración de AWS CLI versión 2 con AWS IAM Identity Center (Centro de identidades de IAM). Puede autenticar a los usuarios y proporcionar credenciales a corto plazo para ejecutar comandos de la AWS CLI. Para obtener más información, consulte Integración de la AWS CLI con el Centro de identidades de IAM en la Guía del usuario de AWS IAM Identity Center y Configuración de la AWS CLI para usar el Centro de identidades de IAM en la Guía del usuario de AWS Command Line Interface.
- No cree claves de acceso a largo plazo para los usuarios humanos que necesiten acceder a las aplicaciones o a los Servicios de AWS: el Centro de identidades de IAM puede generar credenciales de acceso temporales para que los usuarios de proveedores de identidades externos accedan a los Servicios de AWS. De esta manera, no necesita crear y administrar credenciales a largo plazo en IAM. En el Centro de identidades de IAM, cree un conjunto de permisos del Centro de identidades de IAM que conceda acceso a los usuarios de proveedores de identidades externos. A continuación, asigne un grupo del Centro de identidades de IAM al conjunto de permisos de las Cuentas de AWS seleccionadas. Para obtener más información, consulte Qué es AWS IAM Identity Center, Conexión a un proveedor de identidad externo y Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center.
- No almacene claves de acceso a largo plazo en un servicio de cómputos de AWS: en cambio, asigne un rol de IAM a los recursos de cómputo. Esto proporciona automáticamente credenciales temporales para conceder el acceso. Por ejemplo, al crear un perfil de instancia asociado a una instancia de Amazon EC2, puede asignar un rol de AWS a la instancia y ponerla a disposición de todas las aplicaciones. Un perfil de instancia contiene el rol y permite a los programas que se encuentran en ejecución en la instancia de Amazon EC2 obtener credenciales temporales. Para obtener más información, consulte Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias de Amazon EC2.

Acceso a AWS con las credenciales de AWS

AWS requiere diferentes tipos de credenciales de seguridad, según cómo acceda a AWS y qué tipo de usuario de AWS sea. Por ejemplo, usted usa las credenciales de inicio de sesión para la AWS Management Console, pero usa las claves de acceso para realizar llamadas a AWS mediante programación. Además, cada identidad que utilice, ya sea el usuario raíz de la cuenta, un usuario de AWS Identity and Access Management (IAM), un usuario de AWS IAM Identity Center o una identidad federada, contiene credenciales únicas en AWS.

Para obtener instrucciones paso a paso sobre cómo iniciar sesión en AWS según su tipo de usuario, consulte Cómo iniciar sesión en AWS en la Guía de inicio de sesión de AWS.

Directivas de auditoría de seguridad de AWS

De manera periódica, audite su configuración de seguridad para asegurarse de que satisface sus necesidades de negocio actuales. Una auditoría le ofrece la oportunidad de eliminar los usuarios, los grupos, los roles y las políticas de IAM innecesarios y de asegurarse de que los usuarios y el software no posean demasiados permisos.

A continuación encontrará las directrices para revisar y monitorizar sistemáticamente los recursos de AWS en aplicación de las prácticas recomendadas de seguridad.

> **Tip**
>
> Puede monitorear el uso de IAM en relación con las prácticas recomendadas de seguridad con AWS Security Hub. Security Hub utiliza controles de seguridad para evaluar las configuraciones de los recursos y los estándares de seguridad para ayudarle a cumplir varios marcos de conformidad. Para obtener más información sobre el uso de Security Hub para evaluar recursos de IAM, consulte controles de AWS Identity and Access Management en la Guía del usuario de AWS Security Hub.

Contenido

- Revisión de los grupos de IAM
- Revisión de los roles de IAM
- Revisión de los proveedores de IAM; para SAML y OpenID Connect (OIDC)
- Revisión de las aplicaciones móviles
- Sugerencias para revisar las políticas de IAM

Cuándo se debe realizar una auditoría de seguridad

Audite su configuración de seguridad en las siguientes situaciones:

- En forma periódica. Realice los pasos que se describen en este documento a intervalos regulares como práctica recomendada de seguridad.
- Si se producen cambios en su organización; por ejemplo, si se marchan personas.
- Si ha dejado de usar uno o más servicios de AWS individuales para comprobar que ha eliminado los permisos que los usuarios de su cuenta ya no necesitan.
- Si ha agregado o eliminado software en las cuentas, como aplicaciones en instancias de Amazon EC2, pilas de AWS OpsWorks, plantillas de AWS CloudFormation, etc.
- Si sospecha que una persona no autorizada podría haber accedido a su cuenta.

Directrices para la auditoría

Al revisar la configuración de seguridad de su cuenta, siga estas directrices:

- Sea exhaustivo. Fíjese en todos los aspectos de su configuración de seguridad, incluidos aquellos que casi nunca se utilizan.
- No haga suposiciones. Si no conoce bien algún aspecto de la configuración de seguridad (por ejemplo, los motivos para una determinada política o la existencia de un rol), investigue la necesidad de negocio hasta entender el riesgo potencial.
- Simplifique. Para facilitar la auditoría (y la administración), utilice grupos de IAM, roles de IAM, esquemas de nomenclatura coherentes y políticas sencillas.

Revisión de las credenciales de su cuenta de AWS

Siga estos pasos al auditar las credenciales de su cuenta de AWS:

1. Si tiene claves de acceso para el usuario raíz que no está utilizando, puede eliminarlas. Recomendamos encarecidamente no utilizar las claves de acceso raíz para el trabajo cotidiano con AWS. En cambio, utilice usuarios con credenciales temporales, como usuarios en AWS IAM Identity Center.
2. Si necesita claves de acceso para su cuenta, asegúrese de actualizarlas cuando sea necesario.

Revisión de los usuarios de IAM

Siga estos pasos al auditar sus usuarios de IAM existentes:

1. Enumere los usuarios y, a continuación, elimine los usuarios que no se necesitan.
2. Elimine los usuarios de los grupos a los que no necesitan acceso.
3. Revise las políticas adjuntas a los grupos a los que pertenece el usuario. Consulte Sugerencias para revisar las políticas de IAM.
4. Elimine las credenciales de seguridad que el usuario no necesite o que se hayan visto expuestas. Por ejemplo, un usuario de IAM que se utiliza para una aplicación no necesita una contraseña (que solo es necesaria para iniciar sesión en los sitios web de AWS). Del mismo modo, si un usuario ya no utiliza las claves de acceso, no hay motivo para que las tenga. Para obtener más información, consulte Administración de las contraseñas de los usuarios de IAM y Administración de claves de acceso de los usuarios de IAM.

   Puede generar y descargar un informe de credenciales que contenga una lista de todos los usuarios de IAM de su cuenta y el estado de sus credenciales, tales como contraseñas, claves de acceso y dispositivos MFA. Para las contraseñas y claves de acceso, el informe de credenciales muestra la fecha y hora en que se ha utilizado la contraseña o la clave de acceso por última vez. Considere la posibilidad de eliminar de su cuenta las credenciales que no se hayan utilizado recientemente. (No elimine su usuario de acceso de emergencia). Para obtener más información, consulte Obtención de informes de credenciales para la cuenta de AWS.
5. Actualice las contraseñas y claves de acceso cuando sea necesario para los casos de uso que requieren credenciales a largo plazo. Para obtener más información, consulte Administración de las contraseñas de los usuarios de IAM y Administración de claves de acceso de los usuarios de IAM.
6. Como práctica recomendada, exija a los usuarios humanos que utilicen la federación con un proveedor de identidades para acceder a AWS con credenciales temporales. Si es posible, haga la transición de usuarios de IAM a usuarios federados, como los usuarios de IAM Identity Center. Retenga la cantidad mínima de usuarios de IAM necesarios para sus aplicaciones.

Revisión de los grupos de IAM

Siga estos pasos al auditar los grupos de IAM:

1. Enumere los grupos y, a continuación, elimine los grupos que no utiliza.
2. Revise los usuarios de cada grupo y elimine los usuarios que no les pertenecen.
3. Revise las políticas adjuntas al grupo. Consulte Sugerencias para revisar las políticas de IAM.

Revisión de los roles de IAM

Siga estos pasos al auditar los roles de IAM:

1. Enumere los roles y, a continuación, elimine los roles que no utiliza.
2. Revise la política de confianza del rol. Asegúrese de saber quién es la entidad principal y de entender por qué esa cuenta o ese usuario necesitan poder asumir el rol.
3. Revise la política de acceso del rol para asegurarse de que conceda permisos adecuados a quien asuma ese rol; consulte Sugerencias para revisar las políticas de IAM.

Revisión de los proveedores de IAM; para SAML y OpenID Connect (OIDC)

Si ha creado una entidad de IAM para establecer una relación de confianza con un proveedor de identidades (IdP) SAML u OIDC, siga estos pasos:

1. Elimine los proveedores que no se utilicen.
2. Descargue y revise los documentos de metadatos de AWS de cada IdP SAML y asegúrese de que reflejen las necesidades de negocio actuales.
3. Obtenga los últimos documentos de metadatos de los proveedores de identidades SAML y actualice el proveedor en IAM.

Revisión de las aplicaciones móviles

Si ha creado una aplicación móvil que realiza solicitudes a AWS, siga estos pasos:

1. Asegúrese de que la aplicación móvil no contenga claves de acceso integradas, incluso si están en el almacenamiento cifrado.
2. Obtenga credenciales temporales para la aplicación mediante el uso de API diseñadas para ello.

> **Note**
>
> Recomendamos utilizar Amazon Cognito para administrar la identidad de los usuarios en su aplicación. Este servicio permite autenticar a los usuarios mediante Login with Amazon, Facebook, Google o cualquier proveedor de identidad compatible con OpenID Connect (OIDC). Para obtener más información, consulte Grupos de identidades de Amazon Cognito en la Guía para desarrolladores de Amazon Cognito.

Sugerencias para revisar las políticas de IAM

Las políticas son potentes y sutiles, por lo que es importante estudiar y comprender los permisos que concede cada una de ellas. Utilice las siguientes directrices al revisar las políticas:

- Asocie políticas a grupos o roles en lugar de asociarlos a usuarios individuales. Si un usuario individual tiene una política, asegúrese de comprender por qué ese usuario la necesita.
- Asegúrese de que los usuarios, grupos y roles de IAM tengan los permisos que necesitan y no tengan ningún permiso adicional.
- Utilice el simulador de política de IAM para probar las políticas asociadas a usuarios o grupos.
- Recuerde que los permisos de usuario son el resultado de todas las políticas aplicables basadas en identidad (en usuarios, grupos o roles) y basadas en recursos (en recursos como buckets de Amazon S3, colas de Amazon SQS, temas de Amazon SNS y claves de AWS KMS). Es importante examinar todas las políticas aplicables a un usuario y comprender el conjunto completo de permisos concedidos a un usuario individual.
- Tenga en cuenta que permitir que un usuario cree un usuario, grupo, rol o política de IAM y asociar una política a la entidad principal equivale, en la práctica, a conceder a ese usuario todos los permisos sobre todos los recursos de su cuenta. Los usuarios que pueden crear políticas y asociarlas a usuarios, grupos o roles se pueden conceder a sí mismos cualquier permiso. En general, no conceda permisos de IAM a los usuarios o roles en los que no confía lo suficiente para que dispongan de pleno acceso a los recursos de su cuenta. Al realizar la auditoría de seguridad, confirme que se conceden los siguientes permisos de IAM a las identidades de confianza:
  - `iam:PutGroupPolicy`
  - `iam:PutRolePolicy`
  - `iam:PutUserPolicy`
  - `iam:CreatePolicy`

  - `iam:CreatePolicyVersion`
  - `iam:AttachGroupPolicy`
  - `iam:AttachRolePolicy`
  - `iam:AttachUserPolicy`
- Asegúrese de que las políticas no concedan permisos para servicios que no se utilizan. Por ejemplo, si utiliza políticas administradas de AWS, asegúrese de que las políticas administradas de AWS que están en uso en su cuenta sean para servicios que utilice realmente. Para saber qué políticas administradas de AWS se utilizan en su cuenta, utilice la API `GetAccountAuthorizationDetails` de IAM (comando de AWS CLI: `aws iam get-account-authorization-details`).
- Si la política concede a un usuario permiso para lanzar una instancia de Amazon EC2, también podría permitir la acción `iam:PassRole`, pero, en este caso, debería enumerar de forma explícita los roles que el usuario puede pasar a la instancia de Amazon EC2.
- Examine todos los valores del elemento `Action` o `Resource` que incluyan `*`. Cuando sea posible, conceda acceso `Allow` a las acciones y los recursos individuales que los usuarios necesitan. Sin embargo, a continuación se citan algunas razones por las que podría ser conveniente utilizar `*` en una política:
  - La política está diseñada para conceder permisos de nivel administrativo.
  - El comodín se utiliza por comodidad para un conjunto de acciones similares (por ejemplo, `Describe*`) y usted está convencido de la idoneidad de la lista completa de acciones a las que se hace referencia de este modo.
  - El comodín se utiliza para indicar una clase de recursos o una ruta de recursos (por ejemplo, `arn:aws:iam::`*`account-id`*`:users/division_abc/*`) y usted está convencido de la idoneidad de conceder acceso a todos los recursos de esa clase o ruta.
  - Una acción de servicio no admite permisos de nivel de recursos y la única opción para un recurso es `*`.
- Examine los nombres de las políticas para asegurarse de que reflejen la función que cumple cada una de ellas. Por ejemplo, el nombre de una política podría incluir el texto “solo lectura” pero, en realidad, conceder permisos de escritura o cambio.

Para obtener más información sobre cómo planificar su auditoría de seguridad, consulte las Prácticas recomendadas de seguridad, identidad y conformidad en el Centro de arquitectura de AWS.

Protección de los datos en AWS Identity and Access Management

El modelo de responsabilidad compartida de AWS se aplica a la protección de datos de AWS Identity and Access Management. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global que ejecuta toda la Nube de AWS. Usted es responsable de mantener el control sobre el contenido alojado en esta infraestructura. Usted también es responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulte las Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulte la publicación de blog AWS Shared Responsibility Model and GDPR en el Blog de seguridad de AWS.

Con fines de protección de datos, recomendamos proteger las credenciales de la cuenta de Cuenta de AWS y configurar cuentas de usuario individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utilice autenticación multifactor (MFA) en cada cuenta.
- Utilice SSL/TLS para comunicarse con los recursos de AWS. Se recomienda el uso de TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad del usuario con AWS CloudTrail.
- Utilice las soluciones de cifrado de AWS, junto con todos los controles de seguridad predeterminados dentro de los Servicios de AWS.
- Utilice servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita módulos criptográficos validados FIPS 140-2 al acceder a AWS a través de una interfaz de la línea de comandos o una API, utilice un punto de conexión de FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.

Se recomienda encarecidamente no ingresar nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Incluye las situaciones en las que debe trabajar con IAM u otros Servicios de AWS mediante la consola, la API, la AWS CLI o los AWS SDK. Cualquier dato que ingrese en etiquetas o campos de formato libre utilizados para nombres se pueden emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos

encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Cifrado de datos en IAM y AWS STS

El cifrado de datos normalmente se divide en dos categorías: el cifrado en reposo y el cifrado en tránsito.

Cifrado en reposo

Los datos que IAM recopila y almacena se cifran en reposo.

- IAM – Los datos recopilados y almacenados dentro de IAM incluyen las direcciones IP, los metadatos de las cuentas de los clientes y los datos de identificación de los clientes, que contienen las contraseñas. Los metadatos de las cuentas de los clientes y los datos de identificación de estos se cifran en reposo mediante AES 256 o una función hash SHA 256.
- AWS STS – AWS STS no recopila el contenido del cliente excepto los registros de servicio que registran solicitudes correctas, erróneas y defectuosas en el servicio.

Cifrado en tránsito

Los datos de identificación del cliente, incluidas las contraseñas, se cifran en tránsito mediante TLS 1.2 y 1.3. Todos los puntos de enlace de AWS STS admiten HTTPS para cifrar datos en tránsito. Para obtener una lista de puntos de conexión de AWS STS, consulte Regiones y puntos de conexión.

Administración de claves en IAM y AWS STS

No puede administrar las claves de cifrado mediante IAM o AWS STS. Para obtener más información sobre las claves de cifrado y , consulte ¿Qué es AWS KMS? en la guía para desarrolladores de AWS Key Management Service.

Privacidad del tráfico entre redes en IAM y AWS STS

Las solicitudes a IAM deben realizarse utilizando el protocolo Transport Layer Security (TLS). Puede proteger las conexiones al servicio de AWS STS utilizando los puntos de conexión de VPC. Para obtener más información, consulte Puntos de conexión de VPC de tipo interfaz.

Registro y monitoreo en AWS Identity and Access Management

El monitoreo es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el desempeño deAWS Identity and Access Management(IAM),AWS Security Token Service (AWS STS) y sus otras soluciones AWS. AWS proporciona varias herramientas para monitorear sus recursos de AWS y respuesta a posibles incidentes:

- AWS CloudTrail captura todas las llamadas a la API de IAM y AWS STS como eventos, incluidas las llamadas procedentes de la consola y llamadas a la API. Para obtener más información acerca del uso de CloudTrail con IAM y AWS STS, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail. Para obtener más información acerca de CloudTrail, consulte la AWS CloudTrailGuía del usuario de .
- AWS Identity and Access Management Access Analyzer le ayuda a identificar los recursos de su organización y sus cuentas, como buckets de Amazon S3 o roles de IAM, que se comparten con una entidad externa. Esto le ayuda a identificar el acceso no deseado a sus recursos y datos, lo que constituye un riesgo para la seguridad. Para obtener más información, consulte ¿Qué es IAM Access Analyzer?.
- Amazon CloudWatch monitorea los recursos de AWS y las aplicaciones que ejecuta en AWS en tiempo real. Puede recopilar métricas y realizar un seguimiento de las métricas, crear paneles personalizados y definir alarmas que le advierten o que toman medidas cuando una métrica determinada alcanza el umbral que se especifique. Por ejemplo, puede hacer que CloudWatch haga un seguimiento del uso de la CPU u otras métricas de las instancias de Amazon EC2 y lanzar nuevas instancias automáticamente cuando sea necesario. Para obtener más información, consulte la Guía del usuario de Amazon CloudWatch.
- Amazon CloudWatch Logs le ayuda a monitorear, almacenar y tener acceso a los archivos de registro desde instancias de Amazon EC2, CloudTrail u otras fuentes. CloudWatch Logs puede monitorear información en los registros y enviarle una notificación cuando se llega a determinados umbrales. También se pueden archivar los datos de los registros en un almacenamiento de larga duración. Para obtener más información, consulte la Guía del usuario de Amazon CloudWatch Logs.

Para obtener más recursos y prácticas recomendadas de seguridad para IAM, consulte Prácticas de seguridad recomendadas y casos de uso en AWS Identity and Access Management.

Validación de conformidad en AWS Identity and Access Management

Los auditores externos evalúan la seguridad y la conformidad de AWS Identity and Access Management (IAM) en distintos programas de conformidad de AWS. Estos incluyen SOC, PCI, FedRAMP, ISO y otros.

Para saber si un Servicio de AWS está incluido en el ámbito de programas de conformidad específicos, consulte Servicios de AWS en el ámbito del programa de conformidad y escoja el programa de conformidad que le interese. Para obtener información general, consulte Programas de conformidad de AWS.

Puede descargar los informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact.

Su responsabilidad de conformidad al utilizar Servicios de AWS se determina en función de la sensibilidad de los datos, los destinos de cumplimiento de su empresa y la legislación y los reglamentos correspondientes. AWS proporciona los siguientes recursos para ayudar con la conformidad:

- Guías de inicio rápido de seguridad y conformidad: estas guías de implementación tratan consideraciones sobre arquitectura y ofrecen pasos para implementar los entornos de referencia centrados en la seguridad y la conformidad en AWS.
- Architecting for HIPAA Security and Compliance on Amazon Web Services (Arquitectura para la seguridad y el cumplimiento de la HIPAA en Amazon Web Services): en este documento técnico, se describe cómo las empresas pueden utilizar AWS para crear aplicaciones aptas para HIPAA.

  > **Note**
  >
  > No todos los Servicios de AWS son aptos para HIPAA. Para obtener más información, consulte la Referencia de servicios aptos para HIPAA.

- Recursos de conformidad de AWS: este conjunto de manuales y guías podría aplicarse a su sector y ubicación.
- AWSGuías de cumplimiento para clientes: comprenda el modelo de responsabilidad compartida desde la perspectiva del cumplimiento. Las guías resumen las prácticas recomendadas para garantizar la seguridad de Servicios de AWS y orientan los controles de seguridad en varios marcos (incluidos el Instituto Nacional de Estándares y Tecnología (NIST), el Consejo de

Normas de Seguridad del Sector de Tarjetas de Pago (PCI) y la Organización Internacional de Normalización (ISO)).

- Evaluación de recursos con reglas en la Guía para desarrolladores de AWS Config: el servicio AWS Config evalúa en qué medida las configuraciones de sus recursos cumplen las prácticas internas, las directrices del sector y las normativas.
- AWS Security Hub: este Servicio de AWS proporciona una visión completa de su estado de seguridad en AWS. Security Hub utiliza controles de seguridad para evaluar sus recursos de AWS y comprobar su cumplimiento con los estándares y las prácticas recomendadas del sector de la seguridad. Para obtener una lista de los servicios y controles compatibles, consulte la referencia de controles de Security Hub.
- AWS Audit Manager: este Servicio de AWS le ayuda a auditar continuamente el uso de AWS con el fin de simplificar la forma en que administra el riesgo y la conformidad con las normativas y los estándares del sector.

Resiliencia en AWS Identity and Access Management

La infraestructura global de AWS se compone de regiones y zonas de disponibilidad de AWS. AWS Las regiones tienen múltiples zonas de disponibilidad físicamente separadas y aisladas, que están conectadas con redes de baja latencia, alto rendimiento y altamente redundantes. Para obtener más información sobre las regiones y zonas de disponibilidad de AWS, consulte Infraestructura global de AWS.

AWS Identity and Access Management (IAM) y AWS Security Token Service (AWS STS) son servicios autónomos, basados en la región, disponibles en todo el mundo.

IAM es un Servicio de AWS fundamental. IAM debe autenticar y autorizar cada operación que se lleva a cabo en AWS. IAM comprueba cada solicitud con las identidades y políticas almacenadas en IAM para determinar si se permite o se deniega la solicitud. IAM se diseñó con un plano de control y un plano de datos separados para que el servicio se autentique incluso durante fallos inesperados. Los recursos de IAM que se utilizan en las autorizaciones, como los roles y las políticas, se almacenan en el plano de control. Los clientes de IAM pueden cambiar la configuración de estos recursos mediante operaciones de IAM, como `DeletePolicy` y `AttachRolePolicy`. Esas solicitudes de cambio de configuración van al plano de control. Hay un plano de control de IAM para todas las Regiones de AWS comerciales, que se encuentra en la región Este de EE. UU. (Norte de Virginia). A continuación, el sistema de IAM propaga los cambios de configuración a los planos de datos de IAM en cada uno de los planos habilitados para la Región de AWS. El plano de datos de

IAM es, esencialmente, una réplica de solo lectura de los datos de configuración del plano de control de IAM. Cada Región de AWS tiene una instancia completamente independiente del plano de datos de IAM, que autentica y autoriza las solicitudes de la misma región. En cada región, el plano de datos de IAM está distribuido en al menos tres zonas de disponibilidad, y tiene capacidad suficiente para tolerar la pérdida de una zona de disponibilidad sin que el cliente se vea perjudicado. Tanto el plano de control como el de datos de IAM se crearon para que no haya inactividad planificada, con todas las actualizaciones de software y operaciones de escalado que se llevan a cabo de manera invisible para los clientes.

Las solicitudes de AWS STS siempre van a un único punto de conexión global por defecto. Puede utilizar un punto de conexión de AWS STS regional para reducir la latencia o proporcionar redundancia adicional para sus aplicaciones. Para obtener más información, consulte Administrar AWS STS en una Región de AWS.

Ciertos eventos pueden interrumpir la comunicación entre Regiones de AWS a través de la red. No obstante, incluso cuando no puede comunicarse con el punto de conexión de IAM global, AWS STS puede seguir autenticando las entidades principales de IAM, e IAM puede autorizar sus solicitudes. Los detalles específicos de un evento que interrumpe la comunicación determinarán su capacidad de acceso a los servicios de AWS. En la mayoría de los casos, puede seguir utilizando credenciales de IAM en su entorno de AWS. Las siguientes condiciones podrían aplicarse a un evento que interrumpe la comunicación.

Claves de acceso para usuarios de IAM

Puede autenticarse indefinidamente en una región con claves de acceso para usuarios de IAM a largo plazo. Cuando utiliza la AWS Command Line Interface y las API, puede proporcionar claves de acceso de AWS para que AWS pueda verificar su identidad en las solicitudes programáticas.

> **Important**
>
> Como práctica recomendada, le sugerimos que los usuarios inicien sesión con credenciales temporales en lugar de claves de acceso de larga duración.

Credenciales temporales

Puede solicitar nuevas credenciales temporales con el punto de conexión de servicio regional de AWS STS durante al menos 24 horas. Las siguientes operaciones de API generan credenciales temporales.

- AssumeRole
- AssumeRoleWithWebIdentity
- AssumeRoleWithSAML
- GetFederationToken
- GetSessionToken

Entidades principales y permisos

- Es posible que no pueda agregar, modificar o eliminar entidades principales o permisos de IAM.
- Es posible que sus credenciales no reflejen los cambios en los permisos que ha aplicado recientemente en IAM. Para obtener más información, consulte Los cambios que realizo no están siempre visibles inmediatamente.

AWS Management Console

- Es posible que pueda usar un punto de conexión de inicio de sesión regional para iniciar sesión en la AWS Management Console como usuario de IAM. Los puntos de conexión de inicio de sesión regionales tienen el siguiente formato de URL.

  `https://{Account ID}.signin.aws.amazon.com/console?region={Region}`

  Ejemplo: https://111122223333.signin.aws.amazon.com/console?region=us-west-2
- Es posible que no pueda completar la autenticación multifactor (MFA) de segundo factor universal (U2F).

Prácticas recomendadas para la resiliencia de IAM

AWS ha incorporado resiliencia en zonas de disponibilidad y Regiones de AWS. Cuando observa las siguientes prácticas recomendadas de IAM en los sistemas que interactúan con su entorno, aprovecha esa capacidad de resiliencia.

1. Utilice un punto de conexión de servicio regional de AWS STS en lugar del punto de conexión global predeterminado.
2. Revise la configuración de su entorno en busca de recursos vitales que creen o modifiquen rutinariamente los recursos de IAM, y prepare una solución alternativa que utilice los recursos de IAM existentes.

Seguridad de la infraestructura en AWS Identity and Access Management

Como servicio administrado, AWS Identity and Access Management está protegido por la seguridad de la red global de AWS. Para obtener información sobre los servicios de seguridad de AWS y cómo AWS protege la infraestructura, consulte Seguridad en la nube de AWS. Para diseñar su entorno de AWS con las prácticas recomendadas de seguridad de infraestructura, consulte Protección de la infraestructura en Portal de seguridad de AWS Well-Architected Framework.

Puede utilizar llamadas a la API publicadas en AWS para obtener acceso a IAM a través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Nosotros exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) tales como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad de seguridad de IAM. También puede utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Puede acceder a IAM de manera programática mediante la API IAM HTTPS, que le permite emitir solicitudes HTTPS directamente al servicio. La API de consultas devuelve información confidencial, incluidas las credenciales de seguridad. Por lo tanto, debe utilizar HTTPS con todas las solicitudes de API. Cuando utilice la API HTTPS, debe incluir código para firmar digitalmente las solicitudes utilizando sus credenciales.

Puede llamar a estas operaciones de la API desde cualquier ubicación de red, pero IAM admite políticas de acceso basadas en recursos, que pueden incluir restricciones en función de la dirección IP de origen. También puede utilizar políticas de IAM para controlar el acceso desde puntos de enlace específicos de Amazon Virtual Private Cloud (Amazon VPC) o VPC específicas. Este proceso aísla con eficacia el acceso de red a un recurso de IAM determinado únicamente desde la VPC específica de la red de AWS.

Configuración y análisis de vulnerabilidades en AWS Identity and Access Management

AWS gestiona las tareas de seguridad básicas, como la aplicación de parches en la base de datos y el sistema operativo (SO) de invitado, la configuración del firewall y la recuperación de desastres. Estos procedimientos han sido revisados y certificados por los terceros pertinentes. Para obtener más detalles, consulte los siguientes recursos de :

- Modelo de responsabilidad compartida
- Amazon Web Services: información general de procesos de seguridad (documento técnico)

Los siguientes recursos también abordan la configuración y el análisis de vulnerabilidad en AWS Identity and Access Management (IAM):

- Validación de conformidad en AWS Identity and Access Management
- Prácticas de seguridad recomendadas y casos de uso en AWS Identity and Access Management

Políticas administradas de AWS para el Analizador de acceso de AWS Identity and Access Management

Una política administrada de AWS es una política independiente que AWS crea y administra. Las políticas administradas de AWS se diseñan para ofrecer permisos para muchos casos de uso comunes, por lo que puede empezar a asignar permisos a los usuarios, grupos y roles.

Considere que es posible que las políticas administradas por AWS no concedan permisos de privilegio mínimo para los casos de uso concretos, ya que están disponibles para que las utilicen todos los clientes de AWS. Se recomienda definir políticas administradas por el cliente específicas para sus casos de uso a fin de reducir aún más los permisos.

No puede cambiar los permisos definidos en las políticas administradas de AWS. Si AWS actualiza los permisos definidos en una política administrada de AWS, la actualización afecta a todas las identidades de entidades principales (usuarios, grupos y roles) a las que está adjunta la política. Lo más probable es que AWS actualice una política administrada de AWS cuando se lance un nuevo Servicio de AWS o las operaciones de la API nuevas estén disponibles para los servicios existentes.

Para obtener más información, consulte Políticas administradas por AWS en la Guía del usuario de IAM.

IAMReadOnlyAccess

Para permitir el acceso de solo lectura en los recursos IAM, utilice la política administrada `IAMReadOnlyAccess`. Esta política concede permiso para obtener y enumerar todos los recursos de IAM. Permite ver detalles e informes de actividad para usuarios, grupos, roles, políticas, proveedores de identidad y dispositivos MFA. No incluye la capacidad de crear o eliminar recursos ni acceder a recursos del Analizador de acceso de IAM. Vea la política para conocer la lista completa de servicios y acciones que admite esta política.

IAMUserChangePassword

Utilice la política administrada de `IAMUserChangePassword` para permitir a los usuarios de IAM cambiar sus propias contraseñas.

La Configuración de cuenta de IAM y la Política de contraseñas se configuran para permitir a los usuarios de IAM cambiar la contraseña de la cuenta de IAM. Al permitir esta acción, IAM adjunta la siguiente política a cada usuario:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
```

```
        }
    ]
}
```

IAMAccessAnalyzerFullAccess

Utilizar la política administrada `IAMAccessAnalyzerFullAccess` de AWS para permitir que los administradores accedan al Analizador de acceso de IAM.

Grupos de permisos

Esta política se agrupa en instrucciones basadas en el conjunto de permisos proporcionados.

- Analizador de acceso de IAM: Autoriza permisos administrativos completos para todos los recursos del Analizador de acceso de IAM.
- Crear rol vinculado al servicio: Permite al administrador crear un rol vinculado al servicio, que permite que el Analizador de acceso de IAM analice los recursos de otros servicios en su nombre. Este permiso permite crear el rol vinculado al servicio solo para uso del Analizador de acceso de IAM.
- AWS Organizations — Permite a los administradores utilizar el Analizador de acceso de IAM para una organización en AWS Organizations. Después de habilitar el acceso de confianza para el Analizador de acceso de IAM en AWS Organizations, los miembros de la cuenta de administración pueden ver los resultados de su organización.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "access-analyzer:*"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:CreateServiceLinkedRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
```

```
          "iam:AWSServiceName": "access-analyzer.amazonaws.com"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots"
      ],
      "Resource": "*"
    }
  ]
}
```

IAMAccessAnalyzerReadOnlyAccess

Utilice la política administrada `IAMAccessAnalyzerReadOnlyAccess` AWS para permitir el acceso de solo lectura al Analizador de acceso de IAM.

Para permitir también acceso de solo lectura al Analizador de acceso de IAM para AWS Organizations, cree una política administrada por el cliente que permita las acciones Describir y Listar desde la política administrada IAMAccessAnalyzerFullAccess AWS.

Permisos de nivel de servicio

Esta política proporciona acceso de solo lectura al Analizador de acceso de IAM. No se incluyen otros permisos de servicio en esta política.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "IAMAccessAnalyzerReadOnlyAccess",
```

```
      "Effect": "Allow",
      "Action": [
        "access-analyzer:CheckAccessNotGranted",
        "access-analyzer:CheckNoNewAccess",
        "access-analyzer:Get*",
        "access-analyzer:List*",
        "access-analyzer:ValidatePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

AccessAnalyzerServiceRolePolicy

No puede adjuntar AccessAnalyzerServiceRolePolicy a sus entidades de IAM. Esta política está adjunta a un rol vinculado a servicios que permite al Analizador de acceso de IAM realizar acciones en su nombre. Para obtener más información, consulte Uso de roles vinculados a servicios para el Analizador de acceso de AWS Identity and Access Management.

Agrupaciones de permisos

Esta política permite el acceso al Analizador de acceso de IAM para analizar los metadatos de recursos de varios Servicios de AWS.

- Amazon DynamoDB: otorga permisos para ver los flujos y tablas de DynamoDB.
- Amazon Elastic Compute Cloud: otorga permisos para describir direcciones IP, instantáneas y VPC.
- Amazon Elastic Container Registry: Autoriza permisos para describir repositorios de imágenes y recuperar políticas de repositorios.
- Amazon Elastic File System: Autoriza permisos para ver la descripción de un sistema de archivos de Amazon EFS y ver la política de nivel de recursos de un sistema de archivos de Amazon EFS.
- AWS Identity and Access Management: Autoriza permisos para recuperar información sobre un rol especificado y enumerar los roles de IAM que tengan un prefijo de ruta especificado. Permite que los permisos recuperen información sobre los usuarios, los grupos de usuarios, los perfiles de inicio de sesión, las claves de acceso y los últimos datos del servicio a los que se accedió.
- AWS Key Management Service: Autoriza permisos para ver información detallada sobre una clave de KMS y sus principales políticas y autorizaciones.
- AWS Lambda: Autoriza permisos para ver información sobre alias, funciones y capas de Lambda.

- AWS Organizations: Autoriza permisos para Organizations y permite crear un analizador dentro de la organización AWS como zona de confianza.
- Amazon Relational Database Service: Autoriza permisos para ver información detallada sobre instantáneas de base de datos de Amazon RDS e instantáneas de clúster de base de datos de Amazon RDS.
- Amazon Simple Storage Service: otorga permisos para ver información detallada sobre los puntos de acceso de Amazon S3, los buckets y los buckets de directorio de Amazon S3 que utilizan la clase de almacenamiento de Amazon S3 Express One.
- AWS Secrets Manager: otorga permisos para ver información detallada sobre secretos y políticas de recursos adjuntadas a secretos.
- Amazon Simple Notification Service: otorga permisos para ver información detallada sobre un tema.
- Amazon Simple Queue Service: otorga permisos para ver información detallada sobre las colas especificadas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
        "iam:GenerateServiceLastAccessedDetails",
        "iam:GetAccessKeyLastUsed"
        "iam:GetGroup",
```

```
        "iam:GetLoginProfile",
        "iam:GetRole",
        "iam:GetServiceLastAccessedDetails",
        "iam:GetUser",
        "iam:ListAccessKeys",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
        "lambda:GetFunctionUrlConfig",
        "lambda:GetLayerVersionPolicy",
        "lambda:GetPolicy",
        "lambda:ListAliases",
        "lambda:ListFunctions",
        "lambda:ListLayers",
        "lambda:ListLayerVersions",
        "lambda:ListVersionsByFunction",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots",
        "rds:DescribeDBClusterSnapshotAttributes",
        "rds:DescribeDBClusterSnapshots",
        "rds:DescribeDBSnapshotAttributes",
        "rds:DescribeDBSnapshots",
        "s3:DescribeMultiRegionAccessPointOperation",
        "s3:GetAccessPoint",
        "s3:GetAccessPointPolicy",
        "s3:GetAccessPointPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicyStatus",
```

```
        "s3:GetBucketPolicy",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetMultiRegionAccessPoint",
        "s3:GetMultiRegionAccessPointPolicy",
        "s3:GetMultiRegionAccessPointPolicyStatus",
        "s3:ListAccessPoints",
        "s3:ListAllMyBuckets",
        "s3:ListMultiRegionAccessPoints",
        "s3express:GetBucketPolicy",
        "s3express:ListAllMyDirectoryBuckets",
        "sns:GetTopicAttributes",
        "sns:ListTopics",
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:ListSecrets",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues"
      ],
      "Resource": "*"
    }
  ]
}
```

IAM y el Analizador de acceso de IAM se actualizan a las políticas administradas de AWS

Es posible consultar los detalles sobre las actualizaciones de las políticas administradas de AWS e IAM, ya que el servicio comenzó a hacer el seguimiento de estos cambios. Para obtener alertas automáticas sobre cambios en esta página, suscríbase a las notificaciones RSS en las páginas del Historial de revisión de IAM y del Analizador de acceso de IAM.

Cambio	Descripción	Fecha
AccessAnalyzerServiceRolePolicy: permisos agregados	El analizador de acceso de IAM agregó compatibilidad con permisos para recuperar el estado actual del bloqueo	23 de enero de 2024

Cambio	Descripción	Fecha
	de acceso público a las instantáneas de Amazon EC2 a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` .	
AccessAnalyzerServiceRolePolicy: Permisos agregados	El analizador de acceso de IAM agregó compatibilidad con transmisiones y tablas de DynamoDB a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` .	11 de enero de 2024
AccessAnalyzerServiceRolePolicy: Permisos agregados	El analizador de acceso de IAM agregó compatibilidad con buckets de directorio de Amazon S3 a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` .	1 de diciembre de 2023
IAMAccessAnalyzerReadOnlyAccess: Permisos agregados	El Analizador de acceso de IAM agregó permisos para que usted pueda comprobar si las actualizaciones de sus políticas otorgan acceso adicional. El Analizador de acceso de IAM requiere este permiso para realizar comprobaciones de políticas en sus políticas.	26 de noviembre de 2023

Cambio	Descripción	Fecha
AccessAnalyzerServiceRolePolicy: Permisos agregados	El Analizador de acceso de IAM agregó acciones de IAM a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` para admitir las siguientes acciones: • Listado de las entidades de una política • Generación de información sobre los últimos accesos al servicio • Listado de Información de clave de acceso	26 de noviembre de 2023

Cambio	Descripción	Fecha
AccessAnalyzerServiceRolePolicy: Permisos agregados	El Analizador de acceso de IAM agregó compatibilidad con los siguientes tipos de recursos a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` : • Instantáneas de volúmenes de Amazon EBS • Repositorios de Amazon ECR • Sistemas de archivos de Amazon EFS • Instantáneas de base de datos de Amazon RDS • Instantáneas de clúster de base de datos de Amazon RDS • Temas de Amazon SNS	25 de octubre de 2022
AccessAnalyzerServiceRolePolicy: permisos agregados	El Analizador de acceso de IAM agregó la acción `lambda:GetFunctionUrlConfig` a los permisos de nivel de servicio de `AccessAnalyzerServiceRolePolicy` .	6 de abril de 2022

Cambio	Descripción	Fecha
AccessAnalyzerServiceRolePolicy: Permisos agregados	El Analizador de acceso de IAM agregó nuevas acciones de Amazon S3 para analizar los metadatos asociados a los puntos de acceso de varias regiones.	2 de septiembre de 2021
IAMAccessAnalyzerReadOnlyAccess: Permisos agregados	El Analizador de acceso de IAM agregó una nueva acción para conceder permisos de `ValidatePolicy` para permitirle utilizar las verificaciones de políticas para la validación. El Analizador de acceso de IAM requiere este permiso para realizar verificaciones de políticas en sus políticas.	16 de marzo de 2021
El Analizador de acceso de IAM ha iniciado el seguimiento de los cambios	El Analizador de acceso de IAM comenzó a realizar el seguimiento de los cambios de las políticas administradas por AWS.	1 de marzo de 2021

Uso AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer ofrece las siguientes funciones:

- Los analizadores de acceso externo del Analizador de acceso de IAM ayuda a identificar los recursos de su organización y sus cuentas que se comparten con una entidad externa.
- Los analizadores de acceso no utilizados del Analizador de acceso de IAM ayudan a identificar el acceso no utilizado en su organización y sus cuentas.
- El Analizador de acceso de IAM valida las políticas de IAM contra la gramática de las políticas y las AWS prácticas recomendadas.
- Las comprobaciones de políticas personalizadas del Analizador de acceso de IAM ayudan a validar las políticas de IAM frente a los estándares de seguridad especificados.
- El Analizador de acceso de IAM genera políticas de IAM basado en la actividad de acceso de registros de AWS CloudTrail.

Identificar los recursos compartidos con una entidad externa

El Analizador de acceso de IAM le ayuda a identificar los recursos de su organización y sus cuentas, como buckets de Amazon S3 o roles de IAM, que se comparten con una entidad externa. Esto le permite identificar el acceso no deseado a sus recursos y datos, lo que constituye un riesgo para la seguridad. El Analizador de acceso de IAM identifica los recursos compartidos con entidades principales externas mediante el uso de un razonamiento lógico para analizar las políticas basadas en recursos en su entorno de AWS. Para cada instancia de un recurso compartido fuera de su cuenta, el Analizador de acceso de IAM genera un resultado. Los resultados incluyen información sobre el acceso y la entidad principal externa a la que se le concede. Puede revisar los resultados para determinar si el acceso es intencionado y seguro, o si el acceso es no intencionado y supone un riesgo para la seguridad. Además de ayudarle a identificar los recursos compartidos con una entidad externa, puede utilizar los resultados del Analizador de acceso de IAM para previsualizar cómo afecta su política al acceso público y entre cuentas a su recurso antes de implementar los permisos de recursos. Los resultados se organizan en un panel de resumen visual. El panel destaca la división entre los resultados de acceso entre cuentas y público, y proporciona un desglose de los resultados por tipo de recurso. Para obtener más información sobre los paneles, consulte Visualización del panel de resultados del Analizador de acceso de IAM.

> **Note**
>
> Una entidad externa puede ser otra cuenta de AWS, un usuario raíz, un usuario o rol de IAM, un usuario federado, un servicio de AWS, un usuario anónimo u otra entidad que puede utilizar para crear un filtro. Para obtener más información, consulte Elementos de la política de JSON de AWS: Principal.

Cuando se habilita el Analizador de acceso de IAM, se crea un analizador para toda la organización o su cuenta. La organización o cuenta que elija se conoce como la zona de confianza del analizador. El analizador monitorea todos los recursos admitidos dentro de su zona de confianza. Cualquier acceso a los recursos por parte de entidades principales que se encuentren dentro de su zona de confianza se considera de confianza. Una vez habilitadas, el Analizador de acceso de IAM analiza las políticas aplicadas a todos los recursos admitidos en su zona de confianza. Después del primer análisis, el Analizador de acceso de IAM analiza estas políticas periódicamente. Si se agrega una política nueva o se cambia una política existente, el Analizador de acceso de IAM analiza la política nueva o actualizada en unos 30 minutos.

Al analizar las políticas, si el Analizador de acceso de IAM identifica una que concede acceso a una entidad principal externa que no está dentro de su zona de confianza, genera un resultado. Cada resultado incluye detalles sobre el recurso, la entidad externa que tiene acceso al mismo y los permisos concedidos para que pueda tomar las medidas adecuadas. Puede ver los detalles incluidos en el resultado para determinar si el acceso a los recursos es intencional o si es un riesgo potencial que debe resolver. Cuando agrega una política a un recurso o actualiza una política existente, el Analizador de acceso de IAM analiza la política. El Analizador de acceso de IAM también analiza periódicamente todas las políticas basadas en recursos.

En raras ocasiones y bajo determinadas condiciones, el Analizador de acceso de IAM no recibe ninguna notificación de una política agregada o actualizada, lo que puede ocasionar demoras en la generación de resultados. El Analizador de acceso de IAM puede tardar hasta 6 horas en generar o resolver resultados si crea o elimina un punto de acceso de región múltiple asociado a un bucket de Amazon S3 o actualiza la política para el punto de acceso de varias regiones. Además, si hay un problema con la entrega de registros de AWS CloudTrail, el cambio de política no activa un nuevo análisis del recurso que se comunicó en el resultado. Cuando esto sucede, el Analizador de acceso de IAM analiza la política nueva o actualizada durante el siguiente análisis periódico, que es dentro de las 24 horas. Si desea confirmar que un cambio que realice en una política resuelve un problema de acceso notificado en un resultado, puede volver a examinar el recurso notificado en un resultado mediante el enlace Rescan (Volver a examinar) en la página de detalles de los Resultados

o mediante la operación `StartResourceScan` de la API del Analizador de acceso de IAM. Para obtener más información, consulte Resolución de resultados.

> **Important**
>
> El Analizador de acceso de IAM analiza solo las políticas que se aplican a los recursos de la misma región de AWS en la que está habilitada. Para supervisar todos los recursos de su entorno de AWS, debe crear un analizador para habilitar el Analizador de acceso de IAM en cada región en la que utilice los recursos de AWS admitidos.

El Analizador de acceso de IAM analiza los siguientes tipos de recursos:

- Buckets de Amazon Simple Storage Service Batch
- Buckets de directorio de Amazon Simple Storage Service Batch
- Roles de AWS Identity and Access Management
- Claves de AWS Key Management Service
- Funciones y capas de AWS Lambda
- Colas de Amazon Simple Queue Service
- AWS Secrets Manager secretos
- Temas de Amazon Simple Notification Service
- Instantáneas de volúmenes de Amazon Elastic Block Store
- Instantáneas de base de datos de Amazon Relational Database Service
- Instantáneas de clúster de base de datos de Amazon Relational Database Service
- Repositorios de Amazon Elastic Container Registry
- Sistemas de archivos de Amazon Elastic File System
- Amazon DynamoDB Streams
- Tablas de Amazon DynamoDB

Identificar el acceso no utilizado otorgado a roles y usuarios de IAM

IAM Access Analyzer le ayuda a identificar y revisar el acceso no utilizado en su organización de AWS y sus cuentas. IAM Access Analyzer supervisa continuamente todos los usuarios y roles de IAM en su organización de AWS y sus cuentas, y genera resultados sobre el acceso no utilizado.

Los resultados destacan los roles no utilizados, las claves de acceso no utilizadas de los usuarios de IAM y las contraseñas no utilizadas de los usuarios de IAM. En el caso de los usuarios y roles de IAM activos, los resultados proporcionan visibilidad de los servicios y las acciones no utilizados.

Los resultados, tanto de los analizadores de acceso externo como de los no utilizados, se organizan en un panel de resumen visual. El panel destaca los Cuentas de AWS que tienen más resultados y proporciona un desglose de los resultados por tipo. Para obtener más información acerca del panel, consulte Visualización del panel de resultados del Analizador de acceso de IAM.

IAM Access Analyzer revisa la información a la que se accedió por última vez para todos los roles de su organización de AWS y sus cuentas para ayudarlo a identificar los accesos no utilizados. La información sobre las últimas acciones de IAM a las que se accedió le ayuda a identificar las acciones no utilizadas para los roles de su Cuentas de AWS. Para obtener más información, consulte Perfeccionar los permisos con la información sobre los últimos accesos en AWS.

Validar las políticas comparándolas con las prácticas recomendadas de AWS

Puede validar sus políticas con respecto a la gramática de políticas de IAM y las prácticas recomendadas de AWS mediante las comprobaciones básicas de políticas que proporciona la validación de políticas del Analizador de acceso de IAM. Puede crear o editar una política con la AWS CLI, API de AWS o editor de políticas JSON en la consola de IAM. Puede ver los resultados de las verificaciones de validación de políticas que incluyen advertencias de seguridad, errores, advertencias generales y sugerencias para la política. Estos resultados proporcionan recomendaciones procesables que le ayudan a crear políticas funcionales y que se ajustan a las prácticas recomendadas de AWS. Para obtener más información sobre cómo validar políticas mediante validación de políticas, consulte Política de validación de Analizador de acceso de IAM.

Validar las políticas según los estándares de seguridad especificados

Puede validar sus políticas con respecto a los estándares de seguridad especificados mediante las comprobaciones de políticas personalizadas del Analizador de acceso de IAM. Puede crear o editar una política con la AWS CLI, API de AWS o editor de políticas JSON en la consola de IAM. A través de la consola, puede comprobar si la política actualizada concede nuevos accesos en comparación con la versión existente. A través de AWS CLI y de la API de AWS, también puede

comprobar si determinadas acciones de IAM que considera críticas no están permitidas por una política. Estas verificaciones destacan una declaración de las políticas que otorga un nuevo acceso. Puede actualizar la declaración de las políticas y volver a ejecutar las comprobaciones hasta que la política se ajuste a su estándar de seguridad. Para obtener más información sobre cómo validar políticas mediante las comprobaciones de políticas personalizadas, consulte Comprobaciones de políticas personalizadas del Analizador de acceso de IAM.

Generación de políticas

El Analizador de acceso de IAM analiza los registros de AWS CloudTrail para identificar acciones y servicios que han sido utilizados por una entidad (usuario o rol) de IAM dentro de un rango de fecha especificado. A continuación, genera una política de IAM basada en esa actividad de acceso. Puede utilizar la política generada para refinar los permisos de una entidad adjuntándola a un usuario o rol de IAM. Para obtener más información sobre cómo generar políticas mediante el Analizador de acceso de IAM, consulte Generación de políticas del Analizador de acceso de IAM.

Precios del IAM Access Analyzer

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por el analizador por mes.

- Se le cobrará por cada analizador de acceso no utilizado que genere.
- Si crea analizadores de acceso no utilizados en varias regiones, se le cobrará por cada analizador.
- Los roles vinculados a servicios no se analizan para la actividad de acceso no utilizada y no se incluyen en la cantidad total de roles de IAM analizados.

El Analizador de acceso de IAM cobra por las comprobaciones de políticas personalizadas en función del número de solicitudes de API realizadas al Analizador de acceso de IAM para comprobar si hay nuevos accesos.

Para obtener una lista completa de los costos y precios del Analizador de acceso de IAM, consulte Analizador de acceso de IAM.

Para ver su factura, vaya al Panel de Billing and Cost Management en la consola de AWS Billing and Cost Management. La factura contiene vínculos a informes de uso que ofrecen detalles sobre la cuenta. Para obtener más información sobre Cuenta de AWS facturación, consulte la AWS BillingGuía del usuario.

Si tiene alguna pregunta acerca de los eventos, las cuentas y la facturación de AWS, póngase en contacto con AWS Support.

Resultados de acceso externo y no utilizado

El Analizador de acceso de IAM genera resultados sobre el acceso externo y no utilizado en la Cuenta de AWS u organización. Para acceso externo, el Analizador de acceso de IAM genera un resultado para cada instancia de una política basada en recursos que concede a un recurso dentro de su zona de confianza a una entidad principal que no está dentro de su zona de confianza. Cuando crea un analizador de acceso externo, elige una organización o Cuenta de AWS para analizar. Cualquier entidad principal de la organización o cuenta que elija para el analizador se considera de confianza. Dado que las entidades principales de la misma organización o cuenta son de confianza, los recursos y las entidades principales de la organización o cuenta comprenden la zona de confianza del analizador. Cualquier uso compartido que esté dentro de la zona de confianza se considera seguro, por lo que el Analizador de acceso de IAM no genera un resultado. Por ejemplo, si selecciona una organización como zona de confianza para un analizador, todos los recursos y entidades principales de la organización se encuentran dentro de la zona de confianza. Si concede permisos a bucket de Amazon S3 en una de las cuentas de miembro de la organización a una entidad principal en otra cuenta de miembro de la organización, el Analizador de acceso de IAM no genera un resultado. Pero si concede permiso a una entidad principal de una cuenta que no es miembro de la organización, el Analizador de acceso de IAM genera un resultado.

El Analizador de acceso de IAM también genera resultados sobre el acceso concedido no utilizado en la organización de AWS. Cuando crea un analizador de acceso no utilizado, el Analizador de acceso de IAM supervisa continuamente todos los usuarios y roles de IAM en AWS de la organización y las cuentas, y genera resultados sobre el acceso no utilizado. El Analizador de acceso de IAM genera los siguientes tipos de resultados para el acceso no utilizado:

- Roles no utilizados: Roles sin actividad de acceso dentro del período de uso especificado.
- Claves de acceso y contraseñas de los usuarios de IAM no utilizadas: Credenciales que pertenecen a los usuarios de IAM y que les permiten acceder a sus Cuenta de AWS.
- Permisos no utilizados: Permisos de nivel de servicio y de nivel de acción que un rol no utilizó dentro del período de uso especificado. El Analizador de acceso de IAM utiliza políticas basadas en la identidad asociadas a los roles para determinar los servicios y las acciones a los que pueden acceder esos roles. El Analizador de acceso de IAM permite revisar los permisos no utilizados para todos los permisos de nivel de servicio. Para obtener una lista completa de los permisos de nivel

de acción que se admiten para resultados de acceso no utilizado, consulte Servicios y acciones de la información sobre los últimos accesos a la acción de IAM.

> **Note**
>
> El Analizador de acceso de IAM ofrece resultados de acceso externo gratuitos y cobra por los resultados de acceso no utilizado en función del número de usuarios y roles de IAM analizados por el analizador por mes. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

Temas

Cómo funcionan los resultados del Analizador de acceso de IAM

En este tema, se describen los conceptos y términos que se utilizan en el Analizador de acceso de IAM para ayudarle a familiarizarse con la forma en que el Analizador de acceso de IAM supervisa el acceso a los recursos de AWS.

Acceso externo

Para los analizadores de acceso externo, AWS Identity and Access Management Access Analyzer se basa en Zelkova, que traduce las políticas de IAM en declaraciones lógicas equivalentes, y ejecuta un conjunto de solucionadores lógicos de uso general y especializados (teorías de módulo de satisfacibilidad) frente al problema. El Analizador de acceso de IAM aplica Zelkova repetidamente a una política con consultas cada vez más específicas para caracterizar las clases de comportamientos que permite la política, en función del contenido de la política. Para obtener más información sobre las teorías de satisfacibilidad módulo, consulte Teorías de satisfacibilidad módulo.

Para los analizadores externos, el Analizador de acceso de IAM no examina los registros de acceso para determinar si una entidad externa accedió a un recurso dentro de su zona de confianza. Genera un resultado cuando una política basada en recursos permite el acceso a un recurso, incluso si la entidad externa no accedió al recurso. El Analizador de acceso de IAM tampoco tiene en cuenta el estado de ninguna cuenta externa al realizar su determinación. Es decir, si indica que la cuenta 111122223333 puede acceder a su bucket de Amazon S3, no sabe nada sobre el estado de los usuarios, los roles, las políticas de control de servicios (SCP) y otras configuraciones relevantes en esa cuenta. Esto es para la privacidad del cliente: El Analizador de acceso de IAM no tiene en cuenta quién es el propietario de la otra cuenta. También es por seguridad: Si la cuenta no es propiedad del cliente del Analizador de acceso de IAM, todavía es importante saber que una entidad externa podría obtener acceso a sus recursos incluso si actualmente no hay entidades principales en la cuenta que puedan acceder a los recursos.

El Analizador de acceso de IAM solo tiene en cuenta ciertas claves de condición de IAM en las que los usuarios externos no pueden influir directamente o que en caso contrario afectan la autorización. Para obtener ejemplos de claves de condición que el Analizador de acceso de IAM considera, consulte Claves de filtro del Analizador de acceso de IAM.

El Analizador de acceso de IAM no notifica actualmente los resultados de entidades principales de servicios de AWS ni de las cuentas de servicio internas. En casos raros en los que el Analizador de acceso de IAM no es capaz de determinar completamente si una declaración de política concede acceso a una entidad externa, se equivoca al declarar un resultado falso positivo. El Analizador de

acceso de IAM está diseñado para proporcionar una vista completa del uso compartido de recursos en su cuenta y se esfuerza por minimizar los falsos negativos.

Acceso sin utilizar

Debe crear un analizador para los resultados de acceso no utilizado para sus roles, incluso si ya ha creado un analizador para generar los resultados de acceso externo para sus recursos. Tras crear el analizador, el Analizador de acceso de IAM revisa la actividad de acceso para identificar el acceso no utilizado. El Analizador de acceso de IAM revisa la información a la que se accedió por última vez para todos los roles de su organización, claves de acceso y contraseñas de usuarios de su organización de AWS y sus cuentas para ayudarle a identificar los accesos no utilizados. En el caso de los usuarios y roles de IAM activos, el Analizador de acceso de IAM utiliza la información del último servicio y acción de IAM a los que se accedió para identificar los permisos no utilizados. Puede utilizar analizadores de acceso no utilizados para escalar el proceso de revisión a nivel de la organización de AWS y de la cuenta. Puede utilizar la información sobre la acción a la que se accedió por última vez para investigar más a fondo los roles individuales.

Panel Resumen

Tanto para los accesos externos como para los no utilizados, el Analizador de acceso de IAM organiza los resultados en un panel de resumen. Para el acceso externo, el panel de resumen destaca la división entre los resultados de acceso entre cuentas y público, y proporciona un desglose de los resultados por tipo de recurso. Para el acceso no utilizado, el panel destaca sus Cuentas de AWS que tienen más resultados y proporciona un desglose de los resultados por tipo. Tras crear un analizador para el acceso externo o no utilizado, el Analizador de acceso de IAM agrega automáticamente los nuevos resultados al panel de control, centrándose en las funciones con permisos no utilizados.

Introducción a los resultados de AWS Identity and Access Management Access Analyzer

Utilice la información de este tema para obtener información acerca de los requisitos necesarios para utilizar y administrar el Analizador de acceso de IAM AWS Identity and Access Management Access Analyzer y, a continuación, cómo habilitar el Analizador de acceso de IAM. Para obtener más información sobre el rol vinculado a servicios del Analizador de acceso de IAM, consulte Uso de roles vinculados a servicios de AWS Identity and Access Management Access Analyzer.

Permisos requeridos para utilizar el Analizador de acceso de IAM

Para configurar y utilizar correctamente el Analizador de acceso de IAM, se deben conceder los permisos necesarios a la cuenta que utilice.

Políticas de AWS administradas para el Analizador de acceso de IAM

AWS Identity and Access Management Access Analyzer brinda políticas administradas de AWS para ayudarle a comenzar a trabajar rápidamente.

- IAMAccessAnalyzerFullAccess: Permite el acceso completo al Analizador de acceso de IAM para los administradores. Esta política también permite crear los roles vinculados a servicios que son necesarios para permitir que el Analizador de acceso de IAM analice los recursos de su cuenta u organización de AWS.
- IAMAccessAnalyzerReadOnlyAccess: Permite el acceso de solo lectura al Analizador de acceso de IAM. Debe agregar políticas adicionales a sus identidades de IAM (usuarios, grupos de usuarios o roles) para permitirles ver sus resultados.

Recursos definidos por el Analizador de acceso de IAM

Para ver los recursos definidos por el Analizador de acceso de IAM, consulte Tipos de recursos definidos por el Analizador de acceso de IAM en la Referencia de autorización de servicios.

Permisos de servicio del Analizador de acceso de IAM requeridos

El Analizador de acceso de IAM usa un Rol vinculado al servicio (SLR) llamado `AWSServiceRoleForAccessAnalyzer`. Este SLR concede al servicio acceso de solo lectura para analizar recursos de AWS con políticas basadas en los recursos y analizar en su nombre el acceso no utilizado. El servicio crea el rol de su cuenta en los siguientes casos:

- Crea un analizador de acceso externo con su cuenta como la zona de confianza.
- Crea un analizador de acceso no utilizado con su cuenta como la cuenta seleccionada.

Para obtener más información, consulte Uso de roles vinculados a servicios de AWS Identity and Access Management Access Analyzer.

> **Note**
>
> El Analizador de acceso de IAM es regional. Para acceso externo, debe activar Analizador de acceso de IAM en cada región de forma independiente.
> En el caso del acceso no utilizado, los resultados del analizador no cambian en función de la región. No es necesario crear un analizador en cada región en la que tenga recursos.

En algunos casos, después de crear un analizador de acceso externo o de acceso no utilizado en el Analizador de acceso de IAM, la página o el panel Resultados se carga sin ningún resultado ni resumen. Esto puede deberse a un retraso en la consola para rellenar sus resultados. Es posible que tenga que actualizar manualmente el navegador o volver a consultarlo más adelante para ver sus resultados o su resumen. Si sigue sin ver ningún resultado, es porque no tiene recursos compatibles en su cuenta a los que pueda acceder una entidad externa. Si una política que concede acceso a una entidad externa se aplica a un recurso, el Analizador de acceso de IAM genera un resultado.

> **Note**
>
> Para analizadores de acceso externo, puede tardar hasta 30 minutos después de modificar una política para que el Analizador de acceso de IAM analice el recurso y, a continuación, genere un nuevo resultado o actualice un resultado existente para el acceso al recurso.
> Tanto en el caso de los analizadores de acceso externos como en los que no se utilizan, es posible que las actualizaciones de los resultados no se reflejen inmediatamente en el panel de control.

Se requieren permisos del Analizador de acceso de IAM para ver el panel de resultados

Para ver el panel de resultados del Analizador de acceso de IAM, se deben conceder accesos a la cuenta que utilice para realizar las siguientes acciones necesarias:

- `GetAnalyzer`
- `ListAnalyzers`
- `GetFindingsStatistics`

Para ver todas las acciones definidas por el Analizador de acceso de IAM, consulte Acciones definidas por el Analizador de acceso de IAM en la Referencia de autorización de servicios.

Habilitación del Analizador de acceso de IAM

Para crear un analizador de acceso externo con la cuenta Cuenta de AWS como la zona de confianza

Para activar el analizador de acceso externo, debe crear un analizador en dicha región. Debe crear un analizador de acceso externo en cada región en la que desee monitorizar el acceso a los recursos.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso.
3. Elija Configuración del analizador.
4. Elija Create analyzer (Crear analizador).
5. En la sección Análisis, elija Análisis de acceso externo.
6. En la sección Información del analizador confirme que la región mostrada es la región en la que desea activar el Analizador de acceso de IAM.
7. Escriba un nombre para el analizador.
8. Elija Cuenta de AWSActual como zona de confianza para el analizador.

> **Note**
>
> Si su cuenta no es la de administración de AWS Organizations o la de administrador delegado, puede crear un solo analizador con su cuenta como zona de confianza.

9. Opcional. Agregue las etiquetas que desee aplicar al analizador.
10. Elija Enviar.

Cuando crea un analizador de acceso externo para activar el Analizador de acceso de IAM, se crea en su cuenta un rol vinculado a servicios denominado `AWSServiceRoleForAccessAnalyzer`.

Para crear un analizador de acceso externo con la organización como zona de confianza

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso.
3. Elija Configuración del analizador.
4. Elija Create analyzer (Crear analizador).

5. En la sección Análisis, elija Análisis de acceso externo.
6. En la sección Información del analizador confirme que la región mostrada es la región en la que desea activar el Analizador de acceso de IAM.
7. Escriba un nombre para el analizador.
8. Elija Organización actual como la zona de confianza del analizador.
9. Opcional. Agregue las etiquetas que desee aplicar al analizador.
10. Elija Enviar.

Cuando crea un analizador de acceso externo con la organización como zona de confianza, se crea un rol vinculado a servicios denominado `AWSServiceRoleForAccessAnalyzer` en cada cuenta de la organización.

Para crear un analizador de acceso no utilizado para la cuenta actual

Utilice el siguiente procedimiento para crear un analizador de acceso no utilizado para una sola Cuenta de AWS. En el caso del acceso no utilizado, los resultados del analizador no cambian en función de la región. No es necesario crear un analizador en cada región en la que tenga recursos.

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por mes y por analizador. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM .

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso.
3. Elija Configuración del analizador.
4. Elija Create analyzer (Crear analizador).
5. En la sección Análisis, elija Análisis de acceso no utilizado.
6. Escriba un nombre para el analizador.
7. En el Período de seguimiento, introduzca el número de días durante los que se generarán los resultados sobre los permisos no utilizados. Por ejemplo, si introduce 90 días, el analizador generará los resultados de las entidades de IAM de la cuenta seleccionada en relación con los permisos que no se hayan utilizado en 90 días o más desde el último escaneo del analizador. Puede elegir un valor comprendido entre 1 y 180 días.
8. Para las Cuentas seleccionadas, elija Cuenta de Cuenta de AWS actual.

> **Note**
>
> Si su cuenta no es la AWS Organizations cuenta de administrador o cuenta de administrador delegado, puede crear un solo analizador con su cuenta como la cuenta seleccionada.

9. Opcional. Agregue las etiquetas que desee aplicar al analizador.
10. Elija Enviar.

Cuando crea un analizador de acceso no utilizado para activar el Analizador de acceso de IAM, se crea en su cuenta un rol vinculado a servicios denominado `AWSServiceRoleForAccessAnalyzer`.

Para crear un analizador de acceso no utilizado con la organización actual

Utilice el siguiente procedimiento para crear un analizador de acceso no utilizado para que una organización revise de forma centralizada todas las Cuentas de AWS de una organización. En el análisis de acceso no utilizado, los resultados del analizador no cambian en función de la región. No es necesario crear un analizador en cada región en la que tenga recursos.

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por mes y por analizador. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM .

> **Note**
>
> Si se elimina la cuenta de un miembro de la organización, el analizador de acceso no utilizado dejará de generar nuevos resultados y de actualizar los resultados existentes para esa cuenta después de 24 horas. Los resultados asociados a la cuenta de miembro que se elimine de la organización se eliminarán permanentemente después de 90 días.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso.
3. Elija Configuración del analizador.
4. Elija Create analyzer (Crear analizador).

5. En la sección Análisis, elija Análisis de acceso no utilizado.
6. Escriba un nombre para el analizador.
7. En el Período de seguimiento, introduzca el número de días durante los que se generarán los resultados sobre los permisos no utilizados. Por ejemplo, si introduce 90 días, el analizador generará resultados sobre las entidades de IAM de las cuentas de la organización seleccionada en relación con los permisos que no se hayan utilizado en 90 días o más desde el último análisis realizado por el analizador. Puede elegir un valor comprendido entre 1 y 180 días.
8. En Cuentas seleccionadas, elija Organización actual como las cuentas seleccionadas para el analizador.
9. Opcional. Agregue las etiquetas que desee aplicar al analizador.
10. Elija Enviar.

Cuando crea un analizador de acceso no utilizado para activar el Analizador de acceso de IAM, se crea en su cuenta un rol vinculado a servicios denominado `AWSServiceRoleForAccessAnalyzer`.

Estado del Analizador de acceso de IAM

Para ver el estado de los analizadores, elija Analyzers (Analizadores). Los analizadores creados para una organización o cuenta pueden tener el siguiente estado:

Estado	Descripción
Activa	Para analizadores de acceso externo, el analizador monitoriza activamente los recursos dentro de su zona de confianza. El analizador genera activamente nuevos resultados y actualiza los resultados existentes. En el caso de los analizadores de acceso no utilizados, el analizador supervisa activamente el acceso no utilizado dentro de la organización seleccionada o Cuenta de AWS durante el período de seguimiento especificado. El analizador genera activamente nuevos resultados y actualiza los resultados existentes.

Estado	Descripción
Creación	La creación del analizador todavía está en curso. El analizador se activa una vez completada la creación.
Deshabilitado	El analizador está deshabilitado debido a una acción realizada por el administrador de AWS Organizations. Por ejemplo, quitar la cuenta del analizador como administrador delegado para el Analizador de acceso de IAM. Cuando el analizador está en un estado desactivado, no genera nuevos resultados ni actualiza los resultados existentes.
Con error	Error al crear el analizador debido a un problema de configuración. El analizador no generará ningún resultado. Elimine el analizador y cree un nuevo analizador.

Visualización del panel de resultados del Analizador de acceso de IAM

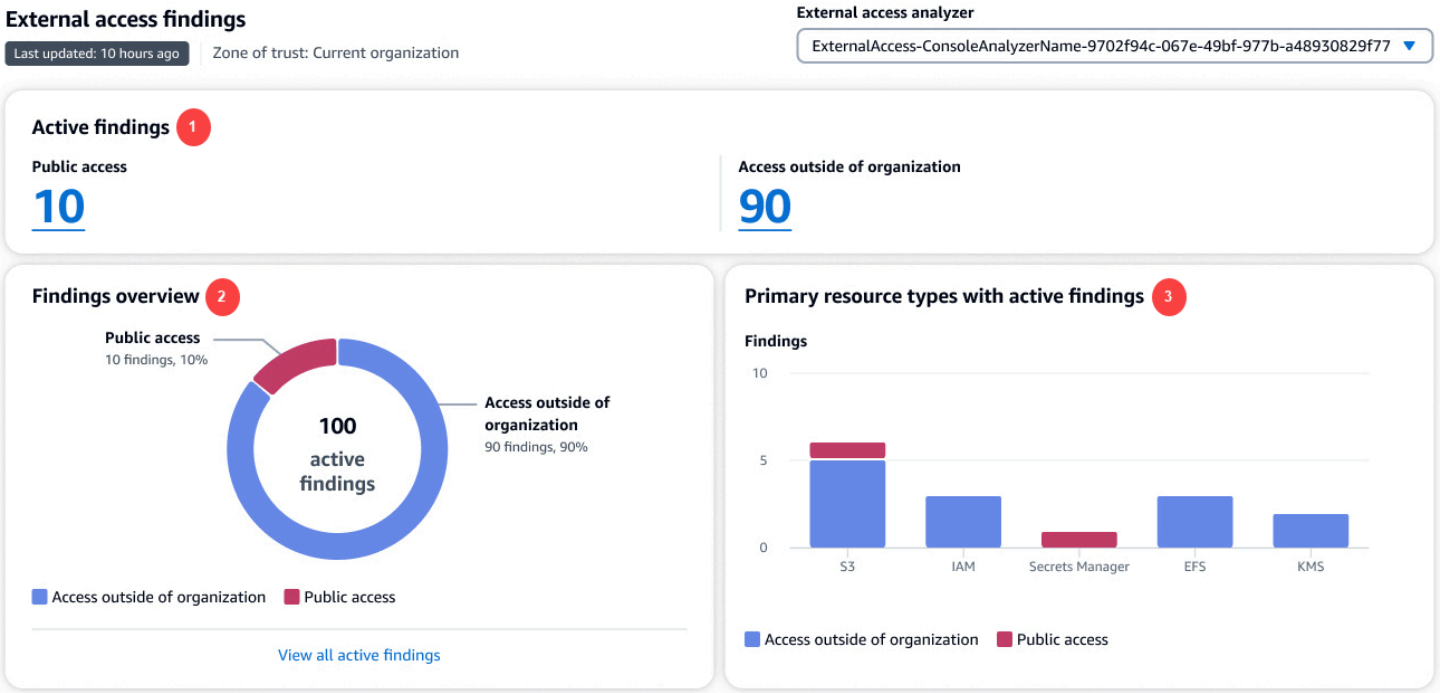
AWS Identity and Access Management Access Analyzer organiza acceso externo y no utilizados, se organizan en un panel de resumen visual. El panel le ayuda a obtener visibilidad sobre el uso efectivo de los permisos a gran escala y a identificar las cuentas que requieren atención. Puede usar el panel para revisar los resultados por la organización AWS, la cuenta y el tipo de resultado.

Ver el panel de resumen de los analizadores de acceso externo

> **Note**
>
> Tras crear o actualizar un analizador, el panel de resumen puede tardar un tiempo en reflejar las actualizaciones de los resultados.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso. Aparece la ventana Resumen.

3. Elija un analizador en el menú desplegable del Analizador de acceso externo. Un resumen de los resultados del analizador se muestra en la sección Resultados del acceso externo.



En la imagen anterior, el panel de resultados de acceso externo está visible desde la página Resumen:

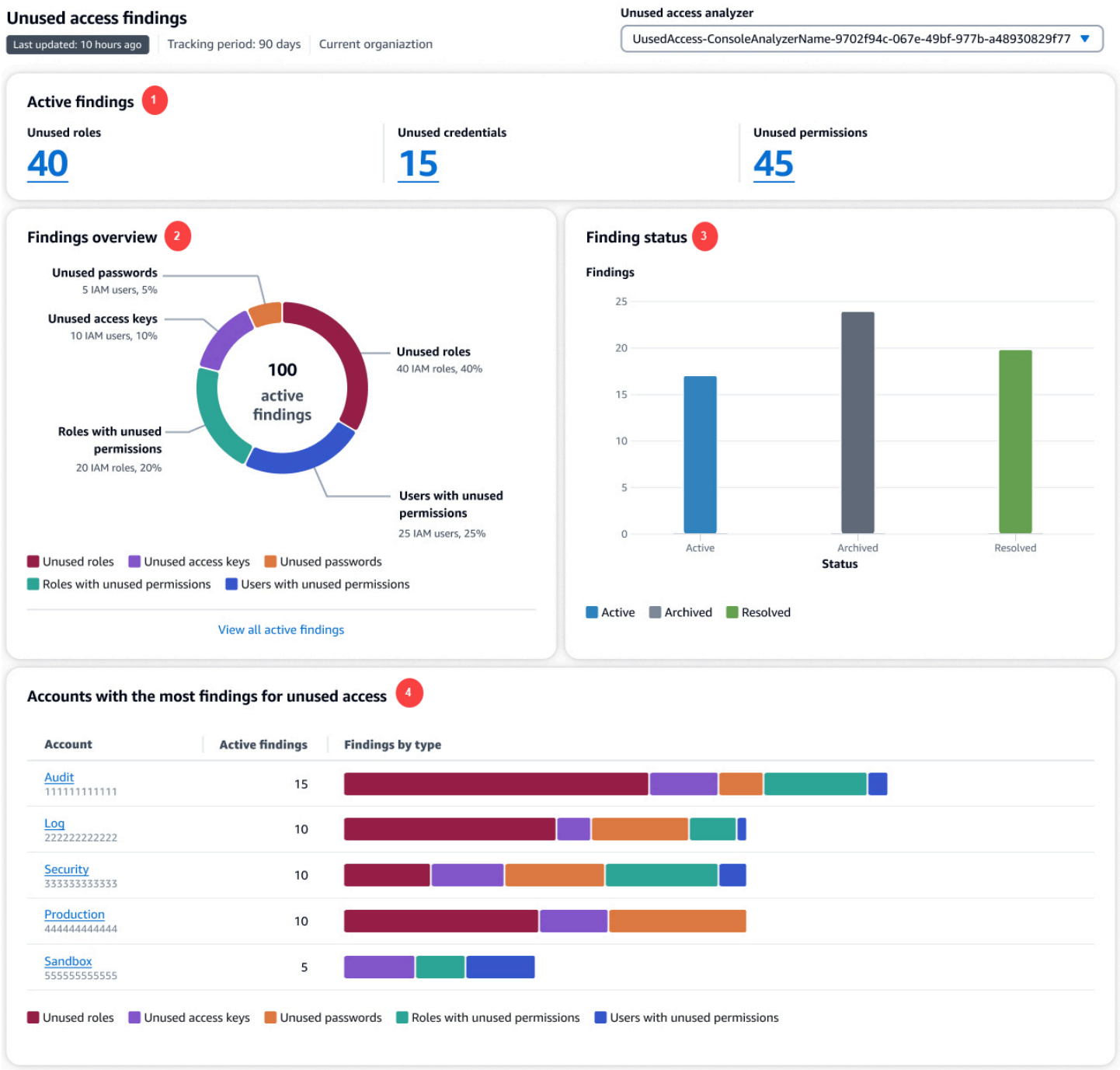
1. La sección de Resultados activos incluye el número de resultados activos para el acceso público y el número de resultados activos a los que se puede acceder desde fuera de la cuenta o la organización. Elija un número para enumerar todos los resultados activos de cada tipo.
2. La sección de Resumen de los resultados incluye un desglose del tipo de resultados activos. Seleccione Ver todos los resultados activos para obtener una lista completa de los hallazgos activos para la organización o la cuenta del analizador.
3. La sección Tipos de recursos principales con resultados activos incluye un desglose de los tipos de recursos principales con resultados activos. Esta información le ayuda a priorizar los resultados principales para los recursos principales. Por ejemplo, Amazon S3, DynamoDB y AWS KMS. No es una lista exhaustiva de todos los tipos de recursos. Es posible que su analizador tenga resultados activos para tipos de recursos que no figuran en esta sección.

Ver el panel de resumen de los analizadores de acceso no utilizados

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por mes. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

> **Note**
>
> Tras crear o actualizar un analizador, en función de la cantidad de usuarios y funciones, el panel de resumen puede tardar un tiempo en reflejar las actualizaciones de los resultados.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso. Aparece la ventana Resumen.
3. Elija un analizador en el menú desplegable del Analizador de acceso externo. Un resumen de los resultados del analizador se muestra en la sección Resultados del acceso no utilizado.



En la imagen anterior, el panel de resultados de acceso externo está visible desde la página Resumen:

1. La sección de Resultados activos incluye el número de resultados activos relacionados con las funciones, las credenciales y los permisos no utilizados en su cuenta u organización. Las credenciales no utilizadas incluyen tanto la clave de acceso como la contraseña no utilizada. Los

permisos no utilizados incluyen tanto a los usuarios como a los roles con permisos no utilizados. Elija un número para enumerar todos los resultados activos de cada tipo.

2. La sección de Resumen de los resultados incluye un desglose del tipo de resultados activos. Seleccione Ver todos los resultados activos para obtener una lista completa de los resultados activos para la organización o la cuenta del analizador.
3. La sección Estado de los resultados incluye un desglose del estado de los resultados (Activo, Archivado y Resuelto) de su cuenta u organización.
4. La sección Cuentas con más resultados de acceso no utilizado solo se muestra si las cuentas seleccionadas del analizador de acceso no utilizado pertenecen a la organización. Incluye un desglose de las cuentas de su organización con los resultados más activos. No es una lista exhaustiva de todas las cuentas de su organización. Es posible que su analizador tenga resultados activos para otras cuentas que no figuran en esta sección.

Trabajar con resultados

Resultados de acceso externo

Los resultados se generan una sola vez para cada instancia de un recurso que se comparte fuera de su zona de confianza. Cada vez que se modifica una política basada en recursos, el Analizador de acceso de IAM analiza la política. Si la política actualizada comparte un recurso que ya se ha identificado en un resultado, pero con permisos o condiciones distintos, se genera un nuevo resultado para esa instancia del recurso compartido. Si se elimina el acceso en el primer resultado, ese resultado se actualiza a un estado de Resuelto.

El estado de todos los resultados permanece Activo hasta que los archive o elimine el acceso que generó el resultado. Al quitar el acceso, el estado del resultado se actualiza a Resuelto.

> **Note**
>
> Puede tardar hasta 30 minutos después de modificar una política para que el Analizador de acceso de IAM analice el recurso y, a continuación, actualice el resultado.

Resultados de acceso sin utilizar

Los resultados de acceso no utilizados se generan para las entidades de IAM de la cuenta u organización seleccionada en función del número de días especificado al crear el analizador. La

próxima vez que el analizador escanee las entidades, se generará un nuevo resultado si se cumple una de las siguientes condiciones:

- Un rol permanece inactivo durante el número especificado de días.
- Un permiso, una contraseña de usuario o una clave de acceso de usuario no utilizados superan el número de días especificado.

Debe revisar todos los resultados de su cuenta para determinar si el uso compartido está previsto y aprobado. Si el acceso externo o no utilizado identificado en el resultado es esperado, puede archivar el resultado. Al archivar un resultado, el estado cambia a Archivado y el resultado se elimina de la lista de resultados Activos. El resultado no se elimina. Puede ver sus resultados archivados en cualquier momento. Examine todos los resultados de su cuenta hasta que no quede ningún resultado activo. Después de llegar a cero resultados, sabrá que cualquier nuevo resultado activo que se genere proviene de un cambio reciente en su entorno.

> **Note**
>
> Los resultados de acceso no utilizados solo están disponibles mediante la acción de la API ListFindingsV2.

Revisión de resultados

Después de habilitar el Analizador de acceso de IAM, el siguiente paso consiste en revisar los resultados para determinar si el acceso identificado en el resultado es intencionado o no. También puede revisar los resultados para determinar resultados similares de acceso que se pretende y, a continuación, crear una regla de archivo para archivar dichos resultados automáticamente. También puede revisar los resultados archivados y resueltos.

Para revisar los resultados

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso.
3. Se muestra el panel de resultados. Seleccione los resultados activos para el analizador de acceso externo o no utilizado.

   Para obtener más información sobre la vista de resultados, consulte Visualización del panel de resultados del Analizador de acceso de IAM.

> **Note**
>
> Los resultados solo se muestran si tiene permiso para ver los resultados del analizador.

Todos los resultados se muestran para el analizador. Para ver otros resultados generados por el analizador, elija el tipo de resultado adecuado en el menú desplegable Estado:

- Seleccione Active (Activo) para ver todos los resultados activos generados por el analizador.
- Seleccione Archived (Archivado) para ver solo los resultados generados por el analizador que se han archivado. Para obtener más información, consulte Archivado de resultados.
- Elija Resolved (Resuelto) para ver solo los resultados generados por el analizador que se han resuelto. Cuando solucione el problema que generó el resultado, el estado del resultado cambia a Resuelto.

  > **Important**
  >
  > Los resultados resueltos se eliminan 90 días después de la última actualización del resultado. Los resultados activos y archivados no se eliminan a menos que elimine el analizador que los generó.

- Seleccione All (Todos) para ver todos los resultados con cualquier estado generados por el analizador.

Resultados de acceso externo

Elija Acceso externo y, a continuación, elija el analizador de acceso externo en el menú desplegable Ver analizador. En la página Resultados de los analizadores de acceso externo, se muestra la siguiente información sobre el recurso compartido y la declaración de política que generó el resultado:

ID del resultado

El ID único asignado al resultado. Elija el ID del resultado para mostrar detalles adicionales sobre el recurso y la declaración de política que generó el resultado.

Recurso

Tipo y nombre parcial del recurso que tiene aplicada una política que concede acceso a una entidad externa que no esté dentro de la zona de confianza.

Cuenta del propietario del recurso

Esta columna solo se muestra si utiliza una organización como zona de confianza. La cuenta de la organización que posee el recurso notificado en el resultado.

External principal (Entidad principal externa)

La entidad principal, que no está dentro de su zona de confianza, al que la política analizada concede acceso. Los valores válidos son:

- Cuenta de AWS – todas las entidades principales en la Cuenta de AWS enumerada con permisos desde dicha cuenta de administrador pueden acceder al recurso.
- Cualquier entidad principal: Todas las entidades principales de cualquiera Cuenta de AWS que cumplen las condiciones incluidas en la columna Condiciones tienen permiso para acceder al recurso. Por ejemplo, si aparece una VPC, significa que cualquier entidad principal de cualquier cuenta que tenga permiso para acceder a la VPC enumerada puede tener acceso al recurso.
- Usuario canónico: Todas las entidades principales en la Cuenta de AWS con el ID de usuario canónico enumerado tienen permiso para acceder al recurso.
- Rol de IAM: El rol de IAM enumerado tiene permiso para acceder al recurso.
- Usuario de IAM: El usuario de IAM tiene permiso para acceder al recurso.

Condición

La condición de la declaración de política que concede el acceso. Por ejemplo, si el campo Condition (Condición) incluye Source VPC (VPC de origen), significa que el recurso se comparte con una entidad principal que tiene acceso al VPC enumerado. Las condiciones pueden ser globales o específicas del servicio. Las claves de condición globales tienen el prefijo `aws:`.

Shared through (Compartido a través de)

El campo Shared through (Compartido a través de) indica cómo se concede el acceso que generó el resultado. Los valores válidos son:

- Política de bucket: La política de bucket asociada al bucket de Amazon S3.
- Lista de control de acceso: Lista de control de acceso (ACL) asociado al bucket de Amazon S3.
- Punto de acceso: punto de acceso o punto de acceso de región múltiple asociado al bucket de Amazon S3. El ARN del punto de acceso se muestra en los detalles de los resultados .

Nivel de acceso

El nivel de acceso concedido a la entidad externa por las acciones de la política basada en recursos. Vea los detalles del resultado para obtener más información. Los valores de nivel de acceso incluyen lo siguiente:

- Enumerar: Permiso para enumerar los recursos dentro del servicio para determinar si existe un objeto. Las acciones con este nivel de acceso pueden enumerar objetos pero no pueden ver el contenido de un recurso.
- Leer: Permiso para leer, pero no editar, el contenido y los atributos de los recursos del servicio.
- Escribir: Permiso para crear, eliminar o modificar los recursos del servicio.
- Permisos: Permiso para conceder o modificar permisos en el nivel de recursos del servicio.
- Etiquetado: Permiso para realizar acciones que solo cambian el estado de etiquetas de recursos.

Actualizado

Una marca de tiempo para la actualización más reciente del estado de resultado, o la hora y la fecha en que se generó el resultado si no se han realizado actualizaciones.

> **Note**
>
> Puede tardar hasta 30 minutos después de modificar una política para que el Analizador de acceso de IAM analice de nuevo el recurso y, a continuación, actualice el resultado.

Estado

El estado del resultado, Activo, Archivado o Resuelto.

Resultados de acceso sin utilizar

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por mes. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

Elija Acceso no utilizado y, a continuación, elija el analizador de acceso no utilizado en el menú desplegable Ver analizador. En la página Resultados de los analizadores de acceso externo, se muestra la siguiente información sobre la entidad de IAM que generó el resultado:

ID del resultado

El ID único asignado al resultado. Elija el ID del resultado para mostrar información adicional sobre la entidad de IAM que generó el resultado.

Tipo de resultado

El tipo de resultado de acceso no utilizado: Clave de acceso no utilizado, Contraseña no utilizada, Permiso no utilizado o Rol no utilizado.

Entidad de IAM

La entidad de IAM informó en el resultado. Puede ser un rol o un usuario de IAM.

ID de Cuenta de AWS

Esta columna solo se muestra si se configura el analizador para todos los Cuentas de AWS de la organización. La Cuenta de AWS de la organización que posee la entidad de IAM notificado en el resultado.

Última actualización

La última vez que la entidad de IAM informó el resultado se actualizó o cuando se creó la entidad si no se ha realizado ninguna actualización.

Estado

El estado del resultado, (Activo, Archivado o Resuelto).

Filtrado de resultados

El filtrado predeterminado de la página de resultados es para mostrar todos los resultados. Para ver los resultados activos, elija el estado Activo en el menú desplegable Estado. Para ver los resultados archivados, elija el estado Archivado en el menú desplegable Estado. La primera vez que comienza a utilizar el Analizador de acceso de IAM, no hay resultados archivados.

Utilice filtros para mostrar solo los resultados que cumplan los criterios de propiedad especificados. Para crear un filtro, seleccione la propiedad en la que desea filtrar y, a continuación, elija si la propiedad equivale o contiene un valor de propiedad para filtrar. Por ejemplo, para crear un filtro que muestre solo los resultados de una Cuenta de AWS específica, elija AWS Account para la propiedad, luego seleccione AWS Cuenta =, luego introduzca el número de cuenta para la Cuenta de AWS de la cual que desea ver los resultados.

Para obtener una lista de claves de filtro para crear o actualizar una regla de archivo, consulte Claves de filtro del Analizador de acceso de IAM.

Filtrado de los resultados del acceso externo

Filtrar los resultados del acceso externo

1. Elija Acceso externo y, a continuación, elija el analizador de acceso externo en el menú desplegable Ver analizador.
2. Haga clic en el cuadro de búsqueda para mostrar una lista de las propiedades disponibles.
3. Elija la propiedad que desea utilizar para filtrar los resultados mostrados.
4. Elija el valor que debe coincidir para la propiedad. Solo se muestran los resultados con ese valor en el resultado.

   Por ejemplo, elija Recurso como la propiedad, luego elija Recurso:, y luego escriba parte o todo el nombre de un bucket y, a continuación, pulse Intro. Solo se muestran los resultados del bucket que coincide con los criterios del filtro. Para crear un filtro que muestre solo los resultados de los recursos que permiten el acceso público, puede elegir la propiedad Acceso público y, a continuación, elija Acceso público =, luego elija Acceso público = verdadero.

Puede agregar propiedades adicionales para filtrar aún más los resultados mostrados. Al agregar propiedades adicionales, solo se muestran los resultados que coinciden con todas las condiciones del filtro. No se admite la definición de un filtro para mostrar resultados que coincidan con una propiedad O con otra propiedad. Seleccione Borrar filtros para borrar los filtros que haya definido y mostrar todos los resultados con el estado especificado para su analizador.

Algunos campos solo se muestran cuando se están viendo los resultados de un analizador con una organización como zona de confianza.

Las siguientes propiedades están disponibles para definir filtros:

- Acceso público: para filtrar por resultados los recursos que permiten el acceso público, filtrar por Acceso público y luego elegir Acceso público: true.
- Recurso: para filtrar por recurso, escriba todo o parte del nombre del recurso.
- Tipo de recurso: para filtrar por tipo de recurso, elija el tipo de la lista mostrada.
- Cuenta del propietario del recurso: utilice esta propiedad para filtrar por cuenta de la organización que posee el recurso notificado en el resultado.

- Cuenta de AWS: utilice esta propiedad para filtrar por Cuenta de AWS a la que se concede acceso en la sección Entidad principal de una declaración de política. Para filtrar por Cuenta de AWS, escriba todo o parte del ID de cuenta de Cuenta de AWS de 12 dígitos, o todo o parte del ARN de cuenta completo del usuario o rol de AWS externo que tiene acceso a los recursos de la cuenta actual.
- Usuario canónico: para filtrar por usuario canónico, escriba el ID de usuario canónico tal como se ha definido para los buckets de Amazon S3. Para obtener más información, consulte Identificadores de la cuenta de AWS.
- Usuario federado: para filtrar por usuario federado, escriba todo o parte del ARN de la identidad federada. Para obtener más información, consulte Federación y proveedores de identidades.
- ID del resultado: para filtrar por ID de búsqueda, escriba todo o parte del ID del resultado.
- ARN principal: utilice esta propiedad para filtrar el ARN de la entidad principal (rol o grupo o usuario de IAM) utilizado en una clave de condición aws:PrincipalArn. Para filtrar por ARN de entidad principal, escriba todo o parte del ARN del usuario de IAM, rol o grupo de IAM desde una Cuenta de AWS externa que se haya notificado en un resultado.
- ID de organización de entidad principal: para filtrar por ID de organización de entidad principal, escriba todo o parte del ID de organización asociado a las entidades principales externas que pertenecen a la organización de AWS especificada como condición en el resultado. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- Rutas de la organización de entidad principal: para filtrar por rutas de la organización de entidad principal, escriba todo o parte del ID de la organización de AWS o unidad organizativa (OU) que permite el acceso a todas las entidades principales externas que son miembros de la cuenta de la organización o unidad organizativa especificada como condición en la política. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- Cuenta de origen: para filtrar por cuenta de origen, escriba todos o parte de los ID de Cuenta de AWS asociados a los recursos, tal como se utiliza en algunos permisos entre servicios en AWS. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- ARN de origen: para filtrar por ARN de origen, escriba todo o parte del ARN especificado como condición en el resultado. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- IP de origen: para filtrar por IP de origen, escriba todo o parte de la dirección IP que permite a las entidades externas acceder a recursos en la cuenta actual cuando se utiliza la dirección IP especificada. Para obtener más información, consulte Claves de contexto de condición global de AWS.

- VPC de origen: para filtrar por VPC de origen, escriba todo o parte del ID de la VPC que permite a las entidades externas acceder a recursos en la cuenta actual cuando se utiliza la VPC especificada. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- ID de la organización de origen: para filtrar por ID de la organización de origen, escriba todos o parte de los ID de la organización asociados a los recursos, tal como se utiliza en algunos permisos entre servicios en AWS. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- Rutas de la organización de origen: para filtrar por rutas de la organización de origen, escriba todo o parte de la unidad organizativa (OU) asociada con los recursos, tal como se utiliza en algunos permisos entre servicios en AWS. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- ID de usuario: para filtrar por ID de usuario, escriba todo o parte del ID de usuario del usuario de IAM desde una Cuenta de AWS externa a la que se permite acceder al recurso en la cuenta actual. Para obtener más información, consulte Claves de contexto de condición global de AWS.
- ID de clave de KMS: para filtrar por ID de clave de KMS, escriba todo o parte del ID de clave para la clave de KMS especificada como condición para acceder a objetos de Amazon S3 cifrados por AWS KMS en su cuenta actual.
- Audiencia de Google: para filtrar por Audiencia de Google, escriba todo o parte del ID de aplicación de Google especificado como condición para el acceso de rol de IAM en su cuenta actual. Para obtener más información, consulte Claves de contexto de condición de AWS STS y de IAM.
- Audiencia de Cognito: para filtrar por Audiencia de Cognito, escriba todo o parte del ID del grupo de identidades de Amazon Cognito especificado como condición para el acceso al rol de IAM en su cuenta actual. Para obtener más información, consulte Claves de contexto de condición de AWS STS y de IAM.
- Cuenta de intermediario: el ID de cuenta de Cuenta de AWS de la cuenta que posee o contiene la entidad que llama, como un usuario, un rol de IAM o usuario raíz de cuenta de IAM. Lo utilizan los servicios que llaman a AWS KMS. Para filtrar por cuenta de intermediario, escriba todo o parte del ID de cuenta de Cuenta de AWS.
- ID de aplicación de Facebook: para filtrar por ID de aplicación de Facebook, escriba todo o parte del ID de aplicación de Facebook (o ID de sitio) especificado como condición para permitir el acceso de federación de Inicio de sesión con Facebook a un rol de IAM en su cuenta actual. Para obtener más información, consulte la sección id en Claves de contexto de condición de IAM y AWS STS.

- ID de aplicación de Amazon: para filtrar por ID de aplicación de Amazon, escriba todo o parte del ID de aplicación de Amazon (o ID del sitio) especificado como condición para permitir el acceso de federación de Login with Amazon para un rol de IAM en su cuenta actual. Para obtener más información, consulte la sección id en Claves de contexto de condición de IAM y AWS STS.
- Token de origen de eventos de Lambda: para filtrar por token de origen de eventos de Lambda transferido con integraciones de Alexa, escriba toda o parte de la cadena del token.

Filtrado de resultados de acceso no utilizados

Para filtrar los resultados de acceso no utilizados

1. Elija Acceso no utilizado y, a continuación, elija el analizador de acceso externo en el menú desplegable Ver analizador.
2. Haga clic en el cuadro de búsqueda para mostrar una lista de las propiedades disponibles.
3. Elija la propiedad que desea utilizar para filtrar los resultados mostrados.
4. Elija el valor que debe coincidir para la propiedad. Solo se muestran los resultados con ese valor en el resultado.

   Por ejemplo, elija Tipo de resultados como propiedad. Luego, elija Tipo de resultados = y, a continuación, elija Tipo de resultados = UnusedIAMRole. Solo se muestran los resultados con un tipo de UnusedIAMRole.

Puede agregar propiedades adicionales para filtrar aún más los resultados mostrados. Al agregar propiedades adicionales, solo se muestran los resultados que coinciden con todas las condiciones del filtro. No se admite la definición de un filtro para mostrar resultados que coincidan con una propiedad O con otra propiedad. Seleccione Borrar filtros para borrar los filtros que haya definido y mostrar todos los resultados con el estado especificado para su analizador.

Los siguientes campos solo se muestran cuando se están viendo los resultados de un analizador que monitorea el acceso no utilizado:

- Tipo de resultados: para filtrar por tipo de resultado, filtre por Tipo de resultados y, a continuación, elija el tipo de resultado.
- Recurso: para filtrar por recurso, escriba todo o parte del nombre del recurso.
- Tipo de recurso: para filtrar por tipo de recurso, elija el tipo de la lista mostrada.

- Cuenta del propietario del recurso: Utilice esta propiedad para filtrar por cuenta de la organización que posee el recurso notificado en el resultado.
- ID del resultado: para filtrar por ID del resultado, escriba todo o parte del ID del resultado.

Archivado de resultados

Cuando obtiene un resultado para acceder a un recurso que es intencionado, puede archivar los resultados. Por ejemplo, un resultado de acceso externo para un rol de IAM que utilizan varios usuarios para flujos de trabajo aprobados o un resultado de acceso no utilizado para una clave de acceso que puede que aún sea necesaria. Al archivar un resultado, se borra de la lista de hallazgos activos. Los resultados archivados no se eliminan. Puede filtrar la página Resultados para mostrar los resultados archivados y desarchivarlos en cualquier momento.

Para archivar los resultados desde la página Resultados

1. Seleccione la casilla de verificación situada junto a uno o varios resultados para archivar.
2. Elija Acciones y, a continuación, elija Archivar.

   Se muestra una confirmación en la parte superior de la pantalla.

Archivar los resultados desde la página Información de resultados

1. Elija el ID del resultado del resultado que desea archivar.
2. Seleccione Archivar.

   Se muestra una confirmación en la parte superior de la pantalla.

Para desarchivar resultados, repita los pasos anteriores, pero elija Unarchive (Desarchivar) en lugar de Archive (Archivar). Al desarchivar un resultado, el estado se establece en Activo.

Resolución de resultados

Resultados de acceso externo

Para resolver los resultados generados a partir de accesos que no pretendía permitir, modifique la declaración de política para quitar los permisos que permiten el acceso al recurso identificado. Por ejemplo, para resultados en buckets de Amazon S3, utilice la consola de Amazon S3 para configurar

los permisos en el bucket. Para los roles de IAM, utilice la consola de IAM para modificar la política de confianza para el rol de IAM enumerado. Utilice la consola para los demás recursos admitidos para modificar las instrucciones de directiva que dieron lugar a un resultado generado.

Después de realizar un cambio para resolver un resultado, como modificar una política aplicada a un rol de IAM, el Analizador de acceso de IAM analiza de nuevo el recurso. Si el recurso ya no se comparte fuera de la zona de confianza, el estado del resultado cambia a Resuelto. El resultado ya no se muestra en la lista Resultados activos y, en su lugar, se muestra en la lista Resultados resueltos.

> **Note**
>
> Esto no se aplica a los resultados Erróneos. Cuando el Analizador de acceso de IAM no puede analizar un recurso, genera un resultado de error. Si resuelve el problema que le impedía al Analizador de acceso de IAM analizar el recurso, el resultado de error se elimina por completo en lugar de cambiar a un resultado resuelto.

Si los cambios realizados dieron como resultado que el recurso se compartiera fuera de su zona de confianza, pero de una manera diferente, por ejemplo con una entidad principal diferente o con un permiso diferente, el Analizador de acceso de IAM generará un nuevo resultado activo.

> **Note**
>
> Puede tardar hasta 30 minutos después de modificar una política para que el Analizador de acceso de IAM analice de nuevo el recurso y, a continuación, actualice el resultado. Los resultados resueltos se eliminan 90 días después de la última actualización del estado del resultado.

Resultados de acceso sin utilizar

Para resolver los problemas de acceso no utilizados, utilice la consola de IAM para eliminar la clave de acceso, la contraseña, el permiso o la función no utilizados. Para obtener más información, consulte los siguientes recursos:

- Para obtener más información sobre cómo eliminar una clave de acceso, consulte Administrar las claves de acceso (consola).

- Para obtener más información sobre la eliminación de la contraseña de un usuario de IAM, consulte Creación, cambio o eliminación de la contraseña de un usuario de IAM (consola).
- Para obtener más información acerca de cómo editar permisos de un usuario, consulte Cambio de los permisos de un usuario (consola) en la Guía del usuario de IAM.
- Para obtener más información sobre la eliminación de un rol de IAM, consulte Eliminar un rol de IAM (consola).

Tras realizar un cambio para resolver un resultado de acceso no utilizado, el estado del resultado se cambia a Resuelto la próxima vez que se ejecute el analizador de acceso no utilizado. El resultado ya no se muestra en la lista Resultados activos y, en su lugar, se muestra en la lista Resultados resueltos. Si realiza un cambio que solo corrija parcialmente un resultado de acceso no utilizada, el resultado existente cambia a Resuelta, pero se genera una nueva. Por ejemplo, se eliminan solo algunos de los permisos sin utilizar de un resultado, pero no todos.

El Analizador de acceso de IAM cobra por el análisis de acceso no utilizado en función del número de usuarios y roles de IAM analizados por mes. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

Tipos de recursos del Analizador de acceso de IAM para acceso externo

Para los analizadores de acceso externo, el Analizador de acceso de IAM analiza las políticas basadas en recursos que se aplican a los recursos de AWS en la región en la que habilitó el Analizador de acceso de IAM. Solo analiza las políticas basadas en recursos. Revise la información sobre cada recurso para obtener detalles sobre cómo genera el Analizador de acceso de IAM resultados para cada tipo de recurso.

> **Note**
>
> Los tipos de recursos compatibles que se muestran son para analizadores de acceso externos. Los analizadores de acceso no utilizado solo admiten roles y usuarios de IAM. Para obtener más información, consulte Trabajar con resultados.

Tipos de recursos compatibles para el acceso externo:

- Buckets de Amazon Simple Storage Service Batch
- Buckets de directorio de Amazon Simple Storage Service Batch

- Roles de AWS Identity and Access Management
- Claves de AWS Key Management Service
- Funciones y capas de AWS Lambda
- Colas de Amazon Simple Queue Service
- AWS Secrets Manager secretos
- Temas de Amazon Simple Notification Service
- Instantáneas de volúmenes de Amazon Elastic Block Store
- Instantáneas de base de datos de Amazon Relational Database Service
- Instantáneas de clúster de base de datos de Amazon Relational Database Service
- Repositorios de Amazon Elastic Container Registry
- Sistemas de archivos de Amazon Elastic File System
- Amazon DynamoDB Streams
- Tablas de Amazon DynamoDB

Buckets de Amazon Simple Storage Service Batch

Cuando el Analizador de acceso de IAM analiza los buckets de S3, genera un resultado cuando una política de bucket de Amazon S3, una ACL o un punto de acceso, lo que incluye un punto de acceso de región múltiple, aplicado a un bucket concede acceso a una entidad externa. Una entidad externa es una entidad principal u otra que puede utilizar para crear un filtro que no esté dentro de su zona de confianza. Por ejemplo, si una política de bucket concede acceso a otra cuenta o permite acceso público, el Analizador de acceso de IAM genera un resultado. Sin embargo, si habilita Bloqueo de acceso público en su bucket, puede bloquear el acceso a la cuenta o al bucket.

> **Note**
>
> El Analizador de acceso de IAM no analiza la política de puntos de acceso adjunta a los puntos de acceso entre cuentas porque el punto de acceso y su política están fuera de la cuenta del analizador. El Analizador de acceso de IAM genera un resultado público cuando un bucket delega el acceso a un punto de acceso entre cuentas y Bloqueo de acceso público no está habilitado en el bucket o la cuenta. Cuando se habilita Bloqueo de acceso público, se resuelve la detección pública y el Analizador de acceso de IAM genera una detección de cuenta cruzada para el punto de acceso entre cuentas.

La configuración de Bloqueo de acceso público de Amazon S3 anula las políticas de bucket que se aplican al bucket. La configuración también anula las políticas de puntos de acceso aplicadas a los puntos de acceso del bucket. El Analizador de acceso de IAM analiza la configuración de Bloqueo de acceso público en el bucket cada vez que cambia una política. Sin embargo, evalúa la configuración de Bloqueo de acceso público en la cuenta solo una vez cada 6 horas. Esto significa que el Analizador de acceso de IAM podría no generar o resolver un resultado para acceso público a un bucket durante un máximo de 6 horas. Por ejemplo, si tiene una política de bucket que permite el acceso público, el Analizador de acceso de IAM genera un resultado para ese acceso. Si, a continuación, habilita Bloqueo de acceso público para bloquear todo el acceso público al bucket en la cuenta, el Analizador de acceso de IAM no resuelve el resultao para la política del bucket durante un máximo de 6 horas, incluso aunque se bloquee todo el acceso público al bucket. La resolución de resultados públicos para puntos de acceso entre cuentas también puede tardar hasta 6 horas una vez que se activa Bloqueo de acceso público en la cuenta.

En el caso de un punto de acceso de región múltiple, el Analizador de acceso de IAM utiliza una política establecida para generar resultados. El Analizador de acceso de IAM evalúa los cambios en los puntos de acceso de región múltiple una vez cada 6 horas. Esto significa que el Analizador de acceso de IAM no genera ni resuelve una búsqueda durante un máximo de 6 horas, incluso si se crea o elimina un punto de acceso de región múltiple o actualiza la política correspondiente.

Buckets de directorio de Amazon Simple Storage Service Batch

Los buckets de directorio de Amazon S3 utilizan la clase de almacenamiento Amazon S3 Express One, que se recomienda para cargas de trabajo o aplicaciones de rendimiento crítico. Para buckets de directorio de Amazon S3, el Analizador de acceso de IAM analiza las políticas de buckets de directorio, incluidas las declaraciones de condición de una política, que permiten a una entidad externa acceder a un bucket de directorio. Para obtener más información acerca de los buckets de directorio de Amazon S3, consulte Buckets de directorio en la Guía del usuario de Amazon Simple Storage Service.

Roles de AWS Identity and Access Management

Para las funciones de IAM, el Analizador de acceso de IAM analiza las Políticas de confianza. En una política de confianza de rol, defina las entidades principales en las que confía para asumir el rol. Una política de confianza de rol es una política basada en recursos requerida que se adjunta a un rol en IAM. El Analizador de acceso de IAM genera resultado para roles dentro de la zona de confianza a la que puede acceder una entidad externa que está fuera de su zona de confianza.

> **Note**
>
> Un rol de IAM es un recurso global. Si una política de confianza de rol concede acceso a una entidad externa, el Analizador de acceso de IAM genera un resultado en cada región habilitada.

Claves de AWS Key Management Service

Para AWS KMS keys, el Analizador de acceso de IAM analiza las políticas clave y las subvenciones aplicadas a una clave. El Analizador de acceso de IAM genera una búsqueda si una política de claves o concesión permite a una entidad externa acceder a la clave. Por ejemplo, si utiliza la clave de condición kms:CallerAccount en una declaración de política para permitir el acceso a todos los usuarios en una cuenta de AWS concreta y especifica una cuenta distinta de la cuenta actual (la zona de confianza del analizador actual), el Analizador de acceso de IAM genera un resultado. Para obtener más información sobre las claves de condición de AWS KMS en las declaraciones de política de IAM, consulte Claves de condición de AWS KMS.

Cuando el Analizador de acceso de IAM analiza una clave de KMS, lee los metadatos de clave, como la política de claves y la lista de concesiones. Si la política de claves no permite que el rol del Analizador de acceso de IAM lea los metadatos de clave, se genera un resultado de error Acceso denegado. Por ejemplo, si la declaración de política de ejemplo siguiente es la única política aplicada a una clave, se produce un resultado de error acceso denegado en el Analizador de acceso de IAM.

```
{
    "Sid": "Allow access for Key Administrators",
    "Effect": "Allow",
    "Principal": {
       "AWS": "arn:aws:iam::111122223333:role/Admin"
    },
    "Action": "kms:*",
    "Resource": "*"
}
```

Dado que esta declaración permite que solo el rol denominado Admin de la cuenta 111122223333 de AWS tenga acceso a la clave, se genera un resultado de error Acceso denegado porque el Analizador de acceso de IAM no puede analizar completamente la clave. Se muestra un hallazgo de error en texto rojo en la tabla Resultados. El resultado es similar al siguiente.

```
{
    "error": "ACCESS_DENIED",
    "id": "12345678-1234-abcd-dcba-111122223333",
    "analyzedAt": "2019-09-16T14:24:33.352Z",
    "resource": "arn:aws:kms:us-
west-2:1234567890:key/1a2b3c4d-5e6f-7a8b-9c0d-1a2b3c4d5e6f7g8a",
    "resourceType": "AWS::KMS::Key",
    "status": "ACTIVE",
    "updatedAt": "2019-09-16T14:24:33.352Z"
}
```

Cuando se crea una clave de KMS, los permisos concedidos para acceder a la clave dependen de cómo se crea la clave. Si recibe un resultado de error Acceso denegado para un recurso clave, aplique la siguiente declaración de política al recurso de la clave para conceder permiso al Analizador de acceso de IAM para acceder a la clave.

```
{
    "Sid": "Allow IAM Access Analyzer access to key metadata",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/aws-service-role/access-
analyzer.amazonaws.com/AWSServiceRoleForAccessAnalyzer"
        },
    "Action": [
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:List*"
    ],
    "Resource": "*"
},
```

Después de recibir un resultado de Acceso denegado para un recurso de clave de KMS y, a continuación, resolver el resultado actualizando la política de claves, el resultado se actualiza al estado Resuelto. Si hay instrucciones de política o concesiones de clave que conceden permiso a la clave a una entidad externa, es posible que vea resultados adicionales para el recurso clave.

Funciones y capas de AWS Lambda

Para las funciones de AWS Lambda, el Analizador de acceso de IAM analiza las políticas, incluidas las declaraciones de condición de una política, que conceden acceso a la función a una entidad

externa. El Analizador de acceso de IAM también analiza los permisos concedidos cuando se utiliza la operación AddPermission del AWS Lambda API con una `EventSourceToken`.

Colas de Amazon Simple Queue Service

Para las colas de Amazon SQS, el Analizador de acceso de IAM analiza las políticas, incluidas las declaraciones de condición de una política, que permiten a una entidad externa acceder a una cola.

AWS Secrets Manager secretos

Para secretos AWS Secrets Manager, el Analizador de acceso de IAM analiza las políticas, incluidas las declaraciones de condición de una política, que permiten a una entidad externa acceder a una cola.

Temas de Amazon Simple Notification Service

El Analizador de acceso de IAM analiza las políticas basadas en recursos adjuntadas a los temas de Amazon SNS, incluidas las declaraciones de condición de las políticas que permiten el acceso externo a un tema. Se puede permitir que cuentas externas realicen acciones de Amazon SNS, tales como suscribirse a temas y publicarlos, mediante una política basada en recursos. Un tema de Amazon SNS es accesible externamente si las entidades principales de una cuenta situada fuera de la zona de confianza pueden realizar operaciones en el tema. Cuando se elige `Everyone` en la política al crear un tema de Amazon SNS, el tema queda accesible al público. `AddPermission` es otra forma de agregar una política basada en recursos a un tema de Amazon SNS que permita el acceso externo.

Instantáneas de volúmenes de Amazon Elastic Block Store

Las instantáneas de volúmenes de Amazon Elastic Block Store no tienen políticas basadas en recursos. Una instantánea se comparte a través de permisos de uso compartido de Amazon EBS. En el caso de las instantáneas de volúmenes de Amazon EBS, el Analizador de acceso de IAM analiza las listas de control de acceso que permiten a una entidad externa acceder a una instantánea. Una instantánea de volumen de Amazon EBS se puede compartir con cuentas externas cuando está cifrada. Una instantánea de volumen sin cifrar se puede compartir con cuentas externas y conceder acceso público. La configuración de uso compartido está en el atributo `CreateVolumePermissions` de la instantánea. Cuando los clientes previsualizan el acceso externo a una instantánea de Amazon EBS, pueden especificar la clave de cifrado como indicador de que la instantánea está cifrada, de manera parecida a la forma en que la vista previa del Analizador de acceso de IAM gestiona los secretos de Secrets Manager.

Instantáneas de base de datos de Amazon Relational Database Service

Las instantáneas de base de datos de Amazon RDS no tienen políticas basadas en recursos. Una instantánea de base de datos se comparte mediante permisos de base de datos de Amazon RDS, y solo se pueden compartir instantáneas de base de datos manuales. En el caso de las instantáneas de base de Amazon RDS, el Analizador de acceso de IAM analiza las listas de control de acceso que permiten a una entidad externa acceder a una instantánea. Las instantáneas de base de datos sin cifrar pueden ser públicas. Las instantáneas de base de datos cifradas no se pueden compartir públicamente, pero se pueden compartir con hasta 20 cuentas más. Para obtener más información, consulte Creación de una instantánea de base de datos. El Analizador de acceso de IAM considera la posibilidad de exportar una instantánea manual de base de datos (por ejemplo, a un bucket de Amazon S3) como acceso de confianza.

> **Note**
>
> El Analizador de acceso de IAM no identifica el acceso público o entre cuentas configurado directamente en la propia base de datos. El Analizador de acceso de IAM solo identifica los resultados de acceso público o entre cuentas configurado en la instantánea de base de datos de Amazon RDS.

Instantáneas de clúster de base de datos de Amazon Relational Database Service

Las instantáneas de clúster de base de datos de Amazon RDS no tienen políticas basadas en recursos. Una instantánea se comparte a través de permisos de clúster de base de datos de Amazon EBS. En el caso de las instantáneas de clúster de base de Amazon RDSel Analizador de acceso de IAM analiza las listas de control de acceso que permiten a una entidad externa acceder a una instantánea. Las instantáneas de clúster sin cifrar pueden ser públicas. Las instantáneas de clúster cifradas no se pueden compartir públicamente. Las instantáneas de clúster, cifradas y sin cifrar, se pueden compartir con hasta 20 cuentas más. Para obtener más información, consulte Creating a DB cluster snapshot (Creación de una instantánea de clúster de base de datos). El Analizador de acceso de IAM considera la posibilidad de exportar una instantánea de clúster de base de datos (por ejemplo, a un bucket de Amazon S3) como acceso de confianza.

> **Note**
>
> Los resultados del Analizador de acceso de IAM no incluyen la supervisión de ningún recurso compartido de clústeres y clones de base de datos de Amazon RDS con otra Cuenta de

AWS u organización que utilice AWS Resource Access Manager. El Analizador de acceso de IAM solo identifica los resultados de acceso público o entre cuentas configurado en la instantánea de clúster de base de datos de Amazon RDS.

Repositorios de Amazon Elastic Container Registry

En el caso de los repositorios de Amazon ECR, el Analizador de acceso de IAM analiza las políticas basadas en recursos, incluidas las declaraciones de condición de una política, que permiten a una entidad externa acceder a un repositorio (de manera similar a otros tipos de recursos, como temas de Amazon SNS y sistemas de archivos de Amazon EFS). En el caso de los repositorios de Amazon ECR, una entidad principal debe tener permiso para `ecr:GetAuthorizationToken` a través de una política basada en identidad para que se considere disponible externamente.

Sistemas de archivos de Amazon Elastic File System

En el caso de los sistemas de archivos de Amazon EFS, el Analizador de acceso de IAM analiza las políticas, incluidas las declaraciones de condición de una política, que permiten a una entidad externa acceder a un sistema de archivos. Un sistema de archivos de Amazon EFS es accesible externamente si las entidades principales de una cuenta ajena a su zona de confianza pueden realizar operaciones en ese sistema de archivos. El acceso lo definen una política de sistema de archivos que utiliza IAM y cómo está montado el sistema de archivos. Por ejemplo, montar el sistema de archivos de Amazon EFS en otra cuenta se considera accesible externamente, a menos que esa cuenta pertenezca a la organización y se haya definido la organización como zona de confianza. Si se monta el sistema de archivos desde una nube privada virtual con una subred pública, el sistema de archivos es accesible externamente. Cuando se utiliza Amazon EFS con AWS Transfer Family, las solicitudes de acceso al sistema de archivos recibidas de un servidor de Transfer Family que sea propiedad de una cuenta diferente a la del sistema de archivos se bloquean si el sistema de archivos permite el acceso público.

Amazon DynamoDB Streams

El Analizador de acceso de IAM genera un resultado si una política de DynamoDB permite al menos una acción entre cuentas que permite que una entidad externa acceda a una secuencia de DynamoDB. Para obtener más información sobre las acciones entre cuentas compatibles con DynamoDB, consulte las acciones de IAM compatibles con las políticas basadas en recursos en la Guía para desarrolladores de Amazon DynamoDB.

Tablas de Amazon DynamoDB

El Analizador de acceso de IAM genera un resultado si una política de DynamoDB permite al menos una acción entre cuentas que permite que una entidad externa acceda a una tabla o indice de DynamoDB. Para obtener más información sobre las acciones entre cuentas compatibles con DynamoDB, consulte las acciones de IAM compatibles con las políticas basadas en recursos en la Guía para desarrolladores de Amazon DynamoDB.

Configuración para IAM Access Analyzer

Si está configurando AWS Identity and Access Management Access Analyzer en su cuenta de administración de AWS Organizations, puede agregar una cuenta de miembro a la organización como administrador delegado para administrar IAM Access Analyzer en su organización. El administrador delegado tiene permisos para crear y administrar analizadores dentro de la organización. Solo la cuenta de administración puede agregar un administrador delegado.

Administrador delegado de IAM Access Analyzer

El administrador delegado para IAM Access Analyzer es la cuenta de un miembro dentro de la organización que tiene permisos para crear y administrar analizadores que analizan el acceso en la organización. Solo la cuenta de administración puede agregar, quitar o cambiar a un administrador delegado.

Si agrega un administrador delegado, puede cambiar posteriormente a una cuenta distinta para el administrador delegado. Cuando lo haga, la cuenta de administrador delegado anterior pierde el permiso para todos los analizadores creados con esa cuenta para analizar el acceso en la organización. Estos analizadores pasan a un estado deshabilitado y ya no generan nuevos hallazgos o actualizan los existentes. Tampoco se puede acceder a las conclusiones actuales de estos analizadores. Puede volver a acceder a ellos en el futuro configurando la cuenta como administrador delegado. Si sabe que no utilizará la misma cuenta que un administrador delegado, considere eliminar los analizadores antes de cambiar el administrador delegado. Esto elimina todos los hallazgos generados. Cuando el nuevo administrador delegado crea nuevos analizadores, se generan nuevas instancias de los mismos hallazgos. No pierde ningún hallazgo, solo se generan para el nuevo analizador en una cuenta distinta. Además, puede seguir accediendo a los hallazgos de la organización mediante la cuenta de administración de la organización, que también tiene permisos de administrador. El nuevo administrador delegado debe crear nuevos analizadores para que IAM Access Analyzer comience a supervisar los recursos de la organización.

Si el administrador delegado abandona la organización de AWS, los privilegios de administración delegados se retiran de la cuenta. Todos los analizadores de la cuenta con la organización como zona de confianza se mueven a un estado deshabilitado. Tampoco se puede acceder a las conclusiones actuales de estos analizadores.

La primera vez que configure analizadores en la cuenta de administración, puede elegir Agregar administrador delegado en la página Configuración del analizador en la consola IAM Access Analyzer.

> **Note**
>
> IAM Access Analyzer cobra por los analizadores de acceso no utilizado en función del número de usuarios y roles de IAM analizados por el analizador por mes. Si crea un analizador de acceso no utilizado en la cuenta de administración y en la cuenta de administrador delegado, se le cobrará por ambos analizadores de acceso no utilizados. Para obtener más información sobre los precios, consulte los precios de IAM Access Analyzer.

Para agregar un administrador delegado mediante la consola

1. Inicie sesión en la consola de AWS con la cuenta de administración de su organización de .
2. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
3. En Analizador de acceso, elija Configuración del analizador.
4. Elija Add delegated administrator (Agregar administrador delegado).
5. En el campo Administrador delegado, introduzca el Cuenta de AWS número de una cuenta de miembro de la organización para definir el administrador delegado.

   La cuenta debe formar parte de la organización.
6. Elija Guardar cambios.

Para agregar un administrador delegado mediante la AWS CLI o los SDK de AWS

Cuando crea un analizador para analizar el acceso en toda la organización en una cuenta de administrador delegado mediante la AWS CLI, AWS la API (utilizando los AWS SDK) o AWS CloudFormation, debe utilizar las API de AWS Organizations para activar el acceso al servicio para IAM Access Analyzer y registrar la cuenta de miembro como administrador delegado.

1. Habilitar el acceso al servicio de confianza para IAM Access Analyzer en AWS Organizations. Consulte Habilitación del acceso de confianza con otros servicios en la Guía del usuario de AWS Organizations.
2. Registre una cuenta de miembro válida de su organización de AWS como administrador delegado mediante la operación de API AWS Organizations `RegisterDelegatedAdministrator` o el comando `register-delegated-administrator` de AWS CLI.

Después de cambiar el administrador delegado, el nuevo administrador debe crear analizadores para empezar a monitorizar el acceso a los recursos de la organización.

Eliminar analizadores

Puede eliminar los analizadores de acceso externos y los no utilizados existentes desde la página Configuración del analizador. Al eliminar un analizador, los recursos especificados en el analizador dejan de supervisarse y no se generan nuevos resultados. Se eliminan todos los resultados generados por el analizador.

Para los resultados que se eliminan porque se elimina el analizador que los generó, el evento se envía a Eventbridge dentro de los dos días siguientes a que se eliminara el analizador. Los resultados del Security Hub pueden tardar hasta 90 días después de eliminar el analizador.

Eliminar un analizador

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En Analizador de acceso, elija Configuración del analizador.
3. Seleccione el analizador que desea eliminar y, luego, elija Eliminar.
4. Escriba **`delete`** en la casilla del texto de confirmación y, a continuación, elija Eliminar.

Reglas de archivado

Las reglas de archivado archivan automáticamente los nuevos hallazgos que cumplen los criterios definidos al crear la regla. También puede aplicar reglas de archivado de forma retroactiva para archivar conclusiones existentes que cumplan los criterios de regla de archivado. Por ejemplo, puede crear una regla de archivado para archivar automáticamente los resultados de un bucket de Amazon S3 específico al que conceda acceso de forma periódica. O si concede acceso a varios recursos a una entidad principal específica, puede crear una regla que archive automáticamente cualquier

nuevo hallazgo generado para el acceso concedido a esa entidad principal. Esto le permite centrarse únicamente en los hallazgos activos que puedan indicar un riesgo de seguridad.

Si crea una regla de archivado, solo se archivarán automáticamente los hallazgos nuevos que coincidan con los criterios de la regla. Los hallazgos existentes no se archivan automáticamente. Al crear una regla, puede incluir hasta 20 valores por criterio en la regla. Para obtener una lista de claves de filtro para crear o actualizar una regla de archivo, consulte Claves de filtro del Analizador de acceso de IAM.

> **Note**
>
> Al crear o editar una regla de archivado, IAM Access Analyzer no valida los valores que incluya en el filtro para la regla. Por ejemplo, si agrega una regla para que coincida con una Cuenta de AWS, IAM Access Analyzer acepta cualquier valor en el campo, incluso si no es un número de cuenta de AWS válido.

Para crear una regla de archivado

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Elija Analizador de acceso y, luego, elija Configuración del analizador.
3. En la sección Analizadores, elija el analizador para el que desea crear una regla de archivo.
4. En la pestaña Reglas de archivado, elija Crear regla de archivado.
5. Escriba un nombre para la regla si desea cambiar el nombre predeterminado.
6. En la sección Rule (Regla), en Criteria (Criterios), seleccione una propiedad que coincida con la regla.
7. Elija una condición para el valor de la propiedad, como Contiene, Es o No es igual.

   Los operadores disponibles dependen de la propiedad que elija.
8. Si lo desea, agregue valores adicionales para la propiedad o agregue criterios adicionales para la regla. Para asegurarse de que su regla no archivará nuevos resultados de acceso público, puede incluir el criterio Acceso público y seleccionar falso.

   Para agregar otro valor para un criterio, elija Add another value (Agregar otro valor). Para agregar otro criterio para la regla, elija Agregar criterio.
9. Cuando termine de agregar criterios y valores, elija Crear regla para aplicar la regla solo a las nuevas conclusiones. Elija Crear y archivar conclusiones activas para archivar las conclusiones

nuevas y existentes según los criterios de la regla. En la sección Hallazgos puede revisar la lista de hallazgos activos a los que se aplica la regla de archivado.

Por ejemplo, para crear una regla que archive automáticamente los resultados de los buckets de Amazon S3: elija Tipo de recurso y, a continuación, elija Es para la condición. A continuación, elija bucket de S3 de la lista de Valores.

Para crear una regla para los resultados de acceso no utilizados que archive automáticamente los resultados de una cuenta concreta: elija Cuenta del propietario del recurso y, a continuación, elija Equivale para la condición. Escriba el ID de Cuenta de AWS en el cuadro de texto Valor.

Continúe definiendo criterios para personalizar la regla según corresponda al entorno y, a continuación, elija Crear regla.

Si crea una nueva regla y agrega varios criterios, puede eliminar un único criterio de la regla seleccionando Remove this criterion (Quitar este criterio). Puede eliminar un valor agregado para un criterio seleccionando Remove value (Quitar valor).

Para editar una regla de archivado

1. Seleccione el nombre de la regla que desea editar en la columna Nombre.

   Solo puede editar una regla de archivado a la vez.
2. Agregue nuevos criterios o elimine los criterios y valores existentes de cada criterio.
3. Elija Guardar cambios para aplicar la regla solo a los nuevos resultados. Elija Guardar y archivar los resultados activos para archivar los resultados nuevos y existentes en función de los criterios de la regla.

Para eliminar una regla de archivado

1. Seleccione la casilla de verificación correspondiente a la cola que desea eliminar.
2. Elija Eliminar.
3. Escriba **`delete`** en el cuadro de diálogo de confirmación Delete archive rule (Eliminar regla de archivado) y, a continuación, elija Delete (Eliminar).

Las reglas solo se eliminan del analizador en la región actual. Debe eliminar las reglas de archivado por separado para cada analizador creado en otras regiones.

Supervisión de AWS Identity and Access Management Access Analyzer con Amazon EventBridge

Utilice la información de este tema para aprender a supervisar resultados del Analizador de acceso de IAM y acceder a vistas previas con Amazon EventBridge. EventBridge es la nueva versión de Amazon CloudWatch Events.

Eventos de resultados

El Analizador de acceso de IAM envía un evento a Eventbridge para cada resultado generado, para un cambio en el estado de un resultado existente y cuando se elimina un resultado. Para recibir resultados y notificaciones sobre resultados, debe crear una regla de evento en Amazon EventBridge. Al crear una regla de evento, también puede especificar una acción de destino que se desencadenará en función de la regla. Por ejemplo, puede crear una regla de evento que desencadena un tema de Amazon SNS cuando se reciba un evento de un nuevo resultado del Analizador de acceso de IAM.

Acceso a eventos de vista previa

El Analizador de acceso de IAM envía un evento a EventBridge para cada vista previa y cambio de acceso de estado. Esto incluye un evento cuando se crea por primera vez la vista previa de acceso (estado Creación), cuando se completa la vista previa de acceso (estado Completado) o cuando la creación de la vista previa de acceso falló (estado Error). Para recibir notificaciones sobre acceso a vista previa, debe crear una regla de evento en Amazon EventBridge. Al crear una regla de evento, puede especificar una acción de destino que se desencadenará en función de la regla. Por ejemplo, puede crear una regla de evento que desencadena un tema de Amazon SNS cuando se reciba un evento de una vista previa de acceso completa desde el Analizador de acceso de IAM.

Frecuencia de notificación de eventos

El Analizador de acceso de IAM envía eventos sobre nuevos resultados y resultados con actualizaciones de estado a EventBridge aproximadamente una hora desde el momento en que se produce el evento en su cuenta. El Analizador de acceso de IAM también envía eventos a EventBridge cuando se elimina un resultado resuelto porque el período de retención ha expirado. Para los resultados que se eliminan porque se elimina el analizador que los generó, el evento se envía a Eventbridge aproximadamente 24 horas después de que se eliminara el analizador. Cuando se elimina un hallazgo, el estado del resultado no cambia. En su lugar, el atributo `isDeleted` se establece en `true`. El Analizador de acceso de IAM también envía eventos sobre las vistas previas de acceso recién creadas y los cambios de estado de la vista previa de acceso a EventBridge.

Ejemplo de eventos de resultados de acceso externo

El siguiente es un evento de ejemplo de resultado de acceso externo del Analizador de acceso de IAM enviado a EventBridge. El `id` listado es el ID del evento en Eventbridge. Para obtener más información, consulte Eventos y patrones de eventos en EventBridge.

En el objeto `detail`, los valores de los atributos `accountId` y `region` hacen referencia a la cuenta y la región notificados en el resultado. El atributo `isDeleted` indica si el evento era del resultado que se eliminó. El `id` es el ID del resultado. La matriz `resources` es un singleton con el ARN del analizador que generó el resultado.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "action": [
            "s3:GetObject"
        ],
        "analyzedAt": "2019-11-21T01:22:22Z",
        "condition": {},
        "createdAt": "2019-11-20T04:58:50Z",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "isPublic": false,
        "principal": {
            "AWS": "999988887777"
        },
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
```

```
    "version": "0"
}
```

El Analizador de acceso de IAM también envía eventos a EventBridge sobre resultados erróneos. Un resultado erróneo es un resultado generado cuando el Analizador de acceso de IAM no puede analizar un recurso. Los eventos para resultados erróneos incluyen un atributo `error` como se muestra en el siguiente ejemplo.

```
{
    "account": "111122223333",
    "detail": {
        "accountId": "111122223333",
        "analyzedAt": "2019-11-21T01:22:22Z",
        "createdAt": "2019-11-20T04:58:50Z",
        "error": "ACCESS_DENIED",
        "id": "22222222-dcba-4444-dcba-333333333333",
        "isDeleted": false,
        "region": "us-west-2",
        "resource": "arn:aws:s3:::my-bucket",
        "resourceType": "AWS::S3::Bucket",
        "status": "ACTIVE",
        "updatedAt": "2019-11-21T01:14:07Z",
        "version": "1.0"
    },
    "detail-type": "Access Analyzer Finding",
    "id": "11111111-2222-4444-aaaa-333333333333",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2019-11-21T01:22:33Z",
    "version": "0"
}
```

Ejemplo de eventos relacionados con los resultados de acceso no utilizados

El siguiente es un evento de ejemplo de resultado de acceso no utilizado del Analizador de acceso de IAM enviado a EventBridge. El `id` listado es el ID del evento en Eventbridge. Para obtener más información, consulte Eventos y patrones de eventos en EventBridge.

En el objeto `detail`, los valores de los atributos `accountId` y `region` hacen referencia a la cuenta y la región notificados en el resultado. El atributo `isDeleted` indica si el evento era del resultado que se eliminó. El `id` es el ID del resultado.

```
{
    "version": "0",
    "id": "dc7ce3ee-114b-3243-e249-7f10f9054b21",
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "123456789012",
    "time": "2023-09-29T17:31:40Z",
    "region": "us-west-2",
    "resources": [
       "arn:aws:access-analyzer:us-west-2:123456789012:analyzer/
integTestLongLivingAnalyzer-DO-NOT-DELETE"
       ],
    "detail": {
        "findingId": "b8ae0460-5d29-4922-b92a-ba956c986277",
        "resource": "arn:aws:iam::111122223333:role/FindingIntegTestFakeRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "111122223333",
        "createdAt": "2023-09-29T17:29:18.758Z",
        "updatedAt": "2023-09-29T17:29:18.758Z",
        "analyzedAt": "2023-09-29T17:29:18.758Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "62160bda-8e94-46d6-ac97-9670930d8ffb",
        "isDeleted": false,
        "findingType": "UnusedPermission",
        "numberOfUnusedServices": 0,
        "numberOfUnusedActions": 1
        }
    }
```

El Analizador de acceso de IAM también envía eventos a EventBridge sobre resultados erróneos. Un resultado erróneo es un resultado generado cuando el Analizador de acceso de IAM no puede analizar un recurso. Los eventos para resultados erróneos incluyen un atributo `error` como se muestra en el siguiente ejemplo.

```
{
    "version": "0",
    "id": "c2e7aa1a-4df7-7652-f33e-64113b8997d4",
```

```
    "detail-type": "Unused Access Finding for IAM entities",
    "source": "aws.access-analyzer",
    "account": "111122223333",
    "time": "2023-10-31T20:26:12Z",
    "region": "us-west-2",
    "resources": [
      "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/ba811f91-
de99-41a4-97c0-7481898b53f2"
      ],
    "detail": {
        "findingId": "b01a34f2-e118-46c9-aef8-0d8526b495c7",
        "resource": "arn:aws:iam::123456789012:role/TestRole",
        "resourceType": "AWS::IAM::Role",
        "accountId": "444455556666",
        "createdAt": "2023-10-31T20:26:08.647Z",
        "updatedAt": "2023-10-31T20:26:09.245Z",
        "analyzedAt": "2023-10-31T20:26:08.525Z",
        "previousStatus": "",
        "status": "ACTIVE",
        "version": "7c7a72a2-7963-4c59-ac71-f0be597010f7",
        "isDeleted": false,
        "findingType": "UnusedIAMRole",
        "error": "INTERNAL_ERROR"
        }
  }
```

Ejemplo de eventos de vista previa de acceso

En el ejemplo siguiente se muestran los datos del primer evento que se envía a EventBridge cuando se crea una vista previa de acceso. La matriz `resources` es solitaria con el ARN del analizador con el que está asociada la vista previa de acceso. En el objeto `detail`, `id` hace referencia al ID de vista previa de acceso y `configuredResources` hace referencia al recurso para el que se creó la vista previa de acceso. El `status` es `Creating` y hace referencia al estado de vista previa de acceso. El `previousStatus` no se especifica porque se acaba de crear la vista previa de acceso.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
```

```
        "createdAt": "2020-02-20T00:00:00.00Z",
        "region": "us-west-2",
        "status": "CREATING",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "aaaabbbb-2222-3333-4444-555566667777",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

En el ejemplo siguiente se muestran los datos de un evento que se envía a EventBridge sobre una vista previa de acceso con un cambio de estado de `Creating` a `Completed`. En el objeto de detalle, el `id` hace referencia al ID de la vista previa de acceso. El `status` y `previousStatus` hacen referencia al estado de vista previa de acceso, donde el estado anterior era `Creating` y el estado actual es `Completed`.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.000Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "COMPLETED",
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "11112222-3333-4444-5555-666677778888",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
```

```
    "version": "0"
}
```

En el ejemplo siguiente se muestran los datos de un evento que se envía a EventBridge sobre una vista previa de acceso con un cambio de estado de `Creating` a `Failed`. En el objeto `detail`, el `id` hace referencia al ID de la vista previa de acceso. El `status` y `previousStatus` hacen referencia al estado de la vista previa de acceso, donde el estado anterior era `Creating` y el estado actual es `Failed`. El campo `statusReason` proporciona el código de motivo que indica que la vista previa de acceso falló debido a una configuración de recursos inválida.

```
{
    "account": "111122223333",
    "detail": {
        "accessPreviewId": "aaaabbbb-cccc-dddd-eeee-ffffaaaabbbb",
        "configuredResources": [
            "arn:aws:s3:::example-bucket"
        ],
        "createdAt": "2020-02-20T00:00:00.00Z",
        "previousStatus": "CREATING",
        "region": "us-west-2",
        "status": "FAILED",
        "statusReason": {
            "code": "INVALID_CONFIGURATION"
        },
        "version": "1.0"
    },
    "detail-type": "Access Preview State Change",
    "id": "99998888-7777-6666-5555-444433332222",
    "region": "us-west-2",
    "resources": [
        "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/MyAnalyzer"
    ],
    "source": "aws.access-analyzer",
    "time": "2020-02-20T00:00:00.00Z",
    "version": "0"
}
```

Creación de una regla de evento mediante la consola

El siguiente procedimiento describe cómo crear regla de eventos utilizando la consola.

1. Abra la consola de Amazon EventBridge en https://console.aws.amazon.com/events/.

2. Con los siguientes valores, cree una regla de EventBridge que supervise la búsqueda de eventos o acceda a los eventos de vista previa:

   - En Tipo de regla, elija Regla con un patrón de evento.
   - En Origen del evento, elija Otro.
   - En Patrón del evento, elija Patrones personalizados (editor de JSON) y pegue uno de los siguientes ejemplos de patrones de eventos en el área de texto:
     - Para crear una regla basada en un evento de resultados de acceso no utilizado o acceso externo, utilice el siguiente ejemplo de patrón:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer Finding"
  ]
}
```

     - Para crear una regla basada únicamente en un evento de resultados de acceso no utilizado, utilice el siguiente ejemplo de patrón:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Unused Access Finding for IAM entities"
  ]
}
```

     > **Note**
     >
     > No puede crear una regla basada únicamente en un evento de resultados de acceso externo.

     - Para crear una regla basada en un evento de vista previa de acceso, utilice el siguiente ejemplo de patrón:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Preview State Change"
  ]
}
```

- En Tipos de destino, elija Servicio de AWS, y en Seleccionar un destino, elija un destino, como un tema de Amazon SNS o una función de AWS Lambda. El destino se activa cuando se recibe un evento que coincide con el patrón de eventos definido en la regla.

Para más información sobre cómo crear reglas de eventos, consulte Creación de reglas de Amazon EventBridge que reaccionan a eventos en la Guía de usuario de Amazon EventBridge.

Creación de una regla de evento mediante la CLI

1. Utilice lo siguiente para crear una regla para Amazon EventBridge mediante la AWS CLI. Reemplace el nombre de regla *TestRule* por el nombre de la regla.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"]}"
```

2. Puede personalizar la regla para desencadenar acciones de destino solo para un subconjunto de resultados generados, como resultados con atributos específicos. En el ejemplo siguiente, se muestra cómo crear una regla que desencadena una acción de destino solo para los resultados con el estado Activo.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-analyzer\"],\"detail-type\":[\"Access Analyzer Finding\"],\"detail\":{\"status\":[\"ACTIVE\"]}}"
```

En el ejemplo siguiente se muestra cómo crear una regla que desencadena una acción de destino solo para las vistas previas de acceso con estado de `Creating` a `Completed`.

```
aws events put-rule --name TestRule --event-pattern "{\"source\":[\"aws.access-
analyzer\"],\"detail-type\":[\"Access Preview State Change\"],\"detail\":{\"status
\":[\"COMPLETED\"]}}"
```

3. Para definir una función de Lambda como destino para la regla que ha creado, utilice el siguiente comando de ejemplo. Reemplace la Región y el nombre de la función en el ARN según corresponda para su entorno.

```
aws events put-targets --rule TestRule --targets Id=1,Arn=arn:aws:lambda:us-
east-1:111122223333:function:MyFunction
```

4. Agregue los permisos necesarios para invocar el destino de regla. En el ejemplo siguiente se muestra cómo conceder permisos a una función de Lambda, siguiendo los ejemplos anteriores.

```
aws lambda add-permission --function-name MyFunction --statement-id 1 --action
 'lambda:InvokeFunction' --principal events.amazonaws.com
```

Integre el analizador de acceso con AWS Security Hub

AWS Security Hub le proporciona una visión completa de su estado de seguridad en AWS y lo ayuda a comprobar su entorno con las prácticas recomendadas y los estándares del sector de seguridad. Security Hub recopila datos de seguridad de todas las cuentas de AWS, de los servicios y de los productos de terceros compatibles y le ayuda a analizar sus tendencias de seguridad y a identificar los problemas de seguridad de mayor prioridad.

La integración de AWS Identity and Access Management Access Analyzer con Security Hub permite enviar los resultados del Analizador de acceso de IAM a Security Hub. Security Hub puede incluir esos resultados en su análisis de la posición de seguridad.

Contenido

- Interpretación de los nombres de los resultados del Analizador de acceso de IAM en Security Hub
- Resultados típicos del Analizador de acceso de IAM
- Habilitación y configuración de la integración
- Cómo dejar de enviar resultados

Cómo el Analizador de acceso de IAM envía los resultados a Security Hub

En Security Hub, los problemas de seguridad se rastrean como resultados. Algunos resultados provienen de problemas detectados por otros servicios de AWS o por socios terceros. Security Hub también cuenta con un conjunto de reglas que utiliza para detectar problemas de seguridad y generar resultados.

Security Hub proporciona herramientas para administrar los resultados de todas estas fuentes. Puede ver y filtrar listas de resultados y ver los detalles de una búsqueda. Consulte Visualización de resultados en la Guía del usuario de AWS Security Hub. También puede realizar un seguimiento del estado de una investigación de un resultado. Consulte Adopción de medidas sobre los resultados en la Guía del usuario de AWS Security Hub.

Todos los resultados en Security Hub usan un formato JSON estándar denominado AWS Security Finding Format (ASFF). El ASFF incluye detalles sobre el origen del problema, los recursos afectados y el estado actual del resultado. Consulte Formato de resultado de seguridad de AWS (ASFF) en la Guía del usuario de AWS Security Hub.

AWS Identity and Access Management Access Analyzer es uno de los AWS servicios que envía los resultados a Security Hub. Para el acceso externo, el Analizador de acceso de IAM genera un resultado cuando detecta una declaración de la política que permite que una entidad principal externa pueda acceder a un recurso admitido en su organización o cuenta. El Analizador de acceso de IAM agrupa todos sus resultados para un recurso y envía un único resultado a Security Hub. Para el acceso no utilizado, el Analizador de acceso de IAM genera un resultado cuando detecta el acceso no utilizado concedido a los usuarios de IAM o sus roles. A continuación, el Analizador de acceso de IAM envía los resultados a Security Hub

Tipos de resultados que envía el Analizador de acceso de IAM

El Analizador de acceso de IAM envía los resultados a Security Hub mediante AWS Formato de resultados de seguridad (ASFF). En ASFF, el campo `Types` proporciona el tipo de resultado. Los resultados del Analizador de acceso de IAM pueden tener los siguientes valores para `Types`.

- Resultados de acceso externo: Efectos/Exposición de datos/Acceso Externo Concedido
- Resultados del acceso externo - Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/Acceso externo concedido
- Resultados de accesos no utilizados: comprobaciones de software y configuración/Prácticas recomendadas de seguridad de AWS/Permiso no utilizado
- Resultados de accesos no utilizados - Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/Rol de IAM no utilizado
- Resultados de accesos no utilizados - Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/Contraseña de usuario de IAM no utilizada
- Resultados de accesos no utilizados - Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/Clave de acceso de usuario de IAM no utilizada

Latencia para el envío de resultados

Cuando el Analizador de acceso de IAM crea un nuevo resultado, generalmente se envía a Security Hub dentro de 30 minutos. En raras ocasiones, y bajo determinadas condiciones, no se notifica al Analizador de acceso de IAM que se ha agregado o actualizado una política. Por ejemplo, un cambio en Amazon S3 la configuración de acceso público de bloqueo a nivel de cuenta puede tardar hasta 12 horas. Además, si hay un problema de entrega con AWS CloudTrail la entrega de registros, el cambio de política no activa un nuevo análisis del recurso que se comunicó en el resultado. Cuando esto sucede, el Analizador de acceso de IAM analiza la política nueva o actualizada durante el siguiente análisis periódico.

Reintento cuando Security Hub no está disponible

Si Security Hub no está disponible, el Analizador de acceso de IAM vuelve a intentar enviar los resultados periódicamente.

Actualización de los resultados existentes en Security Hub

Después de enviar un resultado a Security Hub, AWS Identity and Access Management Access Analyzer envía actualizaciones para reflejar observaciones adicionales de la actividad del resultado a Security Hub. Las actualizaciones se reflejan en el mismo resultado.

Como el Analizador de acceso de IAM agrupa los resultados de acceso externo por recurso, el resultado de un recurso en Security Hub está activo si al menos uno de los resultados del recurso del Analizador de acceso de IAM está activo. Si todos los resultados en el Analizador de acceso de

IAM para un recurso se archivan o resuelven, el resultado de Security Hub se archivará. El resultado de Security Hub se actualiza cuando cambia el acceso a la política entre el acceso público y entre cuentas múltiples. Esta actualización puede incluir cambios en el tipo, título, descripción y gravedad del resultado.

El Analizador de acceso de IAM no agrupa los resultados de acceso no utilizadas por recurso, por lo que si un resultado de acceso no utilizada se resuelve en el Analizador de acceso de IAM, se resuelve el resultado de Security Hub. El resultado de Security Hub se actualiza cuando actualiza el usuario o rol de IAM que generó el resultado de acceso no utilizado.

Visualización de los resultados del Analizador de acceso de IAM en Security Hub

Para ver sus resultados del Analizador de acceso de IAM en Security Hub, elija Ver resultados en la sección AWS: Analizador de acceso de IAM de la página de resumen. También puede seleccionar Resultados en el panel de navegación. A continuación, puede filtrar las conclusiones para mostrar solo los resultados de AWS Identity and Access Management Access Analyzer seleccionando el campo Nombre del producto: con un valor de **`IAM Access Analyzer`**.

Interpretación de los nombres de los resultados del Analizador de acceso de IAM en Security Hub

AWS Identity and Access Management Access Analyzer envía los resultados a Security Hub mediante el AWS Formato de resultados de seguridad (ASFF). En ASFF, el campo Tipos proporciona el tipo de resultado. Los tipos ASFF utilizan un esquema de nomenclatura diferente al de AWS Identity and Access Management Access Analyzer. En la tabla siguiente, se incluyen detalles sobre todos los tipos de ASFF asociados a los resultados de AWS Identity and Access Management Access Analyzer tal como aparecen en Security Hub.

Tipo de resultado de ASFF	Resultado de Security Hub	Descripción
Efectos/Exposición de datos/ Acceso Externo Concedido	<resource ARN>permite el acceso público	Una política basada en recursos adjunta al recurso permite el acceso público del recurso a todos los principales externos.
Comprobaciones de software y configuración/AWS Mejores	<resource ARN>permite el acceso entre cuentas	Una política basada en recursos adjunta al recurso permite el acceso entre

Tipo de resultado de ASFF	Resultado de Security Hub	Descripción
prácticas de seguridad/Acceso externo concedido		cuentas a principales externos fuera de la zona de confianza para el analizador.
Comprobaciones de software y configuración/Prácticas recomendadas de seguridad de AWS/Permisos no utilizados	<resource ARN> contiene permisos no utilizados	Un usuario o rol contiene permisos de servicio y acción no utilizados.
Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/ Rol de IAM no utilizado	<resource ARN> contiene un rol de IAM no utilizado	Un usuario o rol contiene un rol de IAM no utilizado.
Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad/ Contraseña de usuario de IAM no utilizada	<resource ARN> contiene una contraseña de usuario de IAM no utilizada	Un usuario o rol contiene una contraseña de usuario de IAM no utilizada.
Comprobaciones de software y configuración/AWS Prácticas recomendadas de seguridad /Clave de acceso de usuario de IAM no utilizado	<resource ARN> contiene una clave de acceso de usuario de IAM no utilizada	Un usuario o rol contiene una clave de acceso de usuario de IAM no utilizada.

Resultados típicos del Analizador de acceso de IAM

El Analizador de acceso de IAM envía los resultados a Security Hub mediante el Formato de resultados de seguridad AWS (ASFF).

Aquí hay un ejemplo de un resultado típico del Analizador de acceso de IAM para resultados de acceso externo.

```
{
```

```
    "SchemaVersion": "2018-10-08",
    "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/my-analyzer/
arn:aws:s3:::my-bucket",
    "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
    "GeneratorId": "aws/access-analyzer",
    "AwsAccountId": "111122223333",
    "Types": ["Software and Configuration Checks/AWS Security Best Practices/External
 Access Granted"],
    "CreatedAt": "2020-11-10T16:17:47Z",
    "UpdatedAt": "2020-11-10T16:43:49Z",
    "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
    },
    "Title": "AwsS3Bucket/arn:aws:s3:::my-bucket/ allows cross-account access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::my-bucket/ allows cross-account access
 from AWS 444455556666",
    "Remediation": {
        "Recommendation": {"Text": "If the access isn't intended, it indicates a
 potential security risk. Use the console for the resource to modify or remove the
 policy that grants the unintended access. You can use the Rescan button on the Finding
 details page in the Access Analyzer console to confirm whether the change removed the
 access. If the access is removed, the status changes to Resolved."}
    },
    "SourceUrl": "https://console.aws.amazon.com/access-analyzer/home?region=us-
west-2#/findings/details/dad90d5d-63b4-6575-b0fa-ef9c556ge798",
    "Resources": [
        {
            "Type": "AwsS3Bucket",
            "Id": "arn:aws:s3:::my-bucket",
            "Details": {
                "Other": {
                    "External Principal Type": "AWS",
                    "Condition": "none",
                    "Action Granted": "s3:GetObject,s3:GetObjectVersion",
                    "External Principal": "444455556666"
                }
            }
        }
    ],
    "WorkflowState": "NEW",
    "Workflow": {"Status": "NEW"},
    "RecordState": "ACTIVE"
```

```
}
```

Aquí hay un ejemplo de un resultado típico del Analizador de acceso de IAM para resultados de acceso externo.

```
{
    "Findings": [
    {
      "SchemaVersion": "2018-10-08",
      "Id": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-
DO-NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "ProductArn": "arn:aws:securityhub:us-west-2::product/aws/access-analyzer",
      "ProductName": "IAM Access Analyzer",
      "CompanyName": "AWS",
      "Region": "us-west-2",
      "GeneratorId": "aws/access-analyzer",
      "AwsAccountId": "111122223333",
      "Types": [
        "Software and Configuration Checks/AWS Security Best Practices/Unused
 Permission"
      ],
      "CreatedAt": "2023-09-18T16:29:09.657Z",
      "UpdatedAt": "2023-09-21T20:39:16.651Z",
      "Severity": {
        "Product": 1,
        "Label": "LOW",
        "Normalized": 1
      },
      "Title": "AwsIamRole/arn:aws:iam::111122223333:role/IsengardRole-DO-NOT-DELETE/
 contains unused permissions",
      "Description": "AWS::IAM::Role/arn:aws:iam::111122223333:role/IsengardRole-DO-
NOT-DELETE/ contains unused service and action-level permissions",
      "Remediation": {
        "Recommendation": {
          "Text":"If the unused permissions aren’t required, delete the permissions to
 refine access to your account. Use the IAM console to modify or remove the policy that
 grants the unused permissions. If all the unused permissions are removed, the status
 of the finding changes to Resolved."
        }
      },
      "SourceUrl": "https://us-west-2.console.aws.amazon.com/access-analyzer/
home?region=us-west-2#/unused-access-findings?resource=arn%3Aaws%3Aiam%3A
%3A903798373645%3Arole%2FTestRole",
```

```
      "ProductFields": {
      "numberOfUnusedActions": "256",
      "numberOfUnusedServices": "15",
      "resourceOwnerAccount": "111122223333",
      "findingId": "DEMO24d8d-0d3f-4d3d-99f4-299fc8a62ee7",
      "findingType": "UnusedPermission",
      "aws/securityhub/FindingId": "arn:aws:securityhub:us-west-2::product/aws/access-
analyzer/arn:aws:access-analyzer:us-west-2:111122223333:analyzer/integTestAnalyzer-DO-
NOT-DELETE/arn:aws:iam::111122223333:role/TestRole/UnusedPermissions",
      "aws/securityhub/ProductName": "AM Access Analyzer",
      "aws/securityhub/CompanyName": "AWS"
    },
    "Resources": [
    {
      "Type": "AwsIamRole",
      "Id": "arn:aws:iam::111122223333:role/TestRole"
    }
  ],
  "WorkflowState": "NEW",
  "Workflow": {
    "Status": "NEW"
    },
  "RecordState": "ARCHIVED",
  "FindingProviderFields": {
    "Severity": {
      "Label": "LOW"
    },
    "Types": [
    "Software and Configuration Checks/AWS Security Best Practices/Unused Permission"
    ]
  }
  }
]
}
```


Habilitación y configuración de la integración

Para utilizar la integración con Security Hub, debe activar Security Hub. Para obtener información acerca de cómo habilitar Security Hub, consulte la Configuración de Security Hub en la Guía del usuario de AWS Security Hub.

Cuando activa el Analizador de acceso de IAM y Security Hub, la integración se activa automáticamente. El Analizador de acceso de IAM comienza a enviar inmediatamente los resultados a Security Hub.

Cómo dejar de enviar resultados

Para dejar de enviar resultados a Security Hub, puede utilizar la consola de Security Hub o la API.

Consulte Desactivar y habilitar el flujo de resultados desde una integración (consola) o Desactivar el flujo de resultados desde una integración (Security Hub API, AWS CLI) en la Guía del usuario de AWS Security Hub.

Registro de llamadas a la API de IAM Access Analyzer con AWS CloudTrail

IAM Access Analyzer se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones que realiza un usuario, rol o servicio de AWS en IAM Access Analyzer. CloudTrail captura todas las llamadas a la API para IAM Access Analyzer como eventos. Las llamadas capturadas incluyen las llamadas realizadas desde la consola de IAM Access Analyzer y las llamadas de código a las operaciones de la API de IAM Access Analyzer.

Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon S3, incluidos los eventos para IAM Access Analyzer. Si no configura un registro de seguimiento, puede ver los eventos más recientes de la consola de CloudTrail en el Event history (Historial de eventos).

Mediante la información que recopila CloudTrail, puede determinar la solicitud que se realizó a IAM Access Analyzer, la dirección IP desde la que se realizó, quién la realizó y cuándo, entre otros detalles.

Para obtener más información acerca de CloudTrail, consulte la AWS CloudTrailGuía del usuario de .

Información de IAM Access Analyzer en CloudTrail

CloudTrail se habilita en su cuenta de AWS cuando la crea. Cuando se produce una actividad en IAM Access Analyzer, dicha actividad se registra en un evento de CloudTrail junto con los demás eventos de servicios de AWS en Event history (Historial de eventos). Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS. Para obtener más información, consulte Ver eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de los eventos en su cuenta de AWS, incluidos los eventos de IAM Access Analyzer, cree un registro de seguimiento. Un registro de seguimiento permite a

CloudTrail enviar archivos de registro a un bucket de Amazon S3. De manera predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para obtener más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- Servicios e integraciones compatibles con CloudTrail
- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de registro de CloudTrail de varias regiones y Recibir archivos de registro de CloudTrail de varias cuentas

Todas las acciones de IAM Access Analyzer las registra CloudTrail y se documentan en la Referencia de la API de IAM Access Analyzer de IAM. Por ejemplo, las llamadas a las acciones `CreateAnalyzer`, `CreateArchiveRule` y `ListFindings` generan entradas en los archivos de registros de CloudTrail.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con credenciales de usuario AWS Identity and Access Management (IAM) o credenciales de usuario raíz.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro servicio de AWS.

Para obtener más información, consulte el Elemento userIdentity de CloudTrail.

Comprensión de las entradas de los archivos de registros de IAM Access Analyzer

Un registro de seguimiento es una configuración que permite la entrega de eventos como archivos de registros en un bucket de Amazon S3 que especifique. Los archivos log de CloudTrail pueden contener una o varias entradas de log. Un evento representa una solicitud específica realizada desde un origen y contiene información sobre la acción solicitada, la fecha y la hora de la acción, los

parámetros de la solicitud, etc. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el ejemplo siguiente, se muestra una entrada de registro de CloudTrail que muestra la operación `CreateAnalyzer` realizada por una sesión de rol asumido llamada `Alice-tempcreds` el “14 de junio de 2021”. La sesión de rol fue emitida por el rol denominado `admin-tempcreds`.


```
{
  "eventVersion": "1.05",
  "userIdentity":   {
    "type": "AssumedRole",
    "principalId": "AROAIBKEVSQ6C2EXAMPLE:Alice-tempcreds",
    "arn": "arn:aws:sts::111122223333:assumed-role/admin-tempcreds/Alice-tempcreds",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "true",
        "creationDate": "2021-06-14T22:54:20Z"
      },
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "arn:aws:iam::111122223333:role/admin-tempcreds",
        "accountId": "111122223333",
        "userName": "admin-tempcreds"
      },
     "webIdFederationData": {},
    }
  },
  "eventTime": "2021-06-14T22:57:36Z",
  "eventSource": "access-analyzer.amazonaws.com",
  "eventName": "CreateAnalyzer",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "198.51.100.179",
  "userAgent": "aws-sdk-java/1.12.79 Linux/5.4.141-78.230 OpenJDK_64-
Bit_Server_VM/25.302-b08 java/1.8.0_302 vendor/Oracle_Corporation cfg/retry-mode/
standard",
  "requestParameters": {
    "analyzerName": "test",
    "type": "ACCOUNT",
    "clientToken": "11111111-abcd-2222-abcd-222222222222",
        "tags": {
```

```
            "tagkey1": "tagvalue1"
        }
  },
  "responseElements": {
    "arn": "arn:aws:access-analyzer:us-west-2:111122223333:analyzer/test"
  },
  "requestID": "22222222-dcba-4444-dcba-333333333333",
  "eventID": "33333333-bcde-5555-bcde-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",,
  "managementEvent": true,
  "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Claves de filtro del Analizador de acceso de IAM

Puede utilizar las siguientes claves de filtro para definir una regla de archivo (`CreateArchiveRule`), actualizar una regla de archivo (`UpdateArchiveRule`), recuperar una lista de resultados (`ListFindings` y `ListFindingsV2`) o recuperar una lista de resultados de vista previa para un recurso (`ListAccessPreviewFindings`). No hay diferencia entre el uso de la API de IAM y AWS CloudFormation para configurar reglas de archivo.

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
recurso	El ARN identifica de forma única el recurso al que tiene acceso la entidad principal externa. Para obtener más información, consulte los nombres de recursos de Amazon (ARN).	Cadena	Sí	Sí	Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
resourceType `AWS::IAM::Role | AWS::KMS::Key | AWS::Lambda::Function | AWS::Lambda::LayerVersion | AWS::S3::Bucket | AWS::S3Express::DirectoryBucket | AWS::SQS::Queue | AWS::SecretsManager::Secret | AWS::EFS::FileSystem | AWS::EC2::Snapshot`	Tipo de recurso al que tiene acceso la entidad principal externa.	Cadena	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
`| AWS::ECR::Repository | AWS::RDS::DBSnapshot | AWS::RDS::DBClusterSnapshot | AWS::SNS::Topic | AWS::DynamoDB::Stream | AWS::DynamoDB::Table`					
resourceOwnerAccount	El ID de 12 dígitos de la cuenta de AWS que posee el recurso. Para obtener más información, consulte Identificadores de la cuenta de AWS.	Cadena	 Sí	 Sí	 Sí
isPublic	Indica si la búsqueda informa de un recurso que tiene una política que permite el acceso público.	Booleano	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
findingType `UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission`	El tipo del resultado. Solo puede filtrar por tipo de resultado los resultados de acceso no utilizados.	Cadena	 Sí	 Sí	 Sí
estado `ACTIVE | ARCHIVED | RESOLVED`	El estado actual de la tarea.	Cadena	 No	 Sí	 Sí
error	Indica el error notificado para la búsqueda.	Cadena	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
principal.AWS	La cuenta concedió acceso al recurso en el campo `Principal` del resultado. Introduzca el ID de la cuenta de AWS de 12 dígitos o el ARN del usuario de AWS o rol externo. Para obtener más información, consulte Identificadores de la cuenta de AWS.	Cadena	 Sí	 Sí	 Sí
principal.Federated	El ARN de la identidad federada que tiene acceso al recurso en el resultado. Para obtener más información, consulte Federación y proveedores de identidades.	Cadena	 Sí	 Sí	 Sí
condition.aws:PrincipalArn	El ARN de la entidad principal (usuario, rol o grupo de IAM) indicado como condición para el acceso a los recursos. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Cadena	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
condition .aws:PrincipalOrgID	El identificador de organización de la entidad principal indicado como condición para el acceso a los recursos. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Cadena	 Sí	 Sí	 Sí
condition .aws:PrincipalOrgPaths	El identificador de organización o unidad organizativa (OU) indicado como condición para el acceso a los recursos. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Cadena	 Sí	 Sí	 Sí
condition .aws:SourceIp	La dirección IP que permite el acceso de la entidad principal al recurso cuando se utiliza la dirección IP especificada. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Dirección IP	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
condition .aws:SourceVpc	El ID de la VPC que permite el acceso de la entidad principal al recurso cuando se utiliza la VPC especificada. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Cadena	 Sí	 Sí	 Sí
condition .aws:UserId	El ID de usuario del usuario de IAM de una cuenta externa indicada como condición para acceder al recurso. Para obtener más información, consulte Claves de contexto de condición global de AWS.	Cadena	 Sí	 Sí	 Sí
condition .cognito-identity.amazonaws.com:aud	El ID del grupo de identidades de Amazon Cognito especificado como condición para el acceso a roles de IAM en la búsqueda. Para obtener más información, consulte Claves de contexto de condición de AWS STS y de IAM.	Cadena	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
condition.graph.facebook.com:app_id	El ID de la aplicación de Facebook (o ID del sitio) especificado como condición para permitir el acceso de la federación de Facebook al rol de IAM en el resultado. Para obtener más información, consulte Claves de contexto de condición de AWS STS y de IAM.	Cadena	 Sí	 Sí	 Sí
condition.accounts.google.com:aud	El ID de aplicación de Google especificado como condición para acceder al rol de IAM. Para obtener más información, consulte Claves de contexto de condición de AWS STS y de IAM.	Cadena	 Sí	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
condition .kms:CallerAccount	El ID de la cuenta de AWS que posee la entidad que llama (usuario, rol de IAM o usuario raíz de la cuenta) utilizada por los servicios que llaman a AWS KMS. Para obtener más información, consulte Claves de condición de AWS Key Management Service.	Cadena	 Sí	 Sí	 Sí
condition .www.amazon.com:app_id	El ID de la aplicación de Amazon (o ID de sitio) especificado como condición para permitir el acceso de la federación de Login with Amazon al rol. Para obtener más información, consulte	Cadena	 Sí	 Sí	 Sí
id	El ID del resultado.	Cadena	 No	 Sí	 Sí

Criterion	Descripción	Tipo	Regla de archivo	Lista de resultados	Detallar resultados de vista previa de acceso
changeType	Proporciona contexto sobre cómo se compara la búsqueda de vista previa de acceso con el acceso existente identificado en el Analizador de acceso de IAM.	Cadena	 No	 No	 Sí
existingFindingId	El ID existente de la búsqueda en el Analizador de acceso de IAM, proporcionado solo para los resultados existentes en la vista previa de acceso.	Cadena	 No	 No	 Sí
existingFindingStatus	El estado existente de la búsqueda, proporcionado solo para los resultados existentes en la vista previa de acceso.	Cadena	 No	 No	 Sí

Uso de roles vinculados a servicios de AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer utiliza un rol vinculado al servicio de (IAM). Un rol vinculado a un servicio es un tipo único de rol de IAM que está vinculado directamente a Access Analyzer. Los roles vinculados a servicios están predefinidos por IAM Access Analyzer e incluyen todos los permisos que la característica requiere para llamar a otros servicios de AWS en su nombre.

Un rol vinculado a un servicio simplifica la configuración de IAM Access Analyzer porque ya no tendrá que agregar manualmente los permisos necesarios. IAM Access Analyzer define los permisos de sus roles vinculados a servicios y, a menos que esté definido de otra manera, solo Access Analyzer puede asumir sus roles. Los permisos definidos incluyen las políticas de confianza y de permisos y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Para obtener información acerca de otros servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que muestran Yes (Sí) en la columna Service Linked Role (Rol vinculado a servicios). Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios de AWS Identity and Access Management Access Analyzer

AWS Identity and Access Management Access Analyzer IAM Access Analyzer utiliza un rol vinculado al servicio denominado AWSServiceRoleForAccessAnalyzer: permite al analizador de acceso analizar los metadatos del recurso.

El rol vinculado al servicio AWSServiceRoleForAccessAnalyzer confía en que los siguientes servicios asuman el rol:

- `access-analyzer.amazonaws.com`

La política de permisos del rol llamada `AccessAnalyzerServiceRolePolicy` permite que IAM Access Analyzer realice acciones en recursos específicos.

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Crear un rol vinculado a servicios para IAM Access Analyzer

No necesita crear manualmente un rol vinculado a servicios. Cuando habilita Access Analyzer en el AWS Management Console o el AWS API, IAM Access Analyzer crea el rol vinculado al servicio por usted. El mismo rol vinculado al servicio se utiliza en todas las regiones en las que se habilita IAM Access Analyzer. Tanto los resultados de acceso externo como los de acceso no utilizados utilizan la misma función vinculada al servicio.

> **Note**
>
> Analizador de acceso de IAM es regional. Debe habilitar IAM Access Analyzer en cada región de forma independiente.

Si elimina este rol vinculado al servicio, IAM Access Analyzer vuelve a crear el rol la próxima vez que cree un analizador.

También puede utilizar la consola de IAM para crear un rol vinculado al servicio con el caso de uso de Access Analyzer. En la AWS CLI o la API de AWS, cree un rol vinculado al servicio con el nombre de servicio `access-analyzer.amazonaws.com`. Para obtener más información, consulte Crear un rol vinculado a un servicio en la Guía del usuario de IAM. Si elimina este rol vinculado al servicio, puede utilizar este mismo proceso para volver a crear el rol.

Editar un rol vinculado a servicios para IAM Access Analyzer

IAM Access Analyzer no le permite editar el rol vinculado al servicio AWSServiceRoleForAccessAnalyzer. Después de crear un rol vinculado a servicios, no puede cambiarle el nombre, ya que varias entidades pueden hacer referencia al mismo. Sin embargo, puede editar la descripción del rol mediante IAM. Para obtener más información, consulte Editar un rol vinculado a servicios en la Guía del usuario de IAM..

Eliminar un rol vinculado a servicios para IAM Access Analyzer

Si ya no necesita utilizar una característica o servicio que requiere un rol vinculado a servicios, recomendamos que elimine dicho rol. De esta forma no conservará una entidad no utilizada que no se monitorice ni se mantenga de forma activa. Sin embargo, debe limpiar los recursos del rol vinculado al servicio antes de eliminarlo manualmente.

> **Note**
>
> Si IAM Access Analyzer está utilizando el rol cuando se intentan eliminar los recursos, es posible que se produzcan errores en la operación de eliminación. En tal caso, espere unos minutos e intente de nuevo la operación.

Eliminación de los recursos IAM de Access Analyzer que utiliza AWSServiceRoleForAccessAnalyzer

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.

2. En la sección Access reports (Informes de acceso), en Access analyzer (Analizador de acceso), elija Analyzers (Analizadores).
3. Seleccione la casilla de verificación situada en la parte superior izquierda de la lista de analizadores de la tabla Analizadores para seleccionar todos los analizadores.
4. Elija Eliminar.
5. Para confirmar que desea eliminar los analizadores, escriba **`delete`** y, a continuación, elija Delete (Eliminar).

Para eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAM, la AWS CLI o la API de AWS para eliminar el rol vinculado a servicios AWSServiceRoleForAccessAnalyzer. Para obtener más información, consulte Eliminación de un rol vinculado a servicios en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a un servicio de IAM Access Analyzer

IAM Access Analyzer admite el uso de roles vinculados a servicios en todas las regiones en las que el servicio está disponible. Para obtener más información, consulte Regiones y puntos de enlace de AWS.

Vista previa del acceso

Además de ayudarle a identificar los recursos que se comparten con una entidad externa, AWS el Analizador de acceso de IAM también le permite obtener una vista previa de los resultados del Analizador de acceso de IAM antes de implementar permisos de recursos, de modo que pueda validar que los cambios de política solo concedan acceso a sus recursos al público deseado y entre cuentas. Esto le ayuda a comenzar con el acceso externo previsto a sus recursos.

Puede obtener una vista previa y validar el acceso público y entre cuentas a los buckets de Amazon S3 en la consola de Amazon S3. También puede utilizar las API del Analizador de acceso de IAM para obtener una vista previa del acceso público y entre cuentas para sus buckets de Amazon S3, claves de AWS KMS, roles de IAM, colas de Amazon SQS y secretos de Secrets Manager proporcionando permisos propuestos para su recurso.

Temas

- Vista previa del acceso en la consola de Amazon S3

- Vista previa del acceso con las API de IAM Access Analyzer

Vista previa del acceso en la consola de Amazon S3

Después de completar la política de bucket en la consola de Amazon S3, tiene la opción de obtener la vista previa del acceso público y entre cuentas a su bucket de Amazon S3. Puede validar que los cambios de política solo concedan acceso externo previsto antes de elegir Save change (Guardar los cambios). Este paso opcional le permite obtener una vista previa de los hallazgos de AWS Identity and Access Management Access Analyzer para su bucket. Puede validar si el cambio de política presenta nuevos hallazgos o resuelve los hallazgos existentes para el acceso externo. Puede omitir este paso de validación y guardar su política de bucket de Amazon S3 en cualquier momento.

Para obtener una vista previa del acceso externo al bucket, debe tener un analizador de cuentas activo en la región de su bucket con la cuenta como zona de confianza. También debe tener los permisos necesarios para utilizar IAM Access Analyzer y obtener la vista previa del acceso. Para obtener más información acerca de cómo habilitar IAM Access Analyzer y los permisos necesarios, consulte Habilitación del Analizador de acceso de IAM.

Para obtener una vista previa del acceso a su bucket de Amazon S3 cuando cree o edite la política de bucket

1. Una vez que haya terminado de crear o editar su política de bucket, asegúrese de que su política es una política válida de bucket de Amazon S3. El ARN de la política debe coincidir con el ARN del bucket y los elementos de política deben ser válidos.
2. Debajo de la política, en Preview external access (Vista previa del acceso externo), elija un analizador de cuentas activo y, a continuación, elija Preview (Vista previa). Se genera una vista previa de los hallazgos de IAM Access Analyzer para el bucket. La vista previa analiza la política de bucket de Amazon S3 mostrada, junto con los permisos de bucket existentes. Esto incluye la configuración de BPA del bucket y de la cuenta, la ACL del bucket, los puntos de acceso de Amazon S3 y los puntos de acceso multirregión conectados al bucket, así como sus políticas y configuraciones de BPA.
3. Cuando se completa la vista previa del acceso, se muestra una vista previa de los hallazgos de IAM Access Analyzer. Cada hallazgo informa de una instancia de una entidad principal fuera de la cuenta con acceso a su bucket después de guardar la política. Puede validar el acceso a su bucket revisando cada hallazgo. El encabezado del hallazgo proporciona un resumen del acceso y puede expandir el hallazgo para revisar los detalles de hallazgos. Encontrar insignias proporciona contexto sobre cómo guardar la política de bucket que cambiaría el acceso

al bucket. Por ejemplo, puede ayudarlo a validar si el cambio de política introduce nuevos hallazgos o resuelve los hallazgos existentes para el acceso externo:

   a. Nuevo: indica un hallazgo de un acceso externo nuevo que la política presentaría.
   b. Resuelto: indica un hallazgo de un acceso externo existente que la política eliminaría.
   c. Archivado: indica un hallazgo de un acceso externo nuevo que se archivaría automáticamente, en función de las reglas de archivado del analizador que definen cuándo se deben marcar las conclusiones según lo previsto.
   d. Existente: indica un hallazgo existente de un acceso externo que permanecería sin cambios.
   e. Público: si un hallazgo es de un acceso público al recurso, tendrá una insignia Public (Público), además de una de las insignias anteriores.

4. Si identifica el acceso externo que no tiene intención de presentar o eliminar, puede revisar la política y, a continuación, elegir Preview (Vista previa) nuevamente hasta que haya logrado el acceso externo que desea. Si tiene un hallazgo con la etiqueta Public (Público), le recomendamos que revise la política para eliminar el acceso público antes de elegir Save changes (Guardar los cambios). La vista previa del acceso es un paso opcional y puede elegir Save changes (Guardar los cambios) en cualquier momento.

Vista previa del acceso con las API de IAM Access Analyzer

Puede utilizar las API de IAM Access Analyzer para obtener una vista previa del acceso público y entre cuentas para sus bucket de Amazon S3, claves AWS KMS, roles de IAM, colas de Amazon SQS y secretos de Secrets Manager. Puede obtener una vista previa del acceso proporcionando permisos propuestos para un recurso existente que posee o un nuevo recurso que desea implementar.

Para obtener una vista previa del acceso externo al recurso, debe tener un analizador de cuentas activo para la cuenta y la región del recurso. También debe tener los permisos necesarios para utilizar IAM Access Analyzer y obtener la vista previa del acceso. Para obtener más información acerca de cómo habilitar IAM Access Analyzer y los permisos necesarios, consulte Habilitación del Analizador de acceso de IAM.

Para obtener una vista previa del acceso de un recurso, puede utilizar la operación `CreateAccessPreview`, así como proporcionar el ARN del analizador y la configuración del control de acceso para el recurso. El servicio devuelve el ID único para la vista previa del acceso, que puede utilizar para verificar el estado de la vista previa del acceso con la operación `GetAccessPreview`. Cuando el estado es `Completed`, puede utilizar la operación `ListAccessPreviewFindings`

para recuperar los hallazgos generados para la vista previa del acceso. Las operaciones `GetAccessPreview` y `ListAccessPreviewFindings` recuperarán las vistas previas del acceso y los hallazgos creados en aproximadamente 24 horas.

Cada hallazgo recuperado contiene detalles de hallazgos que describen el acceso. Un estado de vista previa del hallazgo que describe si el hallazgo fuera `Active` ,`Archived`, o `Resolved` después de la implementación de permisos, y un `changeType`. `changeType` proporciona contexto sobre cómo se compara la hallazgo de vista previa del acceso con el acceso existente identificado en IAM Access Analyzer:

- Nuevo: el hallazgo es de un acceso recién introducido.
- Sin cambios: el hallazgo de vista previa es un hallazgo existente que permanecería sin cambios.
- Modificado: el hallazgo de vista previa es un hallazgo existente con un cambio de estado.

El `status` y el `changeType` le ayudan a comprender cómo la configuración de recursos cambiaría el acceso a recursos existentes. Si el `changeType` es `Unchanged` (No modificado) o Changed (Modificado), el hallazgo también contendrá el ID y el estado existentes de la búsqueda en IAM Access Analyzer. Por ejemplo, un hallazgo `Changed` con estado de vista previa `Resolved` y `Active` con estado de existente indica que el hallazgo `Active` existente para el recurso se convertiría en `Resolved` como consecuencia del cambio de permisos propuesto.

Puede utilizar la operación `ListAccessPreviews` para recuperar una lista de vistas previas de acceso para el analizador especificado. Esta operación recuperará información sobre la vista previa del acceso creada en aproximadamente una hora.

En general, si la vista previa del acceso es de un recurso existente y deja una opción de configuración sin especificar, la vista previa del acceso utilizará la configuración de recursos existentes de forma predeterminada. Para crear una vista previa de acceso para un nuevo bucket de Amazon S3 o un bucket de Amazon S3 existente de su propiedad, puede proponer una configuración de bucket especificando la política de bucket de Amazon S3, las ACL de bucket, la configuración de BPA del bucket y los puntos de acceso de Amazon S3, incluidos los puntos de acceso multirregión, conectados al bucket. Para obtener información sobre los casos de configuración de cada tipo de recurso, lea a continuación.

Vista previa del acceso a su bucket de Amazon S3

Para crear una vista previa de acceso para un nuevo bucket de Amazon S3 o un bucket de Amazon S3 existente de su propiedad, puede proponer una configuración de bucket especificando la política

de bucket de Amazon S3, las ACL de bucket, la configuración de BPA del bucket y los puntos de acceso de Amazon S3, incluidos los puntos de acceso multirregión, conectados al bucket.

> **Note**
>
> Antes de intentar crear una vista previa de acceso para un nuevo bucket, le recomendamos que llame a la operación Amazon S3 HeadBucket para verificar si el bucket con nombre ya existe. Esta operación es útil para determinar si existe un bucket y si tiene permiso para acceder a él.

Política de bucket: si la configuración es para un bucket de Amazon S3 existente y no especifica la política de bucket de Amazon S3, la vista previa del acceso utiliza la política existente adjunta al bucket. Si la vista previa del acceso es de un recurso nuevo y no especifica la política de bucket de Amazon S3, la vista previa del acceso supone que es un bucket sin una política. Para proponer la eliminación de una política de bucket existente, puede especificar una cadena vacía. Para obtener más información acerca de los límites de política de bucket compatibles, consulte Ejemplos de política de bucket.

Permisos de ACL de bucket: puede proponer hasta 100 permisos de ACL por bucket. Si la configuración de permisos propuesta es para un bucket existente, la vista previa del acceso utiliza la lista propuesta de configuraciones de permisos en lugar de los permisos existentes. De lo contrario, la vista previa del acceso utiliza los permisos existentes para el bucket.

Puntos de acceso del bucket: el análisis admite hasta 100 puntos de acceso por bucket, incluidos los puntos de acceso multirregión, lo que incluye hasta diez puntos de acceso nuevos que puede proponer por bucket. Si la configuración de punto de acceso de Amazon S3 es para un bucket existente, la vista previa de acceso utiliza la configuración de punto de acceso propuesto en lugar de los puntos de acceso existentes. Para proponer un punto de acceso sin una política, puede proveer una cadena vacía como política de punto de acceso. Para obtener más información acerca de los límites de política de puntos de acceso, consulte Restricciones y limitaciones de los puntos de acceso.

Configuración de bloqueo de acceso público: si la configuración propuesta es para un bucket de Amazon S3 existente y no especifica la configuración, la vista previa de acceso utiliza la configuración existente. Si la configuración propuesta es para un nuevo bucket y no especifica la configuración de BPA del bucket, la vista previa de acceso utiliza `false`. Si la configuración

propuesta es para un nuevo punto de acceso o un punto de acceso multirregión y no especifica la configuración de BPA del punto de acceso, la vista previa del acceso utiliza `true`.

Vista previa del acceso a su clave AWS KMS

Para crear una vista previa del acceso para una nueva clave AWS KMS o una clave AWS KMS existente que usted posee, puede proponer una configuración de clave AWS KMS especificando la política de clave y la configuración de permisos de AWS KMS.

Política de clave de AWS KMS: si la configuración es para una clave existente y no especifica la política de clave, la vista previa del acceso utiliza la política existente para la clave. Si la vista previa de acceso es para un recurso nuevo y no especifica la política de clave, la vista previa de acceso utiliza la política de clave predeterminada. La política de claves propuesta no puede ser una cadena vacía.

Permisos de AWS KMS: el análisis admite hasta 100 permisos de KMS por configuración*.* Si la configuración de permisos propuesta es para una clave existente, la vista previa de acceso utiliza la lista propuesta de configuraciones de permisos en lugar de los permisos existentes. De lo contrario, la vista previa de acceso utiliza las concesiones existentes para la clave.

Vista previa del acceso a su rol de IAM

Para crear una vista previa de acceso para un nuevo Rol de IAM o un Rol de IAM existente que usted posee, puede proponer una configuración de Rol de IAM especificando la política de confianza.

Política de confianza de roles: si la configuración es para un nuevo rol de IAM, debe especificar la política de confianza. Si la configuración es para un rol de IAM existente que posee y no propone la política de confianza, la vista previa de acceso utiliza la política de confianza existente para el rol. La política de confianza propuesta no puede ser una cadena vacía.

Vista previa del acceso a la cola de Amazon SQS

Para crear una vista previa de acceso para una nueva cola de Amazon SQS o una cola de Amazon SQS existente de su propiedad, puede proponer una configuración de cola de Amazon SQS especificando la política de Amazon SQS para la cola.

Política de colas de Amazon SQS: si la configuración es para una cola de Amazon SQS existente y no especifica la política de Amazon SQS, la vista previa del acceso utiliza la política existente de Amazon SQS para la cola. Si la vista previa de acceso es para un recurso nuevo y no especifica la

política, la vista previa de acceso supone una cola de Amazon SQS sin una política. Para proponer la eliminación de una política de cola de Amazon SQS existente, puede especificar una cadena vacía para la política de Amazon SQS.

Vista previa del acceso a su secreto de Secrets Manager

Para crear una vista previa de acceso para un nuevo secreto de Secrets Manager o un secreto de Secrets Manager existente de su propiedad, puede proponer una configuración secreta de Secrets Manager especificando la política secreta y la clave de cifrado opcional AWS KMS.

Política secreta: si la configuración es para un secreto existente y no especifica la política secreta, la vista previa de acceso utiliza la política existente para el secreto. Si la vista previa de acceso es para un recurso nuevo y no especifica la política, la vista previa de acceso supone un secreto sin una política. Para proponer la eliminación de una política existente, puede especificar una cadena vacía.

Clave de cifrado de AWS KMS: si la configuración propuesta es para un nuevo secreto y no especifica el ID de clave de AWS KMS, la vista previa de acceso utiliza la clave de KMS predeterminada de la cuenta de AWS. Si especifica una cadena vacía para el ID de clave de AWS KMS, la vista previa de acceso utiliza la clave de KMS predeterminada de la cuenta de AWS.

Comprobaciones para validar políticas

El Analizador de acceso de IAM proporciona comprobaciones de políticas que ayudan a validar sus políticas de IAM antes de adjuntarlas a una entidad. Entre ellas se incluyen las comprobaciones básicas de políticas que proporciona la validación de políticas para validar su política con respecto a la gramática de las políticas y las AWS prácticas recomendadas. Puede ver los resultados de las verificaciones de validación de políticas que incluyen advertencias de seguridad, errores, advertencias generales y sugerencias para la política.

Puede utilizar comprobaciones de políticas personalizadas para comprobar si hay nuevos accesos en función de tus estándares de seguridad. Se aplica un cargo a cada verificación de acceso nuevo. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM .

Temas

- Política de validación de Analizador de acceso de IAM
- Comprobaciones de políticas personalizadas del Analizador de acceso de IAM

Política de validación de Analizador de acceso de IAM

Puede validar sus políticas utilizando validación de políticas de AWS Identity and Access Management Access Analyzer. Puede crear o editar una política con la AWS CLI, API de AWS o editor de políticas JSON en la consola de IAM. Analizador de acceso de IAM valida su política contra la Gramática de la política de IAM y AWSprácticas recomendadas. Puede ver los resultados de las comprobaciones de validación de políticas que incluyen advertencias de seguridad, errores, advertencias generales y sugerencias para la política. Estos resultados proporcionan recomendaciones procesables que le ayudan a crear políticas funcionales y que se ajustan a las prácticas recomendadas de seguridad. Para ver una lista de las comprobaciones de políticas básicas que ejecuta IAM Access Analyzer, consulte Referencia de comprobación de políticas del Analizador de acceso.

Validación de políticas en IAM (consola)

Puede ver los resultados generados por la validación de las políticas al crear o editar una política administrada en la consola de IAM. También puede ver estos resultados para las políticas de usuario insertadas o rol. IAM Access Analyzer no genera estos resultados para las políticas insertadas de grupo.

Para ver los resultados generados por las verificaciones de políticas de IAM JSON

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Comience a crear o editar una política con uno de los siguientes métodos:

    a. Para crear una nueva política administrada, vaya a la página de Políticas y cree una política nueva. Para obtener más información, consulte Creación de políticas mediante el editor JSON.

    b. Para ver las comprobaciones de políticas para una política administrada por el cliente existente, vaya a la página Políticas, elija el nombre de una política y, a continuación, elija Editar. Para obtener más información, consulte Edición de políticas administradas por el cliente (Consola).

    c. Para ver las comprobaciones de políticas para una política insertada en un usuario o rol, vaya a la página Usuarios o Roles, elija el nombre de un usuario o rol, elija el nombre de la política en la pestaña Permisos y luego elija Editar. Para obtener más información, consulte Edición de políticas administradas por el cliente (Consola).
3. En el editor de política, elija la pestaña JSON.

4. En el panel de validación de políticas situado debajo de la política, elija una o varias de las siguientes pestañas. Los nombres de las pestañas también indican el número de cada tipo de búsqueda de la política.

   - Seguridad: ver advertencias si su política permite el acceso que AWS considera un riesgo de seguridad porque el acceso es excesivamente permisivo.
   - Errores: ver los errores si la política incluye líneas que impiden que la misma funcione.
   - Advertencias: consulte las advertencias si su política no se ajusta a las prácticas recomendadas, pero los problemas no implican riesgos para la seguridad.
   - Sugerencias: ver sugerencias si AWS recomienda mejoras que no afectan a los permisos de la política.

5. Revise los detalles de búsqueda proporcionados por la verificación de políticas de Analizador de acceso de IAM. Cada resultado indica la ubicación del problema notificado. Para obtener más información acerca de qué causa el problema y cómo resolverlo, elija el vínculo Más información junto a la conclusión. También puede buscar la verificación de políticas asociada a cada resultado en la página de referencia Verificaciones de políticas de Analizador de acceso.

6. Opcional. Si está editando una política existente, puede ejecutar una comprobación de política personalizada para determinar si la política actualizada concede nuevos accesos en comparación con la versión existente. En el panel de validación de políticas situado debajo de la política, elija la pestaña Comprobar acceso nuevo y, a continuación, elija Comprobar política. Si los permisos modificados otorgan un nuevo acceso, la declaración aparecerá resaltada en el panel de validación de la política. Si no tiene intención de conceder un nuevo acceso, actualice la declaración de política y elija Comprobar la política hasta que no se detecte ningún acceso nuevo. Para obtener más información, consulte Comprobaciones de políticas personalizadas del Analizador de acceso de IAM.

   > **Note**
   >
   > Se aplica un cargo a cada verificación de acceso nuevo. Para obtener más información sobre los precios, consulte los precios de IAM Access Analyzer.

7. Actualice su política para resolver los resultados.

> **Important**
>
> Pruebe minuciosamente las políticas nuevas o editadas antes de implementarlas en el flujo de trabajo de producción.

8. Cuando haya terminado, elija Next. El Validador de políticas informa de cualquier error de sintaxis que no haya informado IAM Access Analyzer.

> **Note**
>
> Puede alternar entre las pestañas Editor visual y JSON en cualquier momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual de la pestaña, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

9. Para las nuevas políticas, en la página Revisar y crear, introduzca el Nombre de la política y la Descripción (opcional) para la política que está creando. Revise los Permisos definidos en esta política para ver los permisos que concede la política. A continuación, elija Create policy (Crear política) para guardar su trabajo.

   Para las políticas existentes, en la página Revisar y guardar, revise los Permisos definidos en esta política para ver los permisos que concede la política. Seleccione Establecer esta nueva versión como predeterminada. casilla de verificación para guardar la versión actualizada como versión predeterminada de la política. A continuación, elija Guardar cambios para guardar su trabajo.

Validación de políticas mediante Analizador de acceso de IAM (AWS CLI o API de AWS)

Puede ver los resultados generados por la validación de políticas de Analizador de acceso de IAM desde el AWS Command Line Interface (AWS CLI).

Para ver los resultados generados por la verificación de políticas de Analizador de acceso de IAM (AWS CLI o API de AWS)

Utilice una de las siguientes:

- AWS CLI: aws accessanalyzer validate-policy

- API de AWS:ValidatePolicy

Referencia de comprobación de políticas del Analizador de acceso

Puede validar sus políticas utilizando validación de políticas de AWS Identity and Access Management Access Analyzer. Puede crear o editar una política con la AWS CLI, API de AWS o editor de políticas JSON en la consola de IAM. El Analizador de acceso de IAM valida su política contra la Gramática de la política de IAM y AWSprácticas recomendadas. Puede ver los resultados de las comprobaciones de validación de políticas que incluyen advertencias de seguridad, errores, advertencias generales y sugerencias para la política. Estos resultados proporcionan recomendaciones procesables que le ayudan a crear políticas funcionales y que se ajustan a las prácticas recomendadas de seguridad. La lista de comprobaciones de políticas básicas proporcionada por el Analizador de acceso de IAM se comparte a continuación. No se aplica ningún cargo adicional asociado a ejecutar las comprobaciones de validación de políticas. Para obtener más información sobre cómo validar políticas mediante validación de políticas, consulte Política de validación de Analizador de acceso de IAM.

Error: no se permite la cuenta ARN

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
ARN account not allowed: The service {{service}} does not support specifying an account
 ID in the resource ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The service {{service}} does not support specifying an account ID in
 the resource ARN."
```

Resolver el error

Elimine el ID de cuenta del ARN del recurso. Los ARN de recursos para algunos servicios AWS no admiten especificar un ID de cuenta.

Por ejemplo, Amazon S3 no admite un ID de cuenta como espacio de nombres en ARN de bucket. El nombre de un bucket de Amazon S3 es único en todo el mundo, y todas las cuentas de AWS comparten el espacio de nombres. Para consultar todos los tipos de recursos disponibles en Amazon

S3, consulte Tipos de recurso definidos por Amazon S3 en la Referencia de autorizaciones de servicio.

Términos relacionados

- Recursos de políticas
- Identificadores de cuenta
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error — Región ARN no permitida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
ARN Region not allowed: The service {{service}} does not support specifying a Region in
 the resource ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The service {{service}} does not support specifying a Region in the
 resource ARN."
```

Resolver el error

Elimine la región del ARN del recurso. Los ARN de recursos para algunos servicios AWS no admiten especificar una región.

Por ejemplo, IAM es un servicio global. La porción de Región de un ARN de recurso de IAM siempre se mantiene en blanco. Los recursos de IAM son globales, como una cuenta AWS lo es hoy. Por ejemplo, después de iniciar sesión como usuario de IAM, puede acceder a servicios AWS en cualquier región geográfica.

- Recursos de políticas
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error: no coincide con el tipo de datos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Data type mismatch: The text does not match the expected JSON data type {{data_type}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The text does not match the expected JSON data type {{data_type}}."
```

Resolver el error

Actualice el texto para utilizar el tipo de datos admitido.

Por ejemplo, la clave de condición global `Version` requiere un tipo de datos `String`. Si proporciona una fecha o un entero, el tipo de datos no coincidirá.

Términos relacionados

- Claves de condición global
- Elementos de la política de JSON de IAM: operadores de condición

Error — Claves duplicadas con diferentes mayúsculas

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Duplicate keys with different case: The condition key {{key}} appears more than once
 with different capitalization in the same condition block. Remove the duplicate
 condition keys.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key {{key}} appears more than once with different
 capitalization in the same condition block. Remove the duplicate condition keys."
```

Resolver el error

Revise las claves de condición similares dentro del mismo bloque de condición y utilice las mismas mayúsculas para todas las instancias.

Un bloque de condición es el texto dentro del elemento `Condition` de una declaración de política. Los nombres de las claves de condición no distinguen entre mayúsculas y minúsculas. El uso de mayúsculas y minúsculas en los valores de la clave de condición depende del operador de condición que utilice. Para obtener más información sobre el uso de mayúsculas y minúsculas en claves de condición, consulte Elementos de política JSON de IAM: Condition.

Términos relacionados

- Condiciones
- El bloque de condición
- Claves de condición global
- Claves de condición de servicio de AWS

Error: acción no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid action: The action {{action}} does not exist. Did you mean {{valid_action}}?
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The action {{action}} does not exist. Did you mean
 {{valid_action}}?"
```

Resolver el error

La acción especificada no es válida. Esto puede suceder si escribe mal el prefijo de servicio o el nombre de la acción. Para algunos problemas comunes, la verificación de políticas devuelve una acción sugerida.

Términos relacionados

- Acciones de las políticas
- Acciones de servicio de AWS

AWS políticas administradas con este error

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Los siguientes ejemplos de políticas administradas de AWS incluyen acciones no válidas en sus afirmaciones de política. Las acciones no válidas no afectan a los permisos concedidos por las políticas. Cuando se utiliza una política administrada de AWS como referencia para crear su política administrada,AWS recomienda que elimine las acciones no válidas de su política.

- AmazonEMRFullAccessPolicy_v2
- CloudWatchSyntheticsFullAccess

Error: cuenta ARN no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid ARN account: The resource ARN account ID {{account}} is not valid. Provide a
 12-digit account ID.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The resource ARN account ID {{account}} is not valid. Provide a 12-
digit account ID."
```

Resolver el error

Actualice el ID de cuenta en el ARN del recurso. Los ID de cuenta son números enteros de 12 dígitos. Para obtener información sobre cómo ver el ID de su cuenta de, consulte Búsqueda del ID de su cuenta de AWS.

Términos relacionados

- Recursos de políticas
- Identificadores de cuenta
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error: prefijo ARN no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid ARN prefix: Add the required prefix (arn) to the resource ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add the required prefix (arn) to the resource ARN."
```

Resolver el error

Los ARN de recursos de AWS deben incluir el prefijo `arn:`.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error — Región ARN no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid ARN Region: The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Region {{region}} is not valid for this resource. Update the
 resource ARN to include a supported Region."
```

Resolver el error

El tipo de recurso no se admite en la región especificada. Para ver una tabla de los servicios de AWS admitidos en cada región, consulte la Tabla de regiones.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Nombres y códigos de regiones

Error: recurso ARN no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid ARN resource: Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Resource ARN does not match the expected ARN format. Update the
 resource portion of the ARN."
```

Resolver el error

El ARN de recursos debe coincidir con las especificaciones de los tipos de recursos conocidos. Para ver el formato ARN esperado para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio para ver sus tipos de recursos y formatos ARN.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error: caso de servicio ARN no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid ARN service case: Update the service name ${service} in the resource ARN to use
 all lowercase letters.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Update the service name ${service} in the resource ARN to use all
 lowercase letters."
```

Resolver el error

El servicio en el ARN del recurso debe coincidir con las especificaciones (incluidas las mayúsculas) de los prefijos de servicio. Para ver el prefijo para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio y busque su prefijo en la primera oración.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error: tipo de datos de condición no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid condition data type: The condition value data types do not match. Use condition
 values of the same JSON data type.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition value data types do not match. Use condition values of
 the same JSON data type."
```

Resolver el error

El valor del par clave-valor de condición debe coincidir con el tipo de datos de la clave de condición y el operador de condición. Para ver el tipo de dato de clave de condición para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio para ver las claves de condición de dicho servicio.

Por ejemplo, la clave de condición global `CurrentTime` admite la el operador de condición `Date`. Si proporciona una cadena o un entero para el valor en el bloque de condición, el tipo de datos no coincidirá.

Términos relacionados

- Condiciones
- El bloque de condición
- Elementos de la política de JSON de IAM: operadores de condición
- Claves de condición global
- Claves de condición de servicio de AWS

Error: formato de clave de condición no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid condition key format: The condition key format is not valid. Use the format
 service:keyname.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key format is not valid. Use the format
 service:keyname."
```

Resolver el error

La clave del par clave-valor de condición debe coincidir con las especificaciones del servicio. Para ver las claves de condición para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio para ver las claves de condición de dicho servicio.

Términos relacionados

- Condiciones
- Claves de condición global
- Claves de condición de servicio de AWS

Error — Condición inválida de varios booleanos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid condition multiple Boolean: The condition key does not support multiple Boolean
 values. Use a single Boolean value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key does not support multiple Boolean values. Use a
 single Boolean value."
```

Resolver el error

La clave del par clave-valor de condición espera un único valor booleano. Cuando proporciona varios valores booleanos, es posible que la coincidencia de condición no devuelva los resultados esperados.

Para ver las claves de condición para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio para ver las claves de condición de dicho servicio.

- Condiciones
- Claves de condición global
- Claves de condición de servicio de AWS

Error: operador de condición no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid condition operator: The condition operator {{operator}} is not valid. Use a
 valid condition operator.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition operator {{operator}} is not valid. Use a valid
 condition operator."
```

Resolver el error

Actualice la condición para utilizar un operador de condición compatible.

Términos relacionados

- Elementos de la política de JSON de IAM: operadores de condición
- Elemento de condición
- Información general de políticas de JSON

Error — Efecto no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid effect: The effect {{effect}} is not valid. Use Allow or Deny.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The effect {{effect}} is not valid. Use Allow or Deny."
```

Resolver el error

Actualizar el elemento `Effect` para utilizar un efecto válido. Los valores válidos para `Effect` son **`Allow`** y **`Deny`**.

Términos relacionados

- Elemento de efecto
- Información general de políticas de JSON

Error: clave de condición global no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid global condition key: The condition key {{key}} does not exist. Use a valid
 condition key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key {{key}} does not exist. Use a valid condition
 key."
```

Resolver el error

Actualice la clave de condición en el par clave-valor de condición para utilizar una clave de condición global compatible.

Las claves de condición globales son claves de condición con un prefijo `aws:`. Los servicios de AWS pueden admitir claves de condición globales o proporcionar claves específicas del servicio que incluyan su prefijo de servicio. Por ejemplo, las claves de condición de IAM incluyen el prefijo `iam:`. Para obtener más información, vea Acciones, recursos y claves de condición para servicios de AWS y elija el servicio cuyas claves desea ver.

Términos relacionados

- Claves de condición global

Error — Partición no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid partition: The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The resource ARN for the service {{service}} does not support the
 partition {{partition}}. Use the supported values: {{partitions}}"
```

Resolver el error

Actualice el ARN del recurso para incluir una partición compatible. Si ha incluido una partición compatible, es posible que el servicio o recurso no admita la partición que ha incluido.

Una partición es un grupo de regiones de AWS. Cada cuenta de AWS está limitada a una partición. En Regiones clásicas, utilice la partición `aws`. En las regiones de China, utilice `aws-cn`.

Términos relacionados

- Nombres de recursos de Amazon (ARN) - Particiones

Error: elemento de política no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid policy element: The policy element {{element}} is not valid.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy element {{element}} is not valid."
```

Resolver el error

Actualice la política para incluir solo los elementos de política JSON compatibles.

Términos relacionados

- Elementos de la política JSON

Error: formato de la entidad principal no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid principal format: The Principal element contents are not valid. Specify a key-
value pair in the Principal element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Principal element contents are not valid. Specify a key-value
 pair in the Principal element."
```

Resolver el error

Actualice la entidad principal para utilizar un formato de par clave-valor compatible.

Puede especificar una entidad principal en una política basada en recursos, pero no una política basada en identidad.

Por ejemplo, para definir el acceso para todos en una cuenta AWS, utilice la siguiente entidad principal en su política:

```
"Principal": { "AWS": "123456789012" }
```

Términos relacionados

- Elementos de la política JSON: entidad principal
- Políticas basadas en identidad y políticas basadas en recursos

Error: clave de la entidad principal no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid principal key: The principal key {{principal-key}} is not valid.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The principal key {{principal-key}} is not valid."
```

Resolver el error

Actualice la clave en el par clave-valor de entidad principal para utilizar una clave de entidad principal compatible. Las siguientes son las claves de entidades principales admitidas:

- AWS
- CanonicalUser
- Federado
- Servicio

Términos relacionados

- Elemento principal

Error: región no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 suported Region.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Region {{region}} is not valid. Update the condition value to a
 suported Region."
```

Resolver el error

Actualice el valor del par clave-valor de condición para incluir una región admitida. Para ver una tabla de los servicios de AWS admitidos en cada región, consulte la Tabla de regiones.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Nombres y códigos de regiones

Error: servicio no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid service: The service {{service}} does not exist. Use a valid service name.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The service {{service}} does not exist. Use a valid service name."
```

Resolver el error

El prefijo de servicio en la clave de acción o condición debe coincidir con las especificaciones (incluidas las mayúsculas) de los prefijos de servicio. Para ver el prefijo para un servicio, consulte

Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio y busque su prefijo en la primera oración.

Términos relacionados

- Los servicios conocidos y sus acciones, los recursos y las claves de condición de los servicios conocidos

Error: clave de condición de servicio no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid service condition key: The condition key {{key}} does not exist in the service
 {{service}}. Use a valid condition key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key {{key}} does not exist in the service {{service}}.
 Use a valid condition key."
```

Resolver el error

Actualice la clave en el par clave-valor de condición para utilizar una clave de condición conocida para el servicio. Los nombres de las claves de condición globales comienzan con el prefijo `aws`. Los servicios de AWS pueden proporcionar claves específicas de servicios que incluyen el prefijo de servicio. Para ver el prefijo para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS.

Términos relacionados

- Claves de condición global
- Los servicios conocidos y sus acciones, los recursos y las claves de condición de los servicios conocidos

Error: servicio no válido en acción

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid service in action: The service {{service}} specified in the action does not
 exist. Did you mean {{service2}}?
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The service {{service}} specified in the action does not exist. Did
 you mean {{service2}}?"
```

Resolver el error

El prefijo de servicio de la acción debe coincidir con las especificaciones (incluidas las mayúsculas) de los prefijos de servicio. Para ver el prefijo para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio y busque su prefijo en la primera oración.

Términos relacionados

- Elemento de acción
- Servicios conocidos y sus acciones

Error: variable no válida para el operador

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid variable for operator: Policy variables can only be used with String and ARN
 operators.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Policy variables can only be used with String and ARN operators."
```

Resolver el error

Puede utilizar variables de política en el elemento `Resource` y en la comparación de cadenas en el elemento `Condition`. Las condiciones admiten variables cuando se utilizan operadores de

cadena u operadores ARN. Los operadores de cadena incluyen `StringEquals`, `StringLike` y `StringNotLike`. Los operadores ARN incluyen `ArnEquals` y `ArnLike`. No se puede utilizar una variable de política con otros operadores como operadores numéricos, fecha, booleanos, binarios, dirección de IP o nulos.

Términos relacionados

- Uso de variables de política en el elemento Condition
- Elemento de condición

Error: versión no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid version: The version ${version} is not valid. Use one of the following
 versions: ${versions}
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The version ${version} is not valid. Use one of the following
 versions: ${versions}"
```

Resolver el error

El elemento de la política `Version` especifica las reglas de sintaxis del lenguaje que AWS va a utilizar para procesar esta política. Para utilizar todas las características disponibles de la política, incluya el último elemento `Version` antes del elemento `Statement` en todas sus políticas.

```
"Version": "2012-10-17"
```

Términos relacionados

- Elementos Version

Error — Error de sintaxis Json

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Json syntax error: Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Fix the JSON syntax error at index {{index}} line {{line}} column
 {{column}}."
```

Resolver el error

La política incluye un error de sintaxis. Compruebe su sintaxis JSON.

Términos relacionados

- Validador de JSON
- Referencia de los elementos de las políticas de JSON de IAM
- Información general de políticas de JSON

Error — Error de sintaxis Json

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Json syntax error: Fix the JSON syntax error.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Fix the JSON syntax error."
```

Resolver el error

La política incluye un error de sintaxis. Compruebe su sintaxis JSON.

Términos relacionados

- Validador de JSON

- Referencia de los elementos de las políticas de JSON de IAM
- Información general de políticas de JSON

Error: falta acción

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing action: Add an Action or NotAction element to the policy statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add an Action or NotAction element to the policy statement."
```

Resolver el error

Las políticas JSON AWS deben incluir un elemento `Action` o `NotAction`.

Términos relacionados

- Elemento de acción
- Elemento NotAction
- Información general de políticas de JSON

Error — Falta el campo ARN

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing ARN field: Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Resource ARNs must include at least {{fields}} fields in the
 following structure: arn:partition:service:region:account:resource"
```

Resolver el error

Todos los campos del ARN del recurso deben coincidir con las especificaciones de un tipo de recurso conocido. Para ver el formato ARN esperado para un servicio, consulte Acciones, recursos y claves de condiciones para Servicios de AWS. Elija el nombre del servicio para ver sus tipos de recursos y formatos ARN.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Recursos de servicio AWS con formatos ARN

Error: falta región ARN

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing ARN Region: Add a Region to the {{service}} resource ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a Region to the {{service}} resource ARN."
```

Resolver el error

Los ARN de recursos para la mayoría de los servicios AWS requieren que especifique una región. Para ver una tabla de los servicios de AWS admitidos en cada región, consulte la Tabla de regiones.

Términos relacionados

- Recursos de políticas
- ARN del recurso
- Nombres y códigos de regiones

Error — Falta el efecto

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing effect: Add an Effect element to the policy statement with a value of Allow or
 Deny.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add an Effect element to the policy statement with a value of Allow
 or Deny."
```

Resolver el error

Las políticas JSON AWS deben incluir un elemento `Effect` con un valor de **`Allow`** y **`Deny`**.

Términos relacionados

- Elemento de efecto
- Información general de políticas de JSON

Error — Falta la entidad principal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing principal: Add a Principal element to the policy statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a Principal element to the policy statement."
```

Resolver el error

Las políticas basadas en recursos deben incluir un elemento `Principal`.

Por ejemplo, para definir el acceso para todos en una cuenta AWS, utilice la siguiente entidad principal en su política:

```
"Principal": { "AWS": "123456789012" }
```

Términos relacionados

- Elemento principal
- Políticas basadas en identidad y políticas basadas en recursos

Error — Falta calificador

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing qualifier: The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The request context key ${key} has multiple values. Use the
 ForAllValues or ForAnyValue condition key qualifiers in your policy."
```

Resolver el error

En el elemento `Condition`, se crean expresiones en las que se usan operadores de condición como igual o menor para comparar una condición en la política con relación a claves y valores en el contexto de la solicitud. Para las solicitudes que incluyen varios valores para una única clave de condición, debe incluir las condiciones entre corchetes, como una matriz ("Key2": ["Value2A", "Value2B"]). También debe utilizar los operadores `ForAllValues` o `ForAnyValue` con el operador de condición `StringLike`. Estos calificadores añaden la funcionalidad de operación de definición al operador de condición para que pueda probar varios valores con varios valores de condición.

Términos relacionados

- Claves de contexto multivalor
- Elemento de condición

AWS políticas administradas con este error

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Las siguientes políticas administradas AWS incluyen un calificador que falta para las claves de condición en sus declaraciones de política. Cuando se utiliza la política administrada AWS como

referencia para crear su política administrada por el cliente, AWS recomienda que agregue los calificadores de clave de condición `ForAllValues` o `ForAnyValue` a su elemento`Condition`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Error: falta recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing resource: Add a Resource or NotResource element to the policy statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a Resource or NotResource element to the policy statement."
```

Resolver el error

Todas las políticas, excepto las políticas de confianza de roles, deben incluir un elemento `Resource` o `NotResource`.

Términos relacionados

- Elemento de recurso
- Elemento NotResource
- Políticas basadas en identidad y políticas basadas en recursos
- Información general de políticas de JSON

Error — Falta la instrucción

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing statement: Add a statement to the policy
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a statement to the policy"
```

Resolver el error

Una política JSON debe incluir una instrucción.

Términos relacionados

- Elementos de la política JSON

Error: nulo con si existe

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Null with if exists: The Null condition operator cannot be used with the IfExists
 suffix. Update the operator or the suffix.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Null condition operator cannot be used with the IfExists suffix.
 Update the operator or the suffix."
```

Resolver el error

Puede agregar `IfExists` al final de cualquier nombre de operador de condición, salvo el operador de condición `Null`. Utilice un operador de condición `Null` para comprobar si una clave de condición está presente en el momento de la autorización. Use `...ifExists` para decir lo siguiente: "Si la clave de la política está presente en el contexto de la solicitud, se debe procesar la clave según se indica en la política. Si la clave no está presente, el elemento de condición se evalúa en verdadero".

Términos relacionados

- Operadores de condición ...IfExists
- Operador de condición nulo
- Elemento de condición

Error: comodín de acción de error de sintaxis de SCP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error action wildcard: SCP actions can include wildcards (*) only at the end
 of a string. Update {{action}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCP actions can include wildcards (*) only at the end of a string.
 Update {{action}}."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations admiten la especificación de valores en elementos `Action` o `NotAction`. Sin embargo, estos valores pueden incluir comodines (*) solo al final de la cadena. Esto significa que puede especificar `iam:Get*` pero no `iam:*role`.

Para especificar varias acciones, AWS recomienda que los enumere individualmente.

Términos relacionados

- Elementos Acción de SCP y NotAction
- Evaluación de SCP
- Políticas de control de servicios AWS Organizations
- Elementos de la política de JSON de IAM: acción

Error: condición de error de sintaxis de SCP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect
 Allow. Update the element Condition or the effect.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCPs do not support the Condition element with effect Allow. Update
 the element Condition or the effect."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations admiten la especificación de valores en el elemento `Condition` solo cuando usted utiliza `"Effect": "Deny"`.

Para permitir una sola acción, puede denegar el acceso a todo excepto a la condición que especifique mediante la versión `...NotEquals` de un operador de condición. Esto niega la comparación hecha por el operador.

Términos relacionados

- Elemento de condición SCP
- Evaluación de SCP
- Políticas de control de servicios AWS Organizations
- Política de ejemplo: denegar el acceso a AWS en función de la región solicitada
- Elementos de la política de JSON de IAM: operadores de condición
- Elementos de la política de JSON de IAM: condición

Error – Error de sintaxis de SCP permitir NotAction

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow.
 Update the element NotAction or the effect.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCPs do not support NotAction with effect Allow. Update the element
 NotAction or the effect."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations no admiten el uso del elemento `NotAction` con `"Effect": "Allow"`.

Debe volver a escribir la lógica para permitir una lista de acciones o para denegar todas las acciones que no se indican en la lista.

Términos relacionados

- Elementos Acción de SCP y NotAction
- Evaluación de SCP
- Políticas de control de servicios AWS Organizations
- Elementos de la política de JSON de IAM: acción

Error – Error – SCP de sintaxis de SCP permitir recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update
 the element Resource or the effect.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCPs do not support Resource with effect Allow. Update the element
 Resource or the effect."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations admiten la especificación de valores en el elemento `Resource` solo cuando usted utiliza `"Effect": "Deny"`.

Debe volver a escribir la lógica para permitir todos los recursos o para denegar todos los recursos que aparecen en la lista.

Términos relacionados

- Elemento de recurso SCP
- Evaluación de SCP
- Políticas de control de servicios AWS Organizations
- Elementos de la política de JSON de IAM;: recurso

Error – Error de SCP de sintaxis NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the
 policy to use Resource instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCPs do not support the NotResource element. Update the policy to
 use Resource instead."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations no admiten el elemento `NotResource`

Debe volver a escribir la lógica para permitir todos los recursos o para denegar todos los recursos que aparecen en la lista.

Términos relacionados

- Elemento de recurso SCP
- Evaluación de SCP
- Políticas de control de servicios AWS Organizations
- Elementos de la política de JSON de IAM;: recurso

Error – Entidad principal de error de sintaxis de SCP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
SCP syntax error principal: SCPs do not support specifying principals. Remove the
 Principal or NotPrincipal element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "SCPs do not support specifying principals. Remove the Principal or
 NotPrincipal element."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations no admiten el `Principal` o elementos `NotPrincipal`

Puede especificar el nombre de recurso de Amazon (ARN) con la clave de condición global `aws:PrincipalArn` en el elemento `Condition`.

Términos relacionados

- Sintaxis de SCP
- Claves de condición globales para entidades principales

Error: se requieren Sids únicos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unique Sids required: Duplicate statement IDs are not supported for this policy type.
 Update the Sid value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Duplicate statement IDs are not supported for this policy type.
 Update the Sid value."
```

Resolver el error

Para algunos tipos de políticas, los ID de instrucción deben ser únicos. El elemento `Sid` (ID de instrucción) le permite ingresar un identificador opcional que se proporciona para la instrucción de la política. Puede asignar un valor de ID de instrucción a cada instrucción de una matriz de instrucciones utilizando el elemento `SID`. En los servicios que le permiten especificar un elemento , como, por ejemplo, SQS y SNS, el valor de `Sid` es simplemente un subID del ID del documento de la política. Por ejemplo, en IAM, el valor de `Sid` debe ser único en la política de JSON.

Términos relacionados

- Elemento de la política de JSON de IAM: Sid

Error: acción no admitida en la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver el error

Algunas acciones no se admiten en el elemento `Action` de la política basada en recursos asociada a otro tipo de recurso. Por ejemplo, las acciones de AWS Key Management Service no son compatibles con las políticas de bucket de Amazon S3. Especifique una acción compatible con el tipo de recurso adjunto a la política basada en recursos.

Términos relacionados

- Elementos de la política de JSON: acción

Error: combinación de elementos no admitidos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported element combination: The policy elements ${element1} and ${element2} can
 not be used in the same statement. Remove one of these elements.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy elements ${element1} and ${element2} can not be used in
 the same statement. Remove one of these elements."
```

Resolver el error

Algunas combinaciones de elementos de política JSON no se pueden utilizar juntas. Por ejemplo, no se puede utilizar `Action` y `NotAction` en la misma instrucción de política. Otros pares que se excluyen mutuamente son `Principal/NotPrincipal` y `Resource/NotResource`.

Términos relacionados

- Referencia de los elementos de las políticas de JSON de IAM
- Información general de políticas de JSON

Error: clave de condición global no admitida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported global condition key: The condition key aws:ARN is not supported. Use
 aws:PrincipalArn or aws:SourceArn instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key aws:ARN is not supported. Use aws:PrincipalArn or
 aws:SourceArn instead."
```

Resolver el error

AWS no admite el uso de la clave de condición global especificada. Dependiendo de su caso de uso, puede utilizar las claves de condición globales de `aws:PrincipalArn` o `aws:SourceArn`. Por ejemplo, en vez de `aws:ARN`, utilice `aws:PrincipalArn` para comparar el Nombre de recurso de Amazon (ARN) de la entidad principal que ha realizado la solicitud con el ARN que se especifique en la política. Alternativamente, utilice la clave de condición global `aws:SourceArn` para comparar el Nombre de recurso de Amazon (ARN) del recurso que realiza una solicitud de servicio a servicio con el ARN que especifique en la política.

Términos relacionados

- Claves de contexto de condición global de AWS

Error – Entidad principal no admitida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported principal: The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy type ${policy_type} does not support the Principal
 element. Remove the Principal element."
```

Resolver el error

El elemento `Principal` especifica la entidad principal que tiene acceso permitido o denegado a un recurso. No puede utilizar el elemento `Principal` en una política basada en identidad de IAM. Puede utilizarlo en las políticas de confianza para los roles de IAM y en las políticas basadas en recursos. Las políticas basadas en recursos son políticas que se integran directamente en un recurso. Por ejemplo, puede integrar las políticas en un bucket de Amazon S3 o en una clave KMS de AWS.

Términos relacionados

- AWS Elementos de la política JSON: entidad principal
- Acceso a recursos entre cuentas en IAM

Error: ARN de recurso no admitido en la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver el error

Algunos ARN de recursos no se admiten en el elemento `Resource` de la política basada en recursos cuando la política se asocia a otro tipo de recurso. Por ejemplo, los ARN de AWS KMS no son compatibles en el elemento `Resource` con las políticas de bucket de Amazon S3. Especifique un ARN de recurso compatible con un tipo de recurso asociado a la política basada en recursos.

Términos relacionados

- Elementos de la política de JSON: acción

Error: Sid no admitido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported Sid: Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Update the characters in the Sid element to use one of the following
 character types: [a-z, A-Z, 0-9]"
```

Resolver el error

El elemento `Sid` admite letras mayúsculas, minúsculas y números.

Términos relacionados

- Elemento de la política de JSON de IAM: Sid

Error: comodín no admitido en la entidad principal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported wildcard in principal: Wildcards (*, ?) are not supported with the
 principal key {{principal_key}}. Replace the wildcard with a valid principal value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Wildcards (*, ?) are not supported with the principal key
 {{principal_key}}. Replace the wildcard with a valid principal value."
```

Resolver el error

La estructura del elemento de `Principal` admite el uso de un par de clave-valor. El valor principal especificado en la política incluye un comodín (*). No se puede incluir un comodín con la clave principal especificada. Por ejemplo, si especifica usuarios en un elemento `Principal`, no puede utilizar un comodín para designar a "todos los usuarios". Debe designar a un usuario o usuarios específicos. De manera similar, cuando especifica una sesión de rol asumido, no puede utilizar un comodín (*) para referirse a "todas las sesiones". Debe nombrar una sesión específica. Además, no puede utilizar un carácter comodín para buscar coincidencias con parte de un nombre o un ARN.

Para resolver este resultado, elimine el comodín y proporcione una entidad principal más específica.

Términos relacionados

- AWS Elementos de la política JSON: entidad principal

Error — Falta tornapunta en la variable

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing brace in variable: The policy variable is missing a closing curly brace. Add }
 after the variable text.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy variable is missing a closing curly brace. Add } after
 the variable text."
```

Resolver el error

La estructura de variables de política admite el uso de un prefijo $ seguido de un par de llaves (`{ }`). Dentro de los caracteres `${ }`, incluya el nombre del valor de la solicitud que quiere utilizar en la política.

Para resolver este resultado, añada la clave faltante para asegurarse de que el conjunto completo de claves de apertura y cierre esté presente.

Términos relacionados

- Elementos de la política de IAM: variables

Error: falta comilla en la variable

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing quote in variable: The policy variable default value must begin and end with a
 single quote. Add the missing quote.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy variable default value must begin and end with a single
 quote. Add the missing quote."
```

Resolver el error

Cuando agrega una variable a la política, puede especificar un valor predeterminado para la variable. Si no hay una variable, AWS utiliza el texto predeterminado que proporcione.

Para agregar un valor predeterminado a una variable, rodee el valor predeterminado entre comillas simples (`' '`), y separe el texto de la variable y el valor predeterminado con una coma y un espacio (`, `).

Por ejemplo, si una entidad principal está etiquetada con `team=yellow`, pueden acceder al bucket de Amazon S3 *DOC-EXAMPLE-BUCKET* con el nombre *DOC-EXAMPLE-BUCKET*`-yellow`. Una política con este recurso podría permitir a los miembros del equipo acceder a sus propios recursos, pero no a los de otros equipos. Para los usuarios sin etiquetas de equipo, puede establecer un valor predeterminado de `company-wide`. Estos usuarios solo pueden acceder al bucket de *DOC-EXAMPLE-BUCKET*`-company-wide` en el que pueden ver información amplia, como instrucciones para unirse a un equipo.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Términos relacionados

- Elementos de la política de IAM: variables

Error: espacio no admitido en la variable

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported space in variable: A space is not supported within the policy variable
 text. Remove the space.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "A space is not supported within the policy variable text. Remove the
 space."
```

Resolver el error

La estructura de variables de política admite el uso de un prefijo $ seguido de un par de llaves ({ }). Dentro de los caracteres ${ }, incluya el nombre del valor de la solicitud que quiere utilizar en la política. Aunque puede incluir un espacio al especificar una variable predeterminada, no puede incluir un espacio en el nombre de la variable.

Términos relacionados

- Elementos de la política de IAM: variables

Error: variable vacía

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty variable: Empty policy variable. Remove the ${ } variable structure or provide a
 variable within the structure.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Empty policy variable. Remove the ${ } variable structure or provide
 a variable within the structure."
```

Resolver el error

La estructura de variables de política admite el uso de un prefijo $ seguido de un par de llaves ({ }). Dentro de los caracteres ${ }, incluya el nombre del valor de la solicitud que quiere utilizar en la política.

Términos relacionados

- Elementos de la política de IAM: Variables

Error: variable no admitida en el elemento

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Variable unsupported in element: Policy variables are supported in the Resource and
 Condition elements. Remove the policy variable {{variable}} from this element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Policy variables are supported in the Resource and Condition
 elements. Remove the policy variable {{variable}} from this element."
```

Resolver el error

Puede utilizar variables de política en el elemento `Resource` y en la comparación de cadenas en el elemento `Condition`.

Términos relacionados

- Elementos de la política de IAM: variables

Error — Variable no admitida en la versión

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Variable unsupported in version: To include variables in your policy, use the policy
 version 2012-10-17 or later.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "To include variables in your policy, use the policy version
 2012-10-17 or later."
```

Resolver el error

Para poder utilizar las variables de políticas, debe incluir el elemento `Version` y establecerlo en una versión que admita las variables de la política. Variables se introdujeron en la versión `2012-10-17`. Las versiones anteriores del lenguaje de políticas no son compatibles con las variables de políticas. Si no establece `Version` a `2012-10-17` o posterior, las variables como `${aws:username}` se tratan como cadenas literales en la política.

El elemento de política `Version` es diferente de la versión de una política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, se crea al cambiar una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada.

Términos relacionados

- Elementos de la política de IAM: variables
- Elementos de la política JSON de IAM: versión

Error: dirección IP privada

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Private IP address: aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "aws:SourceIp works only for public IP address ranges. The values
 for condition key aws:SourceIp include only private IP addresses and will not have the
 desired effect. Update the value to include only public IP addresses."
```

Resolver el error

La clave de condición global `aws:SourceIp` solo funciona para rangos de direcciones IP públicas. Recibe este error cuando su política solo permite direcciones IP privadas. En este caso, la condición nunca coincidiría.

- aws:SourceIp clave de condición global
- Elementos de la política de JSON de IAM: condición

Error — Dirección de notificación privada

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Private NotIpAddress: The values for condition key aws:SourceIp include only private
 IP addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The values for condition key aws:SourceIp include only private IP
 addresses and has no effect. aws:SourceIp works only for public IP address ranges.
 Update the value to include only public IP addresses."
```

Resolver el error

La clave de condición global `aws:SourceIp` solo funciona para rangos de direcciones IP públicas. Recibirá este error cuando utilice el operador de condición `NotIpAddress` y enumere solo las direcciones IP privadas. En este caso, la condición siempre coincidiría y sería ineficaz.

- aws:SourceIp clave de condición global
- Elementos de la política de JSON de IAM: condición

Error: el tamaño de la política supera la cuota de SCP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Policy size exceeds SCP quota: The {{policySize}} characters in the service control
 policy (SCP) exceed the {{policySizeQuota}} character maximum for SCPs. We recommend
 that you use multiple granular policies.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{policySize}} characters in the service control policy (SCP)
 exceed the {{policySizeQuota}} character maximum for SCPs. We recommend that you use
 multiple granular policies."
```

Resolver el error

Las políticas de control de servicios (SCP) AWS Organizations admiten la especificación de valores en elementos `Action` o `NotAction`. Sin embargo, estos valores pueden incluir comodines (*) solo al final de la cadena. Esto significa que puede especificar `iam:Get*` pero no `iam:*role`.

Para especificar varias acciones, AWS recomienda que los enumere individualmente.

Términos relacionados

- Cuotas para Organizations AWS
- Políticas de control de servicios AWS Organizations

Error: formato principal de servicio no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid service principal format: The service principal does not match the expected
 format. Use the format {{expectedFormat}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The service principal does not match the expected format. Use the
 format {{expectedFormat}}."
```

Resolver el error

El valor del par clave-valor de condición debe coincidir con un formato de entidad principal de servicio definido.

Un principal de servicio es un identificador que se utiliza para conceder permisos a un servicio. Puede especificar una entidad principal en el elemento `Principal` o como un valor de algunas claves de condición globales y claves específicas del servicio. El servicio define la entidad principal de cada servicio.

El identificador de una entidad principal de servicio incluye el nombre del servicio y suele tener el siguiente formato en letras minúsculas:

service-name.amazonaws.com

Algunas claves específicas de servicio pueden utilizar un formato diferente para las entidades de servicio. Por ejemplo, la clave de condición `kms:ViaService` requiere el siguiente formato para las entidades principales de servicio en letras minúsculas:

`service-name.AWS_region``.amazonaws.com`

Términos relacionados

- Principales del servicio
- Claves de condición global de AWS
- Clave de condición de `kms:ViaService`

Error – Falta la clave de etiqueta en la condición

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing tag key in condition: The condition key {{conditionKeyName}} must include a tag
 key to control access based on tags. Use the format {{conditionKeyName}}tag-key and
 specify a key name for tag-key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key {{conditionKeyName}} must include a tag key to
 control access based on tags. Use the format {{conditionKeyName}}tag-key and specify a
 key name for tag-key."
```

Resolver el error

Para controlar el acceso según las etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política.

Por ejemplo, para Controlar del acceso a recursos AWS, usted incluye la clave de condición de `aws:ResourceTag`. Esta clave requiere el formato `aws:ResourceTag/`*`tag-key`*. Para especificar la clave de etiqueta `owner` y el valor de la etiqueta `JaneDoe` en una condición, utilice el formato siguiente.

```
"Condition": {
    "StringEquals": {"aws:ResourceTag/owner": "JaneDoe"}
```

```
}
```

Términos relacionados

- Control del acceso mediante etiquetas
- Condiciones
- Claves de condición global
- Claves de condición de servicio de AWS

Error: formato vpc no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid vpc format: The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The VPC identifier in the condition key value is not valid. Use the
 prefix 'vpc-' followed by 8 or 17 alphanumeric characters."
```

Resolver el error

La clave de condición `aws:SourceVpc` debe utilizar el prefijo `vpc-` seguido de 8 o 17 caracteres alfanuméricos, por ejemplo, `vpc-11223344556677889` o `vpc-12345678`.

Términos relacionados

- Claves de condición global de AWS: aws:SourceVpc

Error: formato vpce no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid vpce format: The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The VPCE identifier in the condition key value is not valid.  Use
 the prefix 'vpce-' followed by 8 or 17 alphanumeric characters."
```

Resolver el error

La clave de condición `aws:SourceVpce` debe utilizar el prefijo `vpce-` seguido de 8 o 17 caracteres alfanuméricos, por ejemplo, `vpce-11223344556677889` o `vpce-12345678`.

Términos relacionados

- Claves de condición global de AWS: aws:SourceVpce

Error: no se admite la entidad principal federada

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Federated principal not supported: The policy type does not support a federated
 identity provider in the principal element. Use a supported principal.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The policy type does not support a federated identity provider in
 the principal element. Use a supported principal."
```

Resolver el error

El elemento `Principal` utiliza entidad principales federadas para políticas de confianza asociadas a roles de IAM con el fin de proporcionar acceso a través de la federación de identidades. Las políticas de identidad y otras políticas basadas en recursos no admiten un proveedor de identidad federada en el elemento `Principal`. Por ejemplo, no puede utilizar una entidad principal SAML en una política de bucket de Amazon S3. Cambie el elemento `Principal` a un tipo de entidad principal compatible.

Términos relacionados

- Creación de un rol para la federación de identidades
- Elementos de la política JSON: entidad principal

Error: acción no admitida para la clave de condición

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported action for condition key: The following actions: {{actions}} are not
 supported by the condition key {{key}}. The condition will not be evaluated for these
 actions. We recommend that you move these actions to a different statement without
 this condition key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The following actions: {{actions}} are not supported by the
 condition key {{key}}. The condition will not be evaluated for these actions. We
 recommend that you move these actions to a different statement without this condition
 key."
```

Resolver el error

Asegúrese de que la clave de condición del elemento `Condition` de la instrucción de política se aplique a todas las acciones del elemento `Action`. Para asegurarse de que la política permite o deniega efectivamente las acciones especificadas, debe mover las acciones no admitidas a otra instrucción sin la clave de condición.

> **Note**
>
> Si el elemento `Action` tiene acciones con comodines, el Analizador de acceso de IAM no evalúa esas acciones para este error.

Términos relacionados

- Elementos de la política de JSON: acción

Error: acción no admitida en la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported action in policy: The action {{action}} is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The action {{action}} is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver el error

Algunas acciones no se admiten en el elemento `Action` de la política basada en recursos asociada a otro tipo de recurso. Por ejemplo, las acciones de AWS Key Management Service no son compatibles con las políticas de bucket de Amazon S3. Especifique una acción compatible con el tipo de recurso adjunto a la política basada en recursos.

Términos relacionados

- Elementos de la política de JSON: acción

Error: ARN de recurso no admitido en la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unsupported resource ARN in policy: The resource ARN is not supported for the resource-
based policy attached to the resource type {{resourceType}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The resource ARN is not supported for the resource-based policy
 attached to the resource type {{resourceType}}."
```

Resolver el error

Algunos ARN de recursos no se admiten en el elemento `Resource` de la política basada en recursos cuando la política se asocia a otro tipo de recurso. Por ejemplo, los ARN de AWS KMS no son compatibles en el elemento `Resource` con las políticas de bucket de Amazon S3. Especifique un ARN de recurso compatible con un tipo de recurso asociado a la política basada en recursos.

Términos relacionados

- Elementos de la política de JSON: acción

Error: clave de condición no admitida para la entidad principal de servicio

En el AWS Management Console, el resultado de esta comprobación incluye el siguiente mensaje:

```
Unsupported condition key for service principal: The following condition keys are not
 supported when used with the service principal: {{conditionKeys}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The following condition keys are not supported when used with the
 service principal: {{conditionKeys}}."
```

Resolver el error

Puede especificar Servicios de AWS en el elemento `Principal` de una política basada en recursos mediante una entidad principal de servicio, que es un identificador del servicio. No puede utilizar algunas claves de condición con ciertas entidades principales de servicio. Por ejemplo, no puede utilizar la clave de condición `aws:PrincipalOrgID` con la entidad principal de servicio `cloudfront.amazonaws.com`. Debe eliminar las claves de condición que no se aplican a la entidad principal de servicio en el elemento `Principal`.

Términos relacionados

- Entidades principales del servicio
- Elementos de la política JSON: entidad principal

Error: error de sintaxis de la política de confianza del rol notprincipal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Role trust policy syntax error notprincipal: Role trust policies do not support
 NotPrincipal. Update the policy to use a Principal element instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Role trust policies do not support NotPrincipal. Update the policy
 to use a Principal element instead."
```

Resolver el error

Una política de confianza de rol es una política basada en recursos que se adjunta a un rol de IAM. Las políticas de confianza definen qué entidades principales (cuentas, usuarios, roles y usuarios federados) puede asumir el rol. Las políticas de confianza de rol no admite `NotPrincipal`. Actualice la política para utilizar un elemento `Principal` en su lugar.

Términos relacionados

- Elementos de la política JSON: entidad principal
- Elementos de la política de JSON: NotPrincipal

Error: la política de confianza del rol no admite comodines en la entidad principal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Role trust policy unsupported wildcard in principal: "Principal:" "*" is not supported
 in the principal element of a role trust policy. Replace the wildcard with a valid
 principal value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": ""Principal:" "*" is not supported in the principal element of a role
 trust policy. Replace the wildcard with a valid principal value."
```

Resolver el error

Una política de confianza de rol es una política basada en recursos que se adjunta a un rol de IAM. Las políticas de confianza definen qué entidades principales (cuentas, usuarios, roles y usuarios federados) pueden asumir el rol. `"Principal:" "*"` no se admite en el elemento `Principal` de una política de confianza de rol. Sustituir el comodín por un valor válido de entidad principal.

Términos relacionados

- Elementos de la política JSON: entidad principal

Error: recurso de error de sintaxis de la política de confianza

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Role trust policy syntax error resource: Role trust policies apply to the role
 that they are attached to. You cannot specify a resource. Remove the Resource or
 NotResource element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Role trust policies apply to the role that they are attached to. You
 cannot specify a resource. Remove the Resource or NotResource element."
```

Resolver el error

Una política de confianza de rol es una política basada en recursos que se adjunta a un rol de IAM. Las políticas de confianza definen qué entidades principales (cuentas, usuarios, roles y usuarios federados) puede asumir el rol. Las políticas de confianza de los roles se aplican al rol al que están vinculadas. No se puede especificar un `Resource` o un elemento `NotResource` en una política de confianza de rol. Elimine el `Resource` o el elemento `NotResource`.

- Elementos de la política de JSON: Recurso
- Elementos de la política de JSON: NotResource

Error: discrepancia de tipo de rango de IP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch IP range: The condition operator {{operator}} is used with an invalid IP
 range value. Specify the IP range in standard CIDR format.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition operator {{operator}} is used with an invalid IP range
 value. Specify the IP range in standard CIDR format."
```

Resolver el error

Actualice el texto para utilizar el tipo de datos del operador de condición de dirección IP, en un formato CIDR.

Términos relacionados

- Operadores de condición de dirección IP
- Elementos de la política de JSON de IAM: operadores de condición

Error: falta acción para la clave de condición

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing action for condition key: The {{actionName}} action must be in the action block
 to allow setting values for the condition key {{keyName}}. Add {{actionName}} to the
 action block.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{actionName}} action must be in the action block to allow
 setting values for the condition key {{keyName}}. Add {{actionName}} to the action
 block."
```

Resolver el error

La clave de condición del elemento `Condition` de la declaración de política no se evalúa a menos que la acción especificada esté en elemento `Action`. Para garantizar que las claves de condición que especifique sean efectivamente permitidas o denegadas por su política, añada la acción al elemento `Action`.

Términos relacionados

- Elementos de la política de JSON: acción

Error: sintaxis de la entidad principal federada no válida en la política de confianza del rol

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid federated principal syntax in role trust policy: The principal value specifies
 a federated principal that does not match the expected format. Update the federated
 principal to a domain name or a SAML metadata ARN.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The principal value specifies a federated principal that does not
 match the expected format. Update the federated principal to a domain name or a SAML
 metadata ARN."
```

Resolver el error

El valor de la entidad principal especifica una entidad principal federada que no coincide con el formato esperado. Actualice el formato de la entidad principal federada a un nombre de dominio válido o a un ARN de metadatos SAML.

Términos relacionados

- Usuarios federados y roles

Error: acción incorrecta para la entidad principal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Mismatched action for principal: The {{actionName}} action is invalid with the
 following principal(s): {{principalNames}}. Use a SAML provider principal with
 the sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{actionName}} action is invalid with the following
 principal(s): {{principalNames}}. Use a SAML provider principal with the
 sts:AssumeRoleWithSAML action or use an OIDC provider principal with the
 sts:AssumeRoleWithWebIdentity action. Ensure the provider is Federated if you use
 either of the two options."
```

Resolver el error

La acción especificada en el elemento `Action` de la declaración de la política no es válida para la entidad principal especificada en el elemento `Principal`. Por

ejemplo, no se puede utilizar una entidad principal del proveedor SAML para la acción `sts:AssumeRoleWithWebIdentity`. Debe utilizar una entidad principal de proveedor SAML con la acción `sts:AssumeRoleWithSAML` o utilizar una entidad principal de proveedor OIDC con la acción `sts:AssumeRoleWithWebIdentity`.

Términos relacionados

- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity

Error: falta una acción para la política de confianza de funciones en cualquier lugar

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing action for roles anywhere trust policy: The rolesanywhere.amazonaws.com service
 principal requires the sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession
 permissions to assume a role. Add the missing permissions to the policy.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The rolesanywhere.amazonaws.com service principal requires the
 sts:AssumeRole, sts:SetSourceIdentity, and sts:TagSession permissions to assume a
 role. Add the missing permissions to the policy."
```

Resolver el error

Para que Funciones de IAM en cualquier lugar pueda asumir un rol y suministrar credenciales AWS temporales, el rol debe confiar en la entidad principal del servicio Funciones de IAM en cualquier lugar. La entidad principal del servicio Funciones de IAM en cualquier lugar requiere el `sts:AssumeRole`, `sts:SetSourceIdentity`, y los permisos `sts:TagSession` para asumir un rol. Si falta alguno de los permisos, debe añadirlo a la política.

Términos relacionados

- Modelo de confianza en Funciones en cualquier lugar de AWS Identity and Access Management

Advertencia general: crear SLR con NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with NotResource: Using the iam:CreateServiceLinkedRole action with
 NotResource can allow creation of unintended service-linked roles for multiple
 resources. We recommend that you specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with NotResource can
 allow creation of unintended service-linked roles for multiple resources. We recommend
 that you specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Al utilizar `iam:CreateServiceLinkedRole` en una política con el elemento `NotResource` se puede permitir la creación de roles vinculados a servicios no deseados para varios recursos. AWS recomienda, en su lugar, especificar ARN permitidos en el elemento `Resource`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM recurso

Advertencia general — Crear SLR con estrella en acción y NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with star in action and NotResource: Using an action with a wildcard(*) and
 NotResource can allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using an action with a wildcard(*) and NotResource can
 allow creation of unintended service-linked roles because it can allow
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Las políticas con un carácter comodín (*) en el `Action` y que incluyen el elemento `NotResource` pueden permitir la creación de roles vinculados a servicios no deseados para varios recursos. AWS recomienda, en su lugar, especificar ARN permitidos en el elemento `Resource`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM;: recurso

Advertencia general: crear SLR con NotAction y NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with NotAction and NotResource: Using NotAction with NotResource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on multiple resources. We recommend that you
 specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using NotAction with NotResource can allow creation of unintended
 service-linked roles because it allows iam:CreateServiceLinkedRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Al utilizar el elemento `NotAction` con el elemento `NotResource` se puede permitir la creación de roles vinculados a servicios no deseados para varios recursos. AWS recomienda, en su lugar, que vuelva a escribir la política para permitir `iam:CreateServiceLinkedRole` en una lista limitada de ARN en el elemento `Resource`. También puede agregar `iam:CreateServiceLinkedRole` al elemento `NotAction`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM: NotAction

- Elementos de la política de JSON de IAM: acción
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM;: recurso

Advertencia general: crea SLR con estrella en el recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with star in resource: Using the iam:CreateServiceLinkedRole action with
 wildcards (*) in the resource can allow creation of unintended service-linked roles.
 We recommend that you specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the iam:CreateServiceLinkedRole action with wildcards (*) in
 the resource can allow creation of unintended service-linked roles. We recommend that
 you specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Al utilizar `iam:CreateServiceLinkedRole` en una política con un comodín (*) en el elemento `Resource` se puede permitir la creación de roles vinculados a servicios no deseados para varios recursos. AWS recomienda, en su lugar, especificar ARN permitidos en el elemento `Resource`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM;: recurso

AWS políticas administradas con esta advertencia general

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Algunos de esos casos de uso son para usuarios avanzados de su cuenta. Las siguientes políticas administradas de AWS proporcionan acceso de usuarios avanzados y otorgan permisos para crear roles vinculados al servicio para cualquier servicio de AWS. AWS recomienda que adjunte

las siguientes políticas administradas de AWS a las identidades de IAM que considere usuarios avanzados.

- PowerUserAccess
- AlexaForBusinessFullAccess
- AWSOrganizationsServiceTrustPolicy— Esta política administrada de AWS proporciona permisos para su uso por parte del rol vinculado al servicio de AWS Organizations. Esta función permite a las Organizations a crear roles vinculados a servicios adicionales para otros servicios en su organización AWS.

Advertencia general: cree SLR con estrella en acción y recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with star in action and resource: Using wildcards (*) in the action and
 the resource can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using wildcards (*) in the action and the resource
 can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Las políticas con un carácter comodín (*) en los elementos `Action` y `Resource` pueden permitir la creación de roles vinculados a servicios no deseados para varios recursos. Esto permite crear un rol vinculado al servicio cuando especifica `"Action": "*"`, `"Action": "iam:*"`, o `"Action": "iam:Create*"`. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM: acción

- Elementos de la política de JSON de IAM;: recurso

AWS políticas administradas con esta advertencia general

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Algunos de esos casos de uso son para administradores de su cuenta. Las siguientes políticas administradas de AWS proporcionan acceso de administrador y otorgan permisos para crear roles vinculados al servicio para cualquier servicio de AWS. AWS recomienda que adjunte las siguientes políticas administradas de AWS a las identidades de IAM que considere administradores.

- AdministratorAccess
- IAMFullAccess

Advertencia general — Crear SLR con estrella en el recurso y NotAction

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Create SLR with star in resource and NotAction: Using a resource with wildcards (*)
 and NotAction can allow creation of unintended service-linked roles because it allows
 iam:CreateServiceLinkedRole permissions on all resources. We recommend that you
 specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can allow creation
 of unintended service-linked roles because it allows iam:CreateServiceLinkedRole
 permissions on all resources. We recommend that you specify resource ARNs instead."
```

Resolver la advertencia general

La acción `iam:CreateServiceLinkedRole` otorga permiso para crear un rol de IAM que permite a un servicio de AWS realizar acciones en su nombre. Al utilizar el elemento `NotAction` en una política con un carácter comodín (*) en el elemento `Resource` puede permitir la creación de roles vinculados a servicios no deseados para varios recursos. En su lugar, AWS recomienda que usted especifique los ARN permitidos en el elemento `Resource`. También puede agregar `iam:CreateServiceLinkedRole` al elemento `NotAction`.

- Operación CreateServiceLinkedRole
- Elementos de la política de JSON de IAM: NotAction
- Elementos de la política de JSON de IAM: acción
- Elementos de la política de JSON de IAM;: recurso

Advertencia general: clave de condición global obsoleta

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Deprecated global condition key: We recommend that you update aws:ARN to use the newer
 condition key aws:PrincipalArn.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "We recommend that you update aws:ARN to use the newer condition key
 aws:PrincipalArn."
```

Resolver la advertencia general

La política incluye una clave de condición global obsoleta. Actualice la clave de condición en el par clave-valor de condición para utilizar una clave de condición global compatible.

- Claves de condición global

Advertencia general: valor de fecha no válido

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid date value: The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The date {{date}} might not resolve as expected. We recommend that
 you use the YYYY-MM-DD format."
```

Resolver la advertencia general

El tiempo de Unix Epoch describe un punto en el tiempo que ha transcurrido desde el 1 de enero de 1970, menos segundos bisiestos. Es posible que el tiempo de la fecha de inicio no se resuelva a la hora exacta que espera. AWS recomienda utilizar el estándar W3C para formatos de fecha y hora. Por ejemplo, podría especificar una fecha completa, como AAAA-MM-DD (1997-07-16), o también podría anexar la hora al segundo, como AAAA-MM-DDThh:mm:ssTZD (1997-07-16T19:20:30+01:00).

- Formatos de fecha y hora del W3C
- Elementos de la política JSON de IAM: versión
- aws:CurrentTime global condition key

Advertencia general: referencia de rol no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid role reference: The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Principal element includes the IAM role ID {{roleid}}. We
 recommend that you use a role ARN instead."
```

Resolver la advertencia general

AWS le recomienda que especifique el Nombre de recurso de Amazon (ARN) para un rol de IAM en lugar de su ID principal. Cuando IAM guarda la política, transformará el ARN en el ID principal del rol existente. AWS incluye una precaución de seguridad. Si alguien elimina y vuelve a crear el rol, tendrá un nuevo ID y la política no coincidirá con el ID del nuevo rol.

- Especificación de una entidad principal: roles de IAM
- ARN de IAM
- ID únicos de IAM

Advertencia general: referencia de usuario no válida

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Invalid user reference: The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The Principal element includes the IAM user ID {{userid}}. We
 recommend that you use a user ARN instead."
```

Resolver la advertencia general

AWS le recomienda que especifique el Nombre de recurso de Amazon (ARN) para un usuario de IAM en lugar de su ID principal. Cuando IAM guarda la política, transformará el ARN en el ID principal del usuario existente. AWS incluye una precaución de seguridad. Si alguien elimina y vuelve a crear el usuario, tendrá un nuevo ID y la política no coincidirá con el ID del nuevo usuario.

- Especificación de una entidad principal: usuarios de IAM
- ARN de IAM
- ID únicos de IAM

Advertencia general: falta la versión

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing version: We recommend that you specify the Version element to help you with
 debugging permission issues.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "We recommend that you specify the Version element to help you with
 debugging permission issues."
```

Resolver la advertencia general

AWS le recomienda que incluya el parámetro `Version` opcional en su política. Si no incluye un elemento Version, el sistema toma de forma predeterminada el valor `2012-10-17`, pero las

características más recientes, como, por ejemplo, variables de política, no funcionarán con su política. Por ejemplo, las variables del tipo `${aws:username}` no se reconocerán como variables y se tratarán en la política como si fueran cadenas literales.

- Elementos de la política JSON de IAM: versión

Advertencia general: se recomiendan Sids únicos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Unique Sids recommended: We recommend that you use statement IDs that are unique to
 your policy. Update the Sid value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "We recommend that you use statement IDs that are unique to your
 policy. Update the Sid value."
```

Resolver la advertencia general

AWS recomienda que utilice ID de instrucciones únicos. El elemento `Sid` (ID de instrucción) le permite ingresar un identificador opcional que se proporciona para la instrucción de la política. Puede asignar un valor de ID de instrucción a cada instrucción de una matriz de instrucciones utilizando el elemento `SID`.

Términos relacionados

- Elemento de la política de JSON de IAM: Sid

Advertencia general: comodín sin operador

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Wildcard without like operator: Your condition value includes a * or ? character. If
 you meant to use a wildcard (*, ?), update the condition operator to include Like.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Your condition value includes a * or ? character. If you meant to
 use a wildcard (*, ?), update the condition operator to include Like."
```

Resolver la advertencia general

La estructura del elemento `Condition` requiere que utilice un operador de condición y un par clave-valor. Cuando se especifica un valor de condición que utiliza un comodín (*, ?), debe utilizar la versión `Like` del operador de condición. Por ejemplo, en lugar del operador de condición de cadena `StringEquals`, debe utilizar `StringLike`.

```
"Condition": {"StringLike": {"aws:PrincipalTag/job-category": "admin-*"}}
```

- Elementos de la política de JSON de IAM: operadores de condición
- Elementos de la política de JSON de IAM: condición

AWS políticas administradas con esta advertencia general

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Las siguientes políticas administradas AWS incluyen comodines en su valor de condición sin un operador de condición que incluya `Like` para la coincidencia de patrones. Cuando se utiliza la política administrada AWS como referencia para crear su política administrada por el cliente, AWS recomienda que utilice un operador de condición que admita la coincidencia de patrones con comodines (*, ?), como por ejemplo, `StringLike`.

- AWSGlueConsoleSageMakerNotebookFullAccess

Advertencia general: el tamaño de la política supera la cuota de identidad de la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Policy size exceeds identity policy quota: The {{policySize}} characters in the
 identity policy, excluding whitespace, exceed the {{policySizeQuota}} character
 maximum for inline and managed policies. We recommend that you use multiple granular
 policies.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{policySize}} characters in the identity policy, excluding
 whitespace, exceed the {{policySizeQuota}} character maximum for inline and managed
 policies. We recommend that you use multiple granular policies."
```

Resolver la advertencia general

Puede adjuntar hasta 10 políticas administradas a una identidad de IAM (usuario, grupo de usuarios o rol). Sin embargo, cada política administrada no puede exceder la cuota predeterminada de 6144 caracteres. IAM no cuenta los espacios en blanco al calcular el tamaño de una política frente a esta limitación. Las cuotas, también conocidas como límites en AWS, son el valor máximo de los recursos, acciones y elementos de su cuenta de AWS.

Además, puede agregar tantas políticas insertadas como quiera a una identidad de IAM. Sin embargo, el tamaño de todas las políticas insertadas por identidad no puede superar la cuota especificada.

Si la política es mayor que la cuota, puede organizar la política en varias instrucciones y agruparlas en varias políticas.

Términos relacionados

- IAM y cuotas de caracteres AWS STS
- Varias instrucciones y varias políticas
- Políticas administradas por el cliente de IAM
- Información general de políticas de JSON
- Gramática de políticas JSON de IAM

AWS políticas administradas con esta advertencia general

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Los siguientes ejemplos de las políticas administradas de AWS otorgan permisos a acciones en muchos servicios AWS y exceden el tamaño máximo de la política. Cuando se utiliza la política administrada de AWS como referencia para crear la política administrada, debe dividir la política en varias políticas.

- ReadOnlyAccess
- AWSSupportServiceRolePolicy

Advertencia general: el tamaño de la política supera la cuota de la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Policy size exceeds resource policy quota: The {{policySize}} characters in the
 resource policy exceed the {{policySizeQuota}} character maximum for resource
 policies. We recommend that you use multiple granular policies.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{policySize}} characters in the resource policy exceed the
 {{policySizeQuota}} character maximum for resource policies. We recommend that you use
 multiple granular policies."
```

Resolver la advertencia general

Las políticas basadas en recursos son documentos de política JSON que puede asociar a un recurso, como, por ejemplo, un bucket de Amazon S3. Estas políticas conceden a la entidad principal especificada permiso para ejecutar acciones concretas en el recurso y definen en qué condiciones son aplicables. El tamaño de las políticas basadas en recursos no puede superar la cuota establecida para ese recurso. Las cuotas, también conocidas como límites en AWS, son el valor máximo de los recursos, acciones y elementos de su cuenta de AWS.

Si la política es mayor que la cuota, puede organizar la política en varias instrucciones y agruparlas en varias políticas.

Términos relacionados

- Políticas basadas en recursos
- Políticas de buckets de Amazon S3
- Varias instrucciones y varias políticas
- Información general de políticas de JSON
- Gramática de políticas JSON de IAM

Advertencia general: no coinciden los tipos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch: Use the operator type {{allowed}} instead of operator {{operator}} for
 the condition key {{key}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Use the operator type {{allowed}} instead of operator {{operator}}
 for the condition key {{key}}."
```

Resolver la advertencia general

Actualice el texto para utilizar el tipo de datos del operador de condición admitido.

Por ejemplo, la clave de condición global `aws:MultiFactorAuthPresent` requiere un operador de condición con el tipo de datos `Boolean`. Si proporciona una fecha o un entero, el tipo de datos no coincidirá.

Términos relacionados

- Claves de condición global
- Elementos de la política de JSON de IAM: operadores de condición

Advertencia general: discrepancia de tipo y booleano

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch Boolean: Add a valid Boolean value (true or false) for the condition
 operator {{operator}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a valid Boolean value (true or false) for the condition operator
 {{operator}}."
```

Resolver la advertencia general

Actualice el texto para utilizar un tipo de datos de operador de condición booleana, como `true` o `false`.

Por ejemplo, la clave de condición global `aws:MultiFactorAuthPresent` requiere un operador de condición con el tipo de datos `Boolean`. Si proporciona una fecha o un entero, el tipo de datos no coincidirá.

Términos relacionados

- Operadores de condición booleanos
- Elementos de la política de JSON de IAM: operadores de condición

Advertencia general: discrepancia de tipo y fecha

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch date: The date condition operator is used with an invalid value. Specify
 a valid date using YYYY-MM-DD or other ISO 8601 date/time format.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The date condition operator is used with an invalid value. Specify a
 valid date using YYYY-MM-DD or other ISO 8601 date/time format."
```

Resolver la advertencia general

Actualice el texto para utilizar el tipo de datos del operador de condición de fecha, en un `YYYY-MM-DD` u otro formato de fecha y hora ISO 8601.

Términos relacionados

- Operadores de condición de fecha
- Elementos de la política de JSON de IAM: operadores de condición

Advertencia general: discrepancia de tipo y número

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch number: Add a valid numeric value for the condition operator
 {{operator}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a valid numeric value for the condition operator {{operator}}."
```

Resolver la advertencia general

Actualice el texto para utilizar el tipo de datos del operador de condición numérica.

Términos relacionados

- Operadores de condición numérica
- Elementos de la política de JSON de IAM: operadores de condición

Advertencia general: discrepancia de tipo y cadena

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Type mismatch string: Add a valid base64-encoded string value for the condition
 operator {{operator}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a valid base64-encoded string value for the condition operator
 {{operator}}."
```

Resolver la advertencia general

Actualice el texto para utilizar el tipo de datos del operador de condición de cadena.

Términos relacionados

- Operadores de condición de cadena
- Elementos de la política de JSON de IAM: operadores de condición

Advertencia general: Se recomienda un repositorio y una ramificación de github específicos

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Specific github repo and branch recommended: Using a wildcard (*) in
 token.actions.githubusercontent.com:sub can allow requests from more sources than
 you intended. Specify the value of token.actions.githubusercontent.com:sub with the
 repository and branch name.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using a wildcard (*) in token.actions.githubusercontent.com:sub
 can allow requests from more sources than you intended. Specify the value of
 token.actions.githubusercontent.com:sub with the repository and branch name."
```

Resolver la advertencia general

Si usa GitHub como un IdP de OIDC, la práctica recomendada es limitar las entidades que pueden asumir el rol asociado con el IdP de IAM. Al incluir una declaración `Condition` en una política de confianza de rol, puede limitar el rol a una organización, repositorio o ramificación específica de GitHub. Puede utilizar la clave de condición `token.actions.githubusercontent.com:sub` para limitar el acceso. Le recomendamos que limite la condición a un conjunto específico de repositorios o ramas. Si utiliza un comodín (`*`) en `token.actions.githubusercontent.com:sub`, las Acciones de GitHub de organizaciones o repositorios ajenos a su control podrán asumir los roles asociados al IdP de IAM de GitHub en su cuenta de AWS.

Términos relacionados

- Configuración de un rol para el proveedor de identidades de OIDC de GitHub

Advertencia general: el tamaño de la política supera la cuota de la política de confianza del rol

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Policy size exceeds role trust policy quota: The characters in the role trust policy,
 excluding whitespace, exceed the character maximum. We recommend that you request a
 role trust policy length quota increase using Service Quotas and AWS Support Center.
 If the quotas have already been increased, then you can ignore this warning.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The characters in the role trust policy, excluding whitespace,
 exceed the character maximum. We recommend that you request a role trust policy length
 quota increase using Service Quotas and AWS Support Center. If the quotas have already
 been increased, then you can ignore this warning."
```

Resolver la advertencia general

IAM y AWS STS tienen cuotas que limitan el tamaño de las políticas de confianza de rol. Los caracteres de la política de confianza del rol, excluyendo los espacios en blanco, exceden el máximo de caracteres. Le recomendamos que solicite un aumento de la cuota de longitud de la política de confianza mediante Service Quotas y AWS Support Center Console.

Términos relacionados

- IAM y cuotas de AWS STS, requisitos de nombre y límites de caracteres

Advertencia de seguridad: Permitir con NotPrincipal

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Allow with NotPrincipal: Using Allow with NotPrincipal can be overly permissive. We
 recommend that you use Principal instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using Allow with NotPrincipal can be overly permissive. We recommend
 that you use Principal instead."
```

Resolución de la advertencia de seguridad

El uso de `"Effect": "Allow"` con el `NotPrincipal` puede ser demasiado permisivo. Por ejemplo, esto puede conceder permisos a entidades principales anónimas. AWS recomienda especificar entidades principales a las que se necesita acceso mediante los elementos `Principal`. Alternativamente, puede permitir un acceso amplio y, a continuación, agregar otra instrucción que utilice el elemento `NotPrincipal` con `“Effect”: “Deny”`.

- Elementos de la política JSON de AWS: entidad principal

- AWS Elementos de la política de JSON: NotPrincipal

Advertencia de seguridad: ForAllValues con clave de valor único

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
ForAllValues with single valued key: Using ForAllValues qualifier with the single-
valued condition key {{key}} can be overly permissive. We recommend that you remove
 ForAllValues:.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using ForAllValues qualifier with the single-valued condition key
 {{key}} can be overly permissive. We recommend that you remove ForAllValues:."
```

Resolución de la advertencia de seguridad

AWS recomienda que utilice la opción `ForAllValues` solo con condiciones multivalor. El operador de configuración `ForAllValues` prueba si el valor de cada miembro del conjunto de solicitudes es un subconjunto del conjunto de claves de condición. La condición devuelve true si cada valor de clave de la solicitud coincide con al menos un valor de la política. También devuelve true si no hay claves en la solicitud o si los valores de clave se resuelven en un conjunto de datos nulo, como una cadena vacía.

Para saber si una condición admite un único valor o varios valores, revise la página de Acciones, recursos y claves de condición del servicio. Las claves de condición con el prefijo de tipo de datos `ArrayOf` son claves de condición multivalor. Por ejemplo, Amazon SES admite claves con valores únicos (`String`) y el tipo de datos multivalor `ArrayOfString`.

- Claves de contexto multivalor

Advertencia de seguridad: rol de pase con NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with NotResource: Using the iam:PassRole action with NotResource can be
 overly permissive because it can allow iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the iam:PassRole action with NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). Uso de `iam:PassRole` en una política con el elemento `NotResource` puede permitir que sus entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM;: recurso

Advertencia de seguridad: rol de pase con estrella en acción y NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with star in action and NotResource: Using an action with a wildcard (*) and
 NotResource can be overly permissive because it can allow iam:PassRole permissions on
 multiple resources. We recommend that you specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using an action with a wildcard (*) and NotResource can be overly
 permissive because it can allow iam:PassRole permissions on multiple resources. We
 recommend that you specify resource ARNs instead."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). Las políticas con un carácter comodín (*) en el campo `Action` y que incluyen el elemento `NotResource` pueden permitir que sus entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM;: recurso

Advertencia de seguridad: rol de pase con NotAction y NotResource

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with NotAction and NotResource: Using NotAction with NotResource can
 be overly permissive because it can allow iam:PassRole permissions on multiple
 resources.. We recommend that you specify resource ARNs instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using NotAction with NotResource can be overly permissive because
 it can allow iam:PassRole permissions on multiple resources.. We recommend that you
 specify resource ARNs instead."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). El uso del elemento `NotAction` y listar algunos recursos en el elemento `NotResource` puede permitir que sus entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM: NotAction
- Elementos de la política de JSON de IAM: acción
- Elementos de la política de JSON de IAM: NotResource
- Elementos de la política de JSON de IAM;: recurso

Advertencia de seguridad: rol de pase con estrella en el recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with star in resource: Using the iam:PassRole action with wildcards (*)
 in the resource can be overly permissive because it allows iam:PassRole permissions
 on multiple resources. We recommend that you specify resource ARNs or add the
 iam:PassedToService condition key to your statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the iam:PassRole action with wildcards (*) in the resource can
 be overly permissive because it allows iam:PassRole permissions on multiple resources.
 We recommend that you specify resource ARNs or add the iam:PassedToService condition
 key to your statement."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). Las políticas que permiten `iam:PassRole` y que incluyen un carácter comodín (*) en el elemento `Resource` pueden permitir que sus entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

Algunos servicios AWS incluyen su espacio de nombres de servicio en el nombre de su rol. Esta verificación de políticas tiene en cuenta estas convenciones al analizar la política para generar resultados. Por ejemplo, es posible que el siguiente ARN de recurso no genere un resultado:

```
arn:aws:iam::*:role/Service*
```

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM;: recurso

Políticas administradas AWS con esta advertencia de seguridad

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Uno de esos casos de uso es para administradores de su cuenta. Las siguientes políticas administradas de AWS proporcionan acceso de administrador y otorgan permisos para pasar cualquier rol de IAM a cualquier servicio. AWS recomienda que adjunte las siguientes políticas administradas de AWS a las identidades de IAM que considere administradores.

- AdministratorAccess-Amplify

Los siguientes ejemplos de políticas administradas AWS incluyen permisos para `iam:PassRole` con un carácter comodín (*) en el recurso y se encuentran en una ruta de obsolescencia. Para cada una de estas políticas, actualizamos la guía de permisos, como recomendar una nueva política administrada AWS que admite el caso de uso. Para ver las alternativas a estas políticas, consulte las guías de cada servicio.

- AmazonWebServiceElasticBeanstalkFullAccess
- AWSElasticBeanstalkService
- AWSLambdaFullAccess
- AWSLambdaReadOnlyAccess
- AWSOpsWorksFullAccess
- AWSOpsWorksRole
- AWSDataPipelineRole
- AmazonDynamoDBFullAccesswithDataPipeline
- AmazonElasticMapReduceFullAccess
- AmazonDynamoDBFullAccesswithDataPipeline

- AmazonEC2ContainerServiceFullAccess

Los siguientes ejemplos de políticas administradas AWS proporcionan permisos solo para roles vinculados al servicio, que permiten servicios AWS para realizar acciones en su nombre. No puede adjuntar estas políticas a sus identidades de IAM.

- AWSServiceRoleForAmazonEKSNodegroup

Advertencia de seguridad: rol de pase con estrella en la acción y el recurso

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with star in action and resource: Using wildcards (*) in the action and the
 resource can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using wildcards (*) in the action and the resource can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). Las políticas con un carácter comodín (*) en `Action` y los elementos `Resource` pueden permitir que las entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento `Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM: acción

- Elementos de la política de JSON de IAM;: recurso

Políticas administradas AWS con esta advertencia de seguridad

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Algunos de esos casos de uso son para administradores de su cuenta. Las siguientes políticas administradas de AWS proporcionan acceso de administrador y otorgan permisos para pasar cualquier rol de IAM a cualquier servicio AWS. AWS recomienda que adjunte las siguientes políticas administradas de AWS a las identidades de IAM que considere administradores.

- AdministratorAccess
- IAMFullAccess

Advertencia de seguridad: rol de pase con estrella en recurso y NotAction

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Pass role with star in resource and NotAction: Using a resource with wildcards (*) and
 NotAction can be overly permissive because it allows iam:PassRole permissions on all
 resources. We recommend that you specify resource ARNs or add the iam:PassedToService
 condition key to your statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using a resource with wildcards (*) and NotAction can be overly
 permissive because it allows iam:PassRole permissions on all resources. We recommend
 that you specify resource ARNs or add the iam:PassedToService condition key to your
 statement."
```

Resolución de la advertencia de seguridad

Para configurar muchos servicios de AWS, es necesario transferir un rol de IAM al servicio. Para permitir esto, debe conceder el permiso `iam:PassRole` a una identidad (usuario, grupo de usuarios o rol). El uso del elemento `NotAction` en una política con un carácter comodín (*) en el elemento `Resource` pueden permitir que sus entidades principales accedan a más servicios o características de los que pretendía. En su lugar, AWS recomienda especificar ARN permitidos en el elemento

`Resource`. Además, puede reducir los permisos a un único servicio mediante la clave de condición `iam:PassedToService`.

- Pasar un rol a un servicio
- iam:PassedToService
- Elementos de la política de JSON de IAM: NotAction
- Elementos de la política de JSON de IAM: acción
- Elementos de la política de JSON de IAM;: recurso

Advertencia de seguridad: faltan las claves de condición emparejadas

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing paired condition keys: Using the condition key {{conditionKeyName}}
 can be overly permissive without also using the following condition keys:
 {{recommendedKeys}}. Condition keys like this one are more secure when paired with
 a related key. We recommend that you add the related condition keys to the same
 condition block.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the condition key {{conditionKeyName}} can be overly
 permissive without also using the following condition keys: {{recommendedKeys}}.
 Condition keys like this one are more secure when paired with a related key. We
 recommend that you add the related condition keys to the same condition block."
```

Resolución de la advertencia de seguridad

Algunas claves de condición son más seguras cuando se las empareja con otras claves de condición relacionadas. AWS recomienda incluir las claves de condición relacionadas en el mismo bloque de condición que la clave de condición existente. Esto hace que los permisos otorgados a través de la política sean más seguros.

Por ejemplo, puede utilizar la clave de condición `aws:VpcSourceIp` para comparar la dirección IP desde la que se realizó una solicitud con la dirección IP que ha especificado en la política. AWS recomienda que agregue la clave de condición de `aws:SourceVPC`. Esto comprueba si la solicitud proviene de la VPC que especifique en la política y la dirección IP que especifique.

Términos relacionados

- Clave de condición global `aws:VpcSourceIp`
- Clave de condición global `aws:SourceVPC`
- Claves de condición global
- Elemento de condición
- Información general de políticas de JSON

Advertencia de seguridad: denegar con clave de condición de etiqueta no admitida para el servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Deny with unsupported tag condition key for service: Using the effect Deny with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes can be overly permissive: {{serviceNames}}. Actions for the listed services
 are not denied by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the effect Deny with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes can be
 overly permissive: {{serviceNames}}. Actions for the listed services are not denied
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Resolución de la advertencia de seguridad

El uso de claves de condición de etiqueta no compatibles en el elemento `Condition` de una política con `"Effect": "Deny"` puede ser excesivamente permisiva, porque la condición se ignora para ese servicio. AWSrecomienda quitar las acciones de servicio que no admiten la clave de condición y crear otra instrucción para denegar el acceso a recursos específicos para esas acciones.

Si utiliza la clave de condición `aws:ResourceTag` y no es compatible con una acción de servicio, entonces la clave no se incluye en el contexto de solicitud. En este caso, la condición en el campo `Deny` siempre devuelve `false` y la acción nunca se deniega. Esto sucede incluso si el recurso está etiquetado correctamente.

Cuando un servicio admite la clave de condición `aws:ResourceTag` puede utilizar etiquetas para controlar el acceso a los recursos de dicho servicio. Esto se conoce como control de acceso basado en atributos (ABAC). Los servicios que no admiten estas claves requieren que el usuario controle el acceso a los recursos mediante Control de acceso basado en recursos (RBAC).

> **Note**
>
> Algunos servicios permiten el soporte para la clave de condición `aws:ResourceTag` para un subconjunto de sus recursos y acciones. El Analizador de acceso de IAM devuelve los resultados de las acciones de servicio que no se admiten. Por ejemplo, Amazon S3 admite `aws:ResourceTag` para un subconjunto de sus recursos. Para consultar todos los tipos de recursos disponibles en Amazon S3 que admiten la clave de condición `aws:ResourceTag`, consulte Tipos de recurso definidos por Amazon S3 en la Referencia de autorizaciones de servicio.

Por ejemplo, supongamos que desea denegar el acceso para desetiquetar recursos específicos que están etiquetados con el par de clave-valor `status=Confidential`. También supongamos también que AWS Lambda le permite etiquetar y desetiquetar recursos, pero no admite la clave de condición `aws:ResourceTag`. Para denegar las acciones de eliminación para AWS App Mesh y AWS Backup si esta etiqueta está presente, utilice la clave de condición `aws:ResourceTag`. Para Lambda, utilice una convención de nomenclatura de recursos que incluya el prefijo `"Confidential"`. A continuación, incluya una instrucción separada que impida eliminar recursos con esa convención de nomenclatura.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyDeleteSupported",
            "Effect": "Deny",
            "Action": [
                "appmesh:DeleteMesh",
                "backup:DeleteBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/status": "Confidential"
                }
```

```
            }
        },
        {
            "Sid": "DenyDeleteUnsupported",
            "Effect": "Deny",
            "Action": "lambda:DeleteFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:status-Confidential*"
        }
    ]
}
```

> **Warning**
>
> No utilice la versión ...IfExists del operador de condición como solución alternativa para este resultado. Esto significa «Denegar la acción si la clave está presente en el contexto de la solicitud y los valores coinciden. De lo contrario, deniegue la acción». En el ejemplo anterior, incluida la acción `lambda:DeleteFunction` en la declaración `DenyDeleteSupportedcon` el operador `StringEqualsIfExists` siempre deniega la acción. Para esa acción, la clave no está presente en el contexto y todos los intentos de eliminar ese tipo de recurso se deniegan, independientemente de si el recurso está etiquetado.

Términos relacionados

- Claves de condición global
- Comparación de ABAC con RBAC
- Elementos de la política de JSON de IAM: operadores de condición
- Elemento de condición
- Información general de políticas de JSON

Advertencia de seguridad: denegar NotAction con clave de condición de etiqueta no compatible para el servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Deny NotAction with unsupported tag condition key for service: Using the effect Deny
 with NotAction and the tag condition key {{conditionKeyName}} can be overly permissive
```

```
because some service actions are not denied by this statement. This is because the
condition key doesn't apply to some service actions. We recommend that you use Action
instead of NotAction.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the effect Deny with NotAction and the tag condition key
 {{conditionKeyName}} can be overly permissive because some service actions are not
 denied by this statement. This is because the condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction."
```

Resolución de la advertencia de seguridad

El uso de claves de condición de etiqueta en el `Condition` elemento de una política con el elemento `NotAction` y `"Effect": "Deny"` puede ser demasiado permisivo. La condición se omite para las acciones de servicio que no admiten la clave de condición. AWS recomienda que vuelva a escribir la lógica para denegar una lista de acciones.

Si utiliza la clave de condición `aws:ResourceTag` con `NotAction`, no se deniega ninguna acción de servicio nueva o existente que no admita la clave. AWS le recomienda que indique explícitamente las acciones que desea denegar. el Analizador de acceso de IAM devuelve un resultado independiente para las acciones enumeradas que no admiten la clave de condición `aws:ResourceTag`. Para obtener más información, consulte Advertencia de seguridad: denegar con clave de condición de etiqueta no admitida para el servicio.

Cuando un servicio admite la clave de condición `aws:ResourceTag` puede utilizar etiquetas para controlar el acceso a los recursos de dicho servicio. Esto se conoce como control de acceso basado en atributos (ABAC). Los servicios que no admiten estas claves requieren que el usuario controle el acceso a los recursos mediante Control de acceso basado en recursos (RBAC).

Términos relacionados

- Claves de condición global
- Comparación de ABAC con RBAC
- Elementos de la política de JSON de IAM: operadores de condición
- Elemento de condición
- Información general de políticas de JSON

Advertencia de seguridad: restrinja el acceso a la entidad principal de servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Restrict access to service principal: Granting access to a service principal without
 specifying a source is overly permissive. Use aws:SourceArn, aws:SourceAccount,
 aws:SourceOrgID, or aws:SourceOrgPaths condition key to grant fine-grained access.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Granting access to a service principal without specifying a source
 is overly permissive. Use aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or
 aws:SourceOrgPaths condition key to grant fine-grained access."
```

Resolución de la advertencia de seguridad

Puede especificar Servicios de AWS en el elemento `Principal` de una política basada en recursos mediante una entidad principal de servicio, que es un identificador del servicio. Al conceder acceso a una entidad principal de servicio para que actúe en su nombre, restrinja el acceso. Puede evitar políticas excesivamente permisivas mediante las claves de condición `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` o `aws:SourceOrgPaths` para restringir el acceso a un origen específico, como un ARN de recurso específico, un ID de organización, Cuenta de AWS o rutas de organización. Restringir el acceso lo ayuda a evitar un problema de seguridad denominado problema del suplente confuso.

Términos relacionados

- Entidades principales de Servicio de AWS
- Claves de condición global de AWS: aws:SourceAccount
- Claves de condición global de AWS: aws:SourceArn
- Claves de condición global de AWS: aws:SourceOrgId
- Claves de condición global de AWS: aws:SourceOrgPaths
- Problema del suplente confuso

Advertencia de seguridad: falta la clave de condición para la entidad principal oidc

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing condition key for oidc principal: Using an Open ID Connect principal without a
 condition can be overly permissive. Add condition keys with a prefix that matches your
 federated OIDC principals to ensure that only the intended identity provider assumes
 the role.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using an Open ID Connect principal without a condition can be
 overly permissive. Add condition keys with a prefix that matches your federated OIDC
 principals to ensure that only the intended identity provider assumes the role."
```

Resolución de la advertencia de seguridad

El uso de una entidad principal de Open ID Connect sin una condición puede resultar excesivamente permisivo. Agregue claves de condición con un prefijo que coincida con sus entidades principales de OIDC federadas para garantizar que solo el proveedor de identidad previsto asuma el rol.

Términos relacionados

- Creación de un rol para identidades web o de OpenID Connect Federation (consola)

Advertencia de seguridad: falta la clave de condición del repositorio de github

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Missing github repo condition key: Granting a federated GitHub principal permissions
 without a condition key can allow more sources to assume the role than you intended.
 Add the token.actions.githubusercontent.com:sub condition key and specify the branch
 and repository name in the value.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Granting a federated GitHub principal permissions without a
 condition key can allow more sources to assume the role than you intended. Add the
 token.actions.githubusercontent.com:sub condition key and specify the branch and
 repository name in the value."
```

Resolución de la advertencia de seguridad

Si usa GitHub como un IdP de OIDC, la práctica recomendada es limitar las entidades que pueden asumir el rol asociado con el IdP de IAM. Al incluir una declaración `Condition` en una política de confianza de rol, puede limitar el rol a una organización, repositorio o ramificación específica de GitHub. Puede utilizar la clave de condición `token.actions.githubusercontent.com:sub` para limitar el acceso. Le recomendamos que limite la condición a un conjunto específico de repositorios o ramas. Si no incluye esta condición, GitHub Actions de organizaciones o repositorios fuera de su control pueden asumir roles asociados con el IdP de IAM de GitHub en su cuenta de AWS.

Términos relacionados

- Configuración de un rol para el proveedor de identidades de OIDC de GitHub

Sugerencia: acción de matriz vacía

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array action: This statement includes no actions and does not affect the policy.
 Specify actions.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "This statement includes no actions and does not affect the policy.
 Specify actions."
```

Resolución de la sugerencia

Las instrucciones deben incluir un elemento `Action` o `NotAction` que incluye un conjunto de acciones. Cuando el elemento está vacío, la instrucción de política no proporciona permisos. Especifique acciones en el elemento `Action`

- Elementos de la política de JSON de IAM: acción

Sugerencia: condición de matriz vacía

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array condition: There are no values for the condition key {{key}} and it does
 not affect the policy. Specify conditions.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "There are no values for the condition key {{key}} and it does not
 affect the policy. Specify conditions."
```

Resolución de la sugerencia

La estructura del elemento `Condition` opcional requiere que utilice un operador de condición y un par clave-valor. Cuando el valor de la condición está vacío, la condición devuelve`true` y la declaración de política no proporciona permisos. Especifique un valor de condición.

- Elementos de la política de JSON de IAM: condición

Sugerencia: condición de matriz vacía ForAllValues

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array condition ForAllValues: The ForAllValues prefix with an empty condition key
 matches only if the key {{key}} is missing from the request context. To determine if
 the request context is empty, we recommend that you use the Null condition operator
 with the value of true instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The ForAllValues prefix with an empty condition key matches only
 if the key {{key}} is missing from the request context. To determine if the request
 context is empty, we recommend that you use the Null condition operator with the value
 of true instead."
```

Resolución de la sugerencia

La estructura del elemento `Condition` requiere que utilice un operador de condición y un par clave-valor. El operador de configuración `ForAllValues` prueba si el valor de cada miembro del conjunto de solicitudes es un subconjunto del conjunto de claves de condición.

Cuando utiliza `ForAllValues` con una clave de condición vacía, la condición coincide solo si no hay claves en la solicitud. En su lugar, AWSrecomienda que si desea probar si un contexto de solicitud está vacío, utilice el operador de condición `Null`.

- Claves de contexto multivalor
- Operador de condición nulo
- Elementos de la política de JSON de IAM: condición

Sugerencia — Condición de matriz vacía ForAnyValue

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array condition ForAnyValue: The ForAnyValue prefix with an empty condition key
 {{key}} never matches the request context and it does not affect the policy. Specify
 conditions.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The ForAnyValue prefix with an empty condition key {{key}} never
 matches the request context and it does not affect the policy. Specify conditions."
```

Resolución de la sugerencia

La estructura del elemento `Condition` requiere que utilice un operador de condición y un par clave-valor. El operador de configuración `ForAnyValues` prueba si al menos un miembro del conjunto de valores de la solicitud coincide con al menos un miembro del conjunto de valores de la clave de condición.

Cuando utiliza `ForAnyValues` con una clave de condición vacía, la condición nunca coincide. Esto significa que la declaración no afecta a la política. AWSrecomienda que vuelva a escribir la condición.

- Claves de contexto multivalor
- Elementos de la política de JSON de IAM: condición

Sugerencia: condición de matriz vacía IfExists

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array condition IfExists: The IfExists suffix with an empty condition key matches
 only if the key {{key}} is missing from the request context. To determine if the
 request context is empty, we recommend that you use the Null condition operator with
 the value of true instead.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The IfExists suffix with an empty condition key matches only if the
 key {{key}} is missing from the request context. To determine if the request context
 is empty, we recommend that you use the Null condition operator with the value of true
 instead."
```

Resolución de la sugerencia

El sufijo `...IfExists` edita un operador de condición. Significa que si la clave de la política está presente en el contexto de la solicitud, se debe procesar la clave según se indica en la política. Si la clave no está presente, el elemento de condición se evalúa en verdadero.

Cuando utiliza `...IfExists` con una clave de condición vacía, la condición coincide solo si no hay claves en la solicitud. En su lugar, AWSrecomienda que si desea probar si un contexto de solicitud está vacío, utilice el operador de condición `Null`.

- Operadores de condición ...IfExists
- Elementos de la política de JSON de IAM: condición

Sugerencia: entidad principal de matriz vacía

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Resolución de la sugerencia

Debe utilizar el elemento `Principal` o `NotPrincipal` en las políticas de confianza para los roles de IAM y en las políticas basadas en recursos. Las políticas basadas en recursos son políticas que se integran directamente en un recurso.

Cuando proporciona una matriz vacía en el elemento `Principal`, la instrucción no surte efecto en la política. AWS recomienda especificar las entidades principales que deben tener acceso al recurso.

- Elementos de la política JSON de IAM: Principal
- Elementos de la política de JSON de IAM: NotPrincipal

Sugerencia — Recurso de matriz vacío

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty array resource: This statement includes no resources and does not affect the
 policy. Specify resources.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "This statement includes no resources and does not affect the policy.
 Specify resources."
```

Resolución de la sugerencia

Las instrucciones deben contener un elemento `Resource` o `NotResource`.

Cuando proporciona una matriz vacía en el elemento de recurso de una instrucción, la instrucción no tiene ningún efecto en la política. AWS recomienda que especifique los nombres de recurso de Amazon (ARN) para los recursos.

- Elementos de la política de JSON de IAM;: recurso
- Elementos de la política de JSON de IAM: NotResource

Sugerencia: condición de objeto vacío

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty object condition: This condition block is empty and it does not affect the
 policy. Specify conditions.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "This condition block is empty and it does not affect the policy.
 Specify conditions."
```

Resolución de la sugerencia

La estructura del elemento `Condition` requiere que utilice un operador de condición y un par clave-valor.

Cuando se proporciona un objeto vacío en el elemento de condición de una instrucción, la instrucción no tiene ningún efecto en la política. Elimine el elemento opcional o especifique las condiciones.

- Elementos de la política de JSON de IAM: condición

Sugerencia: entidad principal de objeto vacío

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty object principal: This statement includes no principals and does not affect the
 policy. Specify principals.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "This statement includes no principals and does not affect the
 policy. Specify principals."
```

Resolución de la sugerencia

Debe utilizar el elemento `Principal` o `NotPrincipal` en las políticas de confianza para los roles de IAM y en las políticas basadas en recursos. Las políticas basadas en recursos son políticas que se integran directamente en un recurso.

Cuando proporciona un objeto vacío en el elemento `Principal`, la instrucción no surte efecto en la política. AWS recomienda especificar las entidades principales que deben tener acceso al recurso.

- Elementos de la política JSON de IAM: Principal
- Elementos de la política de JSON de IAM: NotPrincipal

Sugerencia: valor de Sid vacío

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Empty Sid value: Add a value to the empty string in the Sid element.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Add a value to the empty string in the Sid element."
```

Resolución de la sugerencia

El elemento opcional `Sid` (ID de instrucción) le permite ingresar un identificador opcional que se proporciona para la instrucción de la política. Puede asignar un valor de `Sid` a cada instrucción de una matriz de instrucciones. Si opta por utilizar el elemento `Sid`, debe proporcionar un valor de cadena.

Términos relacionados

- Elemento de la política de JSON de IAM: Sid

Sugerencia: mejorar el rango IP

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Improve IP range: The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The non-zero bits in the IP address after the masked bits are
 ignored. Replace address with {{addr}}."
```

Resolución de la sugerencia

Las condiciones de las direcciones IP deben tener el formato CIDR estándar, como 203.0.113.0/24 o 2001:DB8:1234:5678::/64. Cuando se incluyen bits distintos de cero después de los bits enmascarados, no se consideran para la condición. AWS recomienda que utilice la nueva dirección incluida en el mensaje.

- Operadores de condición de dirección IP
- Elementos de la política de JSON de IAM: condición

Sugerencia: nulo con calificador

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Null with qualifier: Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Avoid using the Null condition operator with the ForAllValues or
 ForAnyValue qualifiers because they always return a true or false respectively."
```

Resolución de la sugerencia

En el elemento `Condition`, se crean expresiones en las que se usan operadores de condición como igual o menor para comparar una condición en la política con relación a claves y valores en el contexto de la solicitud. Para las solicitudes que incluyen varios valores para una única clave de condición, debe utilizar los operadores de configuración `ForAllValues` o `ForAnyValue`.

Cuando utiliza el operador de condición `Null` con `ForAllValues`, la declaración siempre devuelve `true`. Cuando utiliza el operador de condición `Null` con `ForAnyValue`, la declaración siempre devuelve `false`. AWS recomienda que utilice la condición `StringLike` con estos operadores de conjunto.

Términos relacionados

- Claves de contexto multivalor
- Operador de condición nulo
- Elemento de condición

Sugerencia: subconjunto de direcciones IP privada

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Private IP address subset: The values for condition key aws:SourceIp include a mix
 of private and public IP addresses. The private addresses will not have the desired
 effect. aws:SourceIp works only for public IP address ranges. To define permissions
 for private IP ranges, use aws:VpcSourceIp.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses will not have the desired effect.
 aws:SourceIp works only for public IP address ranges. To define permissions for
 private IP ranges, use aws:VpcSourceIp."
```

Resolución de la sugerencia

La clave de condición global `aws:SourceIp` solo funciona para rangos de direcciones IP públicas.

Cuando su elemento `Condition` incluye una mezcla de direcciones IP privadas y públicas, la declaración puede no tener el efecto deseado. Usted puede especificar direcciones IP privadas utilizando `aws:VpcSourceIP`.

> **Note**
>
> La clave de condición global `aws:VpcSourceIP` coincide solo si la solicitud proviene de la dirección IP especificada y pasa a través de un punto de enlace de la VPC.

- aws:SourceIp clave de condición global
- aws:VpcSourceIp clave de condición global
- Operadores de condición de dirección IP
- Elementos de la política de JSON de IAM: condición

Sugerencia: subconjunto NotIpAddress privado

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Private NotIpAddress subset: The values for condition key aws:SourceIp include a mix of
 private and public IP addresses. The private addresses have no effect. aws:SourceIp
 works only for public IP address ranges. To define permissions for private IP ranges,
 use aws:VpcSourceIp.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The values for condition key aws:SourceIp include a mix of private
 and public IP addresses. The private addresses have no effect. aws:SourceIp works
 only for public IP address ranges. To define permissions for private IP ranges, use
 aws:VpcSourceIp."
```

Resolución de la sugerencia

La clave de condición global `aws:SourceIp` solo funciona para rangos de direcciones IP públicas.

Cuando su elemento `Condition` incluye el operador de condición `NotIpAddress` y una combinación de direcciones IP privadas y públicas, es posible que la declaración no tenga el efecto deseado. Todas las direcciones IP públicas que no se especifican en la política coincidirán. Ninguna dirección IP privada coincidirá. Para lograr este efecto, puede utilizar `NotIpAddress` con `aws:VpcSourceIP` y especificar las direcciones IP privadas que no deben coincidir.

- aws:SourceIp clave de condición global
- aws:VpcSourceIp clave de condición global
- Operadores de condición de dirección IP
- Elementos de la política de JSON de IAM: condición

Sugerencia: acción redundante

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Redundant action: The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{redundantActionCount}} action(s) are redundant because they
 provide similar permissions. Update the policy to remove the redundant action such as:
 {{redundantAction}}."
```

Resolución de la sugerencia

Cuando utiliza comodines (*) en el elemento `Action`, puede incluir permisos redundantes. AWS recomienda que revise su política e incluya únicamente los permisos que necesite. Esto puede ayudarle a eliminar acciones redundantes.

Por ejemplo, las siguientes acciones incluyen la acción `iam:GetCredentialReport` dos veces.

```
"Action": [
        "iam:Get*",
        "iam:List*",
        "iam:GetCredentialReport"
    ],
```

En este ejemplo, se definen los permisos para todas las acciones de IAM que comienzan con `Get` o `List`. Cuando IAM agrega operaciones adicionales para obtener o listar, esta política las permitirá. Es posible que desee permitir todas estas acciones de solo lectura. La`iam:GetCredentialReport`La acción ya está incluida como parte de`iam:Get*`. Para quitar los permisos duplicados, puede quitar `iam:GetCredentialReport`.

Recibirá un resultado para esta comprobación de política cuando todo el contenido de una acción es redundante. En este ejemplo, si el elemento incluye `iam:*CredentialReport`, no se considera redundante. Eso incluye `iam:GetCredentialReport`, que es redundante, y `iam:GenerateCredentialReport`, que no lo es. Al eliminar `iam:Get*` o `iam:*CredentialReport` se cambiarían los permisos de la política.

- Elementos de la política de JSON de IAM: acción

Políticas administradas AWS con esta sugerencia

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Las acciones redundantes no afectan a los permisos concedidos por las políticas. Cuando se utiliza una política administrada de AWS como referencia para crear su política administrada, AWS recomienda que elimine las acciones redundantes de su política.

Sugerencia: número de valor de condición redundante

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Redundant condition value num: Multiple values in {{operator}} are redundant. Replace
 with the {{greatest/least}} single value for {{key}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Multiple values in {{operator}} are redundant. Replace with the
 {{greatest/least}} single value for {{key}}."
```

Resolución de la sugerencia

Cuando utiliza operadores de condición numérica para valores similares en una clave de condición, puede crear una superposición que dé como resultado permisos redundantes.

Por ejemplo, el siguiente elemento `Condition` incluye varias condiciones `aws:MultiFactorAuthAge` que tienen una superposición de edad de 1200 segundos.

```
"Condition": {
        "NumericLessThan": {
          "aws:MultiFactorAuthAge": [
            "2700",
            "3600"
          ]
        }
      }
```

En este ejemplo, los permisos se definen si la autenticación multifactor (MFA) se completó hace menos de 3600 segundos (1 hora). Podría eliminar el valor `2700` redundante.

- Operadores de condición numérica
- Elementos de la política de JSON de IAM: condición

Sugerencia: recurso redundante

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Redundant resource: The {{redundantResourceCount}} resource ARN(s) are redundant
 because they reference the same resource. Review the use of wildcards (*)
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The {{redundantResourceCount}} resource ARN(s) are redundant because
 they reference the same resource. Review the use of wildcards (*)"
```

Resolución de la sugerencia

Cuando utiliza caracteres comodín (*) en nombres de recurso de Amazon (ARN), puede crear permisos de recursos redundantes.

Por ejemplo, el siguiente elemento `Resource` incluye varios ARN con permisos redundantes.

```
"Resource": [
            "arn:aws:iam::111122223333:role/jane-admin",
            "arn:aws:iam::111122223333:role/jane-s3only",
            "arn:aws:iam::111122223333:role/jane*"
        ],
```

En este ejemplo, se definen los permisos para cualquier rol con un nombre que comience con `jane`. Podría eliminar el `jane-admin` redundante y los `jane-s3only` ARN sin cambiar los permisos resultantes. Esto hace que la política sea dinámica. Definirá permisos para cualquier rol futuro que comience con `jane`. Si la intención de la política es permitir el acceso a un número estático de roles, elimine el último ARN y enumere solo los ARN que deben definirse.

- Elementos de la política de JSON de IAM;: recurso

Políticas administradas AWS con esta sugerencia

AWS políticas administradas le permiten comenzar con AWS asignando permisos basados en casos de uso general AWS.

Los recursos redundantes no afectan a los permisos concedidos por las políticas. Cuando se utiliza una política administrada AWS como referencia para crear su política gestionada por el cliente, AWS recomienda quitar recursos redundantes de la política.

Sugerencia: instrucción redundante

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Redundant statement: The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The statements are redundant because they provide identical
 permissions. Update the policy to remove the redundant statement."
```

Resolución de la sugerencia

El elemento `Statement` es el elemento principal de una política. Este elemento es obligatorio. El elemento `Statement` puede contener una sola instrucción o una matriz de instrucciones individuales.

Cuando se incluye la misma instrucción más de una vez en una política larga, las instrucciones son redundantes. Puede quitar una de las instrucciones sin afectar a los permisos otorgados por la política. Cuando alguien edita una política, puede cambiar una de las instrucciones sin actualizar el duplicado. Esto podría provocar más permisos de los que se pretendía.

- Elementos de la política de JSON de IAM: Instrucción

Sugerencia: comodín en el nombre del servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Wildcard in service name: Avoid using wildcards (*, ?) in the service name because it
 might grant unintended access to other AWS services with similar names.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Avoid using wildcards (*, ?) in the service name because it might
 grant unintended access to other AWS services with similar names."
```

Resolución de la sugerencia

Cuando se incluye el nombre de un servicio AWS en una política, AWSrecomienda no incluir caracteres comodín (*, ?). Esto podría agregar permisos para futuros servicios que no tiene intención. Por ejemplo, hay más de una docena de servicios AWS con la palabra `*code*` en su nombre.

```
"Resource": "arn:aws:*code*::111122223333:*"
```

- Elementos de la política de JSON de IAM;: recurso

Sugerencia: permitir con clave de condición de etiqueta no compatible para el servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Allow with unsupported tag condition key for service: Using the effect Allow with the
 tag condition key {{conditionKeyName}} and actions for services with the following
 prefixes does not affect the policy: {{serviceNames}}. Actions for the listed service
 are not allowed by this statement. We recommend that you move these actions to a
 different statement without this condition key.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the effect Allow with the tag condition key
 {{conditionKeyName}} and actions for services with the following prefixes does not
 affect the policy: {{serviceNames}}. Actions for the listed service are not allowed
 by this statement. We recommend that you move these actions to a different statement
 without this condition key."
```

Resolución de la sugerencia

El uso de claves de condición de etiqueta no compatibles en el elemento `Condition` de una política con `"Effect": "Allow"` no afecta los permisos otorgados por la política, porque la condición se ignora para ese servicio. AWS recomienda quitar las acciones de servicio que no admiten la clave de condición y crear otra instrucción para permitir el acceso a recursos específicos en ese servicio.

Si utiliza la clave de condición `aws:ResourceTag` y no es compatible con una acción de servicio, entonces la clave no se incluye en el contexto de solicitud. En este caso, la condición en el campo `Allow` siempre devuelve `false` y la acción nunca está permitida. Esto sucede incluso si el recurso está etiquetado correctamente.

Cuando un servicio admite la clave de condición `aws:ResourceTag` puede utilizar etiquetas para controlar el acceso a los recursos de dicho servicio. Esto se conoce como control de acceso basado en atributos (ABAC). Los servicios que no admiten estas claves requieren que el usuario controle el acceso a los recursos mediante Control de acceso basado en recursos (RBAC).

> **Note**
>
> Algunos servicios permiten el soporte para la clave de condición `aws:ResourceTag` para un subconjunto de sus recursos y acciones. El Analizador de acceso de IAM devuelve los resultados de las acciones de servicio que no se admiten. Por ejemplo, Amazon S3 admite `aws:ResourceTag` para un subconjunto de sus recursos. Para consultar todos los tipos de recursos disponibles en Amazon S3 que admiten la clave de condición `aws:ResourceTag`, consulte Tipos de recurso definidos por Amazon S3 en la Referencia de autorizaciones de servicio.

Por ejemplo, suponga que desea permitir que los miembros del equipo vean los detalles de recursos específicos etiquetados con el par clave-valor `team=BumbleBee`. También supongamos también que AWS Lambda le permite etiquetar recursos, pero no admite la clave de condición `aws:ResourceTag`. Para permitir la visualización de acciones para AWS App Mesh y AWS Backup si esta etiqueta está presente, utilice la clave de condición `aws:ResourceTag`. Para Lambda, utilice una convención de nomenclatura de recursos que incluya el nombre del equipo como prefijo. A continuación, incluya una instrucción separada que permita ver recursos con esa convención de nomenclatura.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewSupported",
            "Effect": "Allow",
            "Action": [
                "appmesh:DescribeMesh",
                "backup:GetBackupPlan"
                ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/team": "BumbleBee"
                }
```

```
            }
        },
        {
            "Sid": "AllowViewUnsupported",
            "Effect": "Allow",
            "Action": "lambda:GetFunction",
            "Resource": "arn:aws:lambda:*:123456789012:function:team-BumbleBee*"
        }
    ]
}
```

> **Warning**
>
> No utilice la versión Not del operador de condición con "Effect": "Allow" como solución alternativa para este resultado. Estos operadores de condición proporcionan coincidencia denegada. Esto significa que después de que se evalúa la condición, el resultado es negado. En el ejemplo anterior, incluida la acción lambda:GetFunction en la instrucción AllowViewSupportedcon el operador StringNotEquals siempre permite la acción, independientemente de si el recurso está etiquetado.
> No utilice la versión ...IfExists del operador de condición como solución alternativa para este resultado. Esto significa “Permitir la acción si la clave está presente en el contexto de la solicitud y los valores coinciden. De lo contrario, permita la acción”. En el ejemplo anterior, incluida la acción lambda:GetFunction en la declaración AllowViewSupportedcon el operador StringEqualsIfExists siempre permite la acción. Para esa acción, la clave no está presente en el contexto y se permite cada intento de ver ese tipo de recurso, independientemente de si el recurso está etiquetado.

Términos relacionados

- Claves de condición global
- Elementos de la política de JSON de IAM: operadores de condición
- Elemento de condición
- Información general de políticas de JSON

Sugerencia: permitir NotAction con clave de condición de etiqueta no compatible para el servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Allow NotAction with unsupported tag condition key for service: Using the effect Allow
 with NotAction and the tag condition key {{conditionKeyName}} allows only service
 actions that support the condition key. The condition key doesn't apply to some
 service actions. We recommend that you use Action instead of NotAction.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "Using the effect Allow with NotAction and the tag condition key
 {{conditionKeyName}} allows only service actions that support the condition key. The
 condition key doesn't apply to some service actions. We recommend that you use Action
 instead of NotAction."
```

Resolución de la sugerencia

El uso de claves de condición de etiqueta no compatibles en el elemento `Condition` de una política con el elemento `NotAction` y `"Effect": "Allow"` no afecta a los permisos concedidos por la política. La condición se omite para las acciones de servicio que no admiten la clave de condición. AWS recomienda que vuelva a escribir la lógica para admitir una lista de acciones.

Si utiliza la clave de condición `aws:ResourceTag` con `NotAction`, no se deniega ninguna acción de servicio nueva o existente que no admita la clave. AWS le recomienda que indique explícitamente las acciones que desea admitir. El Analizador de acceso de IAM devuelve un resultado independiente para las acciones enumeradas que no admiten la clave de condición `aws:ResourceTag`. Para obtener más información, consulte Sugerencia: permitir con clave de condición de etiqueta no compatible para el servicio.

Cuando un servicio admite la clave de condición `aws:ResourceTag` puede utilizar etiquetas para controlar el acceso a los recursos de dicho servicio. Esto se conoce como control de acceso basado en atributos (ABAC). Los servicios que no admiten estas claves requieren que el usuario controle el acceso a los recursos mediante Control de acceso basado en recursos (RBAC).

Términos relacionados

- Claves de condición global
- Comparación de ABAC con RBAC
- Elementos de la política de JSON de IAM: operadores de condición
- Elemento de condición
- Información general de políticas de JSON

Sugerencia: clave de condición recomendada para la entidad principal de servicio

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Recommended condition key for service principal: To restrict access to the service
 principal {{servicePrincipalPrefix}} operating on your behalf, we recommend
 aws:SourceArn, aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of
 {{key}}.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "To restrict access to the service principal
 {{servicePrincipalPrefix}} operating on your behalf, we recommend aws:SourceArn,
 aws:SourceAccount, aws:SourceOrgID, or aws:SourceOrgPaths instead of {{key}}."
```

Resolución de la sugerencia

Puede especificar Servicios de AWS en el elemento `Principal` de una política basada en recursos mediante una entidad principal de servicio, que es un identificador del servicio. Debe utilizar las claves de condición `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` o `aws:SourceOrgPaths` al conceder acceso a las entidades principales de servicio en lugar de otras claves de condición, como `aws:Referer`. Esto lo ayuda a evitar un problema de seguridad denominado problema del suplente confuso.

Términos relacionados

- Entidades principales de Servicio de AWS
- Claves de condición global de AWS: aws:SourceAccount
- Claves de condición global de AWS: aws:SourceArn
- Claves de condición global de AWS: aws:SourceOrgId
- Claves de condición global de AWS: aws:SourceOrgPaths
- Problema del suplente confuso

Sugerencia: clave de condición irrelevante en la política

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Irrelevant condition key in policy: The condition key {{condition-key}} is not relevant
 for the {{resource-type}} policy.  Use this key in an identity-based policy to govern
 access to this resource.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The condition key {{condition-key}} is not relevant for the
 {{resource-type}} policy.  Use this key in an identity-based policy to govern access
 to this resource."
```

Resolución de la sugerencia

Algunas claves de condición no son relevantes para políticas basadas en recursos. Por ejemplo, la clave de condición `s3:ResourceAccount` no es relevante para la política basada en recursos asociada a un bucket de Amazon S3 o a un tipo de recurso de punto de acceso de Amazon S3.

Puede utilizar la clave de condición de una política basada en la identidad para controlar el acceso al recurso.

Términos relacionados

- Políticas basadas en identidad y políticas basadas en recursos

Sugerencia: Entidad principal redundante en la política de confianza del rol

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Redundant principal in role trust policy: The assumed-role principal
 {{redundant_principal}} is redundant with its parent role {{parent_role}}. Remove the
 assumed-role principal.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The assumed-role principal {{redundant_principal}} is redundant with
 its parent role {{parent_role}}. Remove the assumed-role principal."
```

Resolución de la sugerencia

Si especifica tanto un rol principal asumido como su rol primario en el elemento `Principal` de una política, ésta no permite o deniega los permisos diferentes. Por ejemplo, es redundante si se especifica el elemento `Principal` utilizando el siguiente formato:

```
"Principal": {
            "AWS": [
            "arn:aws:iam::AWS-account-ID:role/rolename",
            "arn:aws:iam::AWS-account-ID:assumed-role/rolename/rolesessionname"
        ]
```

Le recomendamos que elimine el rol principal asumido.

Términos relacionados

- Entidades principales de sesión de rol

Sugerencia: confirme el tipo de reclamación de la audiencia

En el AWS Management Console, el resultado de esta verificación incluye el siguiente mensaje:

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients
 that the JSON web token is intended for. Audience claims can be multivalued or single-
valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If
 the claim is single-valued, do not use a qualifier.
```

En las llamadas programáticas al AWS CLI o API de AWS, el resultado de esta verificación incluye el siguiente mensaje:

```
"findingDetails": "The 'aud' (audience) claim key identifies the recipients that the
 JSON web token is intended for. Audience claims can be multivalued or single-valued.
 If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the claim
 is single-valued, do not use a qualifier."
```

Resolución de la sugerencia

La clave de reclamación `aud` (de la audiencia) es un identificador único para su aplicación que se le emite cuando registra su aplicación con el IdP e identifica a los destinatarios a los que está destinado el token web JSON. La reclamación de la audiencia puede ser de valor único o multivalor. Si la reclamación es multivalor, utilice un `ForAllValues` o un operador de conjunto de

condiciones `ForAnyValue`. Si la reclamación es de valor único, no utilice un operador de conjunto de condiciones.

Términos relacionados

- Creación de un rol para identidades web o de OpenID Connect Federation (consola)
- Claves de contexto multivalor
- Claves de condición de valor único y multivalor

Comprobaciones de políticas personalizadas del Analizador de acceso de IAM

Puede validar sus políticas con respecto a los estándares de seguridad especificados mediante las AWS Identity and Access Management Access Analyzer comprobaciones de políticas personalizadas del Analizador de acceso de IAM. Hay dos tipos de comprobaciones de políticas personalizadas que puede ejecutar:

- Comparación con una política de referencia: al editar una política, puede comprobar si la política actualizada concede nuevos accesos en comparación con una política de referencia, como una versión existente de la política. Puede ejecutar esta comprobación al editar una política con AWS Command Line Interface (AWS CLI), API del Analizador de acceso de IAM (API) de o editor de políticas JSON en la consola de IAM.
- Consulte una lista de acciones de IAM: puede asegurarse de que su política no permita acciones de IAM específicas. Puede ejecutar esta comprobación al crear o editar una política con la AWS CLI o API.

Se asocia un cargo a cada comprobación de política personalizada. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

Cómo funcionan las comprobaciones de políticas personalizadas

Puede ejecutar verificaciones de políticas personalizadas en políticas basadas en identidades y políticas basadas en recursos. Las comprobaciones de políticas personalizadas no se basan en técnicas de coincidencia de patrones ni en el examen de los registros de acceso para determinar si una política permite un acceso nuevo o específico. Al igual que los resultados sobre el acceso externo, las comprobaciones de políticas personalizadas se basan en Zelkova. Zelkova traduce las políticas de IAM en declaraciones lógicas equivalentes, y ejecuta un conjunto de solucionadores

lógicos de uso general y especializados (teorías de módulo de satisfacibilidad) frente al problema. Para comprobar si hay un acceso nuevo o especificado, el Analizador de acceso de IAM aplica Zelkova repetidas veces a una política. Las consultas se vuelven cada vez más específicas para caracterizar las clases de comportamientos que permite la política en función del contenido de la política. Para más información sobre las teorías modulares de la satisfabilidad, consulte Teorías modulares de la satisfabilidad.

En raras ocasiones, el Analizador de acceso de IAM no es capaz de determinar completamente si una declaración de política concede un acceso nuevo o especificado. En esos casos, se equivoca al declarar un falso positivo al no pasar la comprobación de la política personalizada. El Analizador de acceso de IAM está diseñado para proporcionar una evaluación completa de las políticas y se esfuerza por minimizar los falsos negativos. Este enfoque significa que el Analizador de acceso de IAM ofrece un alto grado de seguridad de que una verificación aprobada significa que la política no ha concedido el acceso. Puede inspeccionar manualmente las comprobaciones fallidas consultando la declaración de política que figura en la respuesta del Analizador de acceso de IAM.

Consulte los ejemplos de políticas para comprobar si hay nuevos accesos

Puede encontrar ejemplos de políticas de referencia y aprender a configurar y ejecutar una comprobación de políticas personalizada para nuevos accesos en el repositorio de ejemplos de comprobaciones de políticas personalizadas del Analizador de acceso de IAM en GitHub.

Antes de usar estos ejemplos

Antes de usar estos ejemplos de políticas de referencia, haga lo siguiente:

- Revise las políticas de referencia atentamente y personalícelas para ajustarlas a sus requisitos únicos.
- Pruebe a fondo las políticas de referencia en su entorno con los Servicios de AWS que utilice.

  Las políticas de referencia demuestran la implementación y el uso de comprobaciones de políticas personalizadas. Ellas no son destinadas a ser interpretadas como recomendaciones AWS oficiales o prácticas óptimas que se apliquen exactamente como se indica. Es su responsabilidad probar cuidadosamente las políticas de referencia para comprobar su idoneidad para resolver los requisitos de seguridad de su entorno.
- En su análisis, las comprobaciones de políticas personalizadas son independientes del entorno. Su análisis solo considera la información contenida en las políticas de entrada.

Por ejemplo, las comprobaciones de políticas personalizadas no pueden comprobar si una cuenta es miembro de una AWS organización específica. Por lo tanto, las comprobaciones de políticas personalizadas no pueden comparar los nuevos accesos en función de los valores de las claves para las `aws:PrincipalOrgId` y las `aws:PrincipalAccount` claves de condición

Inspeccionar las comprobaciones de políticas personalizadas fallidas

Cuando se produce un error en una comprobación de una política personalizada, la respuesta del Analizador de acceso de IAM incluye el identificador (`Sid`) de la declaración de política que provocó el error en la comprobación. Aunque el ID de la declaración es un elemento de política opcional, le recomendamos que agregue un ID de declaración para cada declaración de política. La comprobación de política personalizada también devuelve un índice de la declaración para ayudar a identificar el motivo del error de la comprobación. El índice de declaraciones sigue una numeración basada en cero, donde se hace referencia a la primera sentencia como 0. Cuando hay varias declaraciones que provocan un error en una comprobación, la comprobación devuelve solo un identificador de sentencia a la vez. Le recomendamos que corrija la afirmación resaltada en el motivo y vuelva a ejecutar la comprobación hasta que se apruebe.

Validación de políticas con comprobaciones de políticas personalizadas (consola)

Como paso opcional, puede ejecutar una verificación de políticas personalizada al editar una política en el editor de políticas JSON en la consola de IAM. Puede comprobar si la política actualizada concede nuevos accesos en comparación con la versión existente.

Para comprobar si hay nuevos accesos al editar las políticas JSON de IAM

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, seleccione Políticas.
3. En la lista de políticas, seleccione el nombre de la política que desea editar. Puede utilizar el cuadro de búsqueda para filtrar la lista de políticas.
4. Seleccione la pestaña Permisos y, a continuación, Editar.
5. Elija la opción JSON y actualice su política.

6. En el panel de validación de políticas situado debajo de la política, elija la pestaña Comprobar acceso nuevo y, a continuación, elija Comprobar política. Si los permisos modificados otorgan un nuevo acceso, la declaración aparecerá resaltada en el panel de validación de la política.
7. Si no tiene intención de conceder un nuevo acceso, actualice la declaración de política y elija Comprobar la política hasta que no se detecte ningún acceso nuevo.

> **Note**
>
> Se aplica un cargo a cada verificación de acceso nuevo. Para obtener más información sobre los precios, consulte los precios del Analizador de acceso de IAM.

8. Seleccione Siguiente.
9. En la página Revisar y guardar, revise los Permisos definidos en esta política y, a continuación, elija Guardar cambios.

Validación de políticas con comprobaciones de políticas personalizadas (AWS CLI o API)

Puede ejecutar comprobaciones de políticas personalizadas del Analizador de acceso de IAM desde la API del Analizador de acceso de IAM AWS CLI o desde esta.

Ejecutar comprobaciones de políticas personalizadas del Analizador de acceso de IAM (AWS CLI)

- Para comprobar si se permite un nuevo acceso a una política actualizada en comparación con la política existente, ejecute el siguiente comando: check-no-new-access
- Para comprobar si una política no permite el acceso especificado, ejecute el siguiente comando: check-access-not-granted

Ejecutar comprobaciones de políticas personalizadas del Analizador de acceso de IAM (API)

- Para comprobar si se permite un nuevo acceso a una política actualizada en comparación con la política existente, utilice la operación de API CheckNoNewAccess.
- Para comprobar si una política no permite el acceso especificado, utilice la operación de API CheckAccessNotGranted.

Generación de políticas del Analizador de acceso de IAM

Como administrador o desarrollador, puede conceder permisos a entidades (usuarios o roles) de IAM más allá de lo que requieren. IAM proporciona varias opciones para ayudarle a refinar los permisos que concede. Una opción es generar una política de IAM basada en la actividad de acceso de una entidad. El Analizador de acceso de IAM revisa los registros de AWS CloudTrail y genera una plantilla de política que contiene los permisos que la entidad ha utilizado en su intervalo de fechas especificado. Puede utilizar la plantilla para crear una política con permisos detallados que otorguen solo los permisos necesarios para admitir su caso de uso específico.

Temas

Cómo funciona la generación de políticas

El Analizador de acceso de IAM analiza los eventos de CloudTrail para identificar acciones y servicios que han sido utilizados por una entidad (usuario o rol) de IAM. A continuación, genera una política de IAM basada en esa actividad. Puede refinar los permisos de una entidad cuando reemplaza una política de permisos amplia asociada a la entidad por la política generada. A continuación se presenta información general de alto nivel del proceso de generación de políticas.

- Configuración para la generación de plantillas – Especifique un período de tiempo de hasta 90 días para que el Analizador de acceso de IAM analice los eventos históricos de AWS CloudTrail. Debe especificar un rol de servicio existente o crear uno nuevo. El rol de servicio da acceso al Analizador de acceso de IAM a la información del registro de seguimiento de CloudTrail y al último acceso al servicio para identificar los servicios y acciones que se utilizaron. Debe especificar el registro de

seguimiento de CloudTrail que registra eventos para la cuenta antes de poder generar una política. Para obtener más información sobre las cuotas del Analizador de acceso de IAM para los datos de CloudTrail, consulte Cuotas del Analizador de acceso de IAM.

- Generar políticas – El Analizador de acceso de IAM genera una política basada en la actividad de acceso de los eventos de CloudTrail.
- Revisar y personalizar la política: Una vez generada la política, puede revisar los servicios y las acciones que utilizó la entidad durante el intervalo de fechas especificado. Puede personalizar aún más la política, agregando o quitando permisos, especificando recursos y agregando condiciones a la plantilla de política.
- Crear y asociar la política – Tiene la opción de guardar la política generada mediante la creación de una política administrada. Puede asociar la política que cree al usuario o rol cuya actividad se utilizó para generar la política.

Información de nivel de servicio y acción

Cuando el Analizador de acceso de IAM genera una política de IAM, se devuelve información para ayudarle a personalizar aún más la política. Cuando se genera una política, se pueden devolver dos categorías de información:

- Política con información de nivel de acción: Para algunos servicios de AWS, como Amazon EC2, el Analizador de acceso de IAM puede identificar las acciones encontradas en los eventos de CloudTrail y enumera las acciones utilizadas en la política que genera. Para obtener una lista de los servicios compatibles, consulte Servicios de generación de políticas del Analizador de acceso de IAM. Para algunos servicios, el Analizador de acceso de IAM le pide que agregue acciones para los servicios a la política generada.
- Política con información de nivel de servicio: El Analizador de acceso de IAM utiliza la última información a la que se accedió para crear una plantilla de política con todos los servicios utilizados recientemente. Cuando utilice la AWS Management Console, le pedimos que revise los servicios y agregue acciones para completar la política.

Para obtener una lista de acciones de cada servicio, consulte Acciones, recursos y claves de condiciones de servicios de AWS en la Referencia de autorizaciones de servicio.

Cosas que debe saber acerca de la generación de políticas

Antes de generar una política, revise los siguientes detalles importantes.

- Habilite un registro de seguimiento de CloudTrail – Debe tener habilitado un registro de seguimiento de CloudTrail para que su cuenta genere una política basada en la actividad de acceso. Al crear un registro de seguimiento de CloudTrail , CloudTrail envía los eventos relacionados con el registro de seguimiento al bucket Amazon S3 que especifique. Para obtener más información sobre cómo crear un registro de seguimiento de CloudTrail, consulte Creación de un registro de seguimiento para la cuenta de AWS en la Guía del usuario de AWS CloudTrail.
- Eventos de datos no disponibles: El Analizador de acceso de IAM no identifica la actividad de nivel de acción para eventos de datos, como eventos de datos de Amazon S3, en las políticas generadas.
- PassRole – `iam:PassRole` CloudTrail no rastrea la acción, y esta no está incluida en las políticas generadas.
- Reducir el tiempo de generación de políticas – Para generar una política más rápido, reduzca el intervalo de fechas que especifique durante la configuración para la generación de políticas.
- Uso de CloudTrail para auditorías – No utilice la generación de políticas con fines de auditoría; en su lugar, utilice CloudTrail. Para obtener más información sobre cómo utilizar CloudTrail, consulte Registro de IAM y llamadas API AWS STS con AWS CloudTrail.
- Acciones denegadas: La generación de políticas revisa todos los eventos de CloudTrail, incluidas las acciones denegadas.
- Consola de IAM de una política – Puede tener una política generada a la vez en la consola de IAM.
- Disponibilidad de políticas generadas en la consola de IAM – Puede revisar una política generada en la consola de IAM durante un máximo de 7 días después de generarla. Después de 7 días, debe generar una nueva política.
- Cuotas de generación de políticas: Para obtener información adicional acerca de las cuotas de generación de políticas del Analizador de acceso de IAM, consulte Cuotas del Analizador de acceso de IAM.
- Se aplican las tarifas estándares de Amazon S3: Cuando utiliza la característica de generación de políticas, el Analizador de acceso de IAM revisa los registros de CloudTrail en su bucket de S3. No hay cargos adicionales de almacenamiento para acceder a sus registros de CloudTrail para la generación de políticas. AWS cobra las tarifas estándares de Amazon S3 para las solicitudes y la transferencia de datos de los registros de CloudTrail almacenados en su bucket de S3.
- Soporte de AWS Control Tower: La generación de políticas no admite AWS Control Tower para la generación de políticas.

Permisos necesarios para generar una política

Los permisos que necesita para generar una política por primera vez difieren de los que necesita para generar una política para usos posteriores.

Configuración por primera vez

Cuando genere una política por primera vez, debe elegir un rol de servicio existente adecuado en su cuenta o crear un nuevo rol de servicio. El rol de servicio proporciona al Analizador de acceso de IAM acceso a CloudTrail y a la última información del servicio a la que se accedió en su cuenta. Sólo los administradores deben tener los permisos necesarios para crear y configurar roles. Por lo tanto, se recomienda que un administrador cree el rol de servicio durante la primera instalación. Para obtener más información sobre los permisos necesarios para crear roles de servicio, consulte Creación de un rol para delegar permisos a un servicio de AWS.

Permisos necesarios para roles de servicio

Cuando se crea un rol de servicio, se configuran dos políticas para el rol. Asocia una política de permisos de IAM al rol, que especifica lo que éste puede hacer. También puede asociar una política de confianza de rol al rol que especifica la entidad principal que puede utilizar el rol.

En el primer ejemplo de política se muestra la política de permisos para el rol de servicio necesario para generar una política. El segundo ejemplo de política muestra la política de confianza de rol necesaria para el rol de servicio. Puede utilizar estas políticas para poder crear un rol de servicio cuando utiliza la API o AWS CLI de AWS para generar una política. Cuando utiliza la consola de IAM para crear un rol de servicio como parte del proceso de generación de políticas, nosotros generamos estas políticas por usted.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloudtrail:GetTrail",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetServiceLastAccessedDetails",
```

```
                "iam:GenerateServiceLastAccessedDetails"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
        }
    ]
}
```

En el siguiente ejemplo de política se muestra la política de confianza de roles con los permisos que le permiten al Analizador de acceso de IAM asumir el rol.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Usos posteriores

Para generar políticas en la AWS Management Console, un usuario de IAM debe tener una política de permisos que le permita transferir el rol de servicio que se utiliza para la generación de políticas al Analizador de acceso de IAM. `iam:PassRole` suele ir acompañado de `iam:GetRole` para que el usuario pueda obtener los detalles del rol que se va a transferir. En este ejemplo, el usuario puede transferir únicamente los roles que existen en la cuenta especificada con nombres que empiezan con `AccessAnalyzerMonitorServiceRole*`. Para obtener más información sobre la transmisión de

roles de IAM a servicios de AWS, consulte Concesión de permisos a un usuario para transferir un rol a un servicio de AWS.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::123456789012:role/service-role/
AccessAnalyzerMonitorServiceRole*"
    }
  ]
}
```

También debe tener los siguientes permisos del Analizador de acceso de IAM para generar políticas en la API AWS Management Console, AWS o AWS CLI, como se muestra en la instrucción de política a continuación.

```
{
  "Sid": "AllowUserToGeneratePolicy",
  "Effect": "Allow",
  "Action": [
    "access-analyzer:CancelPolicyGeneration",
    "access-analyzer:GetGeneratedPolicy",
    "access-analyzer:ListPolicyGenerations",
    "access-analyzer:StartPolicyGeneration"
  ],
  "Resource": "*"
}
```

Tanto para primer uso como para usos posteriores

Cuando utiliza la AWS Management Console para generar una política, debe tener permiso `cloudtrail:ListTrails` para enumerar los registros de seguimiento de CloudTrail de su cuenta como se muestra en la siguiente instrucción de política.

```
{
```

```
  "Sid": "AllowUserToListTrails",
  "Effect": "Allow",
  "Action": [
    "CloudTrail:ListTrails"
  ],
  "Resource": "*"
}
```

Generar una política basada en la actividad (consola) de CloudTrail

Puede generar una política para un usuario o rol de IAM.

Paso 1: Generar una política basada en la actividad de CloudTrail

En el siguiente procedimiento se explica cómo generar una política para un rol mediante la AWS Management Console.

Generar una política para un rol de IAM

1. Inicie sesión en AWS Management Console y abra la consola IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, seleccione Roles.

   > **Note**
   >
   > Los pasos para generar una política basada en la actividad de un usuario de IAM son casi idénticos. Para ello, elija Usuarios en lugar de Roles.

3. En la lista de roles de su cuenta, elija el nombre del rol cuya actividad desea utilizar para generar una política.
4. En la pestaña Permisos, en la sección Generar política basada en eventos de CloudTrail, elija Generar política.
5. En la página Generar política especifique el período de tiempo en el que desea que el Analizador de acceso de IAM analice los eventos de CloudTrail para las acciones realizadas con el rol. Puede elegir un rango de hasta 90 días. Le recomendamos que elija el período de tiempo más corto posible para reducir el tiempo de generación de políticas.
6. En la sección Acceso a CloudTrail, elija un rol existente adecuado o cree uno nuevo si no existe un rol adecuado. Este rol otorga al Analizador de acceso de IAM permisos para acceder

al registro de seguimiento de CloudTrail en su nombre para revisar la actividad de acceso e identificar los servicios y acciones que se han utilizado. Para obtener más información sobre los permisos necesarios para este rol, consulte Permisos necesarios para generar una política.

7. En la sección del registro de seguimiento de CloudTrail que se va a analizar, especifique el registro de seguimiento de CloudTrail que registra los eventos de la cuenta.

   Si elige un seguimiento de CloudTrail que almacene los registros en una cuenta diferente, se mostrará un cuadro de información sobre el acceso entre cuentas. El acceso entre cuentas requiere una configuración adicional. Para obtener más información, consulte Choose a role for cross-account access más adelante en este tema.

8. Elija Generar política.

9. Mientras la generación de políticas está en curso, volverá a la página Resumen de roles en la pestaña Permisos. Espere hasta que el estado de la sección Detalles de solicitud de política muestre Realizado correctamente, y, a continuación, elija Ver política generada. Puede ver la política generada durante un máximo de siete días. Si genera otra política, la política existente se sustituye por la nueva que genere.

Paso 2: Revise los permisos y agregue acciones para los servicios utilizados

Revise los servicios y acciones utilizados por el rol, de acuerdo a lo que identificó el Analizador de acceso de IAM. Puede agregar acciones para cualquier servicio que se haya utilizado a la plantilla de política generada.

1. Revise las siguientes secciones:

   - En la página Revisar permisos, revise la lista de Acciones incluidas en la política generada. La lista muestra los servicios y las acciones identificados por el Analizador de acceso de IAM que fueron utilizados por el rol en el intervalo de fechas especificado.
   - La sección Servicios utilizados muestra servicios adicionales que el Analizador de acceso de IAM identificó como utilizados por el rol en el intervalo de fechas especificado. Es posible que la información sobre las acciones que se utilizaron no esté disponible para los servicios enumerados en esta sección. Utilice los menús de cada servicio enumerado para seleccionar manualmente las acciones que desea incluir en la política.

2. Cuando haya terminado de agregar acciones, elija Siguiente.

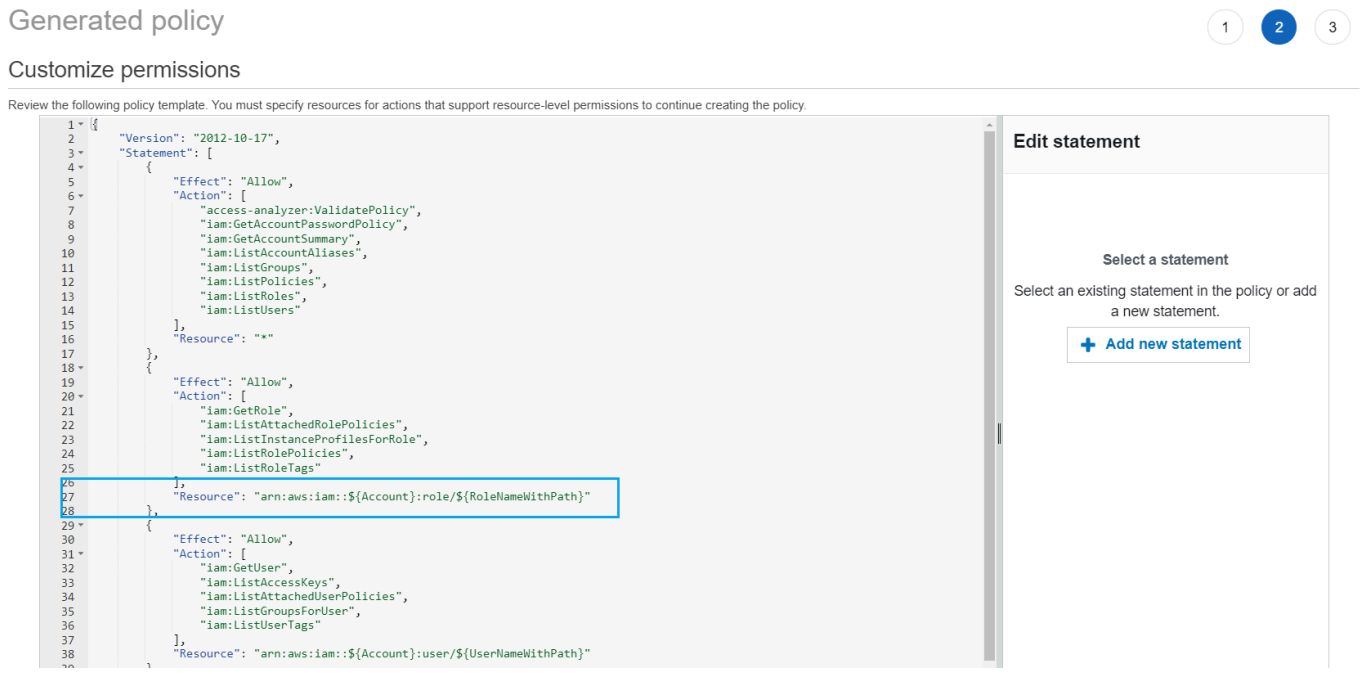
Paso 3: Personalice aún más la política generada

Puede personalizar aún más la política agregando o quitando permisos o especificando recursos.

Para personalizar la política generada

1. Actualice la plantilla de política. La plantilla de política contiene marcadores de posición ARN de recursos para acciones que admiten permisos de nivel de recursos, como se muestra en la siguiente imagen. Los permisos de nivel de recursos hacen referencia a la capacidad de especificar en qué recursos los usuarios tienen permitido realizar acciones. Se recomienda utilizar ARN para especificar los recursos individuales en la política para las acciones que admiten permisos de nivel de recursos. Puede reemplazar los ARN de marcador de posición de recursos por ARN de recursos válidos para su caso de uso.

   Si una acción de la API no admite ARN individuales, debe utilizar un comodín * para especificar que la acción puede afectar a todos los recursos. Para saber qué servicios de AWS admiten permisos en el nivel de recursos, consulte Servicios de AWS que funcionan con IAM. Para obtener una lista de las acciones de cada servicio y para saber qué acciones admiten permisos de nivel de recursos, consulte Acciones, Recursos y Claves de condición para servicios de AWS.



2. (Opcional) Agregue, modifique o elimine instrucciones de política JSON en la plantilla. Para obtener más información sobre cómo escribir políticas JSON, consulte Creación de políticas de IAM (consola).
3. Cuando haya terminado de personalizar la plantilla de política, tiene las siguientes opciones:

- (Opcional) Puede copiar el JSON en la plantilla para utilizarlo por separado fuera de la página de la Política generada. Por ejemplo, si desea utilizar el JSON para crear una política en una cuenta diferente. Si la política de la plantilla supera el límite de 6144 caracteres para las políticas JSON, la política se divide en varias políticas.
- Elija Siguiente para revisar y crear una política administrada en la misma cuenta.

Paso 4: Revisar y crear una política administrada

Si tiene permisos para crear y asociar política de IAM, puede crear una política administrada a partir de la política que se generó. A continuación, puede asociar la política a un usuario o rol de su cuenta.

Para revisar y crear una política

1. En la página Revisar y crear política administrada, especifique Nombre y Descripción (opcional) para la política que está creando.
2. (Opcional) En la sección Resumen, puede revisar los permisos que se incluirán en la política.
3. (Opcional) Agregar metadatos a la política al adjuntar las etiquetas como pares de clave-valor. Para obtener más información acerca del uso de etiquetas en IAM, consulte Etiquetado de los recursos de IAM.
4. Cuando haya terminado, realice una de las acciones siguientes:
   - Puede asociar la nueva política directamente al rol que se utilizó para generar la política. Para ello, cerca de la parte inferior de la página, seleccione la casilla de verificación situada junto a Asociar política a ***YourRoleName***. A continuación, elija Crear y asociar política.
   - De lo contrario, seleccione Crear política. Puede encontrar la política que creó en la lista de políticas en el panel de navegación Políticas de la consola de IAM.
5. Puede asociar la política que creó a una entidad de su cuenta. Después de asociar la política, puede eliminar cualquier otra política demasiado amplia que pueda estar asociada a la entidad. Para saber cómo se adjunta una política administrada, consulte Agregar permisos de identidad de IAM (consola).

Generar una política mediante datos AWS CloudTrail en otra cuenta

Puede crear pistas de CloudTrail que almacenen datos en cuentas centrales para optimizar las actividades gobernantes. Por ejemplo, puede utilizar AWS Organizations para crear un registro de

seguimiento que registra todos los eventos para todos los Cuentas de AWS en esa organización. El seguimiento pertenece a una cuenta central. Si desea generar una política para un usuario o rol en una cuenta que sea diferente de la cuenta donde se almacenan los datos de registro de CloudTrail, debe conceder acceso entre cuentas. Para ello, necesita un rol y una política de bucket que otorguen permisos del Analizador de acceso de IAM a los registros de CloudTrail. Para obtener más información sobre los registros de seguimiento de Organizations, consulte Creación de registros de seguimiento de una organización.

En este ejemplo, suponga que desea generar una política para un usuario o un rol en la cuenta A. El seguimiento de CloudTrail en la cuenta A almacena los registros de CloudTrail en un bucket en la cuenta B. Antes de poder generar una política, debe realizar las siguientes actualizaciones:

1. Elija un rol existente o cree un nuevo rol de servicio que otorgue acceso del Analizador de acceso de IAM al bucket de la cuenta B (donde se almacenan los registros de CloudTrail).
2. Verifique la propiedad de los objetos del bucket de Amazon S3 y la política de permisos del bucket en la cuenta B para que el Analizador de acceso de IAM pueda acceder a los objetos del bucket.

Paso 1 : Elija o cree un rol para el acceso entre cuentas

- En la pantalla Generar políticas, la opción de Uso de un rol existente está preseleccionada para usted si existe un rol con los permisos necesarios en su cuenta. De lo contrario, elija Creación y uso de un nuevo rol de servicio. El nuevo rol se utiliza para conceder acceso al Analizador de acceso de IAM a los registros de CloudTrail en la cuenta B.

Paso 2: Verifique o actualizar la configuración de su bucket de Amazon S3 en la cuenta B

1. Inicie sesión en la AWS Management Console y abra la consola de Amazon S3 en https://console.aws.amazon.com/s3/.
2. En la lista Buckets, elija el nombre del bucket de donde se almacenan sus registros de seguimiento de CloudTrail.
3. Elija la pestaña Permisos y vaya a la sección Propiedad del objeto.

   Use la configuración del bucket Propiedad del objeto de Amazon S3 para controlar la propiedad de los objetos que cargue en sus buckets. De forma predeterminada, si otras Cuentas de AWS cargan objetos en su bucket, los objetos seguirán siendo propiedad de la cuenta de carga. Para generar una política, todos los objetos del bucket deben pertenecer al propietario del bucket. En

función de su caso de uso de ACL, es posible que tenga que cambiar la configuración Propiedad del objeto del bucket. Establezca Propiedad del objeto en una de las siguientes opciones.

- Propietario del bucket obligatorio (recomendado)
- Propietario del bucket preferido

> **Important**
>
> Para generar correctamente una política, los objetos del bucket deben pertenecer al propietario del bucket. Si elige usar Propietario del bucket preferido, solo podrá generar una política para el periodo de tiempo posterior al cambio de propiedad del objeto.

Para más información sobre la propiedad de los objetos en Amazon S3, consulte Control de la propiedad de los objetos y desactivación de las ACL para el bucket en la Guía del usuario de Amazon S3.

4. Agregue permisos a su política de bucket de Amazon S3 en la cuenta B para permitir el acceso al rol en la cuenta A.

   El siguiente ejemplo de política permite `ListBucket` y `GetObject` para un bucket denominado *`DOC-EXAMPLE-BUCKET`*. Permite el acceso si el rol que accede al bucket pertenece a una cuenta de su organización y tiene un nombre que comienza con `AccessAnalyzerMonitorServiceRole`. El uso de `aws:PrincipalArn` como una `Condition` en el elemento `Resource` garantiza que el rol solo puede acceder a la actividad de la cuenta si pertenece a la cuenta A. Puede reemplazar *`DOC-EXAMPLE-BUCKET`* por el nombre del bucket, `optional-prefix` por un prefijo opcional para el bucket y `organization-id` por el ID de su organización.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PolicyGenerationBucketPolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
```

```
        "s3:GetObject",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

5. Si cifra los registros con AWS KMS, actualice la política de claves de AWS KMS en la cuenta donde almacene los registros de CloudTrail para conceder acceso al Analizador de acceso de IAM para utilizar la clave, tal y como se muestra en el siguiente ejemplo de política. Reemplazar `CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN` con el ARN para su seguimiento y `organization-id` con el ID de organización.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "kms:Decrypt",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:cloudtrail:arn":
 "CROSS_ACCOUNT_ORG_TRAIL_FULL_ARN",
          "aws:PrincipalOrgID": "organization-id"
        },
```

```
        "StringLike": {
          "kms:ViaService": [
            "access-analyzer.*.amazonaws.com",
            "s3.*.amazonaws.com"
          ]
          "aws:PrincipalArn": "arn:aws:iam::${aws:PrincipalAccount}:role/service-
role/AccessAnalyzerMonitorServiceRole*"
        }
      }
    }
  ]
}
```

Generar una política basada en la actividad de CloudTrail (CLI de AWS)

Puede utilizar los siguientes comandos para generar una política mediante AWS CLI.

Cómo generar una política

- iniciar generación de políticas de accessanalyzer de aws

Cómo ver una política generada

- obtener política generada de accessanalyzer de aws

Cómo cancelar una solicitud de generación de política

- cancelar generación de política de accessanalyzer de aws

Cómo ver una lista de solicitudes de generación de políticas

- lista de generación de políticas de accessanalyzer de aws

Generar una política basada en la actividad de CloudTrail (API de AWS)

Puede utilizar las siguientes operaciones para generar una política mediante la API de AWS.

Cómo generar una política

- "StartPolicyGeneration" (iniciar generación de políticas)

Cómo ver una política generada

- "GetGeneratedPolicy" (ver política generada)

Cómo cancelar una solicitud de generación de política

- "CancelPolicyGeneration" (cancelar generación de política)

Cómo ver una lista de solicitudes de generación de políticas

- "ListPolicyGenerations" (enumerar generación de políticas)

Servicios de generación de políticas del Analizador de acceso de IAM

En la siguiente tabla, se enumeran los servicios de AWS para los que el Analizador de acceso de IAM genera políticas con información a nivel sobre toda la acción. Para obtener una lista de acciones de cada servicio, consulte Acciones, recursos y claves de condiciones de Servicios de AWS en la Referencia de autorizaciones de servicio.

Servicio	Prefijo de servicio
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	account
AWS Certificate Manager	acm
Amazon Managed Workflows para Apache Airflow	airflow
AWS Amplify	amplify
Creador de UI de AWS Amplify	amplifyuibuilder
Integraciones de aplicaciones de Amazon	app-integrations

Servicio	Prefijo de servicio
AWS AppConfig	appconfig
Amazon AppFlow	appflow
AWS Application Cost Profiler	application-cost-profiler
Información de aplicaciones de Amazon CloudWatch	applicationinsights
AWS App Mesh	appmesh
Amazon AppStream 2.0	appstream
AWS AppSync	appsync
Amazon Managed Service para Prometheus	aps
Amazon Athena	athena
AWS Audit Manager	auditmanager
AWS Auto Scaling	autoscaling
AWS Marketplace	aws-marketplace
AWS Backup	backup
AWS Batch	batch
Amazon Braket	braket
AWS Budgets	budgets
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm

Servicio	Prefijo de servicio
Amazon CloudSearch	cloudsearch
AWS CloudTrail	cloudtrail
Amazon CloudWatch	cloudwatch
AWS CodeArtifact	codeartifact
AWS CodeDeploy	codedeploy
Generador de perfiles de Amazon CodeGuru	codeguru-profiler
Revisor de Amazon CodeGuru	codeguru-reviewer
AWS CodePipeline	codepipeline
AWS CodeStar	codestar
Notificaciones de AWS CodeStar	codestar-notifications
Identidad de Amazon Cognito	cognito-identity
Grupos de usuarios de Amazon Cognito	cognito-idp
Amazon Cognito Sync	cognito-sync
Amazon Comprehend Medical	comprehendmedical
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	connect
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew

Servicio	Prefijo de servicio
AWS Data Exchange	dataexchange
AWS Data Pipeline	datapipeline
DynamoDB Accelerator	dax
AWS Device Farm	devicefarm
Amazon DevOps Guru	devops-guru
AWS Direct Connect	directconnect
Amazon Data Lifecycle Manager	dlm
AWS Database Migration Service	dms
Clústeres elásticos de Amazon DocumentDB	docdb-elastic
AWS Directory Service	ds
Amazon DynamoDB	dynamodb
Amazon Elastic Block Store (EBS)	ebs
Amazon Elastic Compute Cloud	ec2
Amazon Elastic Container Registry	ecr
Amazon Elastic Container Registry Public	ecr-public
Amazon Elastic Container Service	ecs
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk

Servicio	Prefijo de servicio
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR en EKS (Contenedores de EMR)	emr-containers
Amazon EMR sin servidor	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	events
Amazon CloudWatch Evidently	evidently
Amazon FinSpace	finspace
Amazon Data Firehose	firehose
AWS Fault Injection Service	fis
AWS Firewall Manager	fms
Amazon Fraud Detector	frauddetector
Amazon FSx	fsx
Amazon GameLift	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation

Servicio	Prefijo de servicio
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
Almacén de identidad de AWS	identitystore
EC2 Image Builder	imagebuilder
Amazon Inspector Classic	inspector
Amazon Inspector	inspector2
AWS IoT	iot
AWS IoT Analytics	iotanalytics
AWS IoT Core Device Advisor	iotdeviceadvisor
AWS IoT Events	iotevents
AWS IoT Fleet Hub	iotfleethub
AWS IoT SiteWise	iotsitewise
AWS IoT TwinMaker	iottwinmaker
AWS IoT Wireless	iotwireless
Amazon Interactive Video Service	ivs
Amazon Interactive Video Service Chat	ivschat
Amazon Managed Streaming for Apache Kafka	kafka
Amazon Managed Streaming para Kafka Connect	kafkaconnect

Servicio	Prefijo de servicio
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex	lex
Administrador de suscripciones de Linux de AWS License Manager	license-manager-linux-subscriptions
Amazon Lightsail	lightsail
Registros de Amazon CloudWatch	logs
Amazon Lookout for Equipment	lookoutequipment
Amazon Lookout for Metrics	lookoutmetrics
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaPackage	mediapackage
AWS Elemental MediaPackage VOD	mediapackage-vod

Servicio	Prefijo de servicio
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB para Redis	memorydb
AWS Application Migration Service	mgn
AWS Migration Hub	mgh
Recomendaciones de estrategias de AWS Migration Hub	migration hub-strategy
Amazon Pinpoint	mobiletargeting
Amazon MQ	mq
AWS Network Manager	networkmanager
Amazon Nimble Studio	nimble
AWS HealthOmics	omics
AWS OpsWorks	opsworks
AWS OpsWorks CM	opsworks-cm
AWS Outposts	outposts
AWS Organizations	organizations
AWS Panorama	panorama
Información de rendimiento de AWS	pi
Canalizaciones de Amazon EventBridge	pipes
Amazon Polly	polly
Perfiles de clientes de Amazon Connect	profile

Servicio	Prefijo de servicio
Amazon QLDB	qldb
AWS Resource Access Manager	ram
Papelera de reciclaje de AWS	rbin
Amazon Relational Database Service	rds
Amazon Redshift	redshift
API de datos de Amazon Redshift	redshift-data
AWS Migration Hub Refactor Spaces	refactor-spaces
Amazon Rekognition	rekognition
AWS Resilience Hub	resiliencehub
Explorador de recursos de AWS	resource-explorer-2
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
Funciones de AWS Identity and Access Management en cualquier lugar	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	route53-recovery-control-config
Preparación para recuperación de Amazon Route 53	route53-recovery-readiness
Amazon Route 53 Resolver	route53resolver
AWS CloudWatch RUM	rum
Amazon Simple Storage Service	s3

Servicio	Prefijo de servicio
Amazon S3 en Outposts	s3-outposts
Capacidades geoespaciales de Amazon SageMaker	sagemaker-geospatial
Savings Plans	savingsplans
Esquemas de Amazon EventBridge	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	securityhub
Amazon Security Lake	securitylake
AWS Serverless Application Repository	serverlessrepo
AWS Service Catalog	servicecatalog
AWS Cloud Map	servicediscovery
Service Quotas	servicequotas
Amazon Simple Email Service	ses
AWS Shield	shield
AWS Signer	signer
AWS SimSpace Weaver	simspaceweaver
AWS Server Migration Service	sms
Servicio de SMS y voz de Amazon Pinpoint	sms-voice
AWS Snowball	snowball
Amazon Simple Queue Service	sqs

Servicio	Prefijo de servicio
AWS Systems Manager	ssm
AWS Systems Manager Incident Manager	ssm-incidents
AWS Systems Manager para SAP	ssm-sap
AWS Step Functions	states
AWS Security Token Service	sts
Amazon Simple Workflow Service	swf
Amazon CloudWatch Synthetics	synthetics
AWS Resource Groups Tagging API	tag
Amazon Textract	textract
Amazon Timestream	timestream
Creador de redes de telecomunicaciones de AWS	tnb
Amazon Transcribe	transcribe
AWS Transfer Family	transfer
Amazon Translate	translate
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink

Servicio	Prefijo de servicio
Amazon WorkSpaces	workspaces
AWS X-Ray	xray

Cuotas del Analizador de acceso de IAM

El Analizador de acceso de IAM tiene las siguientes cuotas:

Recurso	Cuota predeterminada	Cuota máxima
Máximo de analizadores a nivel de cuenta por tipo de analizador por Cuenta de AWS por Región	1	1
Máximo de analizadores a nivel de organización por tipo de analizador por Cuenta de AWS por Región	5	20[1]
Reglas de archivado máximas por analizador	100 Cada regla de archivo puede tener hasta 20 valores por criterio.	1000[1]
Número máximo de previsualizaciones de acceso por analizador y hora	1 000	1 000
Archivos de registro AWS CloudTrail procesados por generaciones de políticas	100 000	100 000
Generaciones simultáneas de políticas	1	1

Recurso	Cuota predeterminada	Cuota máxima
Generación de políticas de tamaño de datos AWS CloudTrail	25 GB	25 GB
Generación de políticas de intervalo temporal AWS CloudTrail	90 días	90 días
Generación de políticas por día	África (Ciudad del Cabo): 5 Asia-Pacífico (Hong Kong): 5 Europa (Milán): 5 Medio Oriente (Baréin): 5 Todas las demás regiones compatibles: 50 **Note** Las solicitudes de generación de políticas canceladas se aplican a la cuota diaria.	África (Ciudad del Cabo): 5 Asia-Pacífico (Hong Kong): 5 Europa (Milán): 5 Medio Oriente (Baréin): 5 Todas las demás regiones compatibles: 50

[1]El cliente puede configurar algunas cuotas a través de Service Quotas.

Solución de problemas de IAM

Si tiene problemas de acceso denegado o dificultades similares cuando trabaja con AWS Identity and Access Management (IAM), consulte los temas de esta sección.

Temas

Solución de problemas generales de IAM

Utilice la información que se incluye aquí para diagnosticar y solucionar los problemas comunes cuando trabaje con AWS Identity and Access Management (IAM).

Problemas

No puedo iniciar sesión en mi cuenta AWS

Compruebe que tiene las credenciales correctas y que está utilizando el método correcto para iniciar sesión. Para obtener más información, consulte Solución de problemas de inicio de sesión en la Guía del usuario de AWS Sign-In.

He perdido mi claves de acceso

Las claves de acceso se componen de dos partes:

- El identificador de la clave de acceso. No es secreto y se puede ver en la consola de IAM en la que hay una lista de claves de acceso, como, por ejemplo, en la página de resumen del usuario.
- La clave de acceso secreta. Se proporciona cuando crea el par de claves de acceso. Al igual que las contraseñas, no se puede recuperar en otro momento. Si ha perdido su clave de acceso secreta, debe crear un nuevo par de claves de acceso. Si ya ha alcanzado el número máximo de claves de acceso, debe eliminar uno de los pares existentes antes de poder crear otra.

Para obtener más información, consulte Restablecimiento de claves de acceso o contraseñas perdidas u olvidadas para AWS.

Las variables de la política no funcionan

- Compruebe que todas las políticas que contengan variables incluyan el siguiente número de versión en la política: `"Version": "2012-10-17"`. Sin el número de versión correcto, las variables no se sustituyen durante la evaluación. En su lugar, las variables se evalúan literalmente. Las políticas que no incluyan variables seguirán funcionando si incluye el número de versión más reciente.

  El elemento de política `Version` es diferente de la versión de una política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, por otro lado, se crea al realizar cambios en una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada. Para obtener más información sobre el elemento de política `Version`, consulte Elementos de política JSON de IAM: Version. Para obtener más información sobre las versiones de política, consulte the section called “Control de versiones de políticas de IAM”.
- Compruebe que las variables de su política estén escritas respetando mayúsculas y minúsculas. Para obtener más información, consulte Elementos de la política de IAM: variables y etiquetas.

Los cambios que realizo no están siempre visibles inmediatamente

Al ser un servicio al que se obtiene acceso a través de equipos de centros de datos de todo el mundo, IAM utiliza un modelo de computación distribuida llamado consistencia final. Cualquier cambio que realice en IAM (o en otros servicios de AWS), incluidas las etiquetas utilizadas en el control de acceso basado en atributos (ABAC), tardará en aparecer en todos los puntos de conexión posibles. Este retraso se debe en parte al tiempo que se tarda en enviar los datos de un servidor a otro, de una zona de replicación a otra y entre regiones de todo el mundo. IAM también utiliza caché para mejorar el rendimiento, pero en algunos casos esto puede agregar tiempo. Es posible que el cambio no sea visible hasta que se agoten los datos previamente almacenados.

Debe diseñar sus aplicaciones globales teniendo en cuenta estos posibles retrasos. Asegúrese de que funcionan según lo previsto, incluso cuando un cambio realizado en una ubicación no sea visible inmediatamente en otra. Estos cambios incluyen la creación o actualización de usuarios, grupos, roles o políticas. Le recomendamos que no incluya esos cambios de IAM en las rutas de código de gran importancia y alta disponibilidad de su aplicación. En su lugar, realice los cambios de IAM en otra rutina de inicialización o configuración que ejecute con menos frecuencia. Además, asegúrese de comprobar que los cambios se han propagado antes de que los flujos de trabajo de producción dependan de ellos.

Para obtener más información acerca del modo en que esto afecta a otros servicios de AWS, consulte los siguientes recursos:

- Amazon DynamoDB: ¿Cuál es el modelo de consistencia de Amazon DynamoDB? en Preguntas frecuentes sobre DynamoDB, y Consistencia de lectura en la guía para desarrolladores de Amazon DynamoDB.
- Amazon EC2: consistencia final de EC2 en la Referencia de la API de Amazon EC2
- Amazon EMR: la sección sobre cómo asegurar la consistencia el utilizar Amazon S3 y Amazon Elastic MapReduce para flujos de trabajo ETL en el blog Big Data de AWS.
- Amazon Redshift: administración de la consistencia de los datos en la Guía para desarrolladores de bases de datos de Amazon Redshift
- Amazon S3: Modelo de coherencia de datos de Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.

No tengo autorización para realizar la operación iam:DeleteVirtualMFADevice

Es posible que reciba el siguiente error cuando intente asignar o eliminar un dispositivo MFA virtual para usted o para otros usuarios:

```
User: arn:aws:iam::123456789012:user/Diego is not authorized to perform:
 iam:DeleteVirtualMFADevice on resource: arn:aws:iam::123456789012:mfa/Diego with an
 explicit deny
```

Esto podría ocurrir si alguien anteriormente comenzó la asignación de un dispositivo MFA virtual a un usuario en la consola de IAM y después canceló el proceso. Esto crea un dispositivo MFA virtual para el usuario en IAM, pero nunca lo asocia totalmente al usuario. Debe eliminar el dispositivo MFA virtual existente para poder crear un nuevo dispositivo MFA virtual con el mismo nombre de dispositivo.

Para solucionar este problema, un administrador no debe editar los permisos de la política. En su lugar, el administrador debe utilizar la AWS CLI o la API de AWS para eliminar el dispositivo MFA virtual existente pero sin asignar.

Para eliminar un dispositivo MFA virtual existente pero sin asignar

1. Consulte los dispositivos MFA virtuales de su cuenta.
   - AWS CLI: `aws iam list-virtual-mfa-devices`
   - API de AWS: `ListVirtualMFADevices`
2. En la respuesta, localice el ARN del dispositivo MFA virtual del usuario que está intentando corregir.
3. Elimine el dispositivo MFA virtual.
   - AWS CLI: `aws iam delete-virtual-mfa-device`
   - API de AWS: `DeleteVirtualMFADevice`

¿Cómo puedo crear usuarios de IAM de forma segura?

Si tiene empleados que requieren acceso a AWS (), puede elegir crear usuarios de IAM o utilizar IAM Identity Center para la autenticación. Si utiliza IAM, AWS recomienda crear un usuario de IAM y

comunicar las credenciales a los empleados de forma segura. Si no se encuentra físicamente junto a los empleados, utilice un flujo de trabajo seguro para comunicarles las credenciales.

Utilice el siguiente flujo de trabajo para crear un nuevo usuario en IAM de forma segura:

1. Cree un nuevo usuario mediante la AWS Management Console. Elija conceder acceso a la AWS Management Console con una contraseña generada automáticamente. Si es necesario, seleccione la casilla de verificación Users must create a new password at next sign-in (Los usuarios deben crear una nueva contraseña en el siguiente inicio de sesión). No agregue una política de permisos al usuario hasta después de que haya cambiado su contraseña.
2. Después de agregar el usuario, copie la dirección URL de inicio de sesión, el nombre de usuario y la contraseña del nuevo usuario. Para ver la contraseña, elija Show (Mostrar).
3. Envíe la contraseña a su empleado mediante un método de comunicación seguro en su empresa, como email, chat o un sistema de seguimiento de incidentes. Por separado, proporcione a sus usuarios el enlace de la consola de usuario de IAM y su nombre de usuario. Dígale al empleado que confirme si puede iniciar sesión correctamente antes de concederle permisos.
4. Una vez que el empleado lo confirme, agregue los permisos que necesita. Como práctica recomendada de seguridad, agregue una política que requiera que el usuario se autentique mediante MFA para administrar sus credenciales. Para ver una política de ejemplo, consulte AWS: permite a los usuarios de IAM autenticados por MFA administrar sus propias credenciales en la página Credenciales de seguridad.

Recursos adicionales

Los recursos relacionados siguientes pueden ayudarle a solucionar problemas cuando trabaje con AWS.

- Guía del usuario de AWS CloudTrail— Utilice AWS CloudTrail para realizar un seguimiento de un historial de llamadas a la API realizadas a AWS y almacenar esa información en archivos de registro. Esto le ayuda a determinar los usuarios y las cuentas que obtuvieron acceso a los recursos de su cuenta, cuándo se realizaron las llamadas, qué acciones se solicitaron, etc. Para obtener más información, consulte Registro de llamadas a la API de AWS STS y de IAM con AWS CloudTrail.
- AWSCentro de conocimientos: busque preguntas frecuentes y enlaces a otros recursos para ayudarle a solucionar problemas.

- Centro de asistencia de AWS: obtenga asistencia técnica.
- Centro de asistencia prémium de AWS: obtenga asistencia técnica premium.

Solución de problemas de mensajes de error de acceso denegado

Los errores de acceso denegado se muestran cuando AWS deniega explícita o implícitamente una solicitud de autorización. Se produce una denegación explícita cuando una política contiene una declaración `Deny` para la acción de AWS específica. Una denegación implícita se produce cuando no hay declaraciones `Deny` ni `Allow` aplicables. Como una política de IAM deniega una entidad principal de IAM de forma predeterminada, la política debe permitir explícitamente que la entidad principal lleve a cabo una acción. De lo contrario, la política deniega el acceso de forma implícita. Para obtener más información, consulte Diferencia entre denegaciones implícitas y explícitas.

Si varias políticas del mismo tipo de política rechazan una solicitud de autorización, AWS no especifica el número de políticas del mensaje de error de acceso denegado. Si se deniega una solicitud de autorización debido a varios tipos de políticas, AWS incluye solo uno de esos tipos de políticas en el mensaje de error.

> **Important**
>
> ¿Tiene problemas para iniciar sesión en AWS? Asegúrese de que está en la página de inicio de sesión de AWS correcta para su tipo de usuario. Si es el Usuario raíz de la cuenta de AWS (propietario de la cuenta), puede iniciar sesión en AWS con las credenciales que configuró cuando creó la Cuenta de AWS. Si es usuario de IAM, el administrador de su cuenta puede proporcionarle las credenciales que puede utilizar para iniciar sesión en AWS. Si necesita solicitar soporte técnico, no utilice el enlace de comentarios de esta página, ya que el formulario lo recibe el equipo de documentación de AWS, no AWS Support. En lugar de ello, en la página Contacte con nosotros, elija Todavía no es posible iniciar sesión en la cuenta de AWS y, a continuación, elija una de las opciones de asistencia disponibles.

Me aparece un mensaje de "acceso denegado" al realizar una solicitud a un servicio de AWS

- Verifique si el mensaje de error incluye el tipo de política responsable de denegar el acceso. Por ejemplo, si el error menciona que el acceso se deniega debido a una política de control de servicios (SCP), puede centrarse en solucionar problemas de SCP. Cuando conozca el tipo de

política, también puede verificar si faltan instrucciones de denegación o si faltan permisos en la acción específica de las políticas de ese tipo de política. Si el mensaje de error no menciona el tipo de política responsable de denegar el acceso, utilice el resto de las directrices de esta sección para solucionar más problemas.

- Compruebe que tiene el permiso de política basada en identidad para llamar a la acción y a los recursos que ha solicitado. Si hay condiciones establecidas, también debe cumplir dichas condiciones al enviar la solicitud. Para obtener más información sobre cómo consultar o modificar políticas para un usuario, grupo o rol de IAM, consulte Administración de políticas de IAM.
- Si la AWS Management Console da un mensaje diciendo que no está autorizado para llevar a cabo una acción, debe ponerse en contacto con su administrador para recibir ayuda. El administrador le proporcionó sus credenciales de inicio de sesión o enlace de inicio de sesión.

  En el siguiente ejemplo, el error se produce cuando el usuario de IAM `mateojackson` intenta utilizar la consola para consultar los detalles acerca de un recurso ficticio *`my-example-widget`*, pero no tiene los permisos ficticios `widgets:`*`GetWidget`*.

  ```
  User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
   widgets:GetWidget on resource: my-example-widget
  ```

  En este caso, Mateo debe pedirle a su administrador que actualice sus políticas para obtener acceso al recurso *`my-example-widget`* mediante la acción `widgets:`*`GetWidget`*.
- ¿Está intentando obtener acceso a un servicio que admite Políticas basadas en recursos, como por ejemplo Amazon S3, Amazon SNS o Amazon SQS? En caso afirmativo, compruebe que la política le tenga especificado como principal y le dé acceso. Si realiza una solicitud a un servicio dentro de su cuenta, las políticas basadas en identidad o basadas en recursos puede concederle permiso. Si realiza una solicitud a un servicio de una cuenta distinta, tanto las políticas basadas en identidad como las basadas en recursos deben concederle permiso. Para ver qué servicios admiten políticas basadas en recursos, consulte Servicios de AWS que funcionan con IAM.
- Si la política incluye una condición con un par clave-valor, revíselo atentamente. Entre los ejemplos se incluyen la clave de condición global `aws:RequestTag/`*`tag-key`*, la `kms:EncryptionContext:`*`encryption_context_key`* de AWS KMS y la clave de condición `ResourceTag/`*`tag-key`* compatibles con varios servicios. Asegúrese de que el nombre de la clave no coincida con varios resultados. Puesto que los nombres de la clave de condición no distinguen entre mayúsculas y minúsculas, una condición que comprueba una clave denominada `foo` coincidirá con `foo`, `Foo` o `F00`. Si su solicitud incluye varios pares clave-valor con nombres de clave que diferencian únicamente por las mayúsculas o minúsculas, su acceso puede denegarse

de forma inesperada. Para obtener más información, consulte Elementos de política JSON de IAM: Condition.

- Si tiene un límite de permisos, compruebe que la política utilizada para el límite de permisos permite la solicitud. Si las políticas basadas en identidad permiten la solicitud, pero el límite de permisos no la permite, la solicitud se deniega. Un límite de permisos controla los permisos máximos que puede tener una entidad principal de IAM (usuario o rol). Las políticas basadas en recursos no se restringen por los límites de permisos. Los límites de permisos no son comunes. Para obtener más información sobre evalúa estas políticas AWS, consulte Lógica de evaluación de políticas.
- Si va a firmar las solicitudes manualmente (sin utilizar los SDK de AWS), compruebe que haya firmado correctamente la solicitud.

Me aparece un mensaje de "acceso denegado" al realizar una solicitud con credenciales de seguridad temporales

- En primer lugar, asegúrese de que no se le deniega el acceso por un motivo no relacionado con sus credenciales temporales. Para obtener más información, consulte Me aparece un mensaje de "acceso denegado" al realizar una solicitud a un servicio de AWS.
- Compruebe que el servicio acepta credenciales de seguridad temporales, consulte Servicios de AWS que funcionan con IAM.
- Compruebe que las solicitudes se han firmado correctamente y que la solicitud tiene el formato correcto. Para obtener más información, consulte la documentación de su conjunto de herramientas o Uso de credenciales temporales con recursos de AWS.
- Compruebe que sus credenciales de seguridad temporales no hayan caducado. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.
- Compruebe que el usuario o el rol de IAM tenga los permisos adecuados. Los permisos de credenciales de seguridad temporales se obtienen de un usuario o un rol de IAM. Como resultado, los permisos se limitan a los que se conceden al rol cuyas credenciales temporales ha asumido. Para obtener más información sobre cómo se determinan los permisos de las credenciales de seguridad temporales, consulte Control de los permisos para credenciales de seguridad temporales.
- Si asume una función, su sesión de función puede verse limitada por las políticas de la sesión. Al solicitar las credenciales de seguridad temporales mediante programación utilizando AWS STS, tiene la opción de pasar las políticas de sesión administradas o insertadas. Las políticas

de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión de credenciales temporal mediante programación para una función. Puede transferir un único documento de política de sesión insertada JSON utilizando el parámetro `Policy`. Puede utilizar el parámetro `PolicyArns` para especificar hasta 10 políticas de sesión administrada. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. De manera alternativa, si el administrador o un programa personalizado le proporcionan credenciales temporales, es posible que hayan incluido una política de sesión para limitar su acceso.

- Si es un usuario federado, la sesión puede verse limitada por las políticas de la sesión. Puede convertirse en un usuario federado iniciando sesión en AWS como un usuario de IAM y, a continuación, solicitando un token de federación. Para obtener más información acerca de los usuarios federados, consulte GetFederationToken: federación a través de un agente de identidades personalizadas. Si usted o su agente de identidades pasan políticas de sesión al mismo tiempo que solicitan un token de federación, la sesión se verá limitada por esas políticas. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de su usuario de IAM y las políticas de la sesión. Para obtener más información acerca de las políticas de sesión, consulte Políticas de sesión.
- Si obtiene acceso mediante un rol a un recurso que tiene una política basada en recursos, compruebe que la política conceda permisos a dicho rol. Por ejemplo, la política siguiente permite que `MyRole` de la cuenta `111122223333` obtenga acceso a `MyBucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::MyBucket/*"]
  }]
}
```

Ejemplos de mensajes de error de acceso denegado

La mayoría de los mensajes de error de acceso denegado aparecen en el formato `User` *`user`* `is not authorized to perform` *`action`* `on` *`resource`* `because` *`context`*. En este ejemplo, el *`usuario`* es el nombre de recurso de Amazon (ARN) que no recibe acceso, la *`acción`* es la

acción de servicio que la política niega y el *recurso* es el ARN del recurso sobre el que actúa la política. El campo de *contexto* representa un contexto adicional sobre el tipo de política que explica por qué se deniega el acceso.

Cuando una política deniega explícitamente el acceso porque contiene una declaración `Deny`, AWS incluye la frase `with an explicit deny in a` *type* `policy` en el mensaje de error de acceso denegado. Cuando la política deniega implícitamente el acceso, AWS incluye la frase `because no` *type* `policy allows the` *action* `action` en el mensaje de error de acceso denegado.

> **Note**
>
> Algunos servicios de AWS no admiten este formato de mensaje de error de acceso denegado. El contenido de los mensajes de error de acceso denegado puede variar según el servicio que realiza la solicitud de autorización.

En los siguientes ejemplos se muestra el formato de los distintos tipos de mensajes de error de acceso denegado.

Acceso denegado debido a una política de control de servicios: denegación implícita

1. Compruebe si falta una instrucción `Allow` explícita para la acción en sus políticas de control de servicio (SCP). Para el siguiente ejemplo, la acción es `codecommit:ListRepositories`.
2. Actualice su política al agregar la instrucción `Allow`. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS IAM Identity Center.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
 codecommit:ListRespositories action
```

Acceso denegado debido a una política de control de servicios: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en sus políticas de control de servicio (SCP). Para el siguiente ejemplo, la acción es `codecommit:ListRepositories`.
2. Actualice su SCP al eliminar la instrucción `Deny`. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS IAM Identity Center.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service control policy
```

Acceso denegado debido a una política de punto de conexión de VPC: denegación implícita

1. Compruebe si falta la instrucción `Allow` para la acción en sus políticas de punto de conexión de nube privada virtual (VPC). Para el siguiente ejemplo, la acción es `codecommit:ListRepositories`.
2. Actualice su política de punto de conexión de VPC al agregar la instrucción `Allow`. Para obtener más información, consulte Actualizar una política de punto de conexión de VPC en la Guía de AWS PrivateLink.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no VPC endpoint policy allows the
 codecommit:ListRepositories action
```

Acceso denegado debido a una política de punto de conexión de VPC: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en sus políticas de punto de conexión de nube privada virtual (VPC). Para el siguiente ejemplo, la acción es `codedeploy:ListDeployments`.
2. Actualice su política de punto de conexión de VPC al eliminar la instrucción `Deny`. Para obtener más información, consulte Actualizar una política de punto de conexión de VPC en la Guía de AWS PrivateLink.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a VPC endpoint policy
```

Acceso denegado debido a un límite de permisos: denegación implícita

1. Compruebe si falta una instrucción `Allow` para la acción en su límite de permisos. Para el siguiente ejemplo, la acción es `codedeploy:ListDeployments`.

2. Actualice su límite de permisos al agregar la instrucción `Allow` a su política de IAM. Para obtener más información, consulte Límites de permisos para las entidades de IAM y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* because no permissions boundary allows the
 codedeploy:ListDeployments action
```

Acceso denegado debido a un límite de permisos: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en su límite de permisos. Para el siguiente ejemplo, la acción es `sagemaker:ListModels`.
2. Actualice su límite de permisos al eliminar la instrucción `Deny` de su política de IAM. Para obtener más información, consulte Límites de permisos para las entidades de IAM y Edición de políticas de IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sagemaker:ListModels with an explicit deny in a permissions boundary
```

Acceso denegado debido a políticas de sesión: denegación implícita

1. Compruebe si falta una instrucción `Allow` para la acción en sus políticas de sesión. Para el siguiente ejemplo, la acción es `codecommit:ListRepositories`.
2. Actualice su política de sesión al agregar la instrucción `Allow`. Para obtener más información, consulte Políticas de sesión y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no session policy allows the
 codecommit:ListRepositories action
```

Acceso denegado debido a políticas de sesión: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en sus políticas de sesión. Para el siguiente ejemplo, la acción es `codedeploy:ListDeployments`.

2. Actualice su política de sesión al eliminar la instrucción `Deny`. Para obtener más información, consulte Políticas de sesión y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions policy
```

Acceso denegado debido a políticas basadas en recursos: denegación implícita

1. Compruebe si falta una instrucción `Allow` para la acción en su política basada en recursos. Para el siguiente ejemplo, la acción es `secretsmanager:GetSecretValue`.
2. Actualice su política al agregar la instrucción `Allow`. Para obtener más información, consulte Políticas basadas en recursos y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue because no resource-based policy allows the
 secretsmanager:GetSecretValue action
```

Acceso denegado debido a políticas basadas en recursos: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en su política basada en recursos. Para el siguiente ejemplo, la acción es `secretsmanager:GetSecretValue`.
2. Actualice su política al eliminar la instrucción `Deny`. Para obtener más información, consulte Políticas basadas en recursos y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
secretsmanager:GetSecretValue on resource: arn:aws:secretsmanager:us-
east-1:123456789012:secret:* with an explicit deny in a resource-based policy
```

Acceso denegado debido a políticas de confianza de rol: denegación implícita

1. Compruebe si falta una instrucción `Allow` para la acción en su política de confianza de rol. Para el siguiente ejemplo, la acción es `sts:AssumeRole`.
2. Actualice su política al agregar la instrucción `Allow`. Para obtener más información, consulte Políticas basadas en recursos y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
sts:AssumeRole because no role trust policy allows the sts:AssumeRole action
```

Acceso denegado debido a políticas de confianza de rol: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en su política de confianza de rol. Para el siguiente ejemplo, la acción es `sts:AssumeRole`.
2. Actualice su política al eliminar la instrucción `Deny`. Para obtener más información, consulte Políticas basadas en recursos y Edición de políticas de IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
sts:AssumeRole with an explicit deny in the role trust policy
```

Acceso denegado debido a políticas basadas en identidad: denegación implícita

1. Compruebe si falta una instrucción `Allow` para la acción en las políticas basadas en identidad asociadas a la identidad. Para el siguiente ejemplo, la acción es `codecommit:ListRepositories` y está asociada al usuario `JohnDoe`.
2. Actualice su política al agregar la instrucción `Allow`. Para obtener más información, consulte Políticas basadas en identidad y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no identity-based policy allows the
 codecommit:ListRepositories action
```

Acceso denegado debido a políticas basadas en identidad: denegación explícita

1. Compruebe si hay una instrucción `Deny` explícita para la acción en las políticas basadas en identidad asociadas a la identidad. Para el siguiente ejemplo, la acción es `codedeploy:ListDeployments` y está asociada al usuario `JohnDoe`.
2. Actualice su política al eliminar la instrucción `Deny`. Para obtener más información, consulte Políticas basadas en identidad y Edición de políticas de IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
```

```
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in an identity-based policy
```

Access denied when a VPC request fails due to another policy (Acceso denegado cuando una solicitud de VPC falla debido a otra política)

1. Compruebe si hay una instrucción `Deny` explícita para la acción en sus políticas de control de servicio (SCP). Para el siguiente ejemplo, la acción es `SNS:Publish`.
2. Actualice su SCP al eliminar la instrucción `Deny`. Para obtener más información, consulte Actualización de SCP en la Guía del usuario de AWS IAM Identity Center.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
 authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
 policy
```

Solución de problemas de políticas de IAM

Una política es una entidad en AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando una entidad principal, como un usuario, realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. Las políticas se almacenan en AWS como documentos JSON que se asocian a entidades principales como políticas basadas en identidad o a recursos como políticas basadas en recursos. Puede asociar una política basada en la identidad a una entidad principal (o identidad), como un grupo, usuario o rol de IAM. Las políticas basadas en la identidad incluyen políticas administradas por AWS, políticas administradas por el cliente y políticas insertadas. Puede crear y editar políticas administradas por el cliente en la AWS Management Console utilizando las opciones Visual y JSON del editor. Cuando consulte una política en la AWS Management Console, podrá ver un resumen de los permisos concedidos por dicha política. Puede utilizar el editor visual y los resúmenes de políticas para ayudarle a diagnosticar y corregir los errores comunes encontrados al administrar las políticas de IAM.

Tenga en cuenta que todas las políticas de IAM se almacenan mediante la sintaxis que comienza con las reglas de JavaScript Object Notation (JSON). No tiene que entender esta sintaxis para crear o administrar sus políticas. Puede crear y editar una política con el editor visual de la AWS

Management Console. Para obtener más información sobre la sintaxis JSON en las políticas de IAM, consulte Gramática del lenguaje de la política JSON de IAM .

Resolución de problemas de temas relacionados con las políticas de IAM

- Solución de problemas con el editor visual
  - Reestructuración de políticas
  - Selección del ARN de un recurso en el editor visual
  - Denegación de permisos en el editor visual
  - Especificación de varios servicios en el editor visual
  - Reducción del tamaño de su política en el editor visual
  - Corrección de servicios, acciones o tipos de recurso no reconocidos en el editor visual
- Solución de problemas mediante resúmenes de políticas
  - Resumen de la política que falta
  - El resumen de la política incluye servicios, acciones o tipos de recurso no reconocidos
  - El servicio no admite resúmenes de políticas de IAM
  - Mi política no concede los permisos esperados
- Solución de problemas de administración de políticas
  - Asociar o desvincular una política en una cuenta de IAM
  - Cambio de las políticas para las identidades de IAM en función de su actividad
- Solución de problemas con documentos de políticas JSON
  - Validar sus políticas
  - No tengo permisos para validación de políticas en el editor de JSON
  - Más de un objeto de política JSON
  - Más de un elemento de instrucción JSON
  - Más de un elemento de efecto, acción o recurso en un elemento de instrucción JSON
  - Falta el elemento de versión JSON

Solución de problemas con el editor visual

Al crear o editar una política administrada por el cliente, puede utilizar la información del editor Visual como ayuda para solucionar errores de la política. Para ver un ejemplo de cómo utilizar el editor para crear una política, consulte the section called “Control del acceso a identidades”.

Reestructuración de políticas

Cuando crea una política, AWS valida los procesos y transforma la política antes de almacenarla. Cuando AWS devuelve la política en respuesta a una consulta del usuario o la muestra en la consola, AWS vuelve a transformar la política a un formato legible sin cambiar los permisos concedidos por la política. Esto puede provocar diferencias en lo que aparece en el editor visual de la política o en la pestaña JSON: se puede añadir, reordenar o eliminar bloques de permisos del editor virtual y optimizar el contenido de un bloque. En la pestaña JSON puede que se hayan eliminado los espacios en blanco y que se hayan reordenado los elementos de los mapas JSON. Además, los ID de Cuenta de AWS de los elementos principales se pueden reemplazar por el ARN del Usuario raíz de la cuenta de AWS. Debido a estos posibles cambios, no debe comparar los documentos de política JSON como cadenas.

Al crear una política administrada por el cliente en la AWS Management Console, puede decidir trabajar únicamente en el editor JSON. Si nunca realiza cambios en el editor Visual y selecciona Siguiente en el editor JSON, es menos probable que se reestructure la política. No obstante, si crea una política y utiliza la opción Visual del editor para realizar modificaciones, o si selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política con el fin de optimizar su aspecto en el editor visual.

Esta reestructuración se produce únicamente en su sesión de edición y no se guarda automáticamente.

Si la política se ha reestructurado en su sesión de edición, IAM determina si debe guardar la reestructuración en función de las siguientes situaciones:

Uso de esta opción del editor	Si edita la política	Y luego seleccione Siguiente en esta pestaña	Cuando elige Save changes (Guardar cambios)
Visual	Editado	Visual	La política se reestructura
Visual	Editado	JSON	La política se reestructura
Visual	No editado	Visual	La política se reestructura

Uso de esta opción del editor	Si edita la política	Y luego seleccione Siguiente en esta pestaña	Cuando elige Save changes (Guardar cambios)
JSON	Editado	Visual	La política se reestructura
JSON	Editado	JSON	La estructura de la política no cambia
JSON	No editado	JSON	La estructura de la política no cambia

IAM podría reestructurar políticas complejas o políticas que tengan bloques de permisos o instrucciones que permitan varios servicios, tipos de recursos o claves de condición.

Selección del ARN de un recurso en el editor visual

Cuando crea o edita una política con el editor visual, primero debe elegir un servicio y, a continuación, elegir las acciones de dicho servicio. Si el servicio y las acciones que ha seleccionado permiten elegir recursos específicos, el editor visual muestra una lista de los tipos de recursos admitidos. A continuación, puede elegir Add ARN (Agregar ARN) para proporcionar los detalles del recurso. Puede elegir entre las siguientes opciones para añadir un ARN para un tipo de recurso.

- Utilizar el constructor de ARN – Según el tipo de recurso, puede que vea campos diferentes para crear el ARN. También puede elegir Any (Cualquiera) para proporcionar permisos para cualquier valor de la opción especificada. Por ejemplo, si ha seleccionado el grupo de nivel de acceso Read (Lectura) de Amazon EC2, las acciones de la política permiten el tipo de recurso `instance`. Debe proporcionar los valores de Region (Región), Account (Cuenta) e InstanceId (ID de instancia) para el recurso. Si proporciona el ID de cuenta pero elige Any (Cualquiera) para el ID de región e instancia, la política concede permisos sobre todas las instancias de su cuenta.
- Escribir o pegar el ARN: puede especificar los recursos por su Nombre de recurso de Amazon (ARN). Puede incluir un carácter comodín (*****) en cualquier campo del ARN (entre cada par de dos puntos). Para obtener más información, consulte Elementos de política JSON de IAM: Resource.

Denegación de permisos en el editor visual

De forma predeterminada, la política que crea utilizando el editor visual permite las acciones que usted elija. Para denegar las acciones elegidas, seleccione Switch to deny permissions (Cambiar a denegar permisos). Dado que las solicitudes se deniegan de forma predeterminada, por motivos de seguridad recomendamos que permita solo aquellas acciones y recursos a los que un usuario necesita acceso. Debe crear una instrucción para denegar permisos únicamente si desea invalidar separadamente un permiso que otra instrucción o política permita. Le recomendamos que limite al mínimo el número de operaciones de denegación de permisos, ya que pueden aumentar la dificultad de solucionar problemas con los permisos. Para obtener más información acerca de cómo IAM evalúa la lógica de las políticas, consulte Lógica de evaluación de políticas.

> **Note**
>
> De forma predeterminada, solo el Usuario raíz de la cuenta de AWS tiene acceso a todos los recursos de esa cuenta. Por lo tanto, si no ha iniciado sesión como usuario raíz, debe disponer de permisos concedidos por una política.

Especificación de varios servicios en el editor visual

Cuando utiliza el editor visual para crear una política, puede seleccionar solo un servicio a la vez. Se trata de una práctica recomendada, ya que el editor visual le permite elegir acciones para ese único servicio. Seguidamente, deberá elegir los recursos admitidos por dicho servicio y las acciones seleccionadas. Esto facilita la creación de la política y la resolución de problemas.

Si está familiarizado con la sintaxis JSON, también puede utilizar un carácter comodín (*) para especificar manualmente varios servicios. Por ejemplo, escriba **`Code*`** para proporcionar permisos para todos los servicios que comiencen por `Code`, como `CodeBuild` y `CodeCommit`. Sin embargo, debe especificar los ARN de las acciones y los recursos para completar su política. Además, al guardar la política, esta podría reestructurarse para incluir cada servicio en un bloque de permisos distinto.

O bien, si desea utilizar sintaxis JSON (por ejemplo, caracteres comodín) para los servicios, cree, edite y guarde la política utilizando la opción JSON del editor.

Reducción del tamaño de su política en el editor visual

Cuando se utiliza el editor visual para crear una política, IAM crea un documento JSON para almacenar la política. Puede ver este documento cambiando a la opción JSON del editor. Si este documento JSON supera el límite de tamaño de una política, el editor visual muestra un mensaje de error y no le permite revisar y guardar su política. Para ver la limitación de IAM del tamaño de una política administrada, consulte Límites de caracteres de IAM y STS.

Para reducir el tamaño de su política en el editor visual, edite la política o mueva bloques de permiso a otra política. El mensaje de error incluye el número de caracteres que contiene su documento de política, y puede utilizar esta información para ayudarle a reducir el tamaño de la política.

Corrección de servicios, acciones o tipos de recurso no reconocidos en el editor visual

Cuando crea o edita una política en el editor visual, es posible que aparezca una advertencia que indique que la política incluye un servicio, acción o tipo de recurso no reconocido.

> **Note**
>
> IAM comprueba los nombres de los servicios, las acciones y los tipos de recurso para los servicios que admiten resúmenes de políticas. Sin embargo, el resumen de política podría incluir algún valor de recurso o condición que no exista. Pruebe siempre las políticas con el simulador de políticas.

Si la política incluye servicios, acciones o tipos de recurso no reconocidos, se debe a uno de los siguientes errores:

- Servicio de vista previa – Los servicios en vista previa no admiten el editor visual. Si está participando en la vista previa, puede omitir la advertencia y continuar, aunque debe especificar manualmente los ARN de las acciones y los recursos para completar la política. También puede elegir la opción JSON del editor para escribir o pegar un documento de política de JSON.
- Servicio personalizado – Los servicios personalizados no admiten el editor visual. Si usa un servicio personalizado, puede omitir la advertencia y continuar, aunque debe especificar manualmente los ARN de las acciones y los recursos para completar la política. También puede elegir la opción JSON del editor para escribir o pegar un documento de política de JSON.
- El servicio no admite el editor visual – Si su política incluye un servicio disponible de manera general (GA) que no admite el editor visual, puede omitir la advertencia y continuar, aunque

debe especificar manualmente los ARN de las acciones y los recursos para completar la política. También puede elegir la opción JSON del editor para escribir o pegar un documento de política de JSON.

Los servicios disponibles en general son servicios puestos a disposición del público y no son servicios de vista previa ni personalizados. Si un servicio no reconocido está disponible con carácter general y el nombre está escrito correctamente, entonces el servicio no admite el editor visual. Para saber cómo solicitar que se admita el editor visual o el resumen de políticas para un servicio disponible con carácter general, consulte El servicio no admite resúmenes de políticas de IAM.

- La acción no admite el editor visual – Si su política incluye un servicio admitido con una acción no admitida, puede omitir la advertencia y continuar, aunque debe especificar manualmente los ARN de las acciones y los recursos para completar la política. También puede elegir la opción JSON del editor para escribir o pegar un documento de política de JSON.

  Si la política incluye un servicio admitido con una acción no admitida, entonces el servicio no admite el editor visual en su totalidad. Para saber cómo solicitar que se admita el editor visual o el resumen de políticas para un servicio disponible con carácter general, consulte El servicio no admite resúmenes de políticas de IAM.
- El tipo de recurso no admite el editor visual – Si la política incluye una acción admitida con un tipo de recurso no admitido, puede omitir la advertencia y continuar. Sin embargo, IAM no puede confirmar que ha incluido recursos para todas las acciones seleccionadas y es posible que vea advertencias adicionales.
- Error tipográfico – Cuando especifica manualmente un servicio, acción o recurso en el editor visual, puede crear una política que incluya un error tipográfico. Para evitar esto, utilice el editor visual seleccionando servicios y acciones de la lista y, a continuación, rellene la sección de recursos de acuerdo con las instrucciones. Sin embargo, si un servicio no admite plenamente el editor visual, es posible que tenga que especificar manualmente algunas partes de la política.

  Si está seguro de que la política no contiene ninguno de los errores previamente mencionados, es posible que la política tenga un error tipográfico. Compruebe que los nombres del servicio, acción y tipo de recurso estén escritos correctamente. Por ejemplo, puede utilizar `s2` en lugar de `s3` y `ListMyBuckets` en lugar de `ListAllMyBuckets`. Otro error tipográfico habitual en las acciones es la inclusión de texto innecesario en el ARN, como `arn:aws:s3: : :*` o la falta de los dos puntos en las acciones, como `iam.CreateUser`. Puede evaluar una política que es posible que incluya erratas si selecciona Siguiente para revisar el resumen de la política y confirmar que la política proporciona los permisos deseados.

Solución de problemas mediante resúmenes de políticas

Puede diagnosticar y resolver problemas relacionados con los resúmenes de políticas.

Resumen de la política que falta

La consola de IAM incluye tablas de resumen de política que describen el nivel de acceso, los recursos y las condiciones permitidos o rechazados para cada servicio de una política. Las políticas se resumen en tres tablas: el resumen de política, el resumen de servicio y el resumen de acción. La tabla resumen de política incluye una lista de servicios y resúmenes de los permisos que la política elegida define. Puede ver el resumen de política de cualquier política que esté asociada a una entidad en la página Detalles de la política correspondiente a esa política. Puede ver el resumen de política de las políticas administradas en la página Policies (Políticas). Si AWS no puede representar un resumen de una política, verá el documento de la política JSON en lugar del resumen y recibirá el siguiente error:

No se puede generar un resumen para esta política. Pero puede ver o editar el documento de la política JSON.

Si la política no incluye un resumen, se debe a que se ha producido uno de los siguientes errores:

- Elemento de política no admitido – IAM no es compatible con la generación de resúmenes de políticas que incluyan uno de los siguientes elementos de política:
  - `Principal`
  - `NotPrincipal`
  - `NotResource`
- Sin permisos de política – Si una política no proporciona permisos eficaces, no se podrá generar el resumen de política. Por ejemplo, si una política incluye una única instrucción con el elemento `"NotAction": "*"`, entonces concede el acceso a todas las acciones excepto "todas las acciones" (*). Lo que significa que no otorga acceso `Deny` o `Allow` a nada.

> **Note**
>
> Debe ir con cuidado al utilizar estos elementos de política como `NotPrincipal`, `NotAction` y `NotResource`. Para obtener más información sobre el uso de elementos de política, consulte Referencia de los elementos de las políticas de JSON de IAM.

Puede crear una política que no proporcione permisos eficaces si suministra servicios y recursos no coincidentes. Esto puede ocurrir si especifica acciones de un servicio y los recursos de otro servicio. En ese caso, el resumen de la política no aparece. La única indicación de que existe un problema es que la columna de recursos en el resumen puede incluir un recurso de otro servicio. Si esta columna incluye un recurso no coincidente, debe comprobar que la política no tiene errores. Para comprender mejor las políticas, pruébelas siempre con el simulador de políticas.

El resumen de la política incluye servicios, acciones o tipos de recurso no reconocidos

En la consola de IAM, si un resumen de política incluye un símbolo de advertencia (), es posible que la política incluya un servicio, acción o tipo de recurso no reconocido. Para obtener más información acerca de las advertencias dentro de un resumen de política, consulte Resumen de política (lista de servicios).

> **Note**
>
> IAM comprueba los nombres de los servicios, las acciones y los tipos de recurso para los servicios que admiten resúmenes de políticas. Sin embargo, el resumen de política podría incluir algún valor de recurso o condición que no exista. Pruebe siempre las políticas con el simulador de políticas.

Si la política incluye servicios, acciones o tipos de recurso no reconocidos, se debe a uno de los siguientes errores:

- Servicio de vista previa – Los servicios en vista previa no admiten resúmenes de política.
- Servicio personalizado – Los servicios personalizados no admiten resúmenes de políticas.
- El servicio no admite resúmenes – Si su política incluye un servicio disponible de manera general (GA) que no admite resúmenes de políticas, el servicio se incluye en la sección Servicios no reconocidos de la tabla de resumen de la política. Los servicios disponibles en general son servicios puestos a disposición del público y no son servicios de vista previa ni personalizados. Si un servicio no reconocido está disponible en general y el nombre está escrito correctamente, entonces el servicio no admite resúmenes de políticas de IAM. Para aprender a solicitar soporte

de resumen de políticas para un servicio de disponibilidad general, consulte El servicio no admite resúmenes de políticas de IAM.

- La acción no admite resúmenes – Si la política incluye un servicio admitido con una acción no admitida, la acción se incluye en la sección Acciones no reconocidas de la tabla de resumen de servicio. Para obtener más información acerca de las advertencias dentro de un resumen de servicio, consulte Resumen de servicios (lista de acciones).
- El tipo de recurso no admite resúmenes – Si la política incluye una acción admitida con un tipo de recurso no admitido, el recurso se incluye en la sección Tipos de recursos no reconocidos de la tabla de resumen de servicio. Para obtener más información acerca de las advertencias dentro de un resumen de servicio, consulte Resumen de servicios (lista de acciones).
- Tipografía – AWS comprueba que el JSON es sintácticamente correcto y que la política no incluye errores tipográficos u otros errores como parte de la validación de políticas.

> **Note**
>
> Como práctica recomendada, le sugerimos utilizar IAM Access Analyzer para validar sus políticas de IAM y así garantizar la seguridad y funcionalidad de los permisos. Recomendamos que abra las políticas existentes y revise y resuelva cualquier recomendación de validación de políticas.

El servicio no admite resúmenes de políticas de IAM

Cuando los resúmenes de políticas de IAM o el editor visual no reconocen un servicio o acción disponible de manera en general (GA), es posible que el servicio no admita dichas características. Los servicios disponibles en general son servicios puestos a disposición del público y no son servicios de vista previa ni personalizados. Si un servicio no reconocido está disponible con carácter general y el nombre está escrito correctamente, entonces el servicio no admite estas características. Si la política incluye un servicio admitido con una acción no admitida, entonces el servicio no admitirá totalmente los resúmenes de políticas de IAM.

Para solicitar que un servicio añada soporte para el resumen de políticas de IAM o el editor visual

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Localice la política que incluye el servicio no admitido:

- Si la política es una política administrada, seleccione Policies (Políticas) en el panel de navegación. En la lista de políticas, seleccione el nombre de la política que desea ver.
- Si la política es una política en línea asociada al usuario, seleccione Users (Usuarios) en el panel de navegación. En la lista de usuarios, seleccione el nombre del usuario cuya política desea ver. En la tabla de políticas para el usuario, expanda el encabezado del resumen de política que desea ver.

3. A la izquierda del pie de página de la AWS Management Console, seleccione Feedback (Valoración). En el cuadro Comentarios para IAM, escriba **`I request that the <ServiceName> service add support for IAM policy summaries and the visual editor`**. Si quiere que más de un servicio admita los resúmenes, escriba **`I request that the <ServiceName1>, <ServiceName2>, and <ServiceName3> services add support for IAM policy summaries and the visual editor`**.

Para solicitar que un servicio añada soporte para el resumen de política de IAM para una acción faltante

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Localice la política que incluye el servicio no admitido:
   - Si la política es una política administrada, seleccione Policies (Políticas) en el panel de navegación. En la lista de políticas, seleccione el nombre de la política que desea ver.
   - Si la política es una política en línea asociada al usuario, seleccione Users (Usuarios) en el panel de navegación. En la lista de usuarios, seleccione el nombre del usuario cuya política desea ver. En la tabla de políticas para el usuario, seleccione el nombre de la política que quiere ver para expandir el resumen de política.
3. En el resumen de política, elija el nombre del servicio que incluye una acción no compatible.
4. A la izquierda del pie de página de la AWS Management Console, seleccione Feedback (Valoración). En el cuadro Comentarios para IAM, escriba **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName> action`**. Si quiere notificar más de una acción no admitida, escriba **`I request that the <ServiceName> service add IAM policy summary and the visual editor support for the <ActionName1>, <ActionName2>, and <ActionName3> actions`**.

Para solicitar que un servicio diferente incluya las acciones que faltan, repita los últimos tres pasos.

Mi política no concede los permisos esperados

Para asignar permisos a un usuario, grupo, rol o recurso, puede crear una política, que es un documento que define permisos. El documento de políticas incluye los siguientes elementos:

- Efecto – Indica si la política permite o deniega acceso.
- Acción – La lista de acciones permitidas o denegadas por la política
- Recurso – La lista de recursos en los que pueden producirse las acciones
- Condición (opcional) – Las circunstancias en las que la política concede permisos

Para obtener más información sobre este y otros elementos de políticas, consulte Referencia de los elementos de las políticas de JSON de IAM.

Para conceder acceso, la política debe definir una acción con un recurso admitido. Si la política también incluye una condición, esta debe contener una clave de condición global o ser aplicable a la acción. Para saber qué recursos admite una acción, consulte la documentación de AWS de su servicio. Para saber qué condiciones admite una acción, consulte Acciones, recursos y claves de condición de los servicios de AWS.

Para saber si su política define una acción, recurso o condición que no concede permisos, puede ver el resumen de políticas de su política en la consola de IAM, a la que puede obtener acceso desde https://console.aws.amazon.com/iam/. Puede utilizar resúmenes de políticas para identificar y corregir problemas en su política.

Hay varias razones por las que un elemento puede no estar concediendo permisos a pesar de que se define en la política de IAM:

- Se ha definido una acción sin un recurso aplicable
- Se ha definido un recurso sin una acción aplicable
- Se ha definido una condición sin una acción aplicable

Para ver ejemplos de resúmenes de políticas que incluyen advertencias, consulte the section called “Resumen de política (lista de servicios)”.

Se ha definido una acción sin un recurso aplicable

La política que aparece a continuación define todas las acciones `ec2:Describe*` y define un recurso específico. Ninguna de las acciones `ec2:Describe` se conceden porque ninguna de ellas admite permisos de nivel de recursos. Los permisos de nivel de recursos implican que la acción admite recursos que utilicen los ARN en el elemento `Resource` de la política. Si una acción no admite permisos de nivel de recursos, dicha instrucción en la política debe utilizar un comodín (*) en el elemento `Resource`. Para saber qué servicios admiten permisos en el nivel de recursos, consulte Servicios de AWS que funcionan con IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID:instance/*"
    }]
}
```

Esta política no concede permisos y la política de resumen incluye los siguientes errores:

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Para solucionar esta política, debe utilizar * en el elemento `Resource`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
    }]
}
```

Se ha definido un recurso sin una acción aplicable

La política de recursos que aparece a continuación define un bucket de Amazon S3, pero no incluye una acción de S3 que pueda llevarse a cabo en dicho recurso. Esta política también concede acceso completo a todas las acciones de Amazon CloudFront.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Esta política concede permisos para todas las acciones de CloudFront. Sin embargo, como la política define el recurso de S3 `examplebucket` sin definir ninguna acción de S3, el resumen de política incluye la siguiente advertencia:

```
This policy defines some actions, resources, or conditions that do not
provide permissions. To grant access, policies must have an action that has
an applicable resource or condition.
```

Para solucionar esta política con el fin de proporcionar permisos del bucket de S3, debe definir acciones de S3 que puedan llevarse a cabo en un recurso de bucket.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "cloudfront:*",
            "s3:CreateBucket",
            "s3:ListBucket*",
            "s3:PutBucket*",
            "s3:GetBucket*"
        ],
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Otra opción para corregir esta política de modo que proporcione solo permisos de CloudFront es eliminar el recurso de S3.

Se ha definido una condición sin una acción aplicable

La política siguiente define dos acciones de Amazon S3 para todos los recursos de S3, si el prefijo de S3 es `custom` y el ID de la versión es `1234`. Sin embargo, la clave de condición `s3:VersionId` se utiliza para etiquetar la versión de los objetos y no es admitida por las acciones definidas por el bucket. Para saber qué condiciones admite una acción, consulte Acciones, recursos y claves de condición de los servicios de AWS y haga clic en el enlace que dirige a la documentación de servicios de claves de condición.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Esta política concede permisos para las acciones `s3:ListBucketVersions` y `s3:ListBucket` si el nombre del bucket incluye el `custom` prefijo. Sin embargo, como la condición `s3:VersionId` no es compatible con ninguna de las acciones definidas, el resumen de políticas incluye el siguiente error:

```
This policy does not grant any permissions. To grant access, policies must
have an action that has an applicable resource or condition.
```

Para corregir esta política y que utilice el etiquetado de versiones de objetos de S3, debe definir una acción de S3 admita la clave de condición `s3:VersionId`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": [
                        "custom"
                    ],
                    "s3:VersionId": [
                        "1234"
                    ]
                }
            }
        }
    ]
}
```

Esta política concede permisos para cada acción y condición de sí misma. Sin embargo, la política sigue sin conceder permisos porque no hay un caso en el que una única acción coincida con ambas condiciones. En su lugar, debe crear dos instrucciones independientes, y cada una debe incluir únicamente acciones con las condiciones a las que son aplicables.

Para corregir esta política, cree dos instrucciones. La primera instrucción incluye las acciones que admiten la condición `s3:prefix` y la segunda incluye las acciones que admiten la condición `s3:VersionId`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucketVersions",
                "s3:ListBucket"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:prefix": "custom"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObjectVersion",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:VersionId": "1234"
                }
            }
        }
    ]
}
```

Solución de problemas de administración de políticas

Puede diagnosticar y resolver problemas relacionados con la administración de políticas.

Asociar o desvincular una política en una cuenta de IAM

Algunas políticas administradas de AWS están vinculadas a un servicio. Estas políticas se utilizan únicamente con un rol vinculado a un servicio de dicho servicio. En la consola de IAM, cuando se visualiza la página Detalles de la política de una política, dicha página incluye un banner para indicar que la política está vinculada a un servicio. No puede adjuntar esta política a un usuario, grupo o rol en IAM. Si crea un rol vinculado al servicio para este servicio, esta política se adjunta automáticamente al nuevo rol. Dado que la política es necesaria, no puede separar la política del rol vinculado al servicio.

Cambio de las políticas para las identidades de IAM en función de su actividad

Puede actualizar políticas para sus identidades de IAM (usuarios, grupos y roles) en función de su actividad. Para ello, vea los eventos de su cuenta en el Historial de eventos de CloudTrail. Los registros de eventos de CloudTrail de incluyen información detallada que usted puede utilizar para cambiar los permisos de la política. Puede ocurrir que un usuario o función estén intentando realizar una acción en AWS y dicha solicitud se deniegue. En ese caso, puede tener en cuenta si el usuario o la función deben tener permiso para ejecutar la acción. En tal caso, puede añadir a su política la acción e incluso el ARN del recurso al que ha intentado el acceso. Si el usuario o rol tiene permisos que no utiliza, también puede considerar la posibilidad de eliminarlos permisos de su política. Asegúrese de que sus políticas concedan el privilegio mínimo que es necesario para realizar únicamente las acciones necesarias. Para obtener más información acerca del uso de CloudTrail, consulte Ver eventos de CloudTrail en la consola de CloudTrail en la Guía del usuario de AWS CloudTrail.

Solución de problemas con documentos de políticas JSON

Puede diagnosticar y resolver problemas relacionados con documentos de políticas JSON.

Validar sus políticas

Al crear o editar una política JSON, IAM puede realizar la validación de políticas para ayudarle a crear una política eficaz. IAM identifica errores de sintaxis JSON, mientras que IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más las políticas. Para obtener más información acerca la validación de políticas, consulte Validación de políticas de IAM. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer.

No tengo permisos para validación de políticas en el editor de JSON

En AWS Management Console, puede recibir el siguiente error si no tiene permisos para ver los resultados de validación de políticas de IAM Access Analyzer:

```
You need permissions. You do not have the permissions required to perform this operation. Ask your administrator to add permissions.
```

Para solucionar este error, pida al administrador que añada el permiso `access-analyzer:ValidatePolicy` para usted.

Más de un objeto de política JSON

Una política de IAM debe constar de uno y solo un objeto JSON. Los objetos se indican incluyéndolos en llaves { }. Aunque puede anidar otros objetos dentro de un objeto JSON añadiendo llaves ({}) adicionales en el par exterior, una política solo puede contener un par exterior de llaves { }. El siguiente ejemplo es incorrecto porque contiene dos objetos en la parte superior (indicados en *rojo*):

```
{
      "Version": "2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
     "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
     }
   }
```

Sin embargo, podría satisfacer la intención del ejemplo anterior con el uso de la gramática de políticas correcta. En lugar de incluir dos objetos de política completos, cada uno con su propio elemento `Statement`, puede combinar los dos bloques en un único elemento `Statement`. El elemento `Statement` tiene una matriz de dos objetos como valor, tal y como se muestra en el ejemplo siguiente (destacado en negrita):

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":" *"
        },
        {
          "Effect": "Allow",
```

```
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-bucket/*"
        }
      ]
    }
```

Más de un elemento de instrucción JSON

A simple vista, este error podría parecer una variante del error de la sección anterior. Sin embargo, es un tipo de error diferente desde el punto de vista sintáctico. En el siguiente ejemplo solo hay un objeto de política tal como indica el único par de llaves { } en el nivel superior. Sin embargo, ese objeto contiene dos elementos `Statement` en su interior.

Una política de IAM debe contener solo un elemento `Statement`, que consta del nombre (`Statement`) que aparece a la izquierda de un carácter de punto y coma, seguido de su valor a la derecha. El valor de un elemento `Statement` debe ser un objeto, identificado por llaves { }, que contiene un elemento `Effect`, un elemento `Action` y un elemento `Resource`. El siguiente ejemplo es incorrecto porque contiene dos elementos `Statement` en el objeto de política (destacado en *rojo*):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
    }
```

Un objeto de valor puede ser un conjunto de varios objetos de valor. Para solucionar este problema, combine los dos elementos `Statement` en un elemento con una matriz de objetos, tal y como se muestra en el ejemplo siguiente (destacado en negrita):

```
{
      "Version": "2012-10-17",
      "Statement": [    {
```

```
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource":"*"
      },
      {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
   ]
  }
```

El valor del elemento `Statement` es una matriz de objetos. La matriz del ejemplo se compone de dos objetos, cada uno de los cuales es un valor correcto para un elemento `Statement`. Cada objeto de la matriz está separado por comas.

Más de un elemento de efecto, acción o recurso en un elemento de instrucción JSON

En el lado del valor del par nombre/valor `Statement`, el objeto debe constar de un único elemento `Effect`, un elemento `Action` y un elemento `Resource`. La siguiente política es incorrecta porque contiene dos elementos `Effect` en el objeto de valor del `Statement`:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Effect": "Allow",
        "Action": "ec2:* ",
        "Resource": "*"
      }
  }
```

> **Note**
>
> El motor de políticas no permite este tipo de errores en políticas nuevas o editadas. Sin embargo, el motor de políticas continúa aceptando las políticas que se guardaron antes de que se actualizara el motor. El comportamiento de las políticas existentes ante este error es el siguiente:
>
> - Varios elementos `Effect`: solo se considera el último elemento `Effect`. Los demás se omiten.

- Varios elementos `Action`: todos los elementos `Action` se combinan internamente y se tratan como una lista individual.
- Varios elementos `Resource`: todos los elementos `Resource` se combinan internamente y se tratan como una lista individual.

El motor de políticas no le permite guardar ninguna política con errores sintácticos. Debe corregir los errores en la política antes de poder guardarla. Le recomendamos que revise cualquier recomendación de validación de políticas correcta para sus políticas.

En todos los casos, la solución es eliminar el elemento adicional incorrecto. Para los elementos `Effect`, está claro: si desea que en el ejemplo anterior se denieguen los permisos para las instancias de Amazon EC2, deberá eliminar la línea `"Effect": "Allow",` de la política tal y como se indica a continuación:

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Deny",
        "Action": "ec2:* ",
        "Resource": "*"
      }
    }
```

Sin embargo, si el elemento duplicado es `Action` o `Resource`, la solución puede complicarse. Puede disponer de varias acciones para las que desea permitir (o denegar) permisos o puede que quiera controlar el acceso a varios recursos. El siguiente ejemplo es incorrecto porque tiene varios elementos `Resource` (destacados en *rojo*):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket",
        "Resource": "arn:aws:s3:::my-bucket/*"
      }
    }
```

Cada uno de los elementos necesarios en un objeto de valor del elemento `Statement` puede estar presente solo una vez. La solución consiste en colocar cada valor en una matriz. El siguiente ejemplo lo muestra al convertir los dos elementos de recursos independientes en un elemento `Resource` con una matriz como objeto de valor (destacado en negrita):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-bucket/*"
        ]
      }
    }
```

Falta el elemento de versión JSON

El elemento de política `Version` es diferente de la versión de una política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, por otro lado, se crea al realizar cambios en una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada. Para obtener más información sobre el elemento de política `Version`, consulte Elementos de política JSON de IAM: Version. Para obtener más información sobre las versiones de política, consulte the section called “Control de versiones de políticas de IAM”.

A medida que evolucionan las características de AWS, se añaden nuevas funciones a las políticas de IAM para respaldar dichas características. A veces, una actualización en la sintaxis de la política incluye un número de versión nuevo. Si utiliza las características más nuevas de la política gramatical en la política, debe indicarle al motor de análisis de políticas la versión que está utilizando. La versión predeterminada de la política es "2008-10-17". Si desea utilizar una característica de la política introducida más tarde, entonces deberá especificar el número de versión compatible con la característica que desee. Le recomendamos que incluya siempre el número de versión de la sintaxis de política más reciente, que actualmente es `"Version": "2012-10-17"`. Por ejemplo, la siguiente política es incorrecta pues utiliza una variable de política `${...}` en el ARN de un recurso. Sin embargo, no especifica una versión de sintaxis de política que admita las variables de política (destacadas en *`rojo`*):

```
{
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Añadir un elemento `Version` en la parte superior de la política con el valor `2012-10-17`, que es la primera versión de la API de IAM que admite variables de política, soluciona este problema (destacado en negrita):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Solución de problemas con claves de seguridad FIDO

Utilice la información que se indica aquí para diagnosticar y solucionar los problemas más frecuentes que surgen cuando se trabaja con claves de seguridad FIDO2.

Temas

- No puedo habilitar mi clave de seguridad FIDO
- No puedo iniciar sesión con mi clave de seguridad FIDO
- He perdido o he roto la clave de seguridad FIDO
- Otros problemas.

No puedo habilitar mi clave de seguridad FIDO

Consulte las siguientes soluciones según sea su estado, usuario de IAM o administrador del sistema

Usuarios de IAM

Si no puede habilitar su clave de seguridad FIDO, compruebe lo siguiente:

- ¿Está utilizando una configuración admitida?

  Para obtener información acerca de los dispositivos y navegadores que puede utilizar con WebAuthn y AWS, consulte Configuraciones admitidas para usar las claves de seguridad FIDO.

- ¿Está utilizando Mozilla Firefox?

  Las versiones actuales de Firefox admiten WebAuthn de forma predeterminada. Para habilitar la compatibilidad con WebAuthn en Firefox, haga lo siguiente:

  1. En la barra de direcciones de Firefox, escriba **`about:config`**.
  2. En la barra de búsqueda de la pantalla que se abre, escriba **`webauthn`**.
  3. Elija security.webauth.webauthn y cambie su valor a true (verdadero).

- ¿Está utilizando algún complemento de navegador?

  AWS no es compatible con el uso de complementos para agregar compatibilidad de navegador con WebAuthn. En su lugar, utilice un navegador que ofrezca compatibilidad nativa con el estándar WebAuthn.

  Incluso si está utilizando un navegador compatible, puede tener un complemento que sea incompatible con WebAuthn. Un complemento incompatible puede impedir que pueda activar y utilizar su clave de seguridad compatible con FIDO. Debe desactivar todos los complementos que puedan ser incompatibles y reiniciar el navegador. Después intente volver a activar la clave de seguridad FIDO.

- ¿Tiene los permisos adecuados?

  Si no tiene ninguno de los problemas de compatibilidad anteriores, es posible que no tenga los permisos pertinentes. Póngase en contacto con el administrador del sistema.

Administradores de sistemas

Si es el administrador y sus usuarios de IAM no pueden habilitar sus claves de seguridad FIDO a pesar de utilizar una configuración admitida, asegúrese de que tengan los permisos pertinentes. Para un ejemplo detallado, consulte Tutorial de IAM: permitir a los usuarios administrar sus credenciales y configuración de MFA.

No puedo iniciar sesión con mi clave de seguridad FIDO

Si es usuario de IAM y no puede iniciar sesión en la AWS Management Console con su clave de seguridad FIDO, consulte primero Configuraciones admitidas para usar las claves de seguridad FIDO. Si utiliza una configuración admitida pero no puede iniciar sesión, póngase en contacto con su administrador del sistema para obtener ayuda.

He perdido o he roto la clave de seguridad FIDO

Se pueden asignar a un usuario hasta ocho dispositivos MFA de cualquier combinación de los tipos de MFA actualmente compatibles. Con varios dispositivos MFA, solo necesita un dispositivo MFA para iniciar sesión en la AWS Management Console. Reemplazar una clave de seguridad FIDO es similar a reemplazar un token TOTP de hardware. Para obtener información sobre qué debe hacer si pierde o rompe cualquier tipo de dispositivo MFA, consulte ¿Qué pasa si un dispositivo MFA se pierde o deja de funcionar?.

Otros problemas.

Si tiene un problema con las claves de seguridad FIDO que no se aborde aquí, siga una de las indicaciones que se indica a continuación:

- Usuarios de IAM: póngase en contacto con el administrador del sistema.
- usuarios raíz de Cuenta de AWS: contacte Soporte de AWS.

Solución de problemas de roles de IAM

Utilice la información que se indica aquí para diagnosticar y solucionar los problemas comunes que puedan surgir cuando trabaje con roles de IAM.

Temas

- No puedo asumir un rol
- Un nuevo rol ha aparecido en la cuenta de AWS
- No logro editar o eliminar un rol en mi Cuenta de AWS
- No estoy autorizado a realizar la operación: iam: PassRole
- ¿Por qué no puedo asumir un rol con una sesión de 12 horas? (AWS CLI o API de AWS)

- Recibo un error cuando intento cambiar de rol en la consola de IAM
- Mi función tiene una política que me permite realizar una acción, sin embargo, obtengo "acceso denegado"
- El servicio no creó la versión de directiva predeterminada del rol
- No hay ningún caso de uso para un rol de servicio en la consola

No puedo asumir un rol

Compruebe lo siguiente:

- Para permitir que los usuarios vuelvan a asumir el rol actual dentro de una sesión de rol, especifique el ARN del rol o el ARN de la Cuenta de AWS como entidad principal en la política de confianza de rol. Los Servicios de AWS que proporcionan recursos de computación, como Amazon EC2, Amazon ECS, Amazon EKS y Lambda, brindan credenciales temporales y las actualizan automáticamente. Esto garantiza que siempre disponga de un conjunto de credenciales válido. Para estos servicios, no es necesario volver a asumir el rol actual a fin de obtener credenciales temporales. Sin embargo, si tiene la intención de aprobar etiquetas de sesión o una política de sesión, tendrá que volver a asumir el rol actual. Para obtener información sobre cómo modificar una política de confianza de roles a fin de agregar el ARN del rol o el ARN de la Cuenta de AWS para la entidad principal, consulte Modificación de una política de confianza de rol (consola).
- Cuando asume un rol utilizando AWS Management Console, asegúrese de utilizar el nombre exacto de su rol. Los nombres de rol distinguen entre mayúsculas y minúsculas al asumir un rol.
- Cuando asume un rol utilizando API de AWS STS o AWS CLI, asegúrese de utilizar el nombre exacto de su rol en el ARN. Los nombres de rol distinguen entre mayúsculas y minúsculas al asumir un rol.
- Compruebe que la política de IAM le concede permisos para llamar a `sts:AssumeRole` para el rol que desea asumir. El elemento `Action` de su política de IAM debe permitir llamar a la acción `AssumeRole`. Además, el elemento `Resource` de la política de IAM debe especificar el rol que desea asumir. Por ejemplo, el elemento `Resource` puede especificar un rol utilizando el nombre de recurso de Amazon (ARN) o un comodín (*). Por ejemplo, al menos una política aplicable a usted debe conceder permisos similares a los siguientes:

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
```

- Compruebe que su identidad de IAM se etiqueta con las etiquetas que la política de IAM requiere. Por ejemplo, en los siguientes permisos de políticas, el elemento `Condition` requiere que usted, como principal que solicita asumir el rol, tenga una etiqueta específica. Debe ser etiquetados con `department = HR` o `department = CS`. De lo contrario, no puede asumir el rol. Para obtener más información sobre el etiquetado de usuarios y roles de IAM, consulte the section called “Etiquetado de recursos de IAM”.

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "*",
"Condition": {"StringEquals": {"aws:PrincipalTag/department": [
        "HR",
        "CS"
    ]}}
```

- Compruebe que cumple todas las condiciones que se especifican en la política de confianza del rol. Una `Condition` puede especificar una fecha de vencimiento, un ID externo o que una solicitud debe proceder de unas direcciones IP determinadas. Considere el siguiente ejemplo: si la fecha actual es posterior a la fecha especificada, entonces la política nunca coincidirá y no podrá concederle el permiso para asumir el rol.

```
"Effect": "Allow",
"Action": "sts:AssumeRole",
"Resource": "arn:aws:iam::account_id_number:role/role-name-you-want-to-assume"
"Condition": {
    "DateLessThan" : {
        "aws:CurrentTime" : "2016-05-01T12:00:00Z"
    }
}
```

- Compruebe que la Cuenta de AWS desde la que está llamando a `AssumeRole` es una entidad de confianza para el rol que va a tomar. Las entidades de confianza se definen como `Principal` en la política de confianza del rol. El siguiente ejemplo es una política de confianza que está asociada a la función que desea asumir. En este ejemplo, el ID de la cuenta con el usuario de IAM con el que ha iniciado sesión debe ser 123456789012. Si el número de cuenta no está incluido en el elemento `Principal` de la política de confianza del rol, no se puede asumir el rol. No importa los permisos que le otorguen las políticas de acceso. Tenga en cuenta que esta política de ejemplo limita los permisos a acciones que se producen entre el 1 de julio de 2017 y el 31 de diciembre de 2017 (UTC), ambos incluidos. Si inicia sesión antes o después de dichas fechas, entonces la política no coincidirá y no podrá asumir el rol.

```
    "Effect": "Allow",
    "Principal": { "AWS": "arn:aws:iam::123456789012:root" },
    "Action": "sts:AssumeRole",
    "Condition": {
      "DateGreaterThan": {"aws:CurrentTime": "2017-07-01T00:00:00Z"},
      "DateLessThan": {"aws:CurrentTime": "2017-12-31T23:59:59Z"}
    }
```

- Identidad de fuente: los administradores pueden configurar roles para requerir que las identidades pasen una cadena personalizada que identifique a la persona o aplicación que está realizando acciones en AWS, llamados Identidad de fuente. Compruebe si el rol que se asume requiere que se establezca una identidad de fuente. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

Un nuevo rol ha aparecido en la cuenta de AWS

Algunos servicios de AWS exigen que utilice un tipo único de rol de servicio vinculado directamente al servicio. El servicio define previamente este rol vinculado al servicio e incluye todos los permisos que el servicio requiere. Esto simplifica la configuración de un servicio porque ya no tendrá que añadir manualmente los permisos necesarios. Para obtener información general acerca de los roles vinculados a servicios, consulte Uso de roles vinculados a servicios.

Es posible que ya esté utilizando un servicio cuando este comience a admitir roles vinculados a servicios. De ser así, es posible que reciba un mensaje de correo electrónico informándole de un nuevo rol en su cuenta. Este rol incluye todos los permisos que el servicio necesita para realizar acciones en su nombre. No es necesario realizar ninguna acción para admitir este rol. Sin embargo, no debe eliminar el rol de su cuenta. Si lo hace, podría eliminar los permisos que el servicio necesita para obtener acceso a los recursos de AWS. Puede consultar los roles vinculados a servicios en su cuenta en cualquier momento a través de la página de Roles de IAM de la consola de IAM. Los roles vinculados con servicios aparecen con el texto (Service-linked role (Función vinculada al servicio)) en la columna Trusted entities (Entidades de confianza) de la tabla.

Para obtener información sobre los servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes (Sí)en la columna Service-Linked Role (Rol vinculado al servicio). Para obtener más información acerca del uso de roles vinculados a servicios en un servicio, elija el enlace Yes (Sí).

No logro editar o eliminar un rol en mi Cuenta de AWS

No es posible eliminar ni editar los permisos de un rol vinculado a un servicio desde IAM. Estos roles incluyen permisos y políticas de confianza predefinidos exigidos por el servicio para poder realizar acciones en su nombre. Puede utilizar la consola IAM, la AWS CLI o la API para editar solo la descripción de un rol vinculado a un servicio. Puede consultar los roles vinculados a servicios en su cuenta en cualquier momento a través de la página de Roles de IAM de la consola. Los roles vinculados con servicios aparecen con el texto (Service-linked role (Función vinculada al servicio)) en la columna Trusted entities (Entidades de confianza) de la tabla. Un banner en la página Summary (Resumen) del rol también indica que es un tipo de rol vinculado a un servicio. Puede administrar y eliminar estos roles vinculados únicamente a través del servicio vinculado, si dicho servicio admite esa acción. Tenga cuidado a la hora de modificar o eliminar roles vinculados a servicios, ya que podría eliminar permisos que el servicio necesita para acceder a los recursos de AWS.

Para obtener información sobre los servicios que admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM y busque los servicios que tengan Yes (Sí)en la columna Service-Linked Role (Rol vinculado al servicio).

No estoy autorizado a realizar la operación: iam: PassRole

Si crea un rol vinculado a un servicio, debe disponer de permiso para transferir ese rol al servicio. Algunos servicios crean automáticamente un rol vinculado a un servicio en su cuenta al realizar una acción en dicho servicio. Por ejemplo, Amazon EC2 Auto Scaling crea el rol vinculado a un servicio `AWSServiceRoleForAutoScaling` automáticamente la primera vez que se crea un grupo de Auto Scaling. Si intenta crear un grupo de Auto Scaling sin el permiso `PassRole`, recibirá el siguiente error:

```
ClientError: An error occurred (AccessDenied) when calling the
PutLifecycleHook operation: User: arn:aws:sts::111122223333:assumed-role/
Testrole/Diego is not authorized to perform: iam:PassRole on resource:
arn:aws:iam::111122223333:role/aws-service-role/autoscaling.amazonaws.com/
AWSServiceRoleForAutoScaling
```

Para solucionar este error, pida al administrador que añada el permiso `iam:PassRole` para usted.

Para saber qué servicios admiten roles vinculados a servicios, consulte Servicios de AWS que funcionan con IAM. Para saber si un servicio crea automáticamente una función vinculada al servicio por usted, haga clic en el enlace Yes (Sí) para consultar la documentación relacionada con las funciones vinculadas a servicios.

¿Por qué no puedo asumir un rol con una sesión de 12 horas? (AWS CLI o API de AWS)

Cuando se utiliza la API de `AssumeRole*` de AWS STS o las operaciones de CLI `assume-role*` para asumir un rol, es posible especificar un valor para el parámetro `DurationSeconds`. Puede especificar un valor comprendido entre 900 segundos (15 minutos) y la duración máxima de la sesión para el rol. Si especifica un valor superior al indicado en esta opción, la operación producirá un error. Este ajuste puede tener un valor máximo de 12 horas. Por ejemplo, si especifica una duración de 12 horas para la sesión, pero el administrador establece la duración máxima de la sesión en 6 horas, la operación genera un error. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

Si utiliza el encadenamiento de roles (en el que se usa un rol para asumir otro), la sesión tendrá una duración máxima de una hora. Si utiliza a continuación el parámetro `DurationSeconds` para proporcionar un valor superior a una hora, la operación generará un error.

Recibo un error cuando intento cambiar de rol en la consola de IAM

La información que introduzca en la página Cambiar rol debe coincidir con la información del rol. De lo contrario, la operación falla y recibirá el siguiente error:

```
Invalid information in one or more fields. Check your information or
contact your administrator.
```

Si recibe este error, confirme que la siguiente información es correcta:

- ID de cuenta o alias: El ID de la Cuenta de AWS es un número de 12 dígitos. Es posible que su cuenta tenga un alias, que es un identificador descriptivo, como el nombre de su empresa, que se puede utilizar en lugar de su ID de Cuenta de AWS. Puede utilizar el ID de cuenta o el alias en este campo.
- Nombre de rol – Los nombres de rol distinguen entre mayúsculas y minúsculas. El Id. de cuenta y el nombre del rol deben coincidir con la configuración del rol.

Si continúa recibiendo un mensaje de error, póngase en contacto con el administrador para verificar la información anterior. La política de confianza de rol o la política de usuario de IAM pueden limitar el acceso. El administrador puede verificar los permisos de estas directivas.

Mi función tiene una política que me permite realizar una acción, sin embargo, obtengo "acceso denegado"

Puede que su sesión de función se vea limitada por las políticas de la sesión. Al solicitar las credenciales de seguridad temporales mediante programación utilizando AWS STS, tiene la opción de pasar las políticas de sesión administradas o insertadas. Las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión de credenciales temporal mediante programación para una función. Puede transferir un único documento de política de sesión insertada JSON utilizando el parámetro `Policy`. Puede utilizar el parámetro `PolicyArns` para especificar hasta 10 políticas de sesión administrada. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la función y las políticas de la sesión. De manera alternativa, si el administrador o un programa personalizado le proporcionan credenciales temporales, es posible que hayan incluido una política de sesión para limitar su acceso.

El servicio no creó la versión de directiva predeterminada del rol

Un rol de servicio es un rol que asume funciones para realizar acciones en su cuenta en su nombre. Al configurar algunos de los entornos de los servicios de AWS, debe definir un rol que el servicio asumirá. En algunos casos, el servicio crea la función de servicio y su política de IAM para usted. Aunque puede modificar o eliminar la función de servicio y su política desde dentro de IAM, AWS no lo recomienda. El rol y la directiva están pensados para su uso exclusivo de ese servicio. Si edita la directiva y configura otro entorno, cuando el servicio intenta utilizar el mismo rol y la misma directiva, la operación puede fallar.

Por ejemplo, cuando se utiliza AWS CodeBuild por primera vez, el servicio crea un rol denominado `codebuild-RWBCore-service-role`. Ese rol de servicio utiliza la directiva denominada `codebuild-RWBCore-managed-policy`. Si edita la directiva, creará una nueva versión y la guardará como la versión predeterminada. Si realiza una operación posterior en AWS CodeBuild, el servicio podría intentar actualizar la directiva. Si lo hace, recibirá el siguiente error:

```
codebuild.amazon.com did not create the default version (V2) of the
codebuild-RWBCore-managed-policy policy that is attached to the codebuild-
RWBCore-service-role role. To continue, detach the policy from any other
identities and then delete the policy and the role.
```

Si recibe este error, debe realizar cambios en IAM antes de poder continuar con la operación de servicio. En primer lugar, establezca la versión de directiva predeterminada en V1 e intente la

operación de nuevo. Si V1 se eliminó anteriormente, o si V1 no funciona, limpie y elimine la política y el rol existentes.

Para obtener más información sobre la edición de directivas administradas, consulte Edición de políticas administradas por el cliente (Consola). Para obtener más información acerca de las versiones, consulte Control de versiones de políticas de IAM.

Para eliminar un rol de servicio y su directiva

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas).
3. En la lista de políticas, seleccione el nombre de la política que desea ver.
4. Seleccione la pestaña Entidades asociadas para ver qué usuarios, grupos o roles de IAM utilizan esta política. Si alguna de estas identidades utiliza la directiva, complete las siguientes tareas:
    a. Cree una nueva directiva administrada con los permisos necesarios. Para asegurarse de que las identidades tienen los mismos permisos antes y después de las acciones, copie el documento de directiva JSON de la directiva existente. Después, cree la nueva política administrada y pegue el documento JSON tal y como se explica en Creación de políticas mediante el editor JSON.
    b. Para cada identidad afectada, adjunte la nueva directiva y, a continuación, desconecte la anterior. Para obtener más información, consulte Adición y eliminación de permisos de identidad de IAM.
5. Seleccione Roles (Roles) en el panel de navegación.
6. En la lista de roles, elija el nombre del rol que desea eliminar.
7. Seleccione la ficha Relaciones de confianza para ver qué entidades pueden asumir el rol. Si aparece alguna entidad distinta del servicio, complete las siguientes tareas:
    a. Cree un nuevo rol que confíe en esas entidades.
    b. La política que creó en el paso anterior. Si omitió ese paso, cree ahora la nueva directiva administrada.
    c. Notifique a cualquier persona que esté asumiendo el rol que ya no puede hacerlo. Proporcione información acerca de cómo asumir el nuevo rol y tener los mismos permisos.
8. Elimine la directiva.
9. Elimine el rol.

No hay ningún caso de uso para un rol de servicio en la consola

Algunos servicios requieren que cree de forma manual un rol de servicio con el fin de otorgar los permisos de servicio para realizar acciones en su nombre. Si el servicio no aparece en la consola de IAM, debe agregarlo manualmente como la entidad principal de confianza. Si la documentación del servicio o de la característica que utiliza no incluye las instrucciones para agregarlo como la entidad principal de confianza, envíe comentarios sobre la página.

Para crear manualmente un rol de servicio, debe conocer la entidad principal de servicio para el servicio que asumirá la función. Un principal de servicio es un identificador que se utiliza para conceder permisos a un servicio. El servicio define la entidad principal de servicio.

Puede encontrar la entidad principal de algunos servicios comprobando lo siguiente:

1. Abrir Servicios de AWS que funcionan con IAM.
2. Compruebe si el servicio tiene la palabra Yes (Sí) en la columna Service-linked roles (Roles vinculados al servicio).
3. Elija el vínculo Sí para ver la documentación acerca del rol vinculado al servicio en cuestión.
4. Consulte la sección de permisos de rol vinculados a servicios de ese servicio para ver la entidad principal del servicio.

Puede crear manualmente un rol de servicio mediante los comandos de la AWS CLI o de las operaciones de la API de AWS. Para crear manualmente una función de servicio mediante la consola de IAM, realice las siguientes tareas:

1. Cree un rol de IAM con su ID de cuenta. No asocie ninguna política ni otorgue ningún permiso. Para obtener más información, consulte Creación de un rol para delegar permisos a un usuario de IAM.
2. Abra el rol y edite la relación de confianza. En lugar de confiar en la cuenta, el rol debe confiar en el servicio. Por ejemplo, actualice el siguiente elemento de la `Principal`:

   ```
   "Principal": { "AWS": "arn:aws:iam::123456789012:root" }
   ```

   Cambie la entidad principal por el valor de su servicio, como IAM.

   ```
   "Principal": { "Service": "iam.amazonaws.com" }
   ```

3. Agregue los permisos que requiere el servicio asociando políticas de permisos al rol.

4. Regrese al servicio que requiere los permisos y utilice el método documentado para notificar al servicio acerca del nuevo rol de servicio.

Solución de problemas IAM y Amazon EC2

Utilice la información que se indica aquí para diagnosticar y solucionar los problemas de acceso denegado u otros problemas que puedan surgir cuando trabaje con Amazon EC2 e IAM.

Temas

Cuando intento lanzar una instancia, no veo el rol que esperaba en la lista rol de IAM de la consola de Amazon EC2.

Compruebe lo siguiente:

- Si ha iniciado sesión como usuario de IAM, verifique que tenga permiso para llamar a `ListInstanceProfiles`. Para obtener información sobre los permisos necesarios para trabajar con roles, consulte "Permisos necesarios para utilizar roles con Amazon EC2" en Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2. Para obtener información sobre cómo añadir permisos a un usuario, consulte Administración de políticas de IAM.

  Si no puede modificar sus propios permisos, debe ponerse en contacto con un administrador que pueda trabajar con IAM para actualizar los permisos.
- Si ha creado un rol utilizando IAM la CLI o una API, verifique que haya creado un perfil de instancia y que haya añadido el rol a dicho perfil de instancias. Además, si el rol y el perfil de instancias tienen un nombre diferente, no verá el nombre de rol correcto en la lista de roles de IAM de la consola de Amazon EC2. La lista Rol de IAM de la consola de Amazon EC2 establece una lista

de los nombres de perfiles de instancia, pero no de los nombres de rol. Tendrá que seleccionar el nombre del perfil de instancia que contiene el rol que desea. Para obtener más información acerca de los perfiles de instancia, consulte Uso de perfiles de instancia.

> **Note**
>
> Si utiliza la consola de IAM para crear roles, no es necesario que trabaje con perfiles de instancias. Por cada rol que cree en la consola de IAM, se creará un perfil de instancias con el mismo nombre que el rol, y este se agregará automáticamente a dicho perfil de instancias. Un perfil de instancias puede contener un único rol de IAM y este límite no se puede aumentar.

Las credenciales de mi instancia tienen un rol erróneo.

La función del perfil de instancia podría haber sido reemplazada recientemente. En caso afirmativo, la aplicación tendrá que esperar a la siguiente rotación de credenciales programada automáticamente para que las credenciales de la función estén disponibles.

Para forzar el cambio, debe desvincular el perfil de instancia y, a continuación, asociar el perfil de instancia, o bien puede detener la instancia y después reiniciarla.

Cuando intento llamar a **`AddRoleToInstanceProfile`**, recibo el error **`AccessDenied`**.

Si realiza las solicitudes como usuario de IAM, compruebe que los siguientes permisos se cumplan:

- `iam:AddRoleToInstanceProfile` con el recurso que coincide con el ARN del perfil de instancia (por ejemplo, `arn:aws:iam::999999999999:instance-profile/ExampleInstanceProfile`).

Para obtener más información acerca de los permisos necesarios para trabajar con roles, consulte la sección sobre cómo comenzar en Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2. Para obtener información sobre cómo añadir permisos a un usuario, consulte Administración de políticas de IAM.

Amazon EC2: cuando intento lanzar una instancia con un rol, obtengo un error `AccessDenied`

Compruebe lo siguiente:

- Lance una instancia sin un perfil de instancia; Esto le servirá para asegurarse de que el problema se limita a los roles de IAM para instancias Amazon EC2.
- Si realiza las solicitudes como usuario de IAM, compruebe que los siguientes permisos se cumplan:
  - `ec2:RunInstances` con un recurso de comodín ("*")
  - `iam:PassRole` con el recurso que coincide con el ARN del rol (por ejemplo, `arn:aws:iam::999999999999:role/ExampleRoleName`)
- Llame a la acción `GetInstanceProfile` de IAM para asegurarse de que está utilizando un nombre de perfil de instancias válido o un ARN de perfil de instancias válido. Para obtener más información, consulte Uso roles de IAM con instancia de Amazon EC2.
- Llame a la acción `GetInstanceProfile` de IAM para asegurarse de que el perfil de instancias tenga un rol. Los perfiles de instancia vacíos darán un error `AccessDenied`. Para obtener más información sobre cómo crear un rol, consulte Creación de roles de IAM.

Para obtener más información acerca de los permisos necesarios para trabajar con roles, consulte la sección sobre cómo comenzar en Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2. Para obtener información sobre cómo añadir permisos a un usuario, consulte Administración de políticas de IAM.

No puedo obtener acceso a las credenciales de seguridad temporales de mi instancia EC2.

Para acceder a las credenciales de seguridad temporales en la instancia EC2, primero debe utilizar la consola de IAM para crear un rol. A continuación, debe lanzar una instancia EC2 que utilice ese rol y examinar la instancia en ejecución. Para obtener más información, consulte la sección How Do I Get Started? (¿Cómo comenzar?) en Uso de un rol de IAM para conceder permisos a aplicaciones que se ejecutan en instancias Amazon EC2.

Si sigue sin poder acceder a sus credenciales de seguridad temporales en la instancia EC2, compruebe lo siguiente:

- ¿Puede obtener acceso a otra parte del Servicio de metadatos de la instancia (IMDS)? Si no puede, compruebe que el firewall no tenga reglas que bloqueen el acceso a las solicitudes al IMDS.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/
hostname; echo
```

- ¿Existe el subárbol `iam` del IMDS? De lo contrario, verifique que su instancia tenga un perfil de instancia de IAM asociado. Para ello, llame a la operación de la API `DescribeInstances` de EC2 o utilice el comando de la CLI `aws ec2 describe-instances`.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam;
 echo
```

- Consulte el documento `info` del subárbol de IAM por si hay un error. Si detecta un error, consulte ¿Qué significan los errores del documento `info` en el subárbol de IAM? para obtener más información.

```
[ec2-user@domU-12-31-39-0A-8D-DE ~]$ GET http://169.254.169.254/latest/meta-data/iam/
info; echo
```

¿Qué significan los errores del documento **`info`** en el subárbol de IAM?

El documento **`iam/info`** indica **`"Code":"InstanceProfileNotFound"`**

Su perfil de instancias de IAM se ha eliminado y Amazon EC2 ya no puede proporcionar credenciales a su instancia. Debe asociar un perfil de instancias válido a la instancia de Amazon EC2.

Si existe un perfil de instancia con ese nombre, compruebe que el perfil de instancia no se haya eliminado y que no se haya creado otro con el mismo nombre:

1. Llame a la operación IAM de `GetInstanceProfile` para obtener el `InstanceProfileId`.
2. Llame a la operación Amazon EC2 de `DescribeInstances` para obtener el `IamInstanceProfileId` de la instancia.
3. Compruebe que el `InstanceProfileId` de la operación de IAM coincida con el `IamInstanceProfileId` de la operación de Amazon EC2.

Si los ID son diferentes, el perfil de instancia asociado a sus instancias ya no es válido. Debe asociar un perfil de instancia válido a la instancia.

El documento **`iam/info`** indica una operación correcta, pero también indica **`"Message":"Instance Profile does not contain a role..."`**

Se ha eliminado el rol del perfil de instancias mediante la acción `RemoveRoleFromInstanceProfile` de IAM. Puede utilizar la acción `AddRoleToInstanceProfile` de IAM para adjuntar un rol al perfil de instancias. La aplicación tendrá que esperar hasta la siguiente actualización programada para obtener acceso a las credenciales del rol.

Para forzar el cambio, debe desvincular el perfil de instancia y, a continuación, asociar el perfil de instancia, o bien puede detener la instancia y después reiniciarla.

El documento **`iam/security-credentials/[role-name]`** indica **`"Code":"AssumeRoleUnauthorizedAccess"`**

Amazon EC2 no tiene permiso para asumir el rol. El permiso para asumir el rol se controla mediante la política de confianza asociada al rol, como en el ejemplo siguiente. Utilice la API `UpdateAssumeRolePolicy` de IAM para actualizar la política de confianza.

```
{"Version": "2012-10-17","Statement": [{"Effect": "Allow","Principal": {"Service":
 ["ec2.amazonaws.com"]},"Action": ["sts:AssumeRole"]}]}
```

La aplicación tendrá que esperar hasta la siguiente actualización programada automáticamente para obtener acceso a las credenciales del rol.

Para forzar el cambio, debe desvincular el perfil de instancia y, a continuación, asociar el perfil de instancia, o bien puede detener la instancia y después reiniciarla.

Solución de problemas Amazon S3 e IAM

Utilice la información que se indica aquí para diagnosticar y solucionar los problemas que puedan surgir cuando trabaje con Amazon S3 y IAM.

¿Cómo puedo conceder acceso anónimo a un bucket de Amazon S3?

Puede utilizar una política de bucket de Amazon S3 que especifique un asterisco (*) en el elemento `principal`, lo que significa que cualquiera puede obtener acceso al bucket. Con un acceso

anónimo, cualquiera (incluidos los usuarios sin una Cuenta de AWS) podrá obtener acceso al bucket. Para ver ejemplos de políticas, consulte Ejemplos de políticas de bucket de Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.

He iniciado sesión como usuario raíz de una Cuenta de AWS, ¿por qué no puedo obtener acceso a un bucket de Amazon S3 que está en mi cuenta?

En algunos casos es posible que tenga un usuario de IAM con acceso completo a IAM y Amazon S3. Si el usuario de IAM asigna una política de bucket a un bucket de Amazon S3 y no especifica al Usuario raíz de la cuenta de AWS como entidad principal, se denegará al usuario raíz el acceso a ese bucket. Sin embargo, como usuario raíz, sigue pudiendo obtener acceso al bucket. Para ello, modifique la política del bucket para permitir el acceso a desde la consola de Amazon S3 o la AWS CLI. Utilice la siguiente entidad principal, reemplazando *123456789012* por el ID de la Cuenta de AWS.

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

Solución de problemas de la federación SAML 2.0 con AWS

Utilice la información que se indica aquí para diagnosticar y solucionar los problemas que puedan surgir cuando trabaje con SAML 2.0 y la federación con IAM.

Temas

- Error: Your request included an invalid SAML response. To logout, click here.
- Error: RoleSessionName es obligatorio en AuthnResponse (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)
- Error: falta la autorización para realizar sts:AssumeRoleWithSAML (servicio: AWSSecurityTokenService; código de estado: 403; código de error: AccessDenied)
- Error: RoleSessionName en AuthnResponse debe coincidir [a-zA-Z_0-9+=,.@-]{2,64} (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)
- Error: identidad de fuente debe coincidir con [a-zA-Z_0-9+=,.@-]{2,64} y no comenzar con "aws:" (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)
- Error: firma de respuesta no válida (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)

- Error: no se ha podido asumir un rol: el emisor no está presente en el proveedor especificado (servicio: AWSOpenIdDiscoveryService; código de estado: 400; código de error: AuthSamlInvalidSamlResponseException)
- Error: no se ha podido analizar los metadatos.
- Error: el proveedor especificado no existe.
- Error: el valor de DurationSeconds solicitado es mayor que el valor de MaxSessionDuration establecido para este rol.
- Error: la respuesta no contiene la audiencia requerida.
- Cómo ver una respuesta SAML en el navegador para la solución de problemas

Error: Your request included an invalid SAML response. To logout, click here.

Este error puede producirse cuando la respuesta SAML del proveedor de identidades no contiene un atributo con `Name` establecido en `https://aws.amazon.com/SAML/Attributes/Role`. El atributo debe contener uno o varios elementos `AttributeValue`, cada uno con un par de cadenas separadas por comas:

- El ARN de un rol con el que el usuario puede estar mapeado.
- El ARN del proveedor SAML.

Para obtener más información, consulte Configure aserciones SAML para la respuesta de autenticación. Para ver la respuesta de SAML en el navegador, siga los pasos que se indican en Cómo ver una respuesta SAML en el navegador para la solución de problemas.

Error: RoleSessionName es obligatorio en AuthnResponse (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)

Este error puede producirse cuando la respuesta SAML del proveedor de identidades no contiene un atributo con `Name` establecido en `https://aws.amazon.com/SAML/Attributes/RoleSessionName`. El valor de atributo es un identificador para el usuario y suele ser un ID de usuario o una dirección de correo electrónico.

Para obtener más información, consulte Configure aserciones SAML para la respuesta de autenticación. Para ver la respuesta de SAML en el navegador, siga los pasos que se indican en Cómo ver una respuesta SAML en el navegador para la solución de problemas.

Error: falta la autorización para realizar sts:AssumeRoleWithSAML (servicio: AWSSecurityTokenService; código de estado: 403; código de error: AccessDenied)

Este error se produce si el rol de IAM especificado en la respuesta de SAML está mal escrito o no existe. Asegúrese de utilizar el nombre exacto de su rol, ya que los nombres de roles distinguen entre mayúsculas y minúsculas. Corrija el nombre del rol en la configuración del proveedor de servicios SAML.

Solo se le permite el acceso si la política de confianza de rol incluye la acción `sts:AssumeRoleWithSAML`. Si la aserción SAML está configurada para utilizar el atributo `PrincipalTag`, la política de confianza también debe incluir la acción `sts:TagSession`. Para obtener más información acerca de las etiquetas de sesión, consulte Transferencia de etiquetas de sesión en AWS STS.

Este error puede producirse si no tiene `sts:SetSourceIdentity` en su política de confianza de rol. Si la aserción SAML está configurada para utilizar el atributo `SourceIdentity`, la política de confianza también debe incluir la acción `sts:SetSourceIdentity`. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

Este error también puede producirse si los usuarios federados no tienen permisos para asumir el rol. El rol debe tener una política de confianza que especifique el ARN de la identidad SAML de IAM como el elemento `Principal`. El rol también contiene condiciones que controlan qué usuarios pueden asumir el rol. Asegúrese de que sus usuarios cumplan los requisitos de las condiciones.

Este error también puede producirse si la respuesta de SAML no contiene un `Subject` con un `NameID`.

Para obtener más información, consulte la sección Establecer permisos en AWS para usuarios federados y Configure aserciones SAML para la respuesta de autenticación. Para ver la respuesta de SAML en el navegador, siga los pasos que se indican en Cómo ver una respuesta SAML en el navegador para la solución de problemas.

Error: RoleSessionName en AuthnResponse debe coincidir [a-zA-Z_0-9+=,.@-]{2,64} (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)

Este error puede producirse si el valor del atributo `RoleSessionName` es demasiado largo o contiene caracteres no válidos. La longitud válida máxima es de 64 caracteres.

Para obtener más información, consulte Configure aserciones SAML para la respuesta de autenticación. Para ver la respuesta de SAML en el navegador, siga los pasos que se indican en Cómo ver una respuesta SAML en el navegador para la solución de problemas.

Error: identidad de fuente debe coincidir con [a-zA-Z_0-9+=,.@-]{2,64} y no comenzar con **`"aws:"`** (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)

Este error puede producirse si el valor del atributo `sourceIdentity` es demasiado largo o contiene caracteres no válidos. La longitud válida máxima es de 64 caracteres. Para obtener más información acerca de las identidades de fuente, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

Para obtener más información acerca de la creación de aserciones SAML, consulte Configure aserciones SAML para la respuesta de autenticación. Para ver la respuesta de SAML en el navegador, siga los pasos que se indican en Cómo ver una respuesta SAML en el navegador para la solución de problemas.

Error: firma de respuesta no válida (servicio: AWSSecurityTokenService; código de estado: 400; código de error: InvalidIdentityToken)

Este error puede producirse cuando los metadatos de la federación del proveedor de identidades no coinciden con los metadatos del proveedor de identidades de IAM. Por ejemplo, el archivo de metadatos del proveedor de servicios de identidades podría haber cambiado para actualizar un certificado caducado. Descargue el archivo de metadatos SAML actualizado de su proveedor de servicios de identidades. A continuación, actualícelo en la entidad de proveedor de identidades de AWS que usted define en IAM con el comando de la `aws iam update-saml-provider` CLI o el cmdlet de PowerShell `Update-IAMSAMLProvider`.

Error: no se ha podido asumir un rol: el emisor no está presente en el proveedor especificado (servicio: AWSOpenIdDiscoveryService; código de estado: 400; código de error: AuthSamlInvalidSamlResponseException)

Este error puede producirse si el emisor de la respuesta de SAML no coincide con el emisor declarado en el archivo de metadatos de federación. El archivo de metadatos se ha cargado a AWS al crear el proveedor de identidad en IAM.

Error: no se ha podido analizar los metadatos.

Este error se puede producir si el archivo de metadatos no está formateado correctamente.

Cuando crea o administra un proveedor de identidad SAML en la AWS Management Console debe recuperar el documento de metadatos de SAML de su proveedor de identidad.

Este archivo de metadatos incluye el nombre del emisor, información de vencimiento y las claves que se pueden utilizar para validar la respuesta de autenticación SAML (afirmaciones) recibida desde el IdP. El archivo de metadatos debe estar codificado en formato UTF-8 sin una marca de orden de bytes (BOM). Para eliminar la BOM, puede codificar el archivo en UTF-8 con una herramienta de edición de texto, como Notepad++.

El certificado x.509 incluido como parte del documento de metadatos de SAML debe utilizar un tamaño de clave de al menos 1024 bits. Además, el certificado x.509 no debe contener extensiones repetidas. Puede utilizar extensiones, pero estas solo pueden aparecer una vez en el certificado. Si el certificado x.509 no cumple ninguna condición, se produce un error en la creación de proveedor de identidad (IdP) y devuelve el mensaje “Unable to parse metadata” (No se pueden analizar los metadatos).

Según se define en la versión 1.0 del perfil de interoperabilidad de metadatos SAML V2.0, IAM no evalúa ni toma medidas en relación con la caducidad del certificado X.509 del documento de metadatos.

Error: el proveedor especificado no existe.

Este error se produce si el nombre del proveedor que especifique en la aserción SAML no coincide con el nombre del proveedor configurado en IAM. Para obtener más información sobre cómo ver el nombre del proveedor, consulte Crear un proveedor de identidades de SAML en IAM.

Error: el valor de DurationSeconds solicitado es mayor que el valor de MaxSessionDuration establecido para este rol.

Este error puede producirse si se asume un rol desde la AWS CLI o la API.

Cuando utilice las operaciones assume-role-with-saml de la CLI o AssumeRoleWithSAML de la API para asumir un rol, puede especificar un valor para el parámetro `DurationSeconds`. Puede especificar un valor comprendido entre 900 segundos (15 minutos) y la duración máxima de la sesión para el rol. Si especifica un valor superior al indicado en esta opción, la operación producirá un error. Por ejemplo, si especifica una duración de 12 horas para la sesión, pero el administrador establece la duración máxima de la sesión en 6 horas, la operación genera un error. Para obtener información sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

Error: la respuesta no contiene la audiencia requerida.

Este error puede producirse si hay una discrepancia entre la URL de la audiencia y el proveedor de identidades en la configuración de SAML. Asegúrese de que el identificador de la parte de confianza del proveedor de identidades (IdP) coincida exactamente con la URL de la audiencia (ID de entidad) proporcionada en la configuración de SAML.

Cómo ver una respuesta SAML en el navegador para la solución de problemas

Los siguientes procedimientos describen cómo ver la respuesta SAML de su proveedor de servicios en el navegador durante la resolución de un problema relacionado con SAML 2.0.

En todos los navegadores, diríjase a la página en la que pueda reproducir el problema. A continuación, siga los pasos del correspondiente navegador:

Temas

- Google Chrome
- Mozilla Firefox
- Apple Safari
- Qué hacer con la respuesta SAML codificada en Base64

Google Chrome

Para ver una respuesta SAML en Chrome

Estos pasos se probaron con la versión 106.0.5249.103 (versión oficial) (arm64) de Google Chrome. Si utiliza otra versión, es posible que necesite adaptar los pasos como corresponda.

1. Pulse F12 para iniciar la consola para desarrolladores.
2. Seleccione la pestaña Network (Red) y, a continuación, seleccione Preserve log (Conservar registro) en la parte superior izquierda de la ventana Developer Tools (Herramientas para desarrolladores).
3. Reproduzca el problema.
4. (Opcional) Si la columna Method (Método) no está visible en el panel de registro de Developer Tools (Herramientas para desarrolladores) Network (Red), haga clic con el botón derecho en cualquier etiqueta de columna y elija Method (Método) para agregar la columna.
5. Busque una publicación de SAML en el panel de registro de Developer Tools (Herramientas para desarrolladores) Network (Red). Seleccione dicha fila y, a continuación, visualice la pestaña Payload (Carga) en la parte superior. Busque el elemento SAMLResponse que incluye la solicitud codificada. El valor asociado es la respuesta codificada en Base64.

Mozilla Firefox

Para ver una respuesta SAML en Firefox

Este procedimiento se ha probado en la versión 105.0.3 (64 bits) de Mozilla Firefox. Si utiliza otra versión, es posible que necesite adaptar los pasos como corresponda.

1. Pulse F12 para iniciar la consola de herramientas web para desarrolladores.
2. Seleccione la pestaña Network (Red).
3. En la parte superior derecha de la ventana de herramientas para desarrolladores, haga clic en las opciones de la barra de herramientas (icono pequeño con forma de engranaje). Seleccione Persist logs (Conservar registros).
4. Reproduzca el problema.
5. (Opcional) Si la columna Method (Método) no está visible en el panel de registro de Web Developer Tools (Herramientas para desarrolladores) Network (Red), haga clic con el botón derecho en cualquier etiqueta de columna y elija Method (Método) para agregar la columna.

6. Busque un POST SAML en la tabla. Seleccione esa fila y, a continuación, consulte la pestaña Request (Solicitar) y busque el elemento SAMLResponse. El valor asociado es la respuesta codificada en Base64.

Apple Safari

Para ver una respuesta SAML en Safari

Estos pasos se probaron con la versión 16.0 (17614.1.25.9.10, 17614) de Apple Safari. Si utiliza otra versión, es posible que necesite adaptar los pasos como corresponda.

1. Habilite el inspector web en Safari. Abra la ventana Preferences (Preferencias), seleccione la pestaña Advanced (Configuración avanzada) y, a continuación, seleccione Show Develop menu in the menu bar (Mostrar menú de desarrollo en la barra de menús).
2. Ahora puede abrir el inspector web. Elija Develop (Desarrollar) en la barra de menús y, a continuación, seleccione Show Web Inspector (Mostrar inspector web).
3. Seleccione la pestaña Network (Red).
4. En la parte superior izquierda de la ventana de Web Inspector (Inspector web), seleccione las opciones (el icono de círculo pequeño que contiene tres líneas horizontales). Seleccione Preseve Log (Conservar registro).
5. (Opcional) Si la columna Method (Método) no está visible en el panel de registro de Web Inspector (Inspector web) Network (Red), haga clic con el botón derecho en cualquier etiqueta de columna y elija Method (Método) para agregar la columna.
6. Reproduzca el problema.
7. Busque un POST SAML en la tabla. Seleccione dicha fila y, a continuación, visualice la pestaña Headers (Encabezados).
8. Busque el elemento SAMLResponse que incluye la solicitud codificada. Desplácese hacia abajo para buscar `Request Data` con el nombre `SAMLResponse`. El valor asociado es la respuesta codificada en Base64.

Qué hacer con la respuesta SAML codificada en Base64

Una vez que haya encontrado el elemento de respuesta SAML codificada en Base64 en el navegador, cópiela y utilice su herramienta favorita de decodificación en Base64 para extraer la respuesta con etiquetas XML.

> **Consejo sobre seguridad**
>
> Dado que los datos de respuesta SAML que está visualizando pueden incluir datos de seguridad confidenciales, le recomendamos no utilizar un decodificador Base64 online. En cambio, utilice una herramienta instalada en el equipo local que no envíe los datos SAML a través de la red.

Opción integrada para sistemas Windows (PowerShell):

```
PS C:
\> [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String("base64encodedtext
```

Opción integrada para sistemas MacOS y Linux:

```
$ echo "base64encodedtext" | base64 --decode
```

Información de referencia para AWS Identity and Access Management

Utilice los temas de esta sección para encontrar material de referencia detallada sobre distintos aspectos de IAM y AWS STS.

Temas

Nombres de recursos de Amazon (ARN)

Los nombres de recursos de Amazon (ARN) identifican de forma exclusiva los recursos de AWS. Se requiere un ARN cuando sea preciso especificar un recurso de forma inequívoca para todo AWS, como en las políticas de IAM, las etiquetas de Amazon Relational Database Service (Amazon RDS) y las llamadas a la API.

Formato de ARN

A continuación se muestran los formatos generales para los ARN. Los formatos específicos dependen del recurso. Para utilizar un ARN, reemplace el texto en *cursiva* por la información específica del recurso. Tenga en cuenta que los ARN de algunos recursos omiten la región, el ID de la cuenta o ambos.

```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

partition

La partición en la que se encuentra el recurso. Una partición es un grupo de regiones de AWS. Cada cuenta de AWS está limitada a una partición.

Las siguientes son las particiones admitidas:

- `aws`: regiones de AWS
- `aws-cn` - Regiones de China
- `aws-us-gov`: regiones de AWS GovCloud (US)

service

El espacio de nombres del servicio que identifica el producto de AWS.

region

El código de región. Por ejemplo, `us-east-2` para la región Este de EE. UU. (Ohio). Para obtener la lista de códigos de región, consulte Puntos de conexión regionales en la Referencia general de AWS.

account-id

El ID de la cuenta de AWS que posee el recurso, sin los guiones. Por ejemplo, `123456789012`.

resource-type

El tipo de recurso. Por ejemplo, `vpc` para una nube privada virtual (VPC).

resource-id

El identificador del recurso. Es el nombre del recurso, el identificador del recurso o la ruta del recurso. Algunos identificadores de recurso incluyen un recurso principal (tipo-sub-recurso/recurso-principal/sub-recurso) o un calificador como una versión (tipo-recurso:nombre-recurso:calificador).

Ejemplos

Usuario de IAM

arn:aws:iam::*123456789012*:user/*johndoe*

Tema de SNS

arn:aws:sns:*us-east-1*:*123456789012*:*example-sns-topic-name*

VPC

arn:aws:ec2:*us-east-1*:*123456789012*:vpc/*vpc-0e9801d129EXAMPLE*

Consulta del formato de ARN para un recurso

El formato exacto de un ARN depende del tipo de servicio y recurso. Algunos ARN de recursos pueden incluir una ruta, una variable o un comodín. Para consultar el formato de ARN para un recurso de AWS específico, abra la Referencia de autorización de servicios, abra la página del servicio y navegue hasta la tabla de tipos de recursos.

Rutas de los ARN

Algunos ARN de recursos pueden incluir una ruta. Por ejemplo, en Amazon S3, el identificador de recursos es un nombre de objeto que puede incluir barras inclinadas (`/`) para formar una ruta. Del mismo modo, los nombres de usuario de IAM y nombres de grupo pueden incluir rutas. Solo se permiten caracteres alfanuméricos y los siguientes caracteres en las rutas del IAM: barra inclinada (`/`), más (`+`), igual (`=`), coma (`,`), punto final (`.`) arroba (`@`), guion bajo (`_`) y guion (`-`).

Uso de caracteres comodines en rutas

Las rutas pueden incluir un carácter comodín, por ejemplo, un asterisco (`*`). Por ejemplo, si escribe una política de IAM, puede especificar todos los usuarios de IAM que tienen la ruta `product_1234` con un carácter comodín de la siguiente manera:

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

Del mismo modo, puede especificar `user/*` para referirse a todos los usuarios o `group/*` para referirse a todos los grupos, como en los siguientes ejemplos:

```
"Resource":"arn:aws:iam::123456789012:user/*"
"Resource":"arn:aws:iam::123456789012:group/*"
```

En el siguiente ejemplo, se muestran los ARN para un bucket de Amazon S3 en los que el nombre de recurso incluye una ruta:

```
arn:aws:s3:::my_corporate_bucket/*
arn:aws:s3:::my_corporate_bucket/Development/*
```

Uso incorrecto del comodín

No puede utilizar un comodín en la parte del ARN que especifica el tipo de recurso, como el término `user` de un ARN de IAM. Por ejemplo, no se permite lo siguiente.

```
arn:aws:iam::123456789012:u*   <== not allowed
```

Identificadores de IAM

IAM utiliza varios identificadores diferentes para usuarios, grupos de usuarios, roles, políticas y certificados de servidor. En esta sección se describen los identificadores y cuándo se utilizan cada uno de ellos.

Temas

- Nombres fáciles de recordar y rutas
- ARN de IAM
- Identificadores únicos

Nombres fáciles de recordar y rutas

Cuando crea un usuario, un rol, un grupo de usuarios o una política, o cuando carga un certificado de servidor, le asigna un nombre descriptivo. Entre los ejemplos se incluyen Julio, AppPrueba1, Desarrolladores, AdministrarPermisosCredenciales o CertServidorProd.

Si utiliza la API de IAM o AWS Command Line Interface (AWS CLI) para crear recursos de IAM, puede agregar una ruta opcional. Puede utilizar una ruta única o anidar varias rutas como si se tratara de una estructura de carpetas. Por ejemplo, puede utilizar la ruta anidada `/division_abc/subdivision_xyz/product_1234/engineering/` para adaptarla a la estructura organizativa de su empresa. A continuación, puede crear una política para permitir a todos los usuarios de la ruta el acceso a la API del simulador de políticas. Para consultar esta política, visite IAM: obtiene acceso a la API del simulador de políticas en función de la ruta de acceso del usuario. Para obtener información acerca de cómo se puede especificar un nombre descriptivo, consulte la documentación de la API de usuario. Para obtener más ejemplos de cómo utilizar las rutas, consulte ARN de IAM.

Cuando se utiliza AWS CloudFormation para crear recursos, se puede especificar una ruta para los usuarios, los grupos de usuarios, los roles y las políticas administradas por el cliente.

Si tiene un usuario y un grupo de usuarios en la misma ruta, IAM no coloca automáticamente al usuario en ese grupo de usuarios. Por ejemplo, puede crear un grupo de usuarios de desarrolladores y especificar su ruta como `/division_abc/subdivision_xyz/product_1234/engineering/`. Si crea un usuario llamado Bob y le agrega la misma ruta, esto no significa que Bob sea automáticamente miembro del grupo de usuarios Developers. IAM no aplica ningún límite entre los usuarios o los grupos de usuarios en función de sus rutas. Puede haber usuarios con diferentes rutas que usen los mismos recursos, si se les haya concedido permiso sobre dichos recursos. El número y el tamaño de recursos de IAM en una cuenta de AWS son limitados. Para obtener más información, consulte IAM y cuotas de AWS STS.

ARN de IAM

La mayoría de los recursos tienen un nombre fácil de recordar, por ejemplo, un usuario denominado Bob o un grupo de usuarios denominado `Developers`. Sin embargo, el lenguaje de la política de permisos requiere que especifique el recurso o los recursos con el formato de nombre de recurso de Amazon (ARN) siguiente.

```
arn:partition:service:region:account:resource
```

Donde:

- `partition` identifica la partición para el recurso. Para las regiones estándar de AWS, la partición es `aws`. Si tiene recursos en otras particiones, la partición es `aws-`*`partitionname`*. Por ejemplo, la partición de los recursos de la región China (Pekín) es `aws-cn`. No se puede delegar el acceso entre cuentas en particiones diferentes.
- `service` identifica el producto de AWS. Los recursos de IAM siempre usan `iam`.
- `region` identifica la región del recurso. En el caso de los recursos de IAM, se deja siempre en blanco.
- La `account` especifica el ID de Cuenta de AWS sin guiones.
- `resource` identifica el recurso específico por su nombre.

Puede especificar los ARN de IAM y AWS STS utilizando la siguiente sintaxis. La parte de la región del ARN está en blanco porque los recursos de IAM son globales.

Sintaxis:

```
arn:aws:iam::account:root
```

```
arn:aws:iam::account:user/user-name-with-path
arn:aws:iam::account:group/group-name-with-path
arn:aws:iam::account:role/role-name-with-path
arn:aws:iam::account:policy/policy-name-with-path
arn:aws:iam::account:instance-profile/instance-profile-name-with-path
arn:aws:sts::account:federated-user/user-name
arn:aws:sts::account:assumed-role/role-name/role-session-name
arn:aws:iam::account:mfa/virtual-device-name-with-path
arn:aws:iam::account:u2f/u2f-token-id
arn:aws:iam::account:server-certificate/certificate-name-with-path
arn:aws:iam::account:saml-provider/provider-name
arn:aws:iam::account:oidc-provider/provider-name
```

Muchos de los ejemplos siguientes incluyen rutas en la parte del recurso de ARN. Las rutas no se pueden crear ni manipular en la AWS Management Console. Para utilizar rutas, debe trabajar con el recurso utilizando API de AWS, el AWS CLI o Tools for Windows PowerShell.

Ejemplos:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
arn:aws:iam::123456789012:role/S3Access
arn:aws:iam::123456789012:role/application_abc/component_xyz/RDSAccess
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
arn:aws:iam::123456789012:role/service-role/QuickSightAction
arn:aws:iam::123456789012:policy/UsersManageOwnCredentials
arn:aws:iam::123456789012:policy/division_abc/subdivision_xyz/UsersManageOwnCredentials
arn:aws:iam::123456789012:instance-profile/Webserver
arn:aws:sts::123456789012:federated-user/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/default (U2F security key)
arn:aws:iam::123456789012:server-certificate/ProdServerCert
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

```
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

Los siguientes ejemplos proporcionan más detalles para ayudarle a conocer el formato de ARN para los diferentes tipos de recursos de IAM y AWS STS.

- Un usuario de IAM en la cuenta:

> **Note**
>
> Cada nombre de usuario de IAM es único. El nombre de usuario no distingue mayúsculas de minúsculas para el usuario, por ejemplo, durante el proceso de inicio de sesión, pero distingue mayúsculas de minúsculas cuando se usa en una política o como parte de un ARN.

```
arn:aws:iam::123456789012:user/JohnDoe
```

- Otro usuario con una ruta que refleje un organigrama:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- Un grupo de usuarios de IAM:

```
arn:aws:iam::123456789012:group/Developers
```

- Un grupo de usuarios de IAM con una ruta:

```
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/product_A/Developers
```

- Un rol de IAM:

```
arn:aws:iam::123456789012:role/S3Access
```

- Un rol vinculado a un servicio:

```
arn:aws:iam::123456789012:role/aws-service-role/access-analyzer.amazonaws.com/
AWSServiceRoleForAccessAnalyzer
```

- Un rol de servicio:

```
arn:aws:iam::123456789012:role/service-role/QuickSightAction
```

- Una política administrada:

```
arn:aws:iam::123456789012:policy/ManageCredentialsPermissions
```

- Un perfil de instancia que se puede asociar a una instancia Amazon EC2:

```
arn:aws:iam::123456789012:instance-profile/Webserver
```

- Un usuario federado identificado en IAM como "Paulo":

```
arn:aws:sts::123456789012:federated-user/Paulo
```

- La sesión activa de alguien que asume el rol de "Accounting-Role", con el nombre de sesión de rol "Mary":

```
arn:aws:sts::123456789012:assumed-role/Accounting-Role/Mary
```

- El dispositivo de autenticación multifactor asignado al usuario llamado Jorge:

```
arn:aws:iam::123456789012:mfa/Jorge
```

- Un certificado de servidor

```
arn:aws:iam::123456789012:server-certificate/ProdServerCert
```

- Un certificado de servidor con una ruta que refleja un organigrama:

```
arn:aws:iam::123456789012:server-certificate/division_abc/subdivision_xyz/
ProdServerCert
```

- Proveedores de identidades (SAML y OIDC):

```
arn:aws:iam::123456789012:saml-provider/ADFSProvider
arn:aws:iam::123456789012:oidc-provider/GoogleProvider
arn:aws:iam::123456789012:oidc-provider/server.example.org
```

- Proveedor de identidad OIDC con una ruta que refleja la URL del proveedor de identidad OIDC de Amazon EKS:

```
arn:aws:iam::123456789012:oidc-provider/oidc.eks.us-west-2.amazonaws.com/id/
a1b2c3d4567890abcdefEXAMPLE11111
```

Otro ARN importante es el ARN del usuario raíz. Aunque no es un recurso de IAM, debe estar familiarizado con el formato de este ARN. A menudo se utiliza en el elemento `Principal` de un recurso basado en una política.

- La Cuenta de AWS muestra lo siguiente:

```
arn:aws:iam::123456789012:root
```

En el ejemplo siguiente se muestra una política que se podría asignar a Richard para que este pueda administrar sus propias claves de acceso. Observe que el recurso es el usuario de IAM Richard.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ManageRichardAccessKeys",
            "Effect": "Allow",
            "Action": [
                "iam:*AccessKey*",
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/division_abc/subdivision_xyz/Richard"
        },
        {
            "Sid": "ListForConsole",
            "Effect": "Allow",
            "Action": "iam:ListUsers",
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Cuando utiliza ARN para identificar recursos en una política de IAM, puede incluir variables de política. Las variables de política pueden incluir marcadores de posición para la información del entorno de ejecución (como el nombre del usuario) como parte del ARN. Para obtener más información, consulte Elementos de la política de IAM: variables y etiquetas

Uso de comodines y rutas en ARN

Puede utilizar comodines en la parte *recurso* del ARN para especificar varios usuarios o grupos de usuarios o políticas. Por ejemplo, para especificar todos los usuarios que trabajan en product_1234, debe utilizar:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/*
```

Si tiene usuarios cuyos nombres empiezan por la cadena `app_`, puede hacer referencia a todos ellos con el ARN siguiente.

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/product_1234/app_*
```

Para especificar todos los usuarios, grupos de usuarios o políticas de su Cuenta de AWS, utilice un comodín después de la parte `user/`, `group/` o `policy/` del ARN, respectivamente.

```
arn:aws:iam::123456789012:user/*
arn:aws:iam::123456789012:group/*
arn:aws:iam::123456789012:policy/*
```

Si especifica el siguiente ARN para un usuario de `arn:aws:iam::111122223333:user/*`, coincide con los dos ejemplos siguientes.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

Pero, si especifica el siguiente ARN para un usuario `arn:aws:iam::111122223333:user/division_abc*`, coincide con el segundo ejemplo, pero no con el primero.

```
arn:aws:iam::111122223333:user/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

No utilice un comodín en la parte `user/`, `group/` o `policy/` del ARN. Por ejemplo, IAM no permite lo siguiente:

```
arn:aws:iam::123456789012:u*
```

Example Uso de rutas y de ARN en un grupo de usuarios basado en un proyecto

Las rutas no se pueden crear ni manipular en la AWS Management Console. Para utilizar rutas, debe trabajar con el recurso utilizando API de AWS, el AWS CLI o Tools for Windows PowerShell.

En este ejemplo, Jules del grupo de usuarios Marketing_Admin crea un grupo de usuarios basado en proyectos dentro de la ruta /marketing/. Jules asigna usuarios de diferentes partes de la empresa al grupo de usuarios. El ejemplo sirve para mostrar que la ruta de un usuario no tiene ninguna relación con los grupos de usuarios en los que está el usuario.

El grupo de marketing tiene un producto nuevo listo para lanzarlo, y Jules crea un grupo de usuarios nuevo en la ruta /marketing/ llamado Widget_Launch. A continuación, Jules asigna la siguiente política al grupo de usuarios, que concede al grupo de usuarios acceso a objetos de la parte de `example_bucket` designada para este lanzamiento en concreto.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::example_bucket/marketing/newproductlaunch/widget/*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket*",
      "Resource": "arn:aws:s3:::example_bucket",
      "Condition": {"StringLike": {"s3:prefix": "marketing/newproductlaunch/widget/*"}}
    }
  ]
}
```

Después, Jules asigna los usuarios que trabajan en este lanzamiento al grupo de usuarios. Entre ellos figuran Patricia y Eli de la ruta /marketing/. También figuran Chris y Chloe de la ruta /sales/, y Alice y Jim de la ruta /legal/.

Identificadores únicos

Cuando IAM crea un usuario, un grupo de usuarios, un rol, una política, un perfil de instancias o un certificado de servidor, le asigna a cada uno un ID. El ID único se parece a lo siguiente:

```
AIDAJQABLZS4A3QDU576Q
```

Generalmente, se utilizan nombres descriptivos y ARN cuando se trabaja con entidades de IAM. De esta manera, no necesita conocer el ID único de un recurso específico. Sin embargo, el ID único puede ser útil en ocasiones, cuando no es práctico utilizar nombres fáciles de recordar.

Un ejemplo aborda el tema de volver a utilizar nombres fáciles de recordar en su Cuenta de AWS. En la cuenta, un nombre fácil de recordar de un usuario, un grupo de usuarios, un rol o una política tiene que ser único. Por ejemplo, podría crear un usuario de IAM llamado `John`. Su empresa utiliza Amazon S3 y tiene un bucket con carpetas para cada empleado. El usuario de IAM `John` es miembro de un grupo de usuarios de IAM denominado `User-S3-Access` con permisos que permiten a los usuarios obtener acceso únicamente a sus propias carpetas en el bucket. Para ver un ejemplo de cómo podría crear una política basada en identidad que permita a los usuarios de IAM obtener acceso a su propio objeto de bucket en S3 con el nombre descriptivo de los usuarios, consulte Amazon S3: permite a los usuarios de IAM obtener acceso a su directorio principal de S3, mediante programación y en la consola..

Supongamos que el empleado llamado John deja la empresa y usted elimina el usuario de IAM correspondiente denominado `John`. Pero, más tarde otro empleado también llamado John comienza a trabajar en la empresa y usted crea un nuevo usuario de IAM llamado `John`. Usted agrega el nuevo usuario de IAM denominado `John` al grupo de usuarios de IAM existente de `User-S3-Access`. Si la política de bucket especifica al grupo de usuario el nombre de usuario de IAM `John`, la política permite al nuevo usuario John obtener acceso a la información que dejó el anterior John.

En general, se recomienda especificar el ARN del recurso en las políticas en lugar de su ID exclusivo. Ahora bien, todos los usuarios de IAM tienen un ID único, aunque cree un usuario de IAM nuevo que vuelva a utilizar un nombre fácil de utilizar que ya haya eliminado antes. En el ejemplo, el usuario de IAM `John` de antes y el nuevo usuario `John` de IAM tienen ambos ID únicos, pero diferentes. Puede crear políticas de recursos que concedan acceso por ID único y no solo por

nombre de usuario. De esta manera, se reduce la posibilidad de que pueda conceder acceso sin querer a la información que un empleado no debería tener.

En el siguiente ejemplo se muestra cómo se podrían especificar ID exclusivos en el elemento `Principal` de una política basada en recursos.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::111122223333:role/role-name",
    "AIDACKCEVSQ6C2EXAMPLE",
    "AROADBQP57FF2AEXAMPLE"
  }
```

El siguiente ejemplo muestra cómo podría especificar ID únicos en el elemento `Condition` de una política que utiliza la clave de condición global `aws:userid`.

```
"Condition": {
    "StringLike": {
      "aws:userId": [
        "AIDACKCEVSQ6C2EXAMPLE",
        "AROADBQP57FF2AEXAMPLE:role-session-name",
        "AROA1234567890EXAMPLE:*",
        "111122223333"
      ]
    }
  }
```

Los ID de usuario también pueden ser útiles, por ejemplo, si quiere mantener su propia base de datos (u otro almacén) de información de usuarios o roles de IAM. El ID único puede proporcionar un identificador único para cada usuario o rol de IAM que cree. Esto es así cuando tiene usuarios o roles de IAM que reutilizan un nombre, como en el ejemplo anterior.

Descripción de los prefijos de ID único

IAM utiliza los siguientes prefijos para indicar a qué tipo de recurso se aplica cada ID único. Los prefijos pueden variar según el momento en que se crearon.

Prefix	Tipo de recurso
ABIA	Token al portador del servicio AWS STS

Prefix	Tipo de recurso
ACCA	Credenciales específicas del contexto
AGPA	Grupos de usuarios
AIDA	Usuario de IAM
AIPA	El perfil de instancias de Amazon EC2
AKIA	Clave de acceso
ANPA	Política administrada
ANVA	Versión en una política administrada
APKA	Clave pública
AROA	Rol
ASCA	Certificate
ASIA	Los ID de clave de acceso temporales (AWS STS) usan este prefijo, pero son únicos solo en combinación con la clave de acceso secreta y el token de sesión.

Obtener el identificador único

El ID único de un recurso de IAM no está disponible en la consola de IAM. Para obtener el ID único, puede utilizar los siguientes comandos de la AWS CLI o llamadas a la API de IAM.

AWS CLI:

- get-caller-identity
- get-group
- get-role
- get-user
- get-policy

- get-instance-profile
- get-server-certificate

API de IAM:

- GetCallerIdentity
- GetGroup
- GetRole
- GetUser
- GetPolicy
- GetInstanceProfile
- GetServerCertificate

IAM y cuotas de AWS STS

AWS Identity and Access Management (IAM) y AWS Security Token Service (STS) tienen cuotas que limitan el tamaño de los objetos. Estos servicios también limitan el nombre de un objeto, la cantidad de objetos que puede crear y la cantidad de caracteres que puede utilizar al pasar un objeto.

> **Note**
>
> Para obtener información sobre el límite de cuotas y el uso de IAM en el nivel de cuenta, utilice la operación GetAccountSummary de la API o el comando get-account-summary AWS CLI.

Requisitos de nombres de IAM

Los nombres IAM tienen los siguientes requisitos y restricciones:

- Los documentos de políticas solamente pueden contener los siguientes caracteres Unicode: tabulador horizontal (U+0009), salto de línea (U+000A), retorno de carro (U+000D) y caracteres comprendidos entre U+0020 y U+00FF.
- Los nombres de usuarios, grupos, roles, políticas, perfiles de instancias y certificados de servidor deben ser alfanuméricos, incluidos los siguientes caracteres comunes: más (+), igual (=), coma (,),

punto (.), arroba (@), guion bajo (_) y guion (-). Los nombres de ruta de acceso deben comenzar y finalizar con una barra diagonal (/).

- Los nombres de usuarios, grupos, roles y perfiles de instancia deben ser únicos dentro de la cuenta. No distinguen entre mayúsculas y minúsculas, por ejemplo, no puede crear dos grupos denominados **`ADMINS`** y **`admins`**.
- El valor de ID externo que un tercero utiliza para asumir un rol debe tener como mínimo 2 caracteres y como máximo 1224. El valor debe ser alfanumérico sin espacio en blanco. También puede incluir los símbolos siguientes: más (+), igual (=), coma (,), punto (.), arroba (@), dos puntos (:), barra inclinada (/) y guion (-). Para obtener más información acerca del ID externo, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.
- Los nombres de las políticas insertadas deben ser únicos para el usuario, grupo o rol en el que están insertadas. Pueden incluir caracteres básicos del alfabeto latino (ASCII), salvo los siguientes caracteres reservados: barra invertida (\), barra inclinada (/), asterisco (*), signo de interrogación (?) y espacio en blanco. Estos caracteres están reservados según RFC 3986, sección 2.2.
- Las contraseñas de usuarios (perfiles de inicio de sesión) pueden incluir caracteres básicos del alfabeto latino (ASCII).
- Los alias de ID de cuenta de Cuenta de AWS deben ser únicos en todos los productos de AWS y deben ser alfanuméricos de acuerdo con las convenciones de nomenclatura de DNS. Un alias debe ir en minúscula, no debe comenzar ni finalizar por un guion, no puede incluir dos guiones consecutivos y no puede ser un número de 12 dígitos.

Para obtener una lista de caracteres básicos del alfabeto latino (ASCII), diríjase a Library of Congress Basic Latin (ASCII) Code Table.

Cuotas de objetos de IAM

Las cuotas, también conocidas como límites en AWS, son el valor máximo de los recursos, acciones y elementos de su cuenta de Cuenta de AWS. Utilice Service Quotas para administrar sus cuotas de IAM.

Para obtener la lista de puntos de conexión y Service Quotas de IAM, consulte Puntos de conexión y cuotas de AWS Identity and Access Management en la Referencia general de AWS.

Para solicitar un aumento de cuota

1. Siga el procedimiento de inicio de sesión apropiado para su tipo de usuario como se describe en el tema Cómo iniciar sesión en AWS en la Guía del usuario de inicio de sesión en AWS para iniciar sesión en la AWS Management Console.
2. Abra la consola de Service Quotas.
3. En el panel de navegación, elija Servicios de AWS.
4. Elija el menú Este de EE. UU (Norte de Virginia) Región en la barra de navegación. Entonces busquen **IAM**.
5. Elija AWS Identity and Access Management (IAM), elija una cuota y siga las instrucciones para solicitar un aumento de cuota.

Para obtener más información, consulte Solicitud de un aumento de cuota en la Guía del usuario de Service Quotas.

Para ver un ejemplo de cómo solicitar un aumento de cuota de IAM mediante la consola Service Quotas, vea el siguiente vídeo.

Solicitar un aumento de cuota de IAM mediante la consola Service Quotas.

Puede solicitar un aumento de las cuotas predeterminadas para las cuotas ajustables IAM. Las solicitudes hasta maximum quota se aprueban automáticamente y se completan en pocos minutos.

A continuación, se presenta una tabla que enumera los recursos para los cuales las solicitudes de aumento de cuotas se pueden aprobar de forma automática.

Cuotas ajustables para los recursos de IAM

Recurso	Cuota predeterminada	Cuota máxima
Políticas administradas por el cliente por cuenta	1500	5000
Grupos por cuenta	300	500
Perfiles de instancia por cuenta	1 000	5000
Políticas administradas por rol	10	20

Recurso	Cuota predeterminada	Cuota máxima
Políticas administradas por usuario	10	20
Longitud de la política de confianza de rol	2048 caracteres	4096 caracteres
Roles por cuenta	1 000	5000
Certificados de servidor por cuenta	20	1 000

Cuotas del Analizador de acceso de IAM

Para obtener la lista de puntos de conexión y Service Quotas del Analizador de acceso de IAM, consulte Puntos de conexión y cuotas del Analizador de acceso de Iam en la Referencia general de AWS.

Cuotas de Funciones de IAM en cualquier lugar

Para obtener la lista de puntos de conexión de servicio y Service Quotas de Funciones de IAM en cualquier lugar, consulte Puntos de conexión y cuotas de Funciones de AWS Identity and Access Management en cualquier lugar en la Referencia general de AWS.

Límites de caracteres de IAM y STS

A continuación se indica la cantidad máxima de caracteres y los límites de tamaño para IAM y AWS STS. No es posible solicitar un aumento para los siguientes límites.

Descripción	Límite
Alias para un ID de Cuenta de AWS	De 3 a 63 caracteres
En el caso de las políticas insertadas	Puede agregar tantas políticas insertadas como quiera a un usuario de IAM, rol o grupo. Sin embargo, el tamaño total de las políticas agregadas (la suma del tamaño de todas las

Descripción	Límite
	políticas insertadas) por entidad no puede superar los siguientes límites: • El tamaño de política de usuario no puede tener más de 2048 caracteres. • El tamaño de política de rol no puede tener más de 10 240 caracteres. • El tamaño de política de grupo no puede tener más de 5120 caracteres. **Note** IAM no cuenta los espacios en blanco al calcular el tamaño de una política frente a estas limitaciones.
En el caso de las políticas administradas	• Cada política administrada no puede tener más de 6144 caracteres. **Note** IAM no cuenta los espacios en blanco al calcular el tamaño de una política frente a esta limitación.
Nombre del grupo	128 caracteres
Nombre de perfil de instancia	128 caracteres
Contraseña de un perfil de inicio de sesión	1-128 caracteres
Ruta	512 caracteres
Nombre de la política	128 caracteres

Descripción	Límite
Nombre de rol	64 caracteres **Important** Sin embargo, si desea utilizar un rol con la característica Cambiar rol en la AWS Management Console, entonces la combinación de `Path` y `RoleName` no puede superar los 64 caracteres.
Duración de la sesión de rol	12 horas Al asumir un rol desde la API o la AWS CLI, puede utilizar el `duration-seconds` parámetro de la CLI o el `DurationSeconds` parámetro de la API para solicitar una sesión de rol más larga. Puede especificar un valor comprendido entre 900 segundos (15 minutos) y la configuración de la duración máxima de la sesión para el rol, que puede oscilar entre 1-12 horas. Si no especifica un valor para el parámetro `DurationSeconds` , sus credencia les de seguridad serán válidas durante una hora. A los usuarios de IAM que cambian de rol en la consola se les concede la duración máxima de la sesión, o el tiempo restante de la sesión del usuario de IAM, lo que sea menor. La configuración de duración máxima de sesión no limita las sesiones asumidas por los servicios de AWS. Para obtener informaci ón sobre cómo ver el valor máximo para el rol, consulte Cómo consultar la configuración de la duración máxima de la sesión para un rol.

Descripción	Límite
Nombre de sesión de rol	64 caracteres
Políticas de sesión de rol	• El tamaño del documento de política JSON pasado y todos los caracteres del ARN de la política administrada pasados no pueden superar juntos los 2048 caracteres. • Puede pasar un máximo de 10 ARN de políticas administradas al crear una sesión. • Puede pasar una única política de JSON como documento cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. • Además, una conversión de AWS comprime las políticas de sesión pasadas y las etiquetas de sesión en un formato binario empaquetado que tiene un límite separado. El elemento de respuesta `PackedPolicySize` indica por porcentaje lo cerca que están las políticas y etiquetas de su solicitud al límite de tamaño superior. • Recomendamos que apruebe políticas de sesión mediante la AWS CLI o la API de AWS. La AWS Management Console podría añadir información adicional de la sesión de la consola a la política empaquetada.

Descripción	Límite
Etiquetas de sesión de rol	• Las etiquetas de sesión deben cumplir el límite de 128 caracteres de clave de etiqueta y el límite de valor de etiqueta de 256 caracteres. • Puede pasar hasta 50 etiquetas de sesión. • Una conversión de AWS comprime las políticas de sesión pasadas y las etiquetas de sesión en un formato binario empaquetado que tiene un límite separado. Puede pasar etiquetas de sesión utilizando las API de AWS CLI o AWS. El elemento de respuesta `PackedPolicySize` indica por porcentaje lo cerca que están las políticas y etiquetas de su solicitud al límite de tamaño superior.
Respuesta de autenticación de SAML codificada con base64	100 000 caracteres Este límite de caracteres se aplica a la operación `assume-role-with-saml` de la CLI o `AssumeRoleWithSAML` de la API.
Clave de etiqueta	128 caracteres Este límite de caracteres se aplica a las etiquetas de recursos de IAM y etiquetas de sesión.

Descripción	Límite
Valor de etiqueta	256 caracteres Este límite de caracteres se aplica a las etiquetas de recursos de IAM y etiquetas de sesión. Los valores de las etiquetas pueden estar vacíos, lo que significa que los valores de las etiquetas pueden tener una longitud de 0 caracteres.
Identificadores únicos creados por IAM	128 caracteres. Por ejemplo: • ID de usuarios que comienzan con `AIDA` • ID de grupos que comienzan con `AGPA` • ID de roles que comienzan con `AROA` • ID de políticas administradas que comienzan con `ANPA` • ID de certificados de servidor que comienzan con `ASCA` **Note** No pretende ser una lista exhaustiva ni tampoco es una garantía de que los ID de un determinado tipo comiencen únicamente con la combinación especificada de letras.
Nombre de usuario	64 caracteres

Puntos de conexión de VPC de tipo interfaz

Si utiliza Amazon Virtual Private Cloud (Amazon VPC) para alojar sus recursos de AWS, puede establecer una conexión privada entre su VPC y AWS Security Token Service (AWS STS). Puede utilizar esta conexión para permitir que AWS STS se comunique con sus recursos en su VPC sin pasar por la red pública de Internet.

Amazon VPC es un servicio de AWS que puede utilizar para lanzar recursos de AWS en una red virtual que defina. Con una VPC, puede controlar la configuración de la red, como el rango de direcciones IP, las subredes, las tablas de enrutamiento y las puertas de enlace de red. Para conectar su VPC a AWS STS, debe definir un punto de enlace de la VPC de tipo interfaz para AWS STS. El punto de enlace ofrece conectividad escalable de confianza con AWS STS sin necesidad de utilizar una gateway de Internet, una instancia de conversión de las direcciones de red (NAT) o una conexión de VPN. Para obtener más información, consulte ¿Qué es Amazon VPC? en la Guía del usuario de Amazon VPC.

Los puntos de enlace de la VPC de tipo interfaz utilizan la tecnología de AWS PrivateLink, una tecnología de AWS que permite la comunicación privada entre los servicios de AWS mediante una interfaz de red elástica con direcciones IP privadas. Para obtener más información, consulte AWS PrivateLink para servicios de AWS.

La siguiente información está dirigida a los usuarios de Amazon VPC. Para obtener más información, consulte Introducción a Amazon VPC en la Guía del usuario de Amazon VPC.

Disponibilidad

AWS STS actualmente admite puntos de enlace de la VPC en las regiones siguientes:

- US East (Ohio)
- Este de EE. UU. (Norte de Virginia)
- Oeste de EE. UU. (Norte de California)
- Oeste de EE. UU. (Oregón)
- África (Ciudad del Cabo)
- Asia-Pacífico (Hong Kong)
- Asia-Pacífico (Bombay)
- Asia-Pacífico (Osaka)

- Asia-Pacífico (Seúl)
- Asia-Pacífico (Singapur)
- Asia-Pacífico (Sídney)
- Asia-Pacífico (Tokio)
- Canadá (centro)
- China (Pekín)
- China (Ningxia)
- Europa (Fráncfort)
- Europa (Irlanda)
- Europa (Londres)
- Europa (Milán)
- Europa (París)
- Europa (Estocolmo)
- Medio Oriente (Baréin)
- América del Sur (São Paulo)
- AWS GovCloud (Este de EE. UU.)
- AWS GovCloud (Oeste de EE. UU.)

Creación de un punto de enlace de la VPC para AWS STS.

Para comenzar a utilizar AWS STS con su VPC, cree un punto de enlace de la VPC de tipo interfaz para AWS STS. Para obtener más información, consulte Acceso a un servicio de AWS a través de un punto de conexión de VPC de interfaz en la Guía del usuario de Amazon VPC.

Después de crear el punto de enlace de la VPC, debe utilizar el punto de enlace regional correspondiente para enviar sus solicitudes de AWS STS. AWS STS recomienda que utilice los métodos `setEndpoint` y `setRegion` para realizar llamadas a un punto de enlace regional. Puede utilizar el método `setRegion` de forma independiente para regiones habilitadas manualmente, como, por ejemplo, Asia Pacífico (Hong Kong). En este caso, las llamadas se dirigirán punto de enlace regional de STS. Para aprender a activar una región manualmente, consulte Administración de regiones de AWS en la Referencia general de AWS. Si utiliza el método `setRegion` de forma independiente para regiones habilitadas de forma predeterminada, las llamadas se dirigen al punto de enlace global `https://sts.amazonaws.com` .

Al utilizar puntos de enlace regionales, AWS STS llama a otros servicios de AWS con puntos de enlace públicos o puntos de enlace de la VPC de tipo interfaz privados, lo que esté en uso. Por ejemplo, supongamos que ha creado un punto de enlace de VPC de interfaz para AWS STS y que ya ha solicitado credenciales temporales de AWS STS desde los recursos que se encuentran en la VPC. En ese caso, estas credenciales comienzan a fluir a través del punto de enlace de la VPC de la interfaz de forma predeterminada. Para obtener más información acerca de la realización de solicitudes regionales utilizando AWS STS, consulte Administrar AWS STS en una Región de AWS.

Servicios de AWS que funcionan con IAM

Los servicios de AWS que se enumeran a continuación se agrupan alfabéticamente e incluyen información acerca de las características de IAM con las que son compatibles:

- Servicio: Puede elegir el nombre de un servicio para ver la documentación de AWS sobre la autorización y acceso de IAM para dicho servicio.
- Acciones: Puede especificar acciones individuales en una política. Si el servicio no es compatible con esta característica, Todas las acciones estará seleccionado en el editor visual. En un documento de política JSON, debe utilizar `*` en el elemento `Action`. Para obtener una lista de las acciones de cada servicio, vea Acciones, recursos y claves de condición para servicios de AWS.
- Permisos de nivel de recursos: Puede utilizar ARN para especificar recursos individuales en la política. Si el servicio no es compatible con esta característica, Todos los recursos estará seleccionado en el editor visual de la política. En un documento de política JSON, debe utilizar `*` en el elemento `Resource`. Algunas acciones, como, por ejemplo, `List*`, no admiten la especificación de un ARN, ya que están diseñadas para devolver varios recursos. Si un servicio admite esta característica para algunos recursos, pero no para otros, se indica mediante Parcial en la tabla. Consulte la documentación de ese servicio para obtener más información.
- Políticas basadas en recursos: Puede asociar políticas basadas en recursos a un recurso dentro del servicio. Las políticas basadas en recursos incluyen un elemento `Principal` para especificar qué identidades de IAM pueden obtener acceso a dicho recurso. Para obtener más información, consulte Políticas basadas en identidad y políticas basadas en recursos.
- ABAC (autorización basada en etiquetas): Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`. Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial. Para obtener más información sobre cómo definir permisos

basados en atributos como las etiquetas, consulte ¿Qué es ABAC para AWS?. Para ver un tutorial con los pasos para configurar ABAC, consulte Utilizar el control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

- Credenciales temporales: puede utilizar credenciales a corto plazo que se obtienen al iniciar sesión mediante IAM Identity Center, cambiar roles en la consola o que se generan mediante AWS STS en la AWS CLI o la API de AWS. Puede acceder a los servicios con un valor No solamente mientras utiliza sus credenciales de usuario a largo plazo de IAM. Esto incluye un nombre de usuario y una contraseña o sus claves de acceso de usuario. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.
- Roles vinculados a servicios: un rol vinculado a servicios es un tipo especial de rol de servicio que otorga al servicio permiso para obtener acceso a recursos en otros servicios en su nombre. Haga clic en el enlace Sí o Parcial para ver la documentación de los servicios compatibles con estos roles. En esta columna no se indica si el servicio utiliza roles de servicio estándar. Para obtener más información, consulte Uso de roles vinculados a servicios.
- Más información: si un servicio no admite totalmente una característica, puede analizar las notas al pie y buscar una entrada para ver las limitaciones y los enlaces a la información relacionada.

Servicios que funcionan con IAM

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Account Management	Sí	Sí	No	No	Sí	No
AWS Activate Console	Sí	No	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS AmplifyAdministrador	Sí	Sí	No	No	Sí	No
AWS Amplify	Sí	Sí	No	Parcial	Sí	No
Creador de UI de AWS Amplify	Sí	Sí	No	Sí	Sí	No
Las API de Apache Kafka para clústeres de Amazon MSK	Sí	Sí	No	No	Sí	No
Amazon API Gateway	Sí	Sí	Sí	No	Sí	Sí
Administración de Amazon API Gateway	Sí	Sí	No	Sí	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon API Gateway Management V2 (Administración V2)	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS App2Container	 Sí	 No	 No	 No	 Sí	 No
AWS AppConfig	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS App Fabric	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon AppFlow	 Sí	 Sí	 No	 Sí	 Sí	 No
Integraciones de aplicaciones de Amazon	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Aplicación de escalado automático	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Application Cost Profiler	 Sí	 No	 No	 No	 Sí	 No
AWS Application Discovery Arsenal	 Sí	 No	 No	 No	 Sí	 No
AWS Application Discovery Service	 Sí	 No	 No	 No	 Sí	 Sí
AWS Application Migration Service	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Servicio de transformación de aplicaciones de AWS	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS App Mesh	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS App Mesh Vista previa	 Sí	 Sí	 No	 No	 Sí	 Sí
AWS App Runner	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon AppStream 2.0	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS AppSync	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Artifact	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Athena	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Audit Manager	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Auto Scaling	 Sí	 No	 No	 No	 Sí	 Sí
Intercambio de datos B2B de AWS	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Backup	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
Puerta de enlace de AWS Backup	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Almacenamiento de AWS Backup	 Sí	 No	 No	 No	 Sí	 No
AWS Batch	 Sí	 Parcial	 No	 Sí	 Sí	 Sí
Amazon Bedrock	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Billing and Cost Management	 Sí	 No	 No	 No	 Sí	 No
AWS Billing and Cost Management Exportación de datos	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Billing Conductor	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Braket	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Servicio de presupuesto de AWS	 Sí	 Sí	 No	 No	 No	 No
AWS BugBust	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Certificate Manager (ACM)	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Chatbot	 Sí	 Sí	 No	 No	 Sí	 Sí
Amazon Chime	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Clean Rooms	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Clean Rooms ML	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Client VPN	 Sí	 Sí	 No	 No	 Sí	 Sí
AWS Cloud9	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
API de control de Nube de AWS	 Sí	 No	 No	 No	 Sí	 No
Amazon Cloud Directory	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS CloudFormation	Sí	Sí	No	Sí	Sí	No
Amazon CloudFront	Sí	Sí	No	Sí	Sí	Parcial (Información)
Amazon CloudFront KeyValueStore	Sí	Sí	No	No	Sí	No
AWS CloudHSM	Sí	Sí	No	Sí	Sí	Sí
AWS Cloud Map	Sí	Sí	No	Sí	Sí	No
Amazon CloudSearch	Sí	Sí	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS CloudShell	Sí	Sí	No	No	Sí	No
AWS CloudTrail	Sí	Sí	Parcial (Información)	Parcial (Información)	Sí	Sí
Datos de AWS CloudTrail	Sí	Sí	No	Sí	Sí	No
Amazon CloudWatch	Sí	Sí	No	Sí	Sí	Parcial (Información)
Amazon CloudWatch Application Insights (Información de aplicaciones de Amazon CloudWatch).	Sí	No	No	No	Sí	No
Amazon CloudWatch Evidently	Sí	Sí	No	Sí	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon CloudWatch Internet Monitor	 Sí	 Sí	 No	 Sí	 Sí	 No
Registros de Amazon CloudWatch	 Sí	 Sí	 Sí	 Parcial	 Sí	 Sí
Monitor de red Amazon CloudWatch	 Sí	 Sí	 No	 Sí	 Sí	 No
Administrador de acceso a la observabilidad de Amazon CloudWatch	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon CloudWatch RUM	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon CloudWatch Synthetics	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS CodeArtifact	 Sí	 Sí	 Sí	 Sí	 Sí	 No
AWS CodeBuild	 Sí	 Sí	 Sí (Información)	 Parcial (Información)	 Sí	 No
Amazon CodeCatalyst	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS CodeCommit	 Sí	 Sí	 No	 Sí	 Sí	 No
Conexiones de código de AWS	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS CodeDeploy	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Servicio de comandos de host seguro de AWS CodeDeploy	Sí	No	No	No	Sí	No
Amazon CodeGuru Profiler (Generador de perfiles de Amazon CodeGuru)	Sí	Sí	No	Sí	Sí	Sí
Amazon CodeGuru Reviewer (Revisor de Amazon CodeGuru).	Sí	Sí	No	Sí	Sí	Sí
Amazon CodeGuru Security	Sí	Sí	No	Sí	Sí	No
AWS CodePipeline	Sí	Parcial	No	Sí	Sí	No
AWS CodeStar	Sí	Parcial	No	Sí	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS CodeStar Connections	Sí	Sí	No	Sí	Sí	Sí
Notificaciones AWS CodeStar	Sí	Sí	No	Sí	Sí	Sí
Amazon CodeWhisperer	Sí	Sí	No	Sí	Sí	Sí
Amazon Cognito	Sí	Sí	No	Sí	Sí	Sí
Amazon Cognito Sync	Sí	Sí	No	No	Sí	Sí
Grupos de usuarios de Amazon Cognito	Sí	Sí	No	Sí	Sí	Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Comprehend	Sí	Sí	No	Sí	Sí	No
Amazon Comprehend Medical	Sí	No	No	No	Sí	No
AWS Compute Optimizer	Sí	No	No	No	Sí	Sí
AWS Config	Sí	Parcial (Información)	No	Sí	Sí	Sí
Amazon Connect	Sí	Sí	No	Sí	Sí	Sí
Amazon Connect Cases	Sí	Sí	No	Sí	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Connect Customer Profiles (Perfiles de clientes de Amazon Connect).	Sí	Sí	No	Sí	Sí	Sí
Comunicaciones salientes de gran volumen de Amazon Connect	Sí	Sí	No	Sí	Sí	No
Amazon Connect Voice ID	Sí	Sí	No	Sí	Sí	No
AWS Console Mobile Application	Sí	Sí	No	No	Sí	No
Facturación unificada de AWS	Sí	No	No	No	Sí	No
AWS Control Tower	Sí	Sí	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Cost and Usage Report	Sí	Sí	No	No	Sí	No
AWS Cost Explorer	Sí	Sí	No	Sí	Sí	No
Centro de optimización de costes de AWS	Sí	No	No	No	Sí	No
Servicio de verificación de clientes de AWS	Sí	No	No	No	Sí	No
AWS Database Migration Service	Sí	Sí	No (Información)	Sí	Sí	Sí
Database Query Metadata Service	Sí	No	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Data Exchange	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Data Lifecycle Manager	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Data Pipeline	 Sí	 Sí	 No	 Parcial	 Sí	 No
AWS DataSync	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon DataZone	 Sí	 No	 No	 No	 Sí	 No
AWS Deadline Cloud	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS DeepComposer	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS DeepRacer	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Detective	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Device Farm	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon DevOps Guru	 Sí	 Sí	 No	 No	 Sí	 Sí
Herramientas de diagnóstico AWS de	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Direct Connect	Sí	Sí	No	Sí	Sí	Sí
AWS Directory Service	Sí	Sí	No	Sí	Sí	No
Clústeres elásticos de Amazon DocumentDB	Sí	Sí	No	Sí	Sí	No
Amazon DynamoDB Accelerator (DAX) (Acelerador de Amazon DynamoDB (DAX)).	Sí	Sí	No	No	Sí	Sí
Amazon DynamoDB	Sí	Sí	Sí	No	Sí	No
Amazon Elastic Compute Cloud (Amazon EC2)	Sí	Parcial	No	Sí	Sí	Parcial (Información)

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon EC2 Auto Scaling	 Sí	 Sí	 No	 Sí	 Sí	 Sí
EC2 Image Builder	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Conexión de la instancia de Amazon EC2	 Sí	 Sí	 No	 No	 Sí	 Sí
Amazon ElastiCache	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Elastic Beanstalk	 Sí	 Parcial	 No	 Sí	 Sí	 Sí
Amazon Elastic Block Store (Amazon EBS)	 Sí	 Parcial	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Elastic Container Registry (Amazon ECR)	Sí	Sí	Sí	Sí	Sí	Sí
Amazon Elastic Container Registry Público (Público de Amazon ECR)	Sí	Sí	No	Sí	Sí	No
Amazon Elastic Container Service (Amazon ECS)	Sí	Parcial (Información)	No	Sí	Sí	Sí
AWS Elastic Disaster Recovery	Sí	Sí	No	Sí	Sí	Sí
Amazon Elastic File System (Amazon EFS)	Sí	Sí	Sí	Parcial	Sí	Sí
Amazon Elastic Inference	Sí	Sí	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Elastic Kubernetes Service (Amazon EKS)	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Elastic Kubernetes Service (Amazon EKS) Auth	 Sí	 Sí	 No	 No	 Sí	 No
Elastic Load Balancing de AWS	 Sí	 Parcial	 No	 Parcial	 Sí	 Sí
Amazon Elastic Transcoder	 Sí	 Sí	 No	 No	 Sí	 No
Servicio de AWS de activación de software y dispositivos elementales	 Sí	 Sí	 No	 Sí	 Sí	 No
Software y dispositivos de AWS Elemental	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Elemental MediaConnect	Sí	Sí	No	No	Sí	Sí
AWS Elemental MediaConvert	Sí	Sí	No	Sí	Sí	No
AWS Elemental MediaLive	Sí	Sí	No	Sí	Sí	No
AWS Elemental MediaPackage	Sí	Sí	No	Sí	Sí	Parcial (Información)
AWS Elemental MediaPackage V2	Sí	Sí	No	Sí	Sí	No
AWS Elemental MediaPackage VOD	Sí	Sí	No	Sí	Sí	Parcial (Información)

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Elemental MediaStore	 Sí	 Sí	 Sí	 Sí	 Sí	 No
AWS Elemental MediaTailor	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Casos de asistencia de AWS Elemental	 Sí	 No	 No	 No	 Sí	 No
Contenido de asistencia de AWS Elemental	 Sí	 No	 No	 No	 Sí	 No
Amazon EMR	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon EMR en EKS	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon EMR Serverless	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Entity Resolution	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon EventBridge	 Sí	 Sí	 Sí	 Sí	 Sí	 No
Canalizaciones de Amazon EventBridge	 Sí	 Sí	 No	 Sí	 Sí	 No
Programador de Amazon EventBridge	 Sí	 Sí	 No	 Sí	 Sí	 No
Esquemas de Amazon EventBridge	 Sí	 Sí	 Sí	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Fault Injection Service	Sí	Sí	No	Sí	Sí	Sí
Amazon FinSpace	Sí	Sí	No	Sí	Sí	Sí
API de Amazon FinSpace	Sí	Sí	No	No	Sí	No
AWS Firewall Manager	Sí	Sí	No	Sí	Sí	Parcial
Fleet Hub for AWS IoT Device Management	Sí	Sí	No	Sí	Sí	No
Amazon Forecast	Sí	Sí	No	Sí	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Fraud Detector	 Sí	 Sí	 No	 Sí	 Sí	 No
FreeRTOS	 Sí	 Sí	 No	 Sí	 Sí	 No
Nivel gratuito de AWS	 Sí	 No	 No	 No	 Sí	 No
Amazon FSx	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon GameLift	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Global Accelerator	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Glue	 Sí	 Sí	 Sí	 Parcial	 Sí	 No
AWS Glue DataBrew	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Ground Station	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Ground Truth Labeling	 Sí	 No	 No	 No	 Sí	 No
Amazon GuardDuty	 Sí	 Sí	 No	 Sí	 Sí	 Sí
API y notificaciones de AWS Health	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS HealthImaging	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS HealthLake	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS HealthOmics	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Honeycode	 Sí	 Sí	 No	 No	 Sí	 No
AWS IAM Identity Center	 Sí	 Sí	 No	 Parcial	 Sí	 Sí
Directorio de IAM Identity Center	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Almacén de identidades de IAM Identity Center	 Sí	 Sí	 No	 No	 Sí	 No
Servicio OIDC de IAM Identity Center	 Sí	 Sí	 No	 No	 Sí	 No
AWS Identity and Access Management (IAM)	 Sí	 Sí	 Parcial (Información)	 Parcial (Información)	 Parcial (Información)	 No
Analizador de acceso de AWS Identity and Access Management	 Sí	 Sí	 No	 Sí	 Sí	 Parcial
Funciones de AWS Identity and Access Management en cualquier lugar	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Autenticación de almacén de identidades de AWS	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Identity Sync	 Sí	 Sí	 No	 No	 Sí	 No
AWS Import/Export	 Sí	 No	 No	 No	 Sí	 No
Amazon Inspector	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Inspector Classic	 Sí	 No	 No	 No	 Sí	 Sí
Amazon InspectorScan	 Sí	 No	 No	 No	 Sí	 No
Servicio de videos interactivos de Amazon	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Interactive Video Service Chat	 Sí	 Sí	 No	 Sí	 Sí	 No
Facturación de AWS	 Sí	 No	 No	 No	 Sí	 No
AWS IoT 1-Click	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IoT Analytics	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IoT	 Sí	 Sí	 Parcial (Información)	 Sí	 Sí	 No
AWS IoT Core Device Advisor	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS IoT Device Tester	 Sí	 No	 No	 No	 Sí	 No
AWS IoT Events	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IoT FleetWise	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IoT Greengrass	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IoT Greengrass V2	 Sí	 Sí	 No	 Parcial	 Sí	 No
AWS IoT Jobs DataPlane	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS IoT RoboRunner	 Sí	 Sí	 No	 No	 Sí	 No
AWS IoT SiteWise	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS IoT TwinMaker	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS IoT Wireless	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS IQ	 Sí	 Sí	 No	 No	 Sí	 Sí
Permisos de AWS IQ	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Kendra	 Sí	 Sí	 No	 Sí	 Sí	 No
Clasificación de Amazon Kendra Intelligent	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Key Management Service (AWS KMS)	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
Amazon Keyspaces (para Apache Cassandra)	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Managed Service para Apache Flink	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Managed Service para Apache Flink V2	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Data Firehose	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Kinesis Data Streams	 Sí	 Sí	 Sí	 No	 Sí	 No
Amazon Kinesis Video Streams	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Lake Formation	 Sí	 No	 No	 No	 Sí	 Sí
AWS Lambda	 Sí	 Sí	 Sí	 Parcial (Información)	 Sí	 Parcial (Información)
AWS Launch Wizard	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Lex	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Lex V2	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
AWS License Manager	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Administrador de suscripciones de Linux de AWS License Manager	 Sí	 No	 No	 No	 Sí	 No
Suscripciones de usuario de AWS License Manager	 Sí	 No	 No	 No	 Sí	 Sí
Amazon Lightsail	 Sí	 Parcial (Información)	 No	 Parcial (Información)	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Location Service	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Lookout for Equipment	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Lookout for Metrics	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Lookout for Vision	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Machine Learning	 Sí	 Sí	 No	 No	 Sí	 No
Amazon Macie	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Mainframe Modernization	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Managed Blockchain	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Managed Blockchain Query	 Sí	 No	 No	 No	 Sí	 No
Amazon Managed Grafana	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Managed Service para Prometheus	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Managed Streaming for Apache Kafka (MSK)	 Sí	 Sí	 Parcial (Información)	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Managed Streaming para Kafka Connect	 Sí	 Sí	 No	 No	 Sí	 Sí
Amazon Managed Workflows para Apache Airflow	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Marketplace	 Sí	 No	 No	 No	 Sí	 Sí
AWS Marketplace Catálogo	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Marketplace Commerce Analytics	 Sí	 No	 No	 No	 No	 No
AWS Marketplace Servicio de Implementación	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Marketplace Discovery	 Sí	 No	 No	 No	 Sí	 No
AWS Marketplace Entitlement Service	 Sí	 No	 No	 No	 Sí	 No
Servicio de compilación de imágenes de AWS Marketplace	 Sí	 No	 No	 No	 Sí	 No
AWS Marketplace Management Portal	 Sí	 No	 No	 No	 Sí	 No
AWS Marketplace Metering Service	 Sí	 No	 No	 No	 Sí	 No
AWS Marketplace Private Marketplace	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Integración de sistemas de adquisición de AWS Marketplace	 Sí	 No	 No	 No	 Sí	 No
Denunciar al vendedor de AWS Marketplace	 Sí	 Sí	 No	 No	 Sí	 No
AWS Marketplace Vendor Insights	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Mechanical Turk	 Sí	 No	 No	 No	 Sí	 No
Amazon MediaImport	 Sí	 No	 No	 No	 No	 No
Amazon MemoryDB para Redis	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Servicio de entrega de mensajes de Amazon	 Sí	 No	 No	 No	 Sí	 No
Amazon Message Gateway Service	 Sí	 No	 No	 No	 Sí	 No
AWS Microservice Extractor for .NET	 Sí	 No	 No	 No	 Sí	 No
AWS Créditos del programa de aceleración de la migración	 Sí	 Sí	 No	 No	 Sí	 No
AWS Migration Hub	 Sí	 Sí	 No	 No	 Sí	 Sí
AWS Migration Hub Orchestrator	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Migration Hub Refactor Spaces	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
Recomendaciones de estrategia de AWS Migration Hub	 Sí	 No	 No	 No	 Sí	 Sí
Amazon Monitron	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon MQ	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Neptune	 Sí	 Sí	 No	 No	 Sí	 Sí
Análisis de Amazon Neptune	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Network Firewall	Sí	Sí	No	Sí	Sí	Sí
AWS Network Manager	Sí	Sí	No	Sí	Sí	Sí (Información)
AWS Network Manager Chat	Sí	No	No	No	Sí	No
Amazon Nimble Studio	Sí	Sí	No	Sí	Sí	No
Amazon One Enterprise	Sí	Sí	No	Sí	Sí	No
Amazon OpenSearch Ingestion	Sí	Sí	No	Sí	Sí	Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon OpenSearch Serverless	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon OpenSearch Service	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
AWS OpsWorks	 Sí	 Sí	 No	 No	 Sí	 No
Administración de la configuración de AWS OpsWorks	 Sí	 Sí	 No	 No	 Sí	 No
AWS Organizations	 Sí	 Sí	 Sí	 Sí	 No	 Sí
AWS Outposts	 Sí	 Sí	 No	 Sí	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Panorama	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Administración central de cuentas de AWS Partner	 Sí	 No	 No	 No	 Sí	 No
AWS Payment Cryptography	 Sí	 Sí	 No	 Sí	 Sí	 No
Pagos de AWS	 Sí	 No	 No	 No	 Sí	 No
Información de rendimiento de AWS	 Sí	 Sí	 No	 No	 Sí	 No
Amazon Personalize	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Pinpoint	 Sí	 Sí	 No	 Sí	 Sí	 No
Servicio de correo electrónico de Amazon Pinpoint	 Sí	 Sí	 No	 Sí	 Sí	 No
Servicio de SMS y voz de Amazon Pinpoint	 Sí	 No	 No	 No	 Sí	 No
Servicio de SMS y voz de Amazon Pinpoint V2	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Polly	 Sí	 Sí	 No	 No	 Sí	 No
Lista de precios de AWS	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Private 5G	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Private CA Conector para Active Directory	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Private Certificate Authority (AWS Private CA)	 Sí	 Sí	 Sí	 Sí	 Sí	 No
AWS Proton	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Consola de órdenes de compra	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Q	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Q Business	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Q in Connect	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Quantum Ledger Database (Amazon QLDB)	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon QuickSight	 Sí	 Sí	 No	 Sí	 Sí	 No
API de datos de Amazon RDS	 Sí	 Sí	 No	 No	 Sí	 No
Amazon RDS IAM Authentication (Autenticación de IAM de Amazon RDS)	 Sí	 Sí	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Papelera de reciclaje	Sí	Sí	No	Sí	Sí	No
Amazon Redshift	Sí	Sí	No	Sí	Sí	Sí
API de datos de Amazon Redshift	Sí	Sí	No	No	Sí	No
Amazon Redshift Serverless	Sí	Sí	Sí	Sí	Sí	No
Amazon Rekognition	Sí	Sí	Parcial (Información)	Sí	Sí	No
Amazon Relational Database Service (Amazon RDS) (Información)	Sí	Sí	No	Sí	Sí	Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS re:PostPrivate	Sí	Sí	No	Sí	Sí	Sí
AWS Resilience Hub	Sí	Sí	No	Sí	Sí	No
AWS Resource Access Manager (AWS RAM)	Sí	Sí	No	Sí	Sí	Sí
Explorador de recursos de AWS	Sí	Sí	No	Sí	Sí	Sí
AWS Resource Groups	Sí	Sí	No	Sí	Parcial (Información)	No
AWS Resource Groups Tagging API	Sí	No	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon RHEL Knowledgebase Portal	Sí	No	No	No	Sí	No
AWS RoboMaker	Sí	Sí	No	Sí	Sí	Sí
Amazon Route 53	Sí	Sí	No	No	Sí	No
Controlador de recuperación de aplicaciones de Amazon Route 53: Cambio de zona	Sí	Sí	No	No	Sí	No
Amazon Route 53 Dominios	Sí	No	No	No	No	No
Amazon Route 53 Recovery Cluster	Sí	Sí	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Route 53 Recovery Controls	Sí	Sí	No	Sí	Sí	No
Preparación para recuperación de Amazon Route 53	Sí	Sí	No	Sí	Sí	Sí
Amazon Route 53 Resolver	Sí	Sí	No	Sí	Sí	Sí
Amazon S3 Express	Sí	Sí	No	No	Sí	No
Amazon S3 Glacier	Sí	Sí	Sí	Sí	Sí	No
Amazon SageMaker	Sí	Sí	No	Sí	Sí	Parcial (Información)

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Capacidades geoespaciales de Amazon SageMaker	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon SageMaker Ground Truth Synthetic	 Sí	 No	 No	 No	 Sí	 No
AWS Savings Plans	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Secrets Manager	 Sí	 Sí	 Sí	 Sí	 Sí	 No
AWS Security Hub	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Amazon Security Lake	 Sí	 Sí	 No	 No	 Sí	 Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Security Token Service (AWS STS)	 Sí	 Parcial (Información)	 No	 Sí	 Parcial (Información)	 No
AWS Serverless Application Repository	 Sí	 Sí	 Sí	 No	 Sí	 No
AWS Service Catalog	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Service Quotas	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Shield	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Signer	 Sí	 Sí	 Sí	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon SimpleDB	Sí	Sí	No	No	Sí	No
Amazon Simple Email Service (Amazon SES) v2	Sí	Parcial (Información)	Sí	Sí	Parcial (Información)	No
Amazon Simple Notification Service (Amazon SNS)	Sí	Sí	Sí	Sí	Sí	No
Amazon Simple Queue Service (Amazon SQS)	Sí	Sí	Sí	Parcial	Sí	No
Amazon Simple Storage Service (Amazon S3)	Sí	Sí	Sí	Parcial (Información)	Sí	Parcial (Información)

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Simple Storage Service (Amazon S3) Objeto Lambda	 Sí	 Sí	 No	 No	 Sí	 No
Amazon Simple Storage Service (Amazon S3) en AWS Outposts	 Sí	 Sí	 Sí	 No	 Sí	 Sí
Amazon Simple Workflow Service (Amazon SWF)	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS SimSpace Weaver	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Site-to-Site VPN	 Sí	 Sí	 No	 No	 Sí	 Sí
AWS Snowball	 Sí	 No	 No	 No	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Snowball Edge	 Sí	 No	 No	 No	 Sí	 No
AWS Snow Device Management	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS SQL Workbench	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Step Functions	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Storage Gateway	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Supply Chain	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Support App in Slack	 Sí	 No	 No	 No	 Sí	 No
AWS Support	 Sí	 No	 No	 No	 Sí	 Sí
AWS Support Planes	 Sí	 No	 No	 No	 Sí	 No
AWS Sustainability	 Sí	 No	 No	 No	 Sí	 No
AWS Systems Manager	 Sí	 Sí	 No	 Sí	 Sí	 Sí
AWS Systems Manager para SAP	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Systems Manager GUI Connect	 Sí	 No	 No	 No	 Sí	 No
AWS Systems Manager Incident Manager	 Sí	 Sí	 Sí	 Sí	 Sí	 Sí
Contactos de AWS Systems Manager Incident Manager	 Sí	 Sí	 Sí	 No	 Sí	 No
Tag Editor	 Sí	 No	 No	 No	 Sí	 No
AWS Tax Settings	 Sí	 No	 No	 No	 Sí	 No
Creador de redes de telecomunicaciones de AWS	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Textract	 Sí	 No	 No	 No	 Sí	 No
Amazon Timestream	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon Timestream Influxdb	 Sí	 Sí	 No	 Sí	 Sí	 Sí
API de AWS Tiros (para el Analizador de accesibilidad)	 Sí	 No	 No	 No	 No	 No
Amazon Transcribe	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Transfer Family	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Translate	Sí	Sí	No	Sí	Sí	No
AWS Trusted Advisor	Parcial (Información)	Sí	No	No	Parcial	Sí
Notificaciones de usuario de AWS	Sí	Sí	No	Sí	Sí	Sí
Notificaciones de usuario y contactos de AWS	Sí	Sí	No	Sí	Sí	No
Acceso verificado de AWS	Sí	No	No	No	Sí	No
Amazon Verified Permissions	Sí	Sí	No	No	Sí	No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Amazon Virtual Private Cloud (Amazon VPC)	Sí	Parcial (Información)	Parcial (Información)	Sí	Sí	Parcial (Información)
Amazon VPC Lattice	Sí	Sí	No	Sí	Sí	No
Servicios de Amazon VPC Lattice	Sí	Sí	No	No	Sí	No
AWS WAF	Sí	Sí	No	Sí	Sí	Sí
AWS WAF Classic	Sí	Sí	No	Sí	Sí	Sí
AWS WAF Regional	Sí	Sí	No	Sí	Sí	Sí

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
AWS Well-Architected Tool	 Sí	 Sí	 No	 Sí	 Sí	 No
AWS Wickr	 Sí	 Sí	 No	 Sí	 Sí	 No
Amazon WorkDocs	 Sí	 No	 No	 No	 Sí	 No
Amazon WorkMail	 Sí	 Sí	 No	 Sí	 Sí	 Sí
Flujo de mensajes de Amazon WorkMail	 Sí	 Sí	 No	 No	 Sí	 No
Amazon WorkSpaces	 Sí	 Sí	 No	 Sí	 Sí	 No

Servicio	Acciones	Permisos de nivel de recursos	Políticas basadas en recursos	ABAC	Credenciales temporales	Roles vinculados a servicios
Cliente ligero de Amazon WorkSpaces	Sí	Sí	No	Sí	Sí	No
Amazon WorkSpaces Web	Sí	Sí	No	Sí	Sí	Sí
AWS X-Ray	Sí	Parcial (Información)	No	Parcial (Información)	Sí	No

Más información

Amazon CloudFront

CloudFront no tiene roles vinculados a servicios, pero Lambda@Edge sí. Para obtener más información, consulte Uso de roles vinculados a servicios de en la Guía para desarrolladores de Lambda@Edge en la Guía para desarrolladores de Amazon CloudFront.

AWS CloudTrail

CloudTrail admite políticas basadas en recursos solo en los canales de CloudTrail utilizados para las integraciones de CloudTrail Lake con orígenes de eventos externos a AWS.

CloudTrail admite el control de acceso basado en etiquetas para canales y almacenes de datos de eventos de CloudTrail Lake. CloudTrail no admite controles de acceso basados en etiquetas para registros.

Amazon CloudWatch

Los roles vinculados a servicios de CloudWatch no se pueden crear con la AWS Management Console y solo admiten la característica Acciones de alarma.

AWS CodeBuild

CodeBuild admite el uso compartido de recursos entre cuentas mediante AWS RAM.

CodeBuild admite ABAC para acciones basadas en proyectos.

AWS Config

AWS Config admite permisos de nivel de recursos para la agregación de datos de varias cuentas y regiones, y reglas de AWS Config. Para obtener una lista de los recursos admitidos, consulte la sección Agregación de datos de varias cuentas y regiones y la sección Reglas de AWS Config de la Guía de la API de AWS Config.

AWS Database Migration Service

Puede crear y modificar las políticas asociadas a las claves de cifrado de AWS KMS que crea para cifrar los datos migrados a los puntos de conexión de destino compatibles. Los puntos de conexión de destino incluyen Amazon Redshift y Amazon S3. Para obtener más información, consulte Creación y uso de claves AWS KMS para cifrar datos de destino de Amazon Redshift y Creación de claves AWS KMS para cifrar objetos de destino de Amazon S3 en la Guía del usuario de AWS Database Migration Service.

Amazon Elastic Compute Cloud

Los roles vinculados a servicios de Amazon EC2 solo se pueden utilizar para las siguientes características: Solicitudes de instancia de spot, Solicitudes de flota de spot, Flotas de Amazon EC2 y Lanzamiento rápido de instancias de Windows.

Amazon Elastic Container Service

Solo algunas acciones de Amazon ECS admiten los permisos de nivel de recursos.

AWS Elemental MediaPackage

MediaPackage admite roles vinculados a servicios para publicar registros de acceso de clientes en CloudWatch, pero no para otras acciones de la API.

AWS Identity and Access Management

IAM solo admite un tipo de política basada en recursos, el llamado política de confianza de rol, que se asocia a un rol de IAM. Para obtener más información, consulte Conceder permisos de usuario para cambiar de rol.

IAM admite el control de acceso basado en etiquetas para la mayoría de los recursos de IAM. Para obtener más información, consulte Etiquetado de recursos de IAM.

Solo algunas de las acciones de la API de IAM se pueden llamar con credenciales temporales. Para obtener más información, consulte Comparación de opciones de API.

AWS IoT

Los dispositivos conectados a AWS IoT se autentican mediante certificados X.509 o identidades de Amazon Cognito. Puede asociar políticas de AWS IoT a un certificado X.509 o una identidad Amazon Cognito con el fin de controlar qué está autorizado a hacer el dispositivo. Para obtener más información, consulte Seguridad e identidad para AWS IoT en la Guía para desarrolladores de AWS IoT.

AWS Lambda

Lambda admite el control de acceso basado en atributos (ABAC) para acciones de la API que utilizan una función de Lambda como recurso necesario. No se admiten capas, asignaciones de orígenes de eventos ni recursos de configuración de firma de código.

Lambda no tiene roles vinculados a servicios, pero Lambda@Edge sí. Para obtener más información, consulte Uso de roles vinculados a servicios de en la Guía para desarrolladores de Lambda@Edge en la Guía para desarrolladores de Amazon CloudFront.

Amazon Lightsail

Lightsail admite parcialmente permisos de nivel de recursos y ABAC. Para obtener información, consulte Acciones, recursos y claves de condición de Amazon Lightsail.

Amazon Managed Streaming for Apache Kafka (MSK)

Puede adjuntar una política de clúster a un clúster de Amazon MSK que se haya configurado para la conectividad de varias VPC.

AWS Network Manager

WAN en la nube de AWS también admite roles vinculados a servicios. Para obtener más información, consulte Roles vinculados a servicios de WAN en la nube de AWS en la Guía de WAN en la nube de AWS para Amazon VPC.

Amazon Relational Database Service

Amazon Aurora es un motor de base de datos relacional completamente administrado compatible con MySQL y PostgreSQL. Puede elegir Aurora MySQL o Aurora PostgreSQL como opción del motor de base de datos cuando configura servidores de base de datos nuevos mediante Amazon RDS. Para obtener más información, consulte Administración de identidades y accesos para Amazon Aurora en la Guía del usuario de Amazon Aurora.

Amazon Rekognition

Las políticas basadas en recursos solo se admiten para copiar modelos de etiquetas personalizadas de Amazon Rekognition.

AWS Resource Groups

Los usuarios pueden asumir un rol con una política que permita operaciones de Resource Groups.

Amazon SageMaker

Los roles vinculados a servicios están disponibles actualmente para los trabajos de entrenamiento de SageMaker Studio y SageMaker.

AWS Security Token Service

AWS STS no tiene “recursos”, pero permite la restricción del acceso de los usuarios de forma similar. Para obtener más información, consulte Denegar el acceso a las credenciales de seguridad temporales según el nombre.

Solo algunas de las operaciones de la API de AWS STS admiten llamadas con credenciales temporales. Para obtener más información, consulte Comparación de opciones de API.

Amazon Simple Email Service

Solo puede utilizar permisos de nivel de recursos en instrucciones de política que se refieran a las acciones relacionadas con el envío de correo electrónico, como `ses:SendEmail` o `ses:SendRawEmail`. En el caso de instrucciones de política que se refieran a otras acciones, el elemento Resource solo puede contener `*`.

Solo la API de Amazon SES es compatible con las credenciales de seguridad temporales. La interfaz SMTP de Amazon SES no es compatible con las credenciales de SMTP que se derivan de credenciales de seguridad temporales.

Amazon Simple Storage Service

Amazon S3 admite la autorización basada en etiquetas únicamente para los recursos de objetos.

Amazon S3 admite roles vinculados a servicios para Amazon S3 Storage Lens.

AWS Trusted Advisor

El acceso de la API a Trusted Advisor se realiza a través de la API de AWS Support y se controla mediante políticas de IAM de AWS Support.

Amazon Virtual Private Cloud

En una política de usuario de IAM, no puede restringir los permisos a un punto de conexión de VPC de Amazon concreto. Cualquier elemento `Action` que incluya las acciones de API `ec2:*VpcEndpoint*` o `ec2:DescribePrefixLists` debe especificar `""Resource": "*""`. Para obtener más información, consulte Administración de identidades y accesos para puntos de conexión de VPC y servicios de puntos de conexión de VPC en la Guía de AWS PrivateLink.

Amazon VPC admite asociar una única política de recursos a un punto de conexión de VPC para restringir a qué se puede obtener acceso a través de dicho punto de conexión. Para obtener más información sobre el uso de políticas basadas en recursos para controlar el acceso a los recursos desde puntos de conexión de VPC de Amazon específicos, consulte Control del acceso a los servicios con políticas de punto de conexión en la Guía de AWS PrivateLink.

Amazon VPC no tiene roles vinculados a servicios, pero AWS Transit Gateway sí. Para obtener más información, consulte Uso de roles vinculados a servicios para la puerta de enlace de tránsito en la Guía de AWS Transit Gateway de Amazon VPC.

AWS X-Ray

No todas las acciones de X-Ray admiten permisos de nivel de recursos.

X-Ray admite el control de acceso basado en etiquetas para grupos y reglas de muestreo.

Firma de solicitudes API de AWS

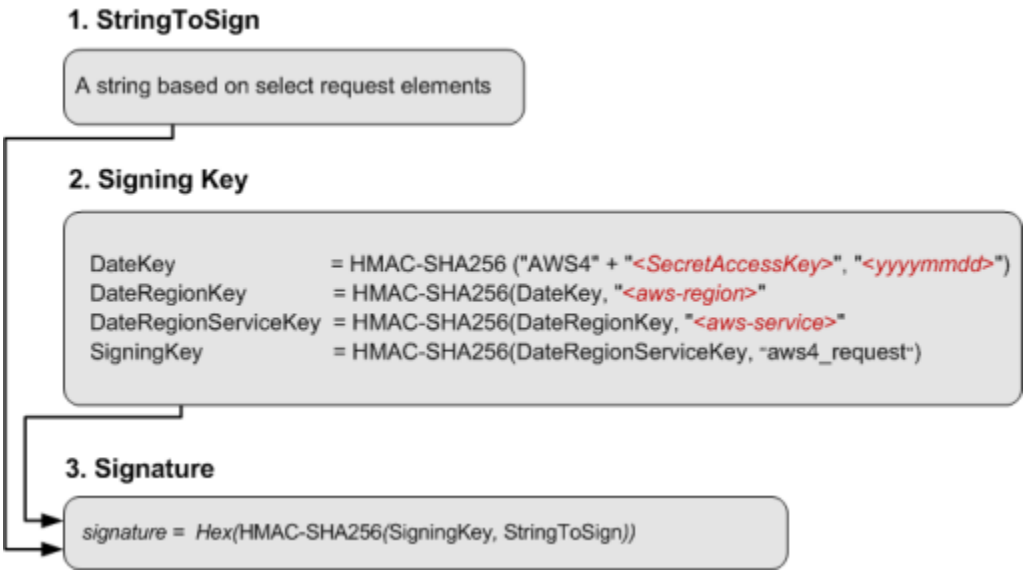
> **Important**
>
> Si usa un SDK de AWS (consulte Código de muestra y bibliotecas) o la herramienta de línea de comandos (CLI) de AWS para enviar solicitudes de API a AWS, puede omitir esta sección porque los clientes del SDK y la CLI autentican sus solicitudes mediante las claves de acceso que usted proporciona. A menos que tenga una razón específica para no hacerlo, le recomendamos que utilice siempre un SDK o la CLI.
> En las regiones en las que se admiten varias versiones de firma, las solicitudes de firma manual significan que debe especificar qué versión de firma se utiliza. Cuando envía solicitudes a puntos de acceso de varias regiones, los SDK y la CLI cambian de forma automática a Signature Version 4A sin configuración adicional.

La información de autenticación que envíe en una solicitud debe incluir una firma. Para calcular una firma, primero debe concatenar los elementos de la solicitud seleccionados para formar una cadena, denominada cadena para firmar. A continuación, debe utilizar una clave de firma para calcular el código de autenticación de mensajes basado en hash (HMAC) de la cadena para firmar.

En AWS Signature Version 4, no usa su clave de acceso secreta para firmar la solicitud. En cambio, primero usa su clave de acceso secreta para derivar una clave de firma. La clave de firma derivada es específica de la fecha, el servicio y la región. Para obtener más información sobre cómo generar una clave de firma en diferentes lenguajes de programación, consulte Ejemplos de firmas de solicitudes.

Signature Version 4 es el protocolo de firma de AWS. AWS también admite una extensión, Signature Version 4A, la cual admite firmas para solicitudes de la API en varias regiones. Para obtener más información, consulte el proyecto sigv4a-signing-examples en GitHub.

En el siguiente diagrama, se ilustra el proceso general de cálculo de una firma.



- La cadena para firmar depende del tipo de solicitud. Por ejemplo, cuando utiliza el encabezado de autorización HTTP o los parámetros de consulta para la autenticación, utiliza una combinación variable de elementos de solicitud a fin de crear la cadena para firmar. Para una solicitud HTTP POST, la política `POST` de la solicitud es la cadena que firma.
- En clave de firma, el diagrama muestra una serie de cálculos, donde el resultado de cada paso se suma al siguiente paso. El último paso es la clave de firma.
- Cuando un servicio de AWS recibe una solicitud autenticada, vuelve a crear la firma con la información de autenticación incluida en la solicitud. Si las firmas coinciden, el servicio procesa la solicitud. De lo contrario, se rechaza la solicitud.

Contenido

Cuándo firmar las solicitudes

Cuando escribe código personalizado que envía solicitudes de la API a AWS, debe incluir código que firme las solicitudes. Puede escribir código personalizado porque:

- Si trabaja con un lenguaje de programación para el que no hay un AWS SDK disponible.
- Necesita control total sobre el modo en que se envían las solicitudes a AWS.

¿Por qué se firman las solicitudes?

El proceso de firma ayuda a proteger las solicitudes de las siguientes formas:

- Comprobación de la identidad del solicitante

  Las solicitudes autenticadas requieren una firma que se crea con las claves de acceso (ID de clave de acceso, clave de acceso secreta). Si utiliza credenciales de seguridad temporales, los cálculos de la firma también requieren un token de seguridad. Para obtener más información, consulte Acceso programático de credenciales de seguridad de AWS.
- Protección de los datos en tránsito

  Para evitar que se altere una solicitud mientras están en tránsito, algunos de sus elementos se utilizan para calcular un resumen (hash) de la solicitud y el valor hash resultante se incluye como parte de la solicitud. Cuando un Servicio de AWS recibe la solicitud, utiliza la misma información para calcular un hash y lo compara con el valor de hash contenido en su solicitud. Si los valores no coinciden, AWS deniega la solicitud.
- Protección contra posibles ataques de reproducción

  En la mayoría de los casos, una solicitud debe llegar a AWS en el plazo de cinco minutos a partir de la marca de tiempo que figura en ella. De lo contrario, AWS deniega la solicitud.

Elementos de una firma de solicitud a la API de AWS

> **Important**
>
> A menos que utilice la CLI o los SDK de AWS, debe escribir código para calcular las firmas que brindan información de autenticación en sus solicitudes. El cálculo de firmas en AWS

Signature Version 4 puede ser una tarea compleja, por lo que le recomendamos que utilice la CLI o los SDK de AWS siempre que sea posible.

Cada solicitud HTTP/HTTPS que utiliza la firma Signature Version 4 debe contener estos elementos.

Elementos

- Especificación de punto de enlace
- Acción
- Parámetros de acciones
- Date
- Información de autenticación

Especificación de punto de enlace

Especifica el nombre de DNS del punto de conexión al que se envía la solicitud. Este nombre suele contener el código de servicio y la región. Por ejemplo, el punto de conexión de Amazon DynamoDB en la región `us-east-1` es `dynamodb.us-east-1.amazonaws.com`.

Para las solicitudes HTTP/1.1, debe incluir el encabezado `Host`. Para solicitudes HTTP/2, puede incluir el encabezado `:authority` o el encabezado `Host`. Utilice únicamente el encabezado `:authority` de conformidad con la especificación HTTP/2. No todos los servicios son compatibles con las solicitudes HTTP/2.

Para conocer los puntos de conexión admitidos por cada servicio, consulte los Puntos de conexión y cuotas del servicio en la Referencia general de AWS.

Acción

Especifica una acción de la API para el servicio. Por ejemplo, la acción de DynamoDB `CreateTable` o la acción de Amazon EC2 `DescribeInstances`.

Para ver las acciones admitidas por cada servicio, consulte la Referencia de autorización del servicio.

Parámetros de acciones

Especifica los parámetros de la acción especificada en la solicitud. Cada acción de la API de AWS tiene un conjunto de parámetros obligatorios y opcionales. La versión de la API suele ser un parámetro obligatorio.

Para ver los parámetros admitidos por una acción de la API, consulte la Referencia de la API del servicio.

Date

Especifica la fecha y la hora de la solicitud. Incluir la fecha y la hora en una solicitud ayuda a impedir que terceros intercepten su solicitud y vuelvan a enviarla posteriormente. La fecha que especifica en el alcance de credencial debe coincidir con la fecha de su solicitud.

La marca de tiempo debe estar en UTC y en el siguiente formato ISO 8601: AAAAMMDDTHHMMSSZ. Por ejemplo, `20220830T123600Z`. No incluya milisegundos en la marca de tiempo.

Puede utilizar un encabezado `date` o `x-amz-date`, o puede incluir `x-amz-date` como parámetro de consulta. Si no podemos encontrar un encabezado `x-amz-date`, buscamos un encabezado `date`.

Información de autenticación

Cada solicitud que envíe debe incluir la siguiente información. AWS usa esta información para garantizar la validez y autenticidad de la solicitud.

- Algoritmo: use `AWS4-HMAC-SHA256` para especificar Signature Version 4 con el algoritmo de hash `HMAC-SHA256`.
- Credencial: una cadena compuesta por el identificador de clave de acceso, la fecha en formato AAAAMMDD, el código de región, el código de servicio y la cadena de terminación `aws4_request`, separados por barras diagonales (/). El código de región, el código de servicio y la cadena de terminación deben utilizar caracteres en minúscula.

  ```
  AKIAIOSFODNN7EXAMPLE/YYYYMMDD/region/service/aws4_request
  ```

- Encabezados firmados: los encabezados HTTP que se incluirán en la firma, separados por punto y coma (;). Por ejemplo, `host;x-amz-date`.
- Firma: una cadena codificada en formato hexadecimal que representa la firma calculada. Debe calcular la firma utilizando el algoritmo especificado en el parámetro `Algorithm`.

Métodos de autenticación

> **Important**
>
> A menos que utilice la CLI o los SDK de AWS, debe escribir código para calcular las firmas que brindan información de autenticación en sus solicitudes. El cálculo de firmas en AWS Signature Version 4 puede ser una tarea compleja, por lo que le recomendamos que utilice la CLI o los SDK de AWS siempre que sea posible.

Puede expresar la información de autenticación mediante uno de los siguientes métodos.

Encabezado de autorización HTTP

El encabezado de `Authorization` HTTP es el método más común para autenticar una solicitud. Todas las operaciones de la API de REST (excepto las cargas basadas en el navegador mediante solicitudes `POST`) requieren este encabezado. Para obtener más información sobre el valor del encabezado de autorización y cómo calcular la firma y las opciones relacionadas, consulte Autenticación de solicitudes: uso del encabezado de autorización (AWS Signature Version 4) en la Referencia de la API de Amazon S3.

A continuación, se muestra un ejemplo de un valor de encabezado de `Authorization`. Se agregan saltos de línea a este ejemplo para facilitar la lectura. En su código, el encabezado debe ser una cadena continua. No hay comas entre el algoritmo y la credencial, pero los demás elementos deben estar separados por comas.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024
```

En la siguiente tabla, se describen los distintos componentes del valor del encabezado de autorización del ejemplo anterior:

Componente	Descripción
Autorización	El algoritmo que se utilizó para calcular la firma. Debe proporcionar este valor cuando

Componente	Descripción
	utilice AWS Signature Version 4 para la autenticación. La cadena especifica AWS Signature Version 4 (`AWS4`) y el algoritmo de firma (`HMAC-SHA256`).
Credential	El ID de clave de acceso y la información sobre el alcance, que incluye la fecha, la región y el servicio que se utilizaron para calcular la firma. Esta cadena tiene el siguiente formato: `<your-access-key-id>/<date>/ <aws-region>/<aws-service>/ aws4_request` Dónde: el valor `<date>` se especifica con el formato DDMMAAAA. El valor `<aws-service>` es s3 al enviar una solicitud a Amazon S3.
SignedHeaders	Una lista de encabezados de solicitud separados por punto y coma que utilizó para calcular `Signature` . La lista solo incluye los nombres de los encabezados y estos nombres deben estar en minúsculas. Por ejemplo: host;range;x-amz-date
Signature	La firma de 256 bits expresada en 64 caracteres hexadecimales en minúsculas. Por ejemplo:`fe5f80f77d5fa3beca038a248ff 027d0445342fe2855ddc9631766 30326f1024` Tenga en cuenta que los cálculos de la firma varían según la opción que elija para transferir la carga.

Parámetros de cadenas de consulta

Puede utilizar una cadena de consulta para expresar una solicitud en su totalidad en una URL. En este caso, utiliza los parámetros de consulta para proporcionar la información de la solicitud, incluida la información de autenticación. Dado que la firma de la solicitud forma parte de la dirección URL, este tipo de URL suele denominarse como URL prefirmada. Puede utilizar URL prefirmadas para incrustar enlaces en los que se puede hacer clic en HTML, que pueden tener una validez de hasta siete días. Para obtener más información, consulte Autenticación de solicitudes: uso de parámetros de consulta (AWS Signature Version 4) en la Referencia de la API de Amazon S3.

A continuación, se muestra un ejemplo de una URL prefirmada. Se agregan saltos de línea a este ejemplo para facilitar la lectura:

```
https://s3.amazonaws.com/examplebucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &X-Amz-Signature=<signature-value>
```

> **Note**
>
> El valor `X-Amz-Credential` de la URL muestra el carácter "/" solo para facilitar la lectura. En la práctica, debe codificarse como %2F. Por ejemplo: `&X-Amz-Credential=<your-access-key-id>%2F20130721%2Fus-east-1%2Fs3%2Faws4_request`

En la siguiente tabla, se describen los parámetros de consulta de la URL que proporcionan información de autenticación.

Nombre del parámetro de la cadena de consulta	Descripción
X-Amz-Algorithm	Identifica la versión de AWS Signature y el algoritmo que utilizó para calcular la firma. En el caso de AWS Signature Version 4, establece este valor de parámetro en `AWS4-HMAC-`

Nombre del parámetro de la cadena de consulta	Descripción
	SHA256. Esta cadena identifica AWS Signature Version 4 (AWS 4) y el algoritmo HMAC-SHA256 (HMAC-SHA256).
X-Amz-Credential	Además del ID de clave de acceso, este parámetro también proporciona el alcance (región y servicio de AWS) para el que es válida la firma. Este valor debe coincidir con el alcance que se utiliza en los cálculos de firmas, como se explica en la siguiente sección. El formato general de este valor de parámetro es el siguiente: <your-access-key-id>/<date>/<AWS Region>/<AWS-service>/aws4_request Por ejemplo: AKIAIOSFODNN7EXAMPLE/20130721/us-east-1/s3/aws4_request . Para obtener una lista de las cadenas regionales de AWS, consulte Puntos de conexión regionales en la Referencia general de AWS.
X-Amz-Date	El formato de fecha y hora debe seguir la norma ISO 8601 y debe tener el formato yyyyMMddTHHmmssZ . Por ejemplo, si la fecha y la hora son “08/01/2016 15:32:41.982-700”, primero se deben convertir a UTC (hora universal coordinada) y, a continuación, se deben enviar como “20160801T223241Z”.

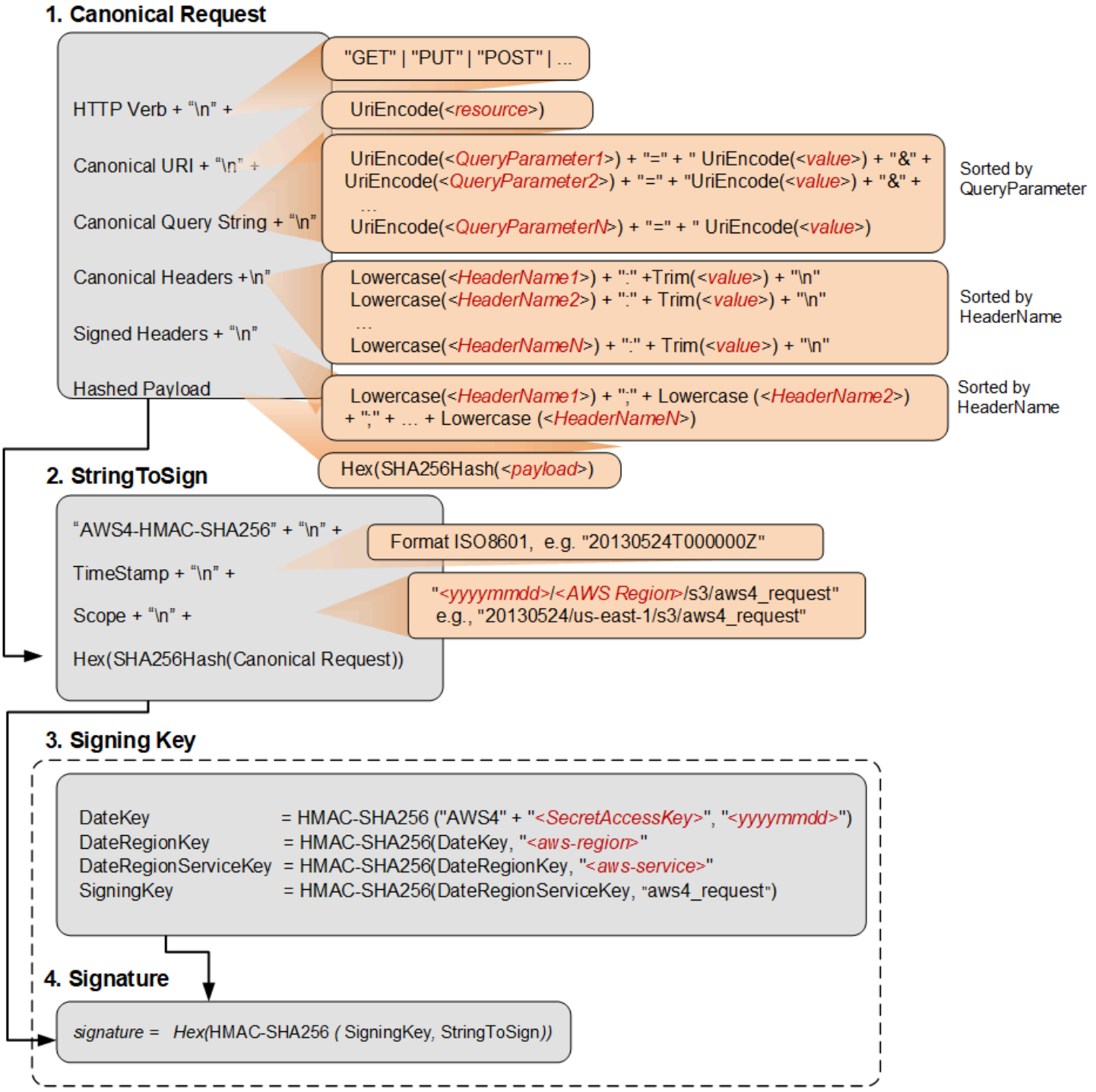
Nombre del parámetro de la cadena de consulta	Descripción
X-Amz-Expires	Proporciona el periodo, en segundos, durante el que es válida la URL prefirmada generada. Por ejemplo, 86 400 (24 horas). Este valor es un entero. El valor mínimo que puede establecer es 1 y el máximo es 604 800 (siete días). Una URL prefirmada puede ser válida durante un máximo de siete días porque la clave de firma que utiliza para calcular la firma es válida durante un máximo de siete días.
X-Amz-SignedHeaders	Muestra los encabezados que utilizó para calcular la firma. Los siguientes encabezados son obligatorios para los cálculos de firmas: • El encabezado del host HTTP. • Cualquier encabezado x-amz-* que planee agregar a la solicitud. Para mayor seguridad, debe firmar todos los encabezados de solicitud que planea incluir en su solicitud.
X-Amz-Signature	Proporciona la firma para autenticar la solicitud . Esta firma debe coincidir con la firma que calcula el servicio; de lo contrario, el servicio deniega la solicitud. Por ejemplo, `733255ef0 22bec3f2a8701cd61d4b371f3f2 8c9f193a1f02279211d48d5193d7` Los cálculos de firmas se describen en la siguiente sección.

Nombre del parámetro de la cadena de consulta	Descripción
X-Amz-Security-Token	Parámetro de credenciales opcional si se utilizan credenciales procedentes del servicio STS.

Creación de una solicitud de API de AWS firmada

> **Important**
>
> Si usa un SDK de AWS (consulte Código de muestra y bibliotecas) o la herramienta de línea de comandos (CLI) de AWS para enviar solicitudes de API a AWS, puede omitir esta sección porque los clientes del SDK y la CLI autentican sus solicitudes mediante las claves de acceso que usted proporciona. A menos que tenga una razón específica para no hacerlo, le recomendamos que utilice siempre un SDK o la CLI.
> En las regiones en las que se admiten varias versiones de firma, las solicitudes de firma manual significan que debe especificar qué versión de firma se utiliza. Cuando envía solicitudes a puntos de acceso de varias regiones, los SDK y la CLI cambian de forma automática a Signature Version 4A sin configuración adicional.

A continuación, se muestra el proceso para crear una solicitud firmada. A fin de calcular una firma, primero necesita una cadena para firmar. Luego, se calcula un hash HMAC-SHA256 de la cadena que se va a firmar mediante una clave de firma. En el siguiente diagrama se ilustra el proceso, incluidos los distintos componentes de la cadena que se crea para la firma.



En la siguiente tabla, se describen las funciones que se muestran en el diagrama. Debe implementar código para estas funciones. Para obtener más información, consulte ejemplos de código en los SDK de AWS.

Función	Descripción
`Lowercase()`	Convierta la cadena de caracteres en minúsculas.
`Hex()`	Codificación en minúsculas en base 16.
`SHA256Hash()`	Función de hash criptográfico del algoritmo de hash seguro (SHA).
`HMAC-SHA256()`	Calcula el HMAC mediante el algoritmo SHA256 con la clave de firma proporcionada. Esta es la firma final.
`Trim()`	Elimine cualquier espacio en blanco inicial o final.
`UriEncode()`	El URI codifica cada byte. UriEncode() debe aplicar las siguientes reglas: • El URI codifica todos los bytes excepto los caracteres no reservados: “A”-“Z”, “a”-“z”, “0”-“9”, “-”, “.”, “_” y “~”. • El carácter de espacio es un carácter reservado y debe codificarse como “%20” (y no como “+”). • Cada byte codificado en URI se encuentra formado por un “%” y el valor hexadecimal de dos dígitos del byte. • Las letras del valor hexadecimal deben estar en mayúsculas, por ejemplo, “%1A”. • Codifique el carácter de barra diagonal, “/”, en todas partes excepto en el nombre de la clave del objeto. Por ejemplo, si el nombre de la clave del objeto es `photos/Jan/sample.jpg` , la barra diagonal del nombre de la clave no se encuentra codificada.

Función	Descripción
	Important Es posible que las funciones estándar de UriEncode que proporciona su plataforma de desarrollo no funcionen debido a las diferencias en la implementación y a la ambigüeda d relacionada en las RFC subyacent es. Le recomendamos que escriba su propia función UriEncode personali zada para asegurarse de que la codificación funcione. Para ver un ejemplo de una función UriEncode en Java, consulte Utilidades de Java en el sitio web de GitHub.

Note

Al firmar sus solicitudes, puede utilizar cualquiera de las siguientes opciones: AWS Signature Version 4 o AWS Signature Version 4A. La diferencia clave entre las dos se determina por la forma en que se calcula la firma. Con AWS Signature Version 4A, la firma no incluye información específica de la región y se calcula mediante el algoritmo `AWS4-ECDSA-P256-SHA256`.

Credenciales de seguridad temporales

En lugar de utilizar credenciales de larga duración para firmar una solicitud, puede utilizar las credenciales de seguridad temporales proporcionadas por AWS Security Token Service (AWS STS).

Cuando utilice credenciales de seguridad temporales, debe agregar `X-Amz-Security-Token` al encabezado de autorización o a la cadena de consulta para contener el token de sesión. Algunos servicios requieren que agregue `X-Amz-Security-Token` a la solicitud canónica. Otros servicios

requieren que solo agregue `X-Amz-Security-Token` al final, después de calcular la firma. Consulte la documentación de cada Servicio de AWS para obtener más detalles.

Resumen de pasos de firma

Paso 1: creación de una solicitud canónica

Organice el contenido de la solicitud (host, acción, encabezados, etc.) en un formato canónico estándar. La solicitud canónica es uno de los datos de entrada utilizados con el fin de crear una cadena para firmar. Para obtener más información, consulte Elementos de una firma de solicitud a la API de AWS.

Paso 2: creación de un hash de la solicitud canónica

Genere una clave de firma al llevar a cabo una sucesión de operaciones hash con clave (operaciones HMAC) en la fecha de la solicitud, la región y el servicio, con su clave de acceso secreta de AWS como clave de la operación hash inicial.

Paso 3: Creación de una cadena para firmar

Cree una cadena para firmar con la solicitud canónica e información adicional, como el algoritmo, la fecha de la solicitud, el ámbito de credenciales y el resumen (hash) de la solicitud canónica.

Paso 4: cálculo de la firma

Una vez generada la clave de firma, se calcula la firma llevando a cabo una operación hash con clave en la cadena para firmar. Utilice la clave de firma generada como clave hash para esta operación.

Paso 5: adición de la firma a la solicitud

Después de calcular la firma, añádasela a un encabezado HTTP o a la cadena de consulta de la solicitud.

Paso 1: creación de una solicitud canónica

Crea una solicitud canónica al concatenar las siguientes cadenas, separadas por caracteres de nueva línea. Esto ayuda a garantizar que la firma que calcula y la firma que calcula AWS puedan coincidir.

```
<HTTPMethod>\n
<CanonicalURI>\n
<CanonicalQueryString>\n
```

```
<CanonicalHeaders>\n
<SignedHeaders>\n
<HashedPayload>
```

- *HTTPMethod*: el método HTTP, como GET, PUT, HEAD y DELETE.
- *CanonicalUri*: la versión codificada en URI del URI del componente de ruta absoluta, que comienza con la "/" que sigue al nombre de dominio y continúa hasta el final de la cadena o del signo de interrogación ("?") si tiene parámetros de cadena de consulta. Si la ruta absoluta está vacía, utilice un carácter de barra diagonal (/). El URI del siguiente ejemplo, `/examplebucket/myphoto.jpg`, es la ruta absoluta y no se codifica la "/" en la ruta absoluta:

  ```
  http://s3.amazonaws.com/examplebucket/myphoto.jpg
  ```

- *CanonicalQueryString*: los parámetros de la cadena de consulta codificados en URI. Codifique en URI cada nombre y valores de forma individual. También debe ordenar los parámetros de la cadena de consulta canónica alfabéticamente por nombre de clave. La clasificación se produce después de la codificación. La cadena de consulta del siguiente ejemplo de URI es:

  ```
  http://s3.amazonaws.com/examplebucket?prefix=somePrefix&marker=someMarker&max-keys=2
  ```

  La cadena de consulta canónica es la siguiente (se agregan saltos de línea a este ejemplo para facilitar la lectura):

  ```
  UriEncode("marker")+"="+UriEncode("someMarker")+"&"+
  UriEncode("max-keys")+"="+UriEncode("20") + "&" +
  UriEncode("prefix")+"="+UriEncode("somePrefix")
  ```

  Cuando una solicitud se dirige a un subrecurso, el valor del parámetro de consulta correspondiente será una cadena vacía (""). Por ejemplo, el siguiente URI identifica el subrecurso ACL en el bucket `examplebucket`:

  ```
  http://s3.amazonaws.com/examplebucket?acl
  ```

  En este caso, la cadena CanonicalQueryString es la siguiente:

  ```
  UriEncode("acl") + "=" + ""
  ```

Si el URI no incluye un “?”, no hay ninguna cadena de consulta en la solicitud y establece la cadena de consulta canónica en una cadena vacía (“”). Aún tendrá que incluir “\n”.

- *CanonicalHeaders*: una lista de los encabezados de las solicitudes con sus valores. Los pares individuales de nombre y valor del encabezado se encuentran separados por el carácter de nueva línea (“\n”). El siguiente es un ejemplo de un encabezado canónico:

```
Lowercase(<HeaderName1>)+":"+Trim(<value>)+"\n"
Lowercase(<HeaderName2>)+":"+Trim(<value>)+"\n"
...
Lowercase(<HeaderNameN>)+":"+Trim(<value>)+"\n"
```

La lista CanonicalHeaders debe incluir lo siguiente:

- Encabezado de `host` HTTP.
- Si el encabezado `Content-Type` se encuentra en la solicitud, debe agregarlo a la lista *CanonicalHeaders*.
- También debe agregar cualquier encabezado `x-amz-*` que planea incluir en su solicitud. Por ejemplo, si utiliza credenciales de seguridad temporales, debe incluir `x-amz-security-token` en su solicitud. Debe agregar este encabezado a la lista de *CanonicalHeaders*.

> **Note**
>
> El encabezado `x-amz-content-sha256` es obligatorio para las solicitudes de AWS de Amazon S3. Proporciona un hash de la carga de solicitud. Si no hay ninguna carga, debe proporcionar el hash de una cadena vacía.

Cada nombre de encabezado debe:

- tener caracteres en minúsculas;
- aparecer en orden alfabético;
- ir seguido de dos puntos (:).

En el caso de los valores, debe:

- recortar los espacios iniciales o finales;
- convertir espacios secuenciales en un solo espacio;
- separar los valores de un encabezado con varios valores mediante comas.

- Debe incluir el encabezado de host (HTTP/1.1) o el encabezado :authority (HTTP/2) y cualquier encabezado `x-amz-*` de la firma. Si lo desea, puede incluir otros encabezados estándar en la firma, como content-type.

Las funciones `Lowercase()` y `Trim()` que se utilizan en este ejemplo se describen en la sección anterior.

La siguiente es una cadena de `CanonicalHeaders` de ejemplo. Los nombres de encabezado se encuentran en minúsculas y ordenados.

```
host:s3.amazonaws.com
x-amz-content-sha256:e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
x-amz-date:20130708T220855Z
```

> **Note**
>
> Para calcular una firma de autorización, solo se requieren el host y cualquier encabezado `x-amz-*`; sin embargo, a fin de evitar la manipulación de datos, debería considerar la posibilidad de incluir todos los encabezados en el cálculo de la firma.

- *`SignedHeaders`*: una lista ordenada alfabéticamente y separada por punto y coma de nombres de encabezados de solicitudes en minúsculas. Los encabezados de las solicitudes de la lista son los mismos que incluyó en la cadena de `CanonicalHeaders`. Por ejemplo, para el caso anterior, el valor de *`SignedHeaders`* sería el siguiente:

```
host;x-amz-content-sha256;x-amz-date
```

- *`HashedPayload`*: una cadena creada con la carga del cuerpo de la solicitud HTTP como entrada para una función de hash. Esta cadena utiliza caracteres hexadecimales en minúscula.

```
Hex(SHA256Hash(<payload>)
```

Si no hay ninguna carga en la solicitud, se calcula un hash de la cadena vacía de la siguiente manera:

```
Hex(SHA256Hash(""))
```

El hash devuelve los siguientes valores:

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Por ejemplo, cuando carga un objeto mediante una solicitud PUT, proporciona los datos del objeto en el cuerpo. Al recuperar un objeto mediante una solicitud GET, se calcula el hash de la cadena vacía.

Paso 2: creación de un hash de la solicitud canónica

Cree un hash (resumen) de la solicitud canónica con el mismo algoritmo que utilizó para crear el hash de la carga. El hash de la solicitud canónica es una cadena de caracteres hexadecimales en minúsculas.

Paso 3: Creación de una cadena para firmar

Cree una cadena al concatenar las siguientes cadenas, separadas por caracteres de nueva línea. No termine esta cadena con un carácter de nueva línea.

```
Algorithm \n
RequestDateTime \n
CredentialScope  \n
HashedCanonicalRequest
```

- *Algorithm*: el algoritmo utilizado para crear el hash de la solicitud canónica. Para SHA-256, el algoritmo es AWS4-HMAC-SHA256.
- *RequestDateTime*: la fecha y la hora utilizadas en el alcance de la credencial. Este valor es la hora UTC actual en formato ISO 8601 (por ejemplo, 20130524T000000Z).
- *CredentialScope*: el alcance de la credencial. Esto restringe la firma resultante a la región y el servicio especificados. La cadena tiene el siguiente formato:*YYYYMMDD*/*region*/*service*/aws4_request.
- *HashedCanonicalRequest*: el hash de la solicitud canónica. Este valor se calcula en el paso 2.

La siguiente es una cadena de ejemplo para firmar.

```
"AWS4-HMAC-SHA256" + "\n" +
```

```
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(SHA256Hash(<CanonicalRequest>))
```

Paso 4: cálculo de la firma

En AWS Signature Version 4, en lugar de utilizar sus claves de acceso de AWS para firmar una solicitud, crea una clave de firma que se limita a una región y un servicio específicos como información de autenticación que agregará a su solicitud.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Para obtener una lista de las cadenas de región, consulte Puntos de conexión regionales en la Referencia general de AWS.

Para cada paso, llame a la función de hash con la clave y los datos necesarios. El resultado de cada llamada a la función de hash se convierte en la entrada de la siguiente llamada a la función de hash.

Datos de ingreso obligatorios

- Una cadena `Key`, la cual contiene su clave de acceso secreta
- Una cadena `Date`, la cual contiene la fecha utilizada en el alcance de la credencial con el formato AAAAMMDD
- Una cadena `Region`, la cual contiene el código de región (por ejemplo, `us-east-1`)
- Una cadena `Service`, la cual contiene el código de servicio (por ejemplo, `ec2`)
- La cadena para firmar que creó en el paso anterior.

Para calcular la firma

1. Concatene “AWS4” y la clave de acceso secreta. Llame a la función de hash con la cadena concatenada como clave y la cadena de fecha como datos.

   ```
   kDate = hash("AWS4" + Key, Date)
   ```

2. Llame a la función de hash con el resultado de la llamada anterior como clave y la cadena de región como datos.

```
kRegion = hash(kDate, Region)
```

3. Llame a la función de hash con el resultado de la llamada anterior como clave y la cadena de servicio como datos.

```
kService = hash(kRegion, Service)
```

4. Llame a la función de hash con el resultado de la llamada anterior como clave y “aws4_request” como datos.

```
kSigning = hash(kService, "aws4_request")
```

5. Llame a la función de hash con el resultado de la llamada anterior como clave y la cadena para firmar como datos. El resultado es la firma como valor binario.

```
signature = hash(kSigning, string-to-sign)
```

6. Convierta la firma de representación binaria a hexadecimal, en caracteres en minúsculas.

Paso 5: adición de la firma a la solicitud

Example Ejemplo: encabezado de autorización

En el ejemplo siguiente se muestra un encabezado `Authorization` para la acción `DescribeInstances`. Para facilitar la lectura, este ejemplo está formateado con saltos de línea. En su código, debe ser una cadena continua. No hay ninguna coma entre el algoritmo y `Credential`. Sin embargo, los demás elementos deben ir separados por comas.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request,
SignedHeaders=host;x-amz-date,
Signature=calculated-signature
```

Example Ejemplo: solicitud con parámetros de autenticación en la cadena de consulta

En el siguiente ejemplo, se muestra una consulta para la acción `DescribeInstances` que incluye la información de autenticación. Para facilitar la lectura, este ejemplo está formateado con saltos de línea y no está codificado en URL. En el código, la cadena de consulta debe ser una cadena continua codificada en URL.

```
https://ec2.amazonaws.com/?
Action=DescribeInstances&
Version=2016-11-15&
X-Amz-Algorithm=AWS4-HMAC-SHA256&
X-Amz-Credential=AKIAIOSFODNN7EXAMPLE/20220830/us-east-1/ec2/aws4_request&
X-Amz-Date=20220830T123600Z&
X-Amz-SignedHeaders=host;x-amz-date&
X-Amz-Signature=calculated-signature
```

Código fuente en los SDK de AWS

Los SDK de AWS incluyen código fuente en GitHub para firmar las solicitudes de la API de AWS. Para obtener ejemplos de código, consulte Proyectos de ejemplo en AWS repositorio de muestras

- AWS SDK for .NET: AWS4Signer.cs
- AWS SDK for C++: AWSAuthV4Signer.cpp
- AWS SDK for Go: v4.go
- AWS SDK for Java: BaseAws4Signer.java
- AWS SDK for JavaScript: v4.js
- AWS SDK for PHP: SignatureV4.php
- AWS SDK for Python (Boto): signers.py
- AWS SDK for Ruby: signer.rb

Ejemplos de firmas de solicitudes

Los siguientes ejemplos de solicitudes de firma de AWS muestran cómo se puede usar SigV4 para firmar las solicitudes enviadas sin el AWS SDK o la herramienta de línea de comandos de AWS.

Carga de Amazon S3 basada en el navegador mediante HTTP POST

Autenticación de solicitudes: cargas basadas en el navegador describe la firma y la información relevante que Amazon S3 utiliza para calcular la firma al recibir la solicitud.

Ejemplo: carga basada en el navegador mediante HTTP POST (con Signature Version 4 de AWS) proporciona más información con un ejemplo de política POST y un formulario que se puede utilizar para cargar un archivo. La política de ejemplo y las credenciales ficticias muestran el flujo de trabajo y el hash de la firma y la política resultantes.

Solicitudes autenticadas de VPC Lattice

Ejemplos de solicitudes autenticadas de Signature Version 4 (SigV4) proporciona ejemplos de Python y Java que muestran cómo se puede realizar la firma de solicitudes con y sin interceptores personalizados.

Uso de la Signature Version 4 con Amazon Translate

El uso de Signature Version 4 con Amazon Translate muestra cómo utilizar un programa de Python para agregar información de autenticación a las solicitudes de Amazon Translate. El ejemplo realiza una solicitud POST, crea una estructura JSON que contiene el texto que se traducirá en el cuerpo (carga) de la solicitud y pasa la información de autenticación en un encabezado Authorization.

Uso de Signature Version 4 con Neptune

Ejemplo: conectarse a Neptune mediante Python con firmas de Signature Version 4 muestra cómo realizar solicitudes firmadas a Neptune mediante Python. En este ejemplo, se incluyen variantes para usar una clave de acceso o credenciales temporales.

Firmar solicitudes HTTP en S3 Glacier

Ejemplo de cálculo de firmas para la API de streaming explica los detalles de la creación de una firma para Upload Archive (archivo POST), una de las dos API de streaming de S3 Glacier.

Realizar solicitudes HTTP a Amazon SWF

Realizar solicitudes HTTP a Amazon SWF muestra el contenido del encabezado de una solicitud JSON a Amazon SWF.

Cálculo de firmas para las API de streaming en Amazon OpenSearch Service

Firmar una solicitud de búsqueda de Amazon OpenSearch Service con AWS SDK para PHP versión 3 incluye un ejemplo de cómo enviar solicitudes HTTP firmadas a Amazon OpenSearch Service.

Proyectos de ejemplo en AWS repositorio de muestras

Los siguientes proyectos de ejemplo muestran cómo firmar solicitudes para realizar solicitudes de la API Rest a AWS servicios con lenguajes comunes como Python, Node.js, Java, C#, Go y Rust.

Proyectos exclusivos de la versión 4a

El proyecto sigv4a-signing-examples proporciona ejemplos de cómo firmar solicitudes con SigV4a para realizar solicitudes de la API Rest a Servicios de AWS con lenguajes comunes como Python, Node.js, Java, C#, Go y Rust.

Realizar solicitudes mediante un punto de acceso multirregional (MRAP) usa la versión 4a de Signature para acceder a los datos de Amazon S3 mediante Python boto 3.

Publicar en AWS IoT Core

Código Python para publicar AWS IoT Core utilizando el protocolo HTTPs proporciona orientación sobre cómo publicar mensajes en AWS IoT Core utilizando el protocolo Https y AWS autenticación SigV4. Tiene dos implementaciones de referencia, una en Python y otra en NodeJS.

Aplicación .Net Framework para publicar en AWS IoT Core utilizando el protocolo HTTPs proporciona orientación sobre cómo publicar mensajes en AWS IoT Core utilizando el protocolo Https y AWS autenticación SigV4. Este proyecto también incluye una implementación equivalente a .NET core.

Solución de problemas de solicitudes firmadas para las API de AWS

> **Important**
>
> A menos que utilice la CLI o los SDK de AWS, debe escribir código para calcular las firmas que brindan información de autenticación en sus solicitudes. El cálculo de firmas de SigV4 puede ser una tarea compleja, por lo que recomendamos utilizar los AWS SDK o la CLI siempre que sea posible.

Cuando desarrolla código que crea una solicitud firmada, es posible que reciba HTTP 403 `SignatureDoesNotMatch` por parte de Servicios de AWS. Estos errores significan que el valor de la firma en la solicitud de HTTP a AWS no coincide con la firma que Servicio de AWS calculó. Los errores HTTP 401 `Unauthorized` se producen cuando los permisos no permiten al intermediario realizar la solicitud.

Las solicitudes de API pueden devolver un error en los siguientes casos:

- La solicitud de API no está firmada y utiliza la autenticación de IAM.
- Las credenciales de IAM utilizadas para firmar la solicitud son incorrectas o no tienen permisos para invocar la API.

- La firma de la solicitud de API firmada no coincide con la firma que calculó el servicio de AWS.
- El encabezado de la solicitud de API es incorrecto.

> **Note**
>
> Actualice el protocolo de firmas de Signature Version 2 (SigV2) de AWS a Signature version 4 (SigV4) de AWS antes de buscar otras soluciones de error. Los servicios (como Amazon S3) y las regiones ya no admiten la firma SigV2.

Causas posibles

- Errores de credencial
- Errores canónicos en la cadena de solicitud y firma
- Errores en el alcance de la credencial
- Errores de firma de clave

Errores de credencial

Asegúrese de que la solicitud de API esté firmada con SigV4. Si la solicitud de API no está firmada, es posible que reciba el siguiente error: `Missing Authentication Token`. Agregue la firma que falta y vuelva a enviar la solicitud.

Verifique que las credenciales de autenticación para la clave de acceso y la clave secreta sean correctas. Si la clave de acceso es incorrecta, es posible que reciba el siguiente error: `Unauthorized`. Asegúrese de que la entidad utilizada para firmar la solicitud esté autorizada para realizar la solicitud. Para obtener más información, consulte Solución de problemas de mensajes de error de acceso denegado.

Errores canónicos en la cadena de solicitud y firma

Si no calculó correctamente la solicitud canónica en Paso 2: creación de un hash de la solicitud canónica o Paso 3: Creación de una cadena para firmar, el paso de verificación de firmas que realiza el servicio falla con el siguiente mensaje de error:

```
The request signature we calculated does not match the signature you provided
```

Cuando el servicio de AWS recibe una solicitud firmada, vuelve a calcular la firma. Si hay diferencias en los valores, las firmas no coinciden. Compare la solicitud canónica y la cadena con la solicitud firmada con el valor en el mensaje de error. Modifique el proceso de firma si hay alguna diferencia.

> **Note**
>
> También puede comprobar que no envió la solicitud a través de un proxy que modifique los encabezados o la solicitud.

Example Ejemplo de solicitud canónica

```
GET                                                    -------- HTTP method
/                                                      -------- Path. For API stage
 endpoint, it should be /{stage-name}/{resource-path}
                                                       -------- Query string key-
value pair. Leave it blank if the request doesn't have a query string.
content-type:application/json                          -------- Header key-value
 pair. One header per line.
host:0123456789.execute-api.us-east-1.amazonaws.com    -------- Host and x-amz-date
 are required headers for all signed requests.
x-amz-date:20220806T024003Z

content-type;host;x-amz-date                           -------- A list of signed
 headers
d167e99c53f15b0c105101d468ae35a3dc9187839ca081095e340f3649a04501       -------- Hash
 of the payload
```

Para comprobar que la clave secreta coincide con el ID de clave de acceso, puede probarla con una implementación funcional conocida. Por ejemplo, utilice un AWS SDK o la AWS CLI para realizar una solicitud a AWS.

Encabezado de solicitud API

Asegúrese de que el encabezado de autorización de SigV4 que agregó en Paso 4: cálculo de la firma incluya la clave de credencial correcta, similar a la siguiente:

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
```

```
Signature=example-generated-signature
```

Si la clave de credencial falta o es incorrecta, es posible que reciba el siguiente mensaje de error: `Authorization header requires 'Credential' parameter. Authorization header requires 'Signature' parameter.` Asegúrese de que la solicitud de autorización de SigV4 también incluya la fecha de la solicitud mediante el uso de `HTTP Date` o del encabezado `x-amz-date`.

Errores en el alcance de la credencial

El alcance de la credencial que creó en Paso 3: Creación de una cadena para firmar restringe la firma a una fecha, región y servicio específicos. Esta cadena tiene el siguiente formato:

```
YYYYMMDD/region/service/aws4_request
```

> **Note**
>
> Si utiliza SigV4a, la región no está incluida en el ámbito de las credenciales.

Date

Si el alcance de la credencial no especifica la misma fecha que el encabezado x-amz-date, el paso de verificación de la firma falla y aparece el siguiente mensaje de error:

```
Date in Credential scope does not match YYYYMMDD from ISO-8601 version of date from
 HTTP
```

Si la solicitud especifica una fecha futura, el paso de verificación de la firma falla con el siguiente mensaje de error:

```
Signature not yet current: date is still later than date
```

Si la solicitud ha caducado, el paso de verificación de la firma falla con el siguiente mensaje de error:

```
Signature expired: date is now earlier than date
```

Región

Si el alcance de la credencial no especifica la misma región que la solicitud, el paso de verificación de la firma falla con el siguiente mensaje de error:

```
Credential should be scoped to a valid Region, not region-code
```

Servicio

Si el alcance de la credencial no especifica el mismo servicio que el encabezado host, el paso de verificación de la firma falla con el siguiente mensaje de error:

```
Credential should be scoped to correct service: 'service'
```

Cadena de terminación

Si el alcance de la credencial no termina con aws4_request, el paso de verificación de la firma falla con el siguiente mensaje de error:

```
Credential should be scoped with a valid terminator: 'aws4_request'
```

Errores de firma de clave

Los errores causados por la generación incorrecta de la clave de firma o por un uso indebido de la criptografía son más difíciles de solucionar. Luego de comprobar que la cadena canónica y la cadena para firmar son correctas, también puede comprobar si existe alguno de los siguientes problemas:

- La clave de acceso secreta no coincide con el ID de clave de acceso que especificó.
- Hay un problema con el código de generación de la clave.

Para comprobar que la clave secreta coincide con el ID de clave de acceso, puede probarla con una implementación funcional conocida. Por ejemplo, utilice un SDK de AWS o la AWS CLI para hacer una solicitud a AWS. Para ver ejemplos, consulte Ejemplos de firmas de solicitudes

Referencia de políticas JSON de IAM

En esta sección se presenta la sintaxis, descripciones y ejemplos detallados de los elementos, variables, y lógica de evaluación de las políticas JSON de IAM. Para obtener más información general, consulte Información general de políticas de JSON.

Esta sección incluye los siguientes temas.

- Referencia de los elementos de las políticas de JSON de IAM - Más información acerca de los elementos que puede utilizar al crear una política. Consulte ejemplos de políticas adicionales y obtenga más información sobre condiciones, tipos de datos admitidos y cómo se utilizan en los diversos servicios.
- Lógica de evaluación de políticas: en esta sección se describen las solicitudes de AWS, cómo se autentican y cómo AWS utiliza políticas para determinar el acceso a los recursos.
- Gramática del lenguaje de la política JSON de IAM : en esta sección se presenta una gramática formal para el lenguaje que se utiliza para crear políticas en IAM.
- Managed Policies de AWS para funciones de trabajo: en esta sección se enumeran todas las políticas administradas de AWS que se asignan directamente a funciones de trabajo habituales en el sector de TI. Con estas políticas puede conceder los permisos necesarios para realizar las tareas que se esperan de quien desempeña una función determinada. Estas políticas consolidan los permisos de muchos servicios en una única política.
- Claves de contexto de condición globales de AWS: esta sección incluye una lista de todas las claves de condición globales de AWS que puede utilizar para limitar los permisos de una política de IAM.
- Claves de contexto de condición de IAM y AWS STS: esta sección incluye una lista de todas las claves de condición de IAM y AWS STS que puede utilizar para limitar los permisos de una política de IAM.
- Acciones, recursos y claves de condiciones de servicios de AWS: en esta sección se presenta una lista de todas las operaciones de API de AWS que puede utilizar como permisos en una política de IAM. También incluye las claves de condición específicas del servicio que pueden utilizarse para especificar más la solicitud.

Referencia de los elementos de las políticas de JSON de IAM

Un documento de política de JSON se compone de elementos. Se indican en el orden que suelen seguir en una política. El orden de los elementos no es importante; por ejemplo, el elemento `Resource` puede ir antes que el elemento `Action`. No es necesario especificar elementos `Condition` en la política. Para obtener más información sobre la estructura general y la finalidad de un documento de política JSON, consulte Información general de políticas de JSON.

Algunos elementos de política JSON se excluyen mutuamente. Esto significa que no se puede crear una política que utilice ambos. Por ejemplo, no se puede utilizar `Action` y `NotAction` en la misma

instrucción de política. Otros pares que se excluyen mutuamente son `Principal`/`NotPrincipal` y `Resource`/`NotResource`.

Los detalles de lo que se incluye en una política pueden variar según el servicio, en función de las acciones que el servicio ponga a disposición, de los tipos de recursos que contenga, etc. Cuando se escriben políticas para un servicio específico, es útil ver ejemplos de políticas para dicho servicio. Para obtener una lista de todos los servicios compatibles con IAM y de enlaces con la documentación de dichos servicios sobre el tema de IAM y las políticas, consulte Servicios de AWS que funcionan con IAM.

Cuando usted crea o edita una política JSON, IAM puede realizar la validación de políticas para ayudarle a crear una política eficaz. IAM identifica errores de sintaxis JSON, mientras que IAM Access Analyzer proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más las políticas. Para obtener más información acerca la validación de políticas, consulte Validación de políticas de IAM. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer.

Temas

Elementos de política JSON de IAM: Version

> **Nota aclaratoria**
>
> El elemento de la política JSON `Version` es diferente de la versión de la política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, por otro lado, se crea al realizar cambios en una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada. Si busca información sobre cómo admitir múltiples versiones de políticas gestionadas, consulte the section called "Control de versiones de políticas de IAM".

El elemento de la política `Version` especifica las reglas de sintaxis del lenguaje que se van a utilizar para procesar esta política. Para utilizar todas las funciones disponibles para políticas, incluya el siguiente elemento `Version` fuera del elemento `Statement` en todas las políticas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

IAM admite los siguientes valores del elemento `Version`:

- `2012-10-17`. Esta es la versión actual del lenguaje de la política y siempre debe incluir un elemento `Version` y establecerlo en `2012-10-17`. De lo contrario, no puede utilizar características como las variables de política que se introdujeron con esta versión.
- `2008-10-17`. Esta es una versión anterior del lenguaje de la política. Puede que vea esta versión en políticas ya existentes. No utilice esta versión en ninguna política nueva ni cuando actualice políticas ya existentes. Las características más recientes, como, por ejemplo, variables de política, no funcionarán con su política. Por ejemplo, las variables del tipo `${aws:username}` no se reconocerán como variables y se tratarán en la política como si fueran cadenas literales.

Elementos de política JSON de IAM: Id

El elemento `Id` especifica un identificador opcional de la política. El ID se utiliza de forma distinta en diferentes servicios. El ID está permitido en políticas basadas en recursos, pero no en políticas basadas en la identidad.

En el caso de los servicios que le permiten definir un elemento `ID`, le recomendamos que utilice un UUID (GUID) para el valor o que incorpore un UUID como parte del ID para garantizar la exclusividad.

```
{
  "Version": "2012-10-17",
  "Id": "cd3ad3d9-2776-4ef1-a904-4c229d1642ee",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Algunos servicios de AWS (por ejemplo, Amazon SQS o Amazon SNS) podrían necesitar este elemento y además que fuera único. Para obtener información específica sobre los servicios para escribir políticas, consulte la documentación del servicio en el que está trabajando.

Elementos de política JSON de IAM: Statement

El elemento `Statement` es el elemento principal de una política. Este elemento es obligatorio. El elemento `Statement` puede contener una sola instrucción o una matriz de instrucciones individuales. Cada bloque de instrucción individual debe estar encerrado entre claves { }. Para instrucciones múltiples, la matriz debe estar entre corchetes [ ].

```
"Statement": [{...},{...},{...}]
```

En el siguiente ejemplo se muestra una política que contiene una matriz de tres instrucciones en un único elemento `Statement`. (La política le permite obtener acceso a su propia "carpeta home" en la consola de Amazon S3). La política incluye la variable `aws:username`, que se sustituirá durante la evaluación de la política por el nombre de usuario de la solicitud. Para obtener más información, consulte Introducción.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

Elementos de política JSON de IAM: Sid

Puede proporcionar un `Sid` (ID de la instrucción) como un identificador opcional para la declaración de política. Puede asignar un valor de `Sid` a cada instrucción de una matriz de instrucciones. Puede

utilizar el valor `Sid` como descripción de la declaración de política. En los servicios que le permiten especificar un elemento `ID`, como, por ejemplo, SQS y SNS, el valor de `Sid` es simplemente un subID del ID del documento de la política. En IAM, el valor de `Sid` debe ser único en la política de JSON.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStatementID",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

El elemento `Sid` admite mayúsculas ASCII (A-Z), minúsculas (a-z) y números (0-9).

IAM no se expone la `Sid` en la API de IAM. No puede recuperar una instrucción determinada basándose en este ID.

> **Note**
>
> Algunos servicios de AWS (por ejemplo, Amazon SQS o Amazon SNS) podrían necesitar este elemento y además que fuera único. Para obtener información específica sobre los servicios para escribir políticas, consulte la documentación del servicio en el que trabaja.

Elementos de política JSON de IAM: Effect

El elemento `Effect` es obligatorio y se utiliza para especificar si la declaración da como resultado un permiso o una denegación explícitos. Los valores válidos para `Effect` son `Allow` y `Deny`. El valor `Effect` distingue entre mayúsculas y minúsculas.

```
"Effect":"Allow"
```

De forma predeterminada, se deniega el acceso a los recursos. Para permitir el acceso a un recurso, debe establecer el elemento `Effect` en `Allow`. Para anular un permiso (por ejemplo, para anular

un permiso que, de lo contrario, estaría en vigor), establezca el elemento `Effect` en `Deny`. Para obtener más información, consulte Lógica de evaluación de políticas.

Elemento de la política de JSON de AWS: Principal

Utilice el elemento `Principal` en una política de JSON basada en recursos para especificar la entidad principal que tiene acceso permitido o denegado a un recurso.

Puede utilizar el elemento `Principal` en las políticas basadas en recursos. Varios servicios admiten políticas basadas en recursos, incluido IAM. El tipo de política basada en recursos de IAM es una política de confianza de roles. En los roles de IAM, utilice el elemento `Principal` en la política de confianza de roles para especificar quién puede asumir el rol. Para obtener acceso entre cuentas, debe especificar el identificador de 12 dígitos de la cuenta de confianza. Consulte ¿Qué es IAM Access Analyzer? para saber si las entidades principales de las cuentas fuera de su zona de confianza (cuenta u organización de confianza) tienen acceso para asumir sus roles.

> **Note**
>
> Tras crear el rol, puede cambiar la cuenta a "*" para permitir que cualquiera pueda asumir el rol. Si lo hace, le recomendamos encarecidamente que limite el acceso al rol a través de otros medios, como, por ejemplo, un elemento `Condition` que limite el acceso únicamente a determinadas direcciones IP. No permita que todo el mundo pueda acceder al rol.

Otros ejemplos de recursos que admiten políticas basadas en recursos incluyen un bucket de Amazon S3 o una AWS KMS key.

No puede utilizar el elemento `Principal` en una política basada en identidad. Las políticas basadas en identidad son políticas de permisos que se adjuntan a identidades de IAM (usuarios, grupos o roles). En esos casos, la entidad principal queda implícita por la identidad a la que está adjunta la política.

Temas

- Especificación de una entidad principal
- Entidades principales de Cuenta de AWS
- Entidades principales de rol de IAM
- Entidades principales de sesión de rol
- Entidades principales de usuario de IAM

- Entidades principales de IAM Identity Center
- Entidades principales de sesión de usuario federado de AWS STS
- Entidad principal del servicio de AWS
- Entidades principales de servicios de AWS en regiones registradas
- Todas las entidades principales
- Más información

Especificación de una entidad principal

Una entidad principal se especifica en el elemento `Principal` de una política basada en recursos o en claves de condición que admiten entidades principales.

Puede especificar cualquiera de las siguientes entidades principales en una política:

- Cuenta de AWS y usuario raíz
- Roles de IAM
- Sesiones de rol
- Usuarios de IAM
- Sesiones de usuarios federados
- Servicios de AWS
- Todas las entidades principales

No puede identificar un grupo de usuarios como entidad principal en una política (como una política basada en recursos) porque los grupos están relacionados con los permisos, no con la autenticación, y las entidades principales las autentica IAM.

Puede especificar más de una entidad principal para cada uno de los tipos de entidad principal en las siguientes secciones mediante una matriz. Las matrices pueden tener uno o varios valores. Cuando se especifica más de una entidad principal en un elemento, se conceden permisos a cada una de ellas. Es un `OR` lógico y no un `AND` lógico, porque se autentica como una entidad principal a la vez. Si incluye más de un valor, utilice corchetes (`[` y `]`) y delimite con comas cada entrada de la matriz. En la siguiente política de ejemplo, se definen los permisos para la cuenta 123456789012 o la cuenta 555555555555.

```
"Principal" : {
```

```
"AWS": [
  "123456789012",
  "555555555555"
  ]
}
```

> **Note**
>
> No puede utilizar un carácter comodín para buscar coincidencias con parte de un nombre de entidad principal o ARN.

Entidades principales de Cuenta de AWS

Puede especificar identificadores de la Cuenta de AWS en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales. Esto delega autoridad en la cuenta. Cuando permite el acceso a otra cuenta, un administrador de esa cuenta debe conceder acceso a una identidad (usuario o rol de IAM) en dicha cuenta. Cuando especifica una Cuenta de AWS, puede usar el ARN de la cuenta (arn:aws:iam::*`account-ID`*:root) o una forma abreviada formada por el prefijo `"AWS":` seguido del ID de la cuenta.

Por ejemplo, si el ID de cuenta es `123456789012`, puede utilizar uno de los métodos siguientes para especificar esa cuenta en el elemento `Principal`:

```
"Principal": { "AWS": "arn:aws:iam::123456789012:root" }
```

```
"Principal": { "AWS": "123456789012" }
```

El ARN de la cuenta y el ID de cuenta abreviado se comportan de la misma manera. Ambos delegan permisos en la cuenta. Usar el ARN de la cuenta en el elemento `Principal` no limita los permisos solo al usuario raíz de la cuenta.

> **Note**
>
> Cuando guarda una política basada en recursos que incluye el ID de cuenta abreviado, el servicio podría convertirla en el ARN de la entidad principal. De este modo, no se modifica la funcionalidad de la política.

Algunos servicios de AWS admiten más opciones para especificar la entidad principal de una cuenta. Por ejemplo, Amazon S3 permite especificar un ID de usuario canónico mediante el siguiente formato:

```
"Principal": { "CanonicalUser":
 "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be" }
```

También puede especificar más de una Cuenta de AWS (o ID de usuario canónico) como entidad principal mediante una matriz. Por ejemplo, puede utilizar los tres métodos para especificar una entidad principal en una política de bucket.

```
"Principal": {
  "AWS": [
    "arn:aws:iam::123456789012:root",
    "999999999999"
  ],
  "CanonicalUser": "79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be"
}
```

Entidades principales de rol de IAM

Puede especificar ARN de entidades principales de rol de IAM en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales. Los roles de IAM son identidades. En IAM, las identidades son recursos a los que puede asignar permisos. Los roles confían en otra identidad autenticada para asumir ese rol. Esto incluye una entidad principal en AWS o un usuario de un proveedor de identidades (IdP) externo. Cuando una entidad principal o una identidad asume un rol, recibe credenciales de seguridad temporales con los permisos del rol asumido. Cuando utiliza esas credenciales de sesión para realizar operaciones en AWS, pasa a ser una entidad principal de sesión de rol.

Los roles de IAM son identidades que existen en IAM. Los roles confían en otra identidad autenticada, como una entidad principal en AWS o un usuario de un proveedor de identidades externo. Cuando una entidad principal o una identidad asume un rol, recibe credenciales de seguridad temporales. A continuación, puede utilizar esas credenciales como entidad principal de sesión de rol para realizar operaciones en AWS.

Cuando especifica una entidad principal de rol en una política basada en recursos, los permisos efectivos de dicha entidad están limitados por cualquier tipo de política que limite los permisos para el rol. Esto incluye políticas de sesión y límites de permisos. Para obtener más información acerca

de cómo se evalúan los permisos efectivos de una sesión de rol, consulte Lógica de evaluación de políticas.

Para especificar el ARN de rol en el elemento `Principal`, utilice el siguiente formato:

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:role/role-name" }
```

> **Important**
>
> Si el elemento `Principal` de una política de confianza de rol contiene un ARN que apunta a un rol de IAM concreto, dicho ARN se transforma en el ID de la entidad principal exclusivo del rol al guardar la política. Esto es útil para reducir el riesgo de que alguien aumente sus privilegios al eliminar y volver a crear el rol. Normalmente este ID no se muestra en la consola, ya que IAM utiliza una transformación inversa al ARN del rol cuando se muestra la política de confianza. Sin embargo, si elimina el rol, la relación se desvincula. La política dejará de aplicarse, aunque vuelva a crear el rol, ya que el ID de la entidad principal del rol nuevo ya no coincide con el ID almacenado en la política de confianza. Cuando esto ocurre, el ID de la entidad principal aparece en las políticas basadas en recursos porque AWS ya no puede asignarlo a un ARN válido. El resultado final es que, si elimina y vuelve a crear un rol al que se hace referencia en un elemento `Principal` de la política de confianza, deberá editar el rol para sustituir el ID de la entidad principal que ahora es incorrecto por el ARN correcto. El ARN se transforma una vez más en el nuevo ID de la entidad principal del rol al guardar la política.

De forma alternativa, puede especificar la entidad principal de rol como la entidad principal en una política basada en recursos o crear una política de permisos amplios que utilice la clave de condición `aws:PrincipalArn`. Cuando utiliza esta clave, a la entidad principal de la sesión de rol se le otorgan los permisos en función del ARN del rol asumido y no del ARN de la sesión resultante. Debido a que AWS no convierte los ARN de clave de condición en ID, los permisos concedidos al ARN del rol persisten si elimina el rol y, a continuación, crea un nuevo rol con el mismo nombre. Los permisos concedidos mediante la clave de condición `aws:PrincipalArn` con un comodín (*) en el elemento `Principal` no están limitados por los tipos de políticas basadas en identidad, tales como límites de permisos o políticas de sesión, a menos que las políticas basadas en la identidad contengan una denegación explícita.

Entidades principales de sesión de rol

Puede especificar sesiones de roles en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales. Cuando una entidad principal o una identidad asume un rol, recibe credenciales de seguridad temporales con los permisos del rol asumido. Cuando utiliza esas credenciales de sesión para realizar operaciones en AWS, pasa a ser una entidad principal de sesión de rol.

El formato que utilice para una entidad principal de sesión de rol depende de la operación de AWS STS que se utilizó para asumir el rol.

Además, los administradores pueden diseñar un proceso para controlar cómo se emiten las sesiones de roles. Por ejemplo, pueden proporcionar una solución con un solo clic para sus usuarios que cree un nombre de sesión predecible. Si su administrador lo hace, puede utilizar las entidades principales de sesión de rol en sus políticas o claves de condición. De lo contrario, puede especificar el ARN del rol como entidad principal en la clave de condición `aws:PrincipalArn`. La forma en que especifica el rol como entidad principal puede cambiar los permisos efectivos para la sesión resultante. Para obtener más información, consulte Entidades principales de rol de IAM.

Entidades principales de sesión de rol asumido

Una entidad principal de sesión de rol asumido es una entidad de sesión que resulta del uso de la operación de AWS STS `AssumeRole`. Para obtener más información acerca de qué entidades principales pueden asumir un rol mediante esta operación, consulte Comparación de las operaciones de la API de AWS STS.

Para especificar el ARN de sesión de rol asumido en el elemento `Principal`, utilice el siguiente formato:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Al especificar una sesión de rol asumido en un elemento `Principal`, no puede utilizar un comodín "*" para referirse a todas las sesiones. Las entidades principales siempre deben designar a una sesión específica.

Entidades principales de sesión OIDC

Una entidad principal de sesión OIDC es una entidad de sesión que resulta del uso de la operación de AWS STS `AssumeRoleWithWebIdentity`. Puede utilizar un proveedor de OIDC externo (IdP)

para iniciar sesión y, a continuación, asumir un rol de IAM mediante esta operación. Esto aprovecha la federación de identidades y emite una sesión de rol. Para obtener más información acerca de qué entidades principales pueden asumir un rol mediante esta operación, consulte Comparación de las operaciones de la API de AWS STS.

Cuando emite un rol de un proveedor de OIDC, obtiene este tipo especial de entidad principal de sesión que incluye información sobre el proveedor OIDC.

Utilice este tipo de entidad principal en su política para permitir o denegar el acceso con base en el proveedor de identidad web de confianza. Para especificar el ARN de sesión de rol de identidad OICD en el elemento `Principal` de una política de confianza de rol, utilice el siguiente formato:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Entidades principales de sesión de SAML

Una entidad principal de sesión de SAML es una entidad de sesión que resulta del uso de la operación de AWS STS `AssumeRoleWithSAML`. Puede utilizar un proveedor de identidad (IdP) SAML externo para iniciar sesión y, a continuación, asumir un rol de IAM mediante esta operación. Esto aprovecha la federación de identidades y emite una sesión de rol. Para obtener más información acerca de qué entidades principales pueden asumir un rol mediante esta operación, consulte Comparación de las operaciones de la API de AWS STS.

Cuando emite un rol de un proveedor de identidad SAML, obtiene este tipo especial de entidad principal de sesión que incluye información sobre el proveedor de identidad SAML.

Utilice este tipo de entidad principal en su política para permitir o denegar el acceso con base en el proveedor de identidad SAML de confianza. Para especificar el ARN de sesión de rol de identidad SAML en el elemento `Principal` de una política de confianza de rol, utilice el siguiente formato:

```
"Principal": { "Federated": "arn:aws:iam::AWS-account-ID:saml-provider/provider-name" }
```

Entidades principales de usuario de IAM

Puede especificar a los usuarios de IAM en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales.

> **Note**
>
> En un elemento `Principal`, la parte del nombre de usuario del Nombre de recurso de Amazon (ARN) distingue entre mayúsculas y minúsculas.

```
"Principal": { "AWS": "arn:aws:iam::AWS-account-ID:user/user-name" }
```

```
"Principal": {
  "AWS": [
    "arn:aws:iam::AWS-account-ID:user/user-name-1",
    "arn:aws:iam::AWS-account-ID:user/user-name-2"
  ]
}
```

Cuando especifica usuarios en un elemento `Principal`, no puede utilizar un comodín (*) para designar a “todos los usuarios”. Las entidades principales siempre tienen que designar a usuarios específicos.

> **Important**
>
> Si el elemento `Principal` de una política de confianza de rol contiene un ARN que apunta a un usuario de IAM concreto, IAM transforma al ARN en el ID principal exclusivo del usuario al guardar la política. Esto es útil para reducir el riesgo de que alguien aumente sus privilegios al eliminar y volver a crear el rol. Normalmente este ID no se muestra en la consola, ya que también existe una transformación inversa al ARN del usuario cuando se muestra la política de confianza. Sin embargo, si elimina el usuario, la relación se desvincula. La política ya no se aplica, incluso aunque vuelva a crear el usuario. Esto se debe a que el usuario nuevo tiene un ID de entidad principal nuevo que no coincide con el ID almacenado en la política de confianza. Cuando esto ocurre, el ID de la entidad principal aparece en las políticas basadas en recursos porque AWS ya no puede asignarlo a un ARN válido. El resultado final es que, si elimina y vuelve a crear un rol al que se hace referencia en un elemento `Principal` de la política de confianza, deberá editar el rol para sustituir el ID de la entidad principal que

ahora es incorrecto por el ARN correcto. IAM transforma una vez más el ARN en el nuevo ID principal del usuario al guardar la política.

Entidades principales de IAM Identity Center

En IAM Identity Center, la entidad principal en una política basada en recursos debe definirse como la entidad principal de Cuenta de AWS. Para especificar el acceso, haga referencia al ARN del rol del conjunto de permisos en el bloque de condiciones. Para obtener más información, consulte Hacer referencia a los conjuntos de permisos en las políticas de recursos, Amazon EKS y AWS KMS en la Guía del usuario de IAM Identity Center.

Entidades principales de sesión de usuario federado de AWS STS

Puede especificar sesiones de usuarios federados en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales.

> **Important**
>
> AWS recomienda que utilice sesiones de usuarios federados de AWS STS solo cuando sea necesario, por ejemplo, cuando se requiere acceso de usuario raíz. En su lugar, utilice roles para delegar permisos.

Una entidad principal de usuario federado de AWS STS es una entidad de sesión que resulta del uso de la operación de AWS STS `GetFederationToken`. En este caso, AWS STS utiliza la federación de identidad como método para obtener tokens de acceso temporales en lugar de utilizar roles de IAM.

En AWS, los usuarios de IAM o un Usuario raíz de la cuenta de AWS pueden autenticarse mediante claves de acceso a largo plazo. Para conocer las entidades principales que pueden federarse mediante esta operación, consulte Comparación de las operaciones de la API de AWS STS.

- Usuario federado de IAM: un usuario de IAM se federa mediante la operación `GetFederationToken` que da como resultado una entidad principal de sesión de usuario federado para dicho usuario de IAM.
- Usuario raíz federado: un usuario raíz se federa mediante la operación `GetFederationToken` que da como resultado una entidad principal de sesión de usuario federado para dicho usuario raíz.

Cuando un usuario de IAM o usuario raíz solicita credenciales temporales de AWS STS mediante esta operación, inicia una sesión de usuario federado temporal. El ARN de esta sesión se basa en la identidad original que se federó.

Para especificar el ARN de sesión de usuario federado en el elemento `Principal`, utilice el siguiente formato:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:federated-user/user-name" }
```

Entidad principal del servicio de AWS

Puede especificar servicios de AWS en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales. Una entidad principal del servicio es un identificador de un servicio.

Los roles de IAM que puede asumir un servicio de AWS se denominan roles de servicio. Los roles de servicio deben incluir una política de confianza. Las políticas de confianza son políticas basadas en recursos que se asocian a un rol que define qué entidades principales pueden asumir el rol. Algunos roles de servicio tienen políticas de confianza predefinidas. Sin embargo, en algunos casos, debe especificar la entidad principal del servicio en la política de confianza. La entidad principal de servicio de una política de IAM no puede ser `"Service": "*"`.

El identificador de la entidad principal de un servicio incluye el nombre del servicio y suele tener el siguiente formato:

service-name`.amazonaws.com`

El servicio define la entidad principal de servicio. Puede encontrar la entidad principal del servicio para algunos servicios al abrir Servicios de AWS que funcionan con IAM, comprobar si el servicio tiene Sí en la columna Rol vinculado al servicio y abrir el vínculo Sí a fin de ver la documentación del rol vinculado al servicio para ese servicio. Consulte la sección Permisos de rol vinculados a servicios de ese servicio para ver la entidad principal del servicio.

En el siguiente ejemplo, se muestra una política que puede asociarse a un rol de servicio. La política permite que dos servicios, Amazon ECS y Elastic Load Balancing, asuman el rol. Los servicios pueden realizar las tareas concedidas por la política de permisos asignada al rol (no se muestra). Para especificar varios elementos principales del servicio, no debe especificar dos elementos `Service`; solo puede tener uno. En cambio, utilice una gama de varios elementos principales del servicio como el valor de un único elemento `Service`.

```
"Principal": {
    "Service": [
        "ecs.amazonaws.com",
        "elasticloadbalancing.amazonaws.com"
   ]
}
```

Entidades principales de servicios de AWS en regiones registradas

Puede lanzar recursos en varias regiones de AWS, y debe registrarse en algunas de esas regiones. Para obtener una lista completa de las regiones en las que debe registrarse, consulte Administración de regiones de AWS en la Guía de Referencia general de AWS.

Cuando un servicio de AWS de una región registrada realiza una solicitud dentro de la misma región, el formato del nombre de la entidad principal del servicio se identifica como la versión no regionalizada del nombre de la entidad principal del servicio:

service-name.amazonaws.com

Cuando un servicio de AWS de una región registrada realiza una solicitud interregional a otra región, el formato del nombre de la entidad principal del servicio se identifica como la versión regionalizada del nombre de la entidad principal del servicio:

service-name.*{region}*.amazonaws.com

Por ejemplo, supongamos que tiene un tema de Amazon SNS ubicado en la región `ap-southeast-1` y un bucket de Amazon S3 ubicado en la región registrada `ap-east-1`. Desea configurar las notificaciones del bucket de S3 para que puedan publicar mensajes en el tema de SNS. Para permitir que el servicio S3 publique mensajes en el tema de SNS, debe conceder permiso `sns:Publish` a la entidad principal del servicio S3 mediante la política de acceso basada en recursos del tema.

Si especifica la versión no regionalizada de la entidad principal del servicio S3, `s3.amazonaws.com`, en la política de acceso del tema, se producirá un error en la solicitud `sns:Publish` del bucket al tema. En el siguiente ejemplo, se especifica la entidad principal del servicio S3 no regionalizada en el elemento de política `Principal` de la política de acceso del tema de SNS.

```
"Principal": { "Service": "s3.amazonaws.com" }
```

Dado que el bucket se encuentra en una región registrada y la solicitud se realiza fuera de esa misma región, la entidad principal del servicio S3 aparece como nombre la entidad principal del servicio regionalizado, `s3.ap-east-1.amazonaws.com`. Debe utilizar el nombre de la entidad principal del servicio regionalizado cuando un servicio de AWS de una región registrada realice una solicitud a otra región. Una vez que se especifica el nombre de la entidad principal del servicio regionalizado, si el bucket realiza una solicitud `sns:Publish` al tema de SNS ubicado en otra región, la solicitud se procesará correctamente. En el siguiente ejemplo, se especifica la entidad principal del servicio S3 regionalizada en el elemento de política `Principal` de la política de acceso del tema de SNS.

```
"Principal": { "Service": "s3.ap-east-1.amazonaws.com" }
```

Las listas de autorizaciones basadas en políticas de recursos o entidades principales de servicios para solicitudes interregionales de una región registrada a otra región solo se procesarán correctamente si se especifica el nombre de la entidad principal del servicio regionalizado.

> **Note**
>
> Para las políticas de confianza de roles de IAM, se recomienda utilizar el nombre de la entidad principal del servicio no regionalizado. Los recursos de IAM son globales, de modo que se puede utilizar el mismo rol en cualquier región.

Todas las entidades principales

Puede utilizar un comodín (*) para especificar todas las entidades principales en el elemento `Principal` de una política basada en recursos o en claves de condición que admitan entidades principales. Los permisos de concesión de Políticas basadas en recursos y las claves de condición se emplean para limitar las condiciones de una declaración de política.

> **Important**
>
> Recomendamos encarecidamente que no utilice un comodín (*) en el elemento `Principal` de una política basada en recursos con un efecto `Allow` a menos que tenga la intención de conceder acceso público o anónimo. De lo contrario, especifique las entidades principales, servicios o cuentas de AWS en el elemento `Principal` y, a continuación, restrinja aún más el acceso en el elemento `Condition`. Esto es especialmente cierto para las políticas de

confianza del rol de IAM, porque permiten que otras entidades principales lo sean también en su cuenta.

Para las políticas basadas en recursos, usar un comodín (*) con un efecto `Allow` concede acceso a todos los usuarios, incluidos los usuarios anónimos (acceso público). No se requieren otros permisos para las entidades principales de usuarios y rol de IAM dentro de su cuenta. Para las entidades principales de otras cuentas, también deben tener permisos basados en la identidad en su cuenta que les permitan acceder a su recurso. Esto se denomina acceso entre cuentas.

Para los usuarios anónimos, los siguientes elementos son equivalentes:

```
"Principal": "*"
```

```
"Principal" : { "AWS" : "*" }
```

No puede utilizar un carácter comodín para buscar coincidencias con parte de un nombre de entidad principal o ARN.

En el siguiente ejemplo se muestra una política basada en recursos que pueden utilizarse en lugar de Especificar `NotPrincipal` con `Deny` para denegar explícitamente todas las entidades principales excepto para los especificados en el elemento `Condition`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME/*",
        "arn:aws:s3:::BUCKETNAME"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
```

```
    }
  ]
}
```

Más información

Para obtener más información, consulte los siguientes enlaces:

- Ejemplos de política de bucket en la Guía del usuario de Amazon Simple Storage Service
- Políticas de ejemplo para Amazon SNS en la Guía para desarrolladores de Amazon Simple Notification Service
- Ejemplos de políticas de Amazon SQS en la Guía para desarrolladores de Amazon Simple Queue Service
- Políticas de claves en la Guía para desarrolladores de AWS Key Management Service
- Identificadores de cuenta en la Referencia general de AWS
- Federación OIDC

Elemento de la política de JSON de AWS: NotPrincipal

Puede utilizar el elemento `NotPrincipal` para denegar el acceso a todas las entidades principales, excepto el usuario de IAM, el usuario federado, el rol de IAM, la Cuenta de AWS, el servicio de AWS u otra entidad principal especificada en el elemento `NotPrincipal`.

Puede utilizarlo en políticas basadas en recursos para servicios de AWS, incluidos los puntos de conexión de VPC. Las políticas basadas en recursos son políticas que se integran directamente en un recurso. No puede utilizar el elemento `NotPrincipal` en una política basada en identidad de IAM ni en una política de confianza de rol de IAM.

`NotPrincipal` se debe usar con `"Effect":"Deny"`. No se admite su uso con `"Effect":"Allow"`.

> **Important**
>
> En muy pocas situaciones se exige el uso de `NotPrincipal`. Le recomendamos que sopese otras opciones de autorización antes de decidirse por utilizar `NotPrincipal`. Cuando utilice `NotPrincipal`, puede ser difícil solucionar los efectos de varios tipos de política. En su lugar, recomendamos utilizar la clave de contexto `aws:PrincipalArn` con

los operadores de condición de ARN. Para obtener más información, consulte Todas las entidades principales.

Especificar **`NotPrincipal`** con **`Deny`**

Cuando se utiliza `NotPrincipal` con `Deny`, también debe especificar el ARN de la cuenta del principal no denegado. De lo contrario, la política podría denegar el acceso a toda la cuenta que contenga el principal. En función del servicio que incluya en la política, AWS podría validar primero la cuenta y después al usuario. Si se está evaluando a un usuario de rol asumido (alguien que utiliza un rol), AWS podría validar primero la cuenta, después el rol y, por último, al usuario de rol asumido. El usuario de rol asumido se identifica mediante el nombre de la sesión de rol que se especifica cuando asumieron el rol. Por lo tanto, le recomendamos encarecidamente que incluya de forma explícita el ARN de la cuenta de un usuario, o que incluya tanto el ARN de un rol como el ARN de la cuenta que contenga dicho rol.

Important

No utilice instrucciones de política basadas en recursos que incluyan un elemento de política `NotPrincipal` con un efecto `Deny` para los usuarios o roles de IAM que tengan una política de límite de permisos adjunta. El elemento `NotPrincipal` con efecto `Deny` siempre denegará cualquier entidad principal de IAM que tenga una política de límite de permisos adjunta, independientemente de los valores especificados en el elemento `NotPrincipal`. Esto provoca que algunos usuarios o roles de IAM que de otro modo tendrían acceso al recurso pierdan dicho acceso. Recomendamos cambiar las instrucciones de política basadas en recursos y utilizar el operador de condición `ArnNotEquals` con la clave de contexto `aws:PrincipalArn` para limitar el acceso en lugar del elemento `NotPrincipal`. Para obtener información sobre los límites de permisos, consulte Límites de permisos para las entidades de IAM.

Note

Como práctica recomendada, debe incluir los ARN de la cuenta en su política. Algunos servicios requieren el ARN de la cuenta, aunque esto no es necesario en todos los casos. Cualquier política existente sin el ARN necesario seguirá funcionando, pero las nuevas políticas que incluyan estos servicios deben cumplir este requisito. IAM no realiza un

seguimiento de estos servicios y, por lo tanto, recomienda que incluya siempre la cuenta ARN.

En los ejemplos siguientes se muestra cómo utilizar `NotPrincipal` y `"Effect": "Deny"` en la misma instrucción de política efectivamente.

Example Ejemplo de usuario de IAM de la misma cuenta o de otra cuenta

En el siguiente ejemplo, se deniega explícitamente a todos las entidades principales el acceso a un recurso, excepto al usuario denominado Bob de la Cuenta de AWS 444455556666. Tenga en cuenta que, como práctica recomendada, el elemento `NotPrincipal` contiene el ARN tanto del usuario Bob como de la Cuenta de AWS a la que pertenece Bob (`arn:aws:iam::444455556666:root`). Si el elemento `NotPrincipal` contuviera únicamente el ARN de Bob, el efecto de la política podría ser denegar explícitamente el acceso a la Cuenta de AWS que contiene al usuario Bob. En algunos casos, un usuario no puede tener más permisos que su cuenta principal, de modo que si se deniega explícitamente el acceso a la cuenta de Bob, Bob tampoco podría obtener acceso al recurso.

Este ejemplo funciona según lo previsto cuando forma parte de una instrucción de una política basada en recursos asociada a un recurso en la misma Cuenta de AWS o en otra cuenta (no 444455556666). Este ejemplo por sí mismo no concede acceso a Bob, se limita a no mencionar a Bob en la lista de principales a los que se deniega el acceso de forma explícita. Para permitir a Bob el acceso al recurso, deberá haber otra instrucción en la política que permita el acceso de forma explícita con `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:iam::444455556666:user/Bob",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::BUCKETNAME",
            "arn:aws:s3:::BUCKETNAME/*"
        ]
    }]
}
```

Example Ejemplo de rol de IAM de la misma cuenta o de otra cuenta

En el siguiente ejemplo, se deniega explícitamente a todos las entidades principales el acceso a un recurso, excepto al usuario de rol asumido denominado cross-account-audit-app en la Cuenta de AWS 444455556666. Como práctica recomendada, el elemento `NotPrincipal` contiene el ARN del usuario del rol asumido (cross-account-audit-app), el rol (cross-account-read-only-role) y la Cuenta de AWS a la que pertenece el rol (444455556666). Si el elemento `NotPrincipal` no tuviese el ARN del rol, el efecto de la política podría ser denegar explícitamente el acceso al rol. Del mismo modo, si el elemento `NotPrincipal` no tuviese el ARN de la cuenta de Cuenta de AWS a la que pertenece el rol, el efecto de la política podría ser denegar explícitamente el acceso a la Cuenta de AWS y a todas las entidades de esa cuenta. En algunos casos, los usuarios de rol asumido no pueden tener más permisos que su rol principal y los roles no pueden tener más permisos que su Cuenta de AWS principal, por lo que cuando se deniega el acceso explícitamente al rol o la cuenta, el usuario del rol asumido podría no obtener acceso al recurso.

Este ejemplo funciona según lo previsto cuando forma parte de una instrucción de una política basada en recursos asociada a un recurso de otra Cuenta de AWS (no en 444455556666). En este ejemplo en sí no se permite el acceso al usuario de rol asumido cross-account-audit-app, simplemente no se menciona cross-account-audit-app en la lista de principales a los que se deniega el acceso explícitamente. Para dar a cross-account-audit-app acceso al recurso, la política debe tener otra instrucción que permita de forma explícita el acceso con `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
            "arn:aws:iam::444455556666:role/cross-account-read-only-role",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::Bucket_AccountAudit",
            "arn:aws:s3:::Bucket_AccountAudit/*"
        ]
    }]
}
```

Al especificar una sesión de rol asumido en un elemento `NotPrincipal`, no puede utilizar un comodín (*) para referirse a "todas las sesiones". Las entidades principales siempre deben designar a una sesión específica.

Elementos de política JSON de IAM: Action

El elemento `Action` describe la acción o las acciones específicas que se permitirán o denegarán. Las instrucciones deben incluir un elemento `Action` o un elemento `NotAction`. Cada servicio de AWS tiene su propio conjunto de acciones que describen las tareas que se pueden realizar con dicho servicio. Por ejemplo, la lista de acciones para Amazon S3 se puede encontrar en Especificación de permisos en una política en la Guía del usuario de Amazon Simple Storage Service. Puede encontrar la lista de acciones de Amazon EC2 en la Referencia de la API de Amazon EC2 y la lista de acciones de AWS Identity and Access Management se puede encontrar en la Referencia de la API de IAM. Para encontrar la lista de acciones de otros servicios, consulte la documentación de referencia de la API correspondiente al servicio.

Los valores se especifican utilizando un espacio de nombres de servicio como un prefijo de acción (`iam`, `ec2`, `sqs`, `sns`, `s3`, etc.) seguido del nombre de la acción que debe permitirse o denegarse. El nombre debe coincidir con una acción compatible con el servicio. El prefijo y el nombre de acción no distinguen entre mayúsculas y minúsculas. Por ejemplo, `iam:ListAccessKeys` es igual que `IAM:listaccesskeys`. En los siguientes ejemplos se muestran elementos `Action` de diferentes servicios.

Acción de Amazon SQS

```
"Action": "sqs:SendMessage"
```

Acción de Amazon EC2

```
"Action": "ec2:StartInstances"
```

Acción de IAM

```
"Action": "iam:ChangePassword"
```

Acción de Amazon S3

```
"Action": "s3:GetObject"
```

Puede especificar varios valores para el elemento `Action`.

```
"Action": [ "sqs:SendMessage", "sqs:ReceiveMessage", "ec2:StartInstances",
 "iam:ChangePassword", "s3:GetObject" ]
```

Puede utilizar un asterisco (*) para otorgar acceso a todas las acciones que el producto de AWS específico ofrece. Por ejemplo, el elemento `Action` siguiente se aplica a todas las acciones de S3.

```
"Action": "s3:*"
```

También puede utilizar caracteres comodín (*) como parte del nombre de la acción. Por ejemplo, el elemento `Action` siguiente se aplica a todas las acciones de IAM que contienen la cadena `AccessKey`, como `CreateAccessKey`, `DeleteAccessKey`, `ListAccessKeys` y `UpdateAccessKey`.

```
"Action": "iam:*AccessKey*"
```

Algunos servicios le permiten limitar las acciones que están disponibles. Por ejemplo, Amazon SQS le permite hacer que esté disponible solo un subconjunto de todas las acciones de Amazon SQS posibles. En ese caso, el comodín `*` no permite un control completo de la cola, simplemente permite únicamente el subconjunto de acciones que ha compartido. Para obtener más información, consulte Explicación de permisos en la Guía del desarrollador de Amazon Simple Queue Service.

Elementos de política JSON de IAM: NotAction

`NotAction` es un elemento de política avanzada que hace coincidir explícitamente todo, salvo la lista de acciones especificada. El uso de `NotAction` puede traducirse en una política abreviada; se publican únicamente unas cuantas acciones que no deben coincidir, en vez de publicar una larga lista de acciones que sí coincidirán. Las acciones especificadas en `NotAction` no se ven afectadas por el efecto `Allow` o `Deny` de una declaración de política. Esto, a su vez, significa que todas las acciones o servicios aplicables que no están en la lista están permitidos cuando se usa el efecto `Allow`. Además, las acciones o servicios que no están en la lista se deniegan cuando se usa el efecto `Deny`. Cuando se utiliza `NotAction` con el elemento `Resource`, proporciona el ámbito de la política. Esta es la forma en que AWS determina qué acciones o servicios son aplicables. Para obtener más información, consulte la siguiente política de ejemplo.

NotAction con Allow

Puede utilizar el elemento `NotAction` en una instrucción con `"Effect": "Allow"` para proporcionar acceso a todas las acciones de un servicio de AWS, salvo las acciones especificadas en `NotAction`. Puede utilizarla con el elemento `Resource` para proporcionar el ámbito de la política, limitando las acciones permitidas a las acciones que puede llevar a cabo en el recurso especificado.

En el siguiente ejemplo se permite a los usuarios obtener acceso a todas las acciones de Amazon S3 que se pueden realizar en cualquier recurso de S3, salvo la eliminación de un bucket. Esto no permite a los usuarios utilizar la operación del API `ListAllMyBuckets` de S3, ya que dicha acción requiere el recurso "*". Esta política además no permite acciones en otros servicios, ya que otras acciones de servicio no son aplicables a los recursos de S3.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

En ocasiones, es posible que desee permitir el acceso a un gran número de acciones. El elemento `NotAction` le permite invertir la instrucción, lo que se traducirá en una lista de acciones más corta. Por ejemplo, debido al gran número de servicios de AWS, es posible que desee crear una política que permita al usuario hacer todo salvo obtener acceso a acciones de IAM.

En el siguiente ejemplo se permite a los usuarios obtener acceso a todas las acciones de todos los servicios de AWS, salvo IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Sea precavido al utilizar los elementos `NotAction` y `"Effect": "Allow"` en la misma instrucción o en otra instrucción de una misma política. `NotAction` hace coincidir todos los servicios y acciones que no se mencionan explícitamente en la lista o son aplicables al recurso especificado, lo que podría dar como resultado que se concediera a los usuarios más permisos de los que en realidad se pretendía.

NotAction con Deny

Puede utilizar el elemento `NotAction` en una instrucción con `"Effect": "Deny"` para denegar el acceso a todas los recursos de una lista, salvo las acciones especificadas en el elemento

`NotAction`. Esta combinación no permite los elementos de la lista, pero deniega de forma explícita las acciones que no figuran en la lista. Debe seguir dando permiso las acciones que quiere permitir.

En el siguiente ejemplo condicional se deniega el acceso a las acciones no relacionadas con IAM cuando el usuario no ha iniciado la sesión con MFA. Si el usuario ha iniciado sesión con MFA, la condición `"Condition"` no se cumple y la instrucción `"Deny"` final no tiene efecto. Tenga en cuenta, sin embargo, que esto no supone conceder al usuario acceso a ninguna acción. Simplemente se deniegan explícitamente todas las acciones salvo las de IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllUsersNotUsingMFA",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

Para ver una política de ejemplo que deniegue el acceso a las acciones fuera de las regiones especificadas, excepto las acciones de servicios determinados, consulte AWS: deniega el acceso a AWS en función de la región solicitada..

Elementos de política JSON de IAM: Resource

El elemento `Resource` especifica el objeto u objetos que la instrucción abarca. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Los recursos se especifican mediante un ARN. Para obtener más información sobre el formato de los ARN, consulte ARN de IAM.

Cada servicio tiene su propio conjunto de recursos. Aunque siempre se utiliza un ARN para especificar un recurso, los detalles del ARN de un recurso dependen del servicio y el recurso. Para obtener información sobre cómo especificar un recurso, consulte la documentación del servicio para quiere escribir una instrucción.

> **Note**
>
> Algunos servicios no le permiten especificar acciones para recursos individuales; en su lugar, todas las acciones que publique en los elementos `Action` o `NotAction` se aplican

a todos los recursos de ese servicio. En dichos casos, utilice el asterisco * en el elemento `Resource`.

El siguiente ejemplo hace referencia a una cola Amazon SQS específica.

```
"Resource": "arn:aws:sqs:us-east-2:account-ID-without-hyphens:queue1"
```

El siguiente ejemplo hace referencia al usuario de IAM llamado Bob de una Cuenta de AWS.

> **Note**
>
> En un elemento de `Resource`, el nombre de usuario de IAM distingue entre mayúsculas y minúsculas.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"
```

Uso de comodines en ARN de recursos

Puede utilizar comodines como parte del ARN del recurso. Puede utilizar caracteres comodín (* y ?) dentro de segmentos de ARN (las partes separadas por dos puntos) para representar cualquier combinación de caracteres con un asterisco (*) y cualquier carácter individual con un signo de interrogación (?). Puede utilizar varios caracteres * o ? de cada segmento. Si el comodín (*) es el último carácter en un segmento ARN de recurso, puede expandirse para que coincida más allá de los límites de los dos puntos. Le recomendamos que utilice comodines (* y ?) dentro de los segmentos ARN, separados por dos puntos.

> **Note**
>
> No se puede utilizar un carácter comodín en el segmento de servicio que identifica el producto de AWS. Para obtener más información sobre segmentos de ARN, consulte Nombres de recursos de Amazon (ARN).

El siguiente ejemplo hace referencia a todos los usuarios de IAM cuya ruta es `/accounting`.

```
"Resource": "arn:aws:iam::account-ID-without-hyphens:user/accounting/*"
```

El siguiente ejemplo hace referencia a todos los elementos dentro de un bucket de Amazon S3 específico.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
```

El carácter asterisco (*) puede expandirse para reemplazar todo dentro de un segmento, incluidos caracteres como una barra diagonal (/) que, de otro modo, podrían parecer delimitadores dentro de un espacio de nombres de servicio dado. Por ejemplo, considere el siguiente ARN de Amazon S3 como la misma lógica de expansión comodín que se aplica a todos los servicios.

```
"Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*/test/*"
```

Los comodines del ARN se aplican a todos los siguientes objetos del bucket, no solo al primer objeto de la lista.

```
DOC-EXAMPLE-BUCKET/1/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/3/object.jpg
DOC-EXAMPLE-BUCKET/1/2/3/test/4/object.jpg
DOC-EXAMPLE-BUCKET/1///test///object.jpg
DOC-EXAMPLE-BUCKET/1/test/.jpg
DOC-EXAMPLE-BUCKET//test/object.jpg
DOC-EXAMPLE-BUCKET/1/test/
```

Considere los dos últimos objetos de la lista anterior. Un nombre de objeto de Amazon S3 puede comenzar o finalizar de manera válida con el carácter de barra diagonal (/) que es delimitador convencional. Mientras que “/” funciona como delimitador, no hay ningún significado específico cuando este carácter se utiliza dentro de un ARN de recurso. Se trata de la misma manera que a cualquier otro carácter válido. El ARN no coincidiría con los siguientes objetos:

```
DOC-EXAMPLE-BUCKET/1-test/object.jpg
DOC-EXAMPLE-BUCKET/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test.jpg
```

Especificación de varios recursos

Puede especificar varios recursos. El siguiente ejemplo hace referencia a dos tablas de DynamoDB.

```
"Resource": [
```

```
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/books_table",
    "arn:aws:dynamodb:us-east-2:account-ID-without-hyphens:table/magazines_table"
]
```

Uso de variables de política en ARN de recursos

En el elemento `Resource`, puede utilizar variables de política de JSON en la parte del ARN que identifica los recursos específicos (es decir, en la parte final del ARN). Por ejemplo, puede utilizar la clave `{aws:username}` como parte de un ARN de recurso para indicar que el nombre del usuario actual debe incluirse como parte del nombre del recurso. El siguiente ejemplo muestra cómo puede utilizar la clave `{aws:username}` en un elemento `Resource`. La política permite obtener acceso a una tabla de Amazon DynamoDB que coincide con el nombre de usuario actual.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:account-id:table/${aws:username}"
  }
}
```

Para obtener más información sobre las variables de las políticas de JSON, consulte Elementos de la política de IAM: variables y etiquetas.

Elementos de política JSON de IAM: NotResource

`NotResource` es un elemento de política avanzada que coincide explícitamente con todos los recursos excepto los especificados. El uso de `NotResource` puede traducirse en una política más abreviada; se publican únicamente unos cuantos recursos que no deben coincidir, en vez de incluir una larga lista de recursos que sí coincidirán. Esto resulta especialmente útil para las políticas que se aplican en un único servicio de AWS.

Por ejemplo, supongamos que tiene un grupo denominado `HRPayroll`. Los miembros de `HRPayroll` no deben tener permiso para obtener acceso a todos los recursos de Amazon S3, a excepción de la carpeta `Payroll` del bucket `HRBucket`. La política siguiente deniega de forma explícita el acceso a todos los recursos de Amazon S3 que no sean los recursos publicados en la lista. Tenga en cuenta, sin embargo, que esta política no concede al usuario acceso a ningún recurso.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": "s3:*",
    "NotResource": [
      "arn:aws:s3:::HRBucket/Payroll",
      "arn:aws:s3:::HRBucket/Payroll/*"
    ]
  }
}
```

Normalmente, para denegar explícitamente el acceso a un recurso debe escribir una política que utilice `"Effect":"Deny"` y que incluya un elemento `Resource` que enumere cada carpeta individualmente. Sin embargo, en dicho caso, cada vez que agregue una carpeta a `HRBucket` o agregue un recurso a Amazon S3; al que no debe poderse obtener acceso, deberá agregar su nombre a la lista de `Resource`. Si, en su lugar, utiliza un elemento `NotResource`, se deniega automáticamente a los usuarios el acceso a las carpetas nuevas, salvo que añada los nombres de las carpetas al elemento `NotResource`.

Cuando utilice `NotResource`, debe tener en cuenta que los recursos especificados en este elemento son los únicos recursos que no están limitados. Esto, a su vez, limita todos los recursos que se aplicarían a la acción. En el ejemplo anterior, la política solo afecta a las acciones de Amazon S3 y, por lo tanto, solo a los recursos de Amazon S3. Si la acción también incluía acciones de Amazon EC2, la política no denegaría el acceso a los recursos de EC2. Para saber qué acciones de un servicio permiten especificar el ARN de un recurso, consulte Acciones, recursos y claves de condición de los servicios de AWS.

NotResource con otros elementos

Nunca debe utilizar los elementos `"Effect": "Allow"`, `"Action": "*"` y `"NotResource": "arn:aws:s3:::HRBucket"` de forma conjunta. Esta instrucción es muy peligrosa, ya que permite todas las acciones en AWS en todos los recursos excepto el bucket de S3 `HRBucket`. Esto incluso permitiría al usuario añadir una política a sí mismo que le permitiera obtener acceso a `HRBucket`. No lo haga.

Sea precavido al utilizar los elementos `NotResource` y `"Effect": "Allow"` en la misma instrucción o en otra instrucción de una misma política. `NotResource` permite todos los servicios y recursos que no se mencionan explícitamente en la lista, lo que podría dar como resultado que

se concediera a los usuarios más permisos de los que en realidad se pretendía. Con los elementos `NotResource` y `"Effect": "Deny"` en la misma declaración se deniegan los servicios y recursos que no se mencionan explícitamente.

Elementos de política JSON de IAM: Condition

El elemento `Condition` (o el bloque `Condition`) permite especificar condiciones que se aplican cuando la política surte efecto. El elemento `Condition` es opcional. En el elemento `Condition`, se crean expresiones en las que se usan operadores de condición (igual, menor que, etc.) para hacer coincidir las claves de contexto y los valores de la política con las claves y valores en el contexto de la solicitud. Para obtener más información sobre el contexto de la solicitud, consulte Solicitud.

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

La clave de contexto que especifique en una condición de política puede ser una clave de contexto de condición global o una clave de contexto específica de un servicio. Las claves de contexto de condición globales tienen el prefijo `aws:`. Las claves de contexto específicas de un servicio tienen el prefijo del servicio. Por ejemplo, Amazon EC2 le permite escribir una condición con la clave de contexto `ec2:InstanceType`, que es exclusiva de dicho servicio. Para ver las claves de contexto de IAM específicas del servicio con el prefijo `iam:`, consulte Claves de contexto de condición de IAM y AWS STS.

Los nombres de las claves de contexto no distinguen entre mayúsculas y minúsculas. Por ejemplo, la inclusión de la clave de contexto `aws:SourceIP` es equivalente a las pruebas de `AWS:SourceIp`. El uso de mayúsculas y minúsculas en los valores de la clave de contexto depende del operador de condición que utilice. Por ejemplo, la siguiente condición incluye el operador `StringEquals` para garantizar que únicamente coincidan las solicitudes que realice `johndoe`. A los usuarios denominados `JohnDoe` se les niega el acceso.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

La siguiente condición utiliza el operador `StringEqualsIgnoreCase` para que coincida con los usuarios `johndoe` o `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

Algunas de las claves de contexto admiten los pares clave-valor, con los que puede especificar una parte del nombre de la clave. Entre los

ejemplos se incluyen la clave de contexto `aws:RequestTag/tag-key`, la `kms:EncryptionContext:encryption_context_key` de AWS KMS y la clave de contexto `ResourceTag/tag-key` compatibles con varios servicios.

- Si utiliza la clave de contexto `ResourceTag/tag-key` para un servicio como Amazon EC2, debe especificar un nombre de clave para `tag-key`.
- Los nombres de las claves no distinguen entre mayúsculas y minúsculas. Esto significa que si especifica `"aws:ResourceTag/TagKey1": "Value1"` en el elemento de condición de su política, la condición coincidirá con una clave de etiqueta de recurso denominada `TagKey1` o `tagkey1`, pero no con ambas.
- Los servicios de AWS que admiten estos atributos pueden permitirle crear varios nombres de clave que solo difieran por caso. Por ejemplo, puede etiquetar una instancia de Amazon EC2 con `ec2=test1` y `EC2=test2`. Cuando se utiliza una condición como `"aws:ResourceTag/EC2": "test1"` para permitir el acceso a dicho recurso, el nombre de clave coincide con ambas etiquetas, pero solo un valor coincide. Esto puede generar errores inesperados de la condición.

> **Important**
>
> La práctica recomendada es que se asegure de que los miembros de su cuenta sigan una convención de nomenclatura coherente al nombrar los atributos de par clave-valor. Entre los ejemplos se incluyen etiquetas o contextos de cifrado de AWS KMS. Para imponer este comportamiento, utilice la clave de contexto `aws:TagKeys` para el etiquetado o `kms:EncryptionContextKeys` para el contexto de cifrado de AWS KMS.

- Para obtener una lista de todos los operadores de condición y una descripción de cómo funcionan, consulte Operadores de condición.
- A menos que se especifique lo contrario, todas las claves de contexto pueden tener varios valores. Para ver una descripción de cómo gestionar las claves de contexto que tienen varios valores, consulte Claves de contexto multivalor.
- Para obtener una lista de todas las claves de contexto disponibles en todo el mundo, consulte Claves de contexto de condición globales de AWS.
- Para conocer las claves de contexto de condición definidas por cada servicio, consulte Acciones, recursos y claves de condición para servicios de AWS.

El contexto de la solicitud

Cuando una entidad principal realiza una solicitud a AWS, AWS recopila la información de la solicitud en un contexto de solicitud. La información se utiliza para evaluar y autorizar la solicitud. Puede utilizar el elemento `Condition` de una política JSON para probar claves de contexto específicas con respecto al contexto de la solicitud. Por ejemplo, puede crear una política que utilice la clave de contexto aws:CurrentTime para permitir a un usuario realizar acciones específicas solo durante un intervalo de fechas específico.

Cuando se envía una solicitud, AWS evalúa cada clave de contexto de la política y devuelve un valor de true, false, not present y, de vez en cuando, null (una cadena de datos vacía). Una clave de contexto que no está presente en la solicitud no se considera una discordancia. Por ejemplo, la siguiente política permite eliminar su propio dispositivo de autenticación multifactor (MFA), pero solo si ha iniciado sesión con MFA en la última hora (3600 segundos).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

El contexto de la solicitud puede devolver los siguientes valores:

- True: si el solicitante ha iniciado sesión con MFA en la última hora o menos, la condición devuelve true.
- False: si el solicitante ha iniciado sesión con MFA hace más de una hora, la condición devuelve false.
- Not present: si el solicitante realizó una solicitud con sus claves de acceso de usuario de IAM en la AWS CLI o la API de AWS, la clave no está presente. En este caso, la clave no está presente y no coincidirá.

- Null: para las claves de contexto definidas por el usuario, como la transferencia de etiquetas en una solicitud, es posible incluir una cadena vacía. En este caso, el valor en el contexto de la solicitud es null. Un valor null puede devolver true en algunos casos. Por ejemplo, si utiliza el operador de condición `ForAllValues` con varios valores con la clave de contexto `aws:TagKeys`, puede experimentar resultados inesperados si el contexto de la solicitud devuelve un valor null. Para obtener más información, consulte aws:TagKeys y Claves de contexto multivalor.

El bloque de condición

En el siguiente ejemplo se muestra el formato básico de un elemento `Condition`:

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

Un valor de la solicitud está representado por una clave de contexto, en este caso `s3:prefix`. El valor de clave de contexto se compara con un valor que especifique como valor literal, como `janedoe/*`. El tipo de comparación que debe realizarse se especifica con el operador de condición (aquí, `StringLike`). Puede crear condiciones para comparar cadenas, fechas, números, y más, mediante el uso de comparaciones booleanas típicas como igual, superior a e inferior a. Cuando se utilizan operadores de cadena u operadores de ARN, también se puede utilizar una variable de política en el valor de clave de contexto. El siguiente ejemplo incluye la variable `aws:username`.

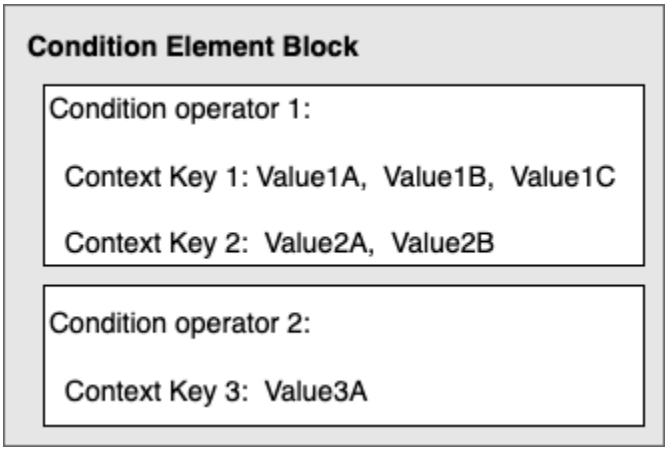
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

En determinadas circunstancias, las claves de contexto pueden contener múltiples valores. Por ejemplo, una solicitud a Amazon DynamoDB podría pedir la devolución o actualización de varios atributos de una tabla. Una política para obtener acceso a tablas de DynamoDB puede incluir la clave de contexto `dynamodb:Attributes`, que contiene todos los atributos indicados en la solicitud. Puede probar los diversos atributos de la solicitud con una lista de atributos permitidos de una política utilizando un conjunto de operadores del elemento `Condition`. Para obtener más información, consulte Claves de contexto multivalor.

Cuando se evalúa la política durante una solicitud, AWS sustituye la clave por el valor correspondiente de la solicitud. (En este ejemplo, AWS utilizaría la fecha y la hora de la solicitud). La condición se evalúa para devolver true o false, lo cual, a su vez, se examina para saber si la política en su conjunto permite o deniega la solicitud.

Múltiples valores en un elemento Condition

Un elemento `Condition` puede contener varios operadores de condición y cada uno de ellos puede contener, a su vez, varios pares de clave-valor de contexto. La siguiente figura ilustra este caso.



Para obtener más información, consulte Claves de contexto multivalor.

Elementos de la política de JSON de IAM: operadores de condición

Utilice operadores de condición en el elemento `Condition` para hacer coincidir la clave y el valor de la política con los valores del contexto de la solicitud. Para obtener más información sobre el parámetro `Condition`, consulte Elementos de política JSON de IAM: Condition.

El operador de condición que puede utilizar en una política depende de la clave de condición que elija. Puede elegir una clave de condición global o una clave de condición específica del servicio. Para saber qué operador de condición puede utilizar para una clave de condición global, consulte Claves de contexto de condición globales de AWS. Para saber qué operador de condición puede utilizar para una clave de condición específica de un servicio, consulte Acciones, recursos y claves de condición para servicios de AWS y elija el servicio que desea ver.

> **Important**
>
> Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán y la condición es falsa. Si la condición de la política exige que la clave nocoincida, como `StringNotLike` o `ArnNotLike` y la clave correcta no está presente, la condición es verdadera. Esta lógica se aplica a todos los operadores de condición excepto ...IfExists y Null check. Estos operadores prueban si la clave está presente (existe) en el contexto de la solicitud.

Los operadores de condición pueden agruparse en las categorías siguientes:

- Cadena
- Numérico
- Fecha y hora
- Booleano
- Binario
- Dirección IP
- Nombre de recurso de Amazon (ARN) (disponible solo para determinados servicios).
- ...IfExists (comprueba si el valor de clave existe como parte de otra comprobación)
- Null check (comprueba si el valor de clave existe como comprobación independiente)

Operadores de condición de cadena

Los operadores de condición de cadena le permiten desarrollar elementos `Condition` que restringen el acceso comparando una clave con el valor de una cadena.

Operador de condición	Descripción
`StringEquals`	Coincidencia exacta; distingue entre mayúsculas y minúsculas.
`StringNotEquals`	Coincidencia negada.
`StringEqualsIgnoreCase`	Coincidencia exacta; no distingue entre mayúsculas y minúsculas.
`StringNotEqualsIgnoreCase`	Coincidencia negada; no distingue entre mayúsculas y minúsculas.
`StringLike`	Coincidencia que distingue entre mayúsculas y minúsculas. Los valores pueden incluir comodines de coincidencia de varios caracteres (*) y comodines de coincidencia de un único carácter (?) en cualquier parte de la cadena. Debe especificar caracteres comodín para lograr coincidencias de cadenas parciales.

Operador de condición	Descripción
	Note Si una clave contiene varios valores, `StringLike` se puede calificar con un conjunto de operadores: `ForAllValues:StringLike` y `ForAnyValue:StringLike`. Para obtener más información, consulte Claves de contexto multivalor.
`StringNotLike`	Coincidencia negada que distingue entre mayúsculas y minúsculas. Los valores pueden incluir comodines de coincidencia de varios caracteres (*) o comodines de coincidencia de un único carácter (?) en cualquier parte de la cadena.

Por ejemplo, la instrucción siguiente contiene un elemento `Condition` que utiliza la clave `aws:PrincipalTag` para especificar que el principal que realiza la solicitud debe etiquetarse con la categoría de trabajo `iamuser-admin`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  }
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. En este ejemplo, la clave `aws:PrincipalTag/job-category` está presente en el contexto de la solicitud si la entidad principal utiliza un usuario IAM con etiquetas asociadas. También se incluye para un principal con un rol de IAM con etiquetas o etiquetas de sesión asociadas. Si un usuario sin la etiqueta intenta ver o editar una clave de acceso, la condición devuelve `false` y la instrucción deniega la solicitud implícitamente.

Puede utilizar una variable de política con el operador de condición `String`.

En el siguiente ejemplo se utiliza el operador de condición `StringLike` para establecer una coincidencia de cadena con una variable de política y crear una política que permita a un usuario de IAM utilizar la consola de Amazon S3 para administrar su propio "directorio principal" en un bucket de Amazon S3. La política permite las acciones especificadas en un bucket de S3, siempre y cuando `s3:prefix` coincida con uno de los patrones especificados.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Resource": "arn:aws:s3:::*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

Para ver un ejemplo de una política que muestre cómo utilizar el elemento `Condition` para restringir el acceso a recursos en función de un ID de aplicación y un ID de usuario de la federación de OIDC, consulte Amazon S3: permite a los usuarios de Amazon Cognito obtener acceso a los objetos de su bucket.

Coincidencia de caracteres comodín

Los operadores de condición de cadena realizan una coincidencia sin patrones que no impone un formato predefinido. Los operadores de condición de ARN y fecha son un subconjunto de operadores de cadena que imponen una estructura al valor de la clave de condición. Al utilizar los operadores StringLike o StringNotLike para las coincidencias parciales de cadenas de un ARN o una fecha, la coincidencia ignora qué parte de la estructura aparece como comodín.

Por ejemplo, las siguientes condiciones buscan una coincidencia parcial de un ARN mediante diferentes operadores de condición.

Cuando se usa ArnLike, las partes del ARN de partición, servicio, ID de cuenta, tipo de recurso e ID de recurso parcial deben coincidir exactamente con el ARN en el contexto de la solicitud. Solo la región y la ruta del recurso permiten la coincidencia parcial.

```
"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

Cuando se usa StringLike en lugar de ArnLike, la coincidencia ignora la estructura del ARN y permite la coincidencia parcial, independientemente de la parte marcada como comodín.

```
"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

ARN	ArnLike	StringLike
ar:aws:cloudtrail:us-west-2 :111122223333:trail/finance	Match	Match
arn:aws:cloudtrail:us-east-2:111122223333:trail/finance/archive	Match	Match
ar:aws:cloudtrail:us-east-2:444455556 666:user/111122223333:trail/finance	Sin coincidencia	Match

Operadores de condición numérica

Los operadores de condición numérica le permiten desarrollar elementos `Condition` que restringen el acceso comparando una clave con un valor entero o un valor decimal.

Operador de condición	Descripción
`NumericEquals`	Coincidencia
`NumericNotEquals`	Coincidencia negada.
`NumericLessThan`	Coincidencia "menos que"
`NumericLessThanEquals`	Coincidencia "menos que o igual"
`NumericGreaterThan`	Coincidencia "más que"
`NumericGreaterThanEquals`	Coincidencia "superior a o igual"

Por ejemplo, la siguiente instrucción contiene un elemento `Condition` que utiliza el operador de condición `NumericLessThanEquals` con la clave `s3:max-keys` para especificar que el solicitante puede incluir en la lista hasta 10 objetos en `example_bucket` a la vez.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_bucket",
    "Condition": {"NumericLessThanEquals": {"s3:max-keys": "10"}}
  }
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. En este ejemplo, la clave `s3:max-keys` siempre está presente en la solicitud cuando se realiza la operación `ListBucket`. Si esta política permitía todas las operaciones

de Amazon S3, solo se permitirían las operaciones que incluyen la clave de contexto `max-keys` con un valor inferior o igual a 10.

No puede utilizar una variable de política con el operador de condición `Numeric`.

Operadores de condición de fecha

Los operadores de condición de fecha le permiten desarrollar elementos `Condition` que restringen el acceso comparando una clave con el valor de una fecha/hora. Los operadores de condición se usan con la clave `aws:CurrentTime` o la clave `aws:EpochTime`. Debe especificar valores de fecha y hora con una de las implementaciones W3C de los formatos de fecha ISO 8601 o en la fecha de inicio (UNIX).

> **Note**
>
> Los comodines no están permitidos en los operadores de condición de fecha.

Operador de condición	Descripción
`DateEquals`	Coincidencia con una fecha específica.
`DateNotEquals`	Coincidencia negada.
`DateLessThan`	Coincidencia antes de una fecha y hora específicas.
`DateLessThanEquals`	Coincidencia en una fecha y hora específicas o antes.
`DateGreaterThan`	Coincidencia después de una fecha y hora específicas.
`DateGreaterThanEqu als`	Coincidencia en una fecha y hora específicas o después.

Por ejemplo, la instrucción siguiente contiene un elemento `Condition` que utiliza el operador de condición `DateGreaterThan` con la clave `aws:TokenIssueTime`. Esta condición especifica que las credenciales de seguridad temporales utilizadas para realizar la solicitud se emitieron en 2020. Esta política se puede actualizar mediante programación todos los días para asegurarse de que los miembros de la cuenta utilizan credenciales nuevas.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"DateGreaterThan": {"aws:TokenIssueTime": "2020-01-01T00:00:01Z"}}
  }
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. La clave `aws:TokenIssueTime` está presente en el contexto de la solicitud solo cuando el principal utiliza credenciales temporales para realizar la solicitud. La clave no está presente en solicitudes de la AWS CLI, la API de AWS o el SDK de AWS realizadas con claves de acceso. En este ejemplo, si un usuario de IAM intenta ver o editar una clave de acceso, se deniega la solicitud.

No puede utilizar una variable de política con el operador de condición `Date`.

Operadores de condición booleanos

Las condiciones booleanas le permiten crear elementos `Condition` que restringen el acceso comparando una clave con "true" o "false".

Operador de condición	Descripción
`Bool`	Coincidencia booleana

Por ejemplo, esta política basada en identidad utiliza el operador de condición `Bool` con la clave `aws:SecureTransport` para denegar la replicación de objetos y etiquetas de objetos en el bucket de destino y su contenido si la solicitud no se lleva a cabo a través de SSL.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
        }
      }
    }
  ]
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. El contexto de la solicitud de `aws:SecureTransport` devuelve verdadero o falso.

Puede utilizar una variable de política con el operador de condición `Boolean`.

Operadores de condición binaria

El operador de condición `BinaryEquals` le permite crear elementos `Condition` que prueban valores de clave que están en formato binario. Compara el valor del byte de la clave especificada con una representación codificada en base 64 del valor binario de la política.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán.

No puede utilizar una variable de política con el operador de condición `Binary`.

Operadores de condición de dirección IP

Los operadores de condición de la dirección IP le permiten crear elementos `Condition` que restringen el acceso basándose en una comparación de una clave con una dirección IPv4 o IPv6 o un rango de direcciones IP. Puede utilizarlos con la clave `aws:SourceIp`. El valor debe tener el formato CIDR estándar (por ejemplo, 203.0.113.0/24 o 2001: DB8:1234:5678::/64). Si especifica una dirección IP sin el prefijo de enrutamiento asociado, IAM utiliza el valor del prefijo predeterminado `/32`.

Algunos dispositivos AWS admiten IPv6 utilizando :: para representar un rango de 0. Para saber si un servicio es compatible con IPv6, consulte la documentación correspondiente a dicho servicio.

Operador de condición	Descripción
`IpAddress`	La dirección o el rango IP especificado
`NotIpAddress`	Todas las direcciones IP, salvo la dirección o el rango IP especificado

Por ejemplo, la siguiente instrucción utiliza el operador de condición `IpAddress` con la clave `aws:SourceIp` para especificar que la solicitud debe provenir del rango IP 203.0.113.0 a 203.0.113.255.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id:user/*",
    "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
  }
}
```

La clave de condición `aws:SourceIp` se resuelve en la dirección IP de la que proviene la solicitud. Si la solicitud se origina en una instancia de Amazon EC2, `aws:SourceIp` toma el valor de la dirección IP pública de la instancia.

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. La clave `aws:SourceIp` se incluye siempre en el contexto de la solicitud,

excepto cuando el solicitante utiliza un punto de enlace de la VPC para realizar la solicitud. En este caso, la condición devuelve `false` y la instrucción deniega la solicitud implícitamente.

No puede utilizar una variable de política con el operador de condición `IpAddress`.

En el siguiente ejemplo se muestra cómo combinar direcciones IPv4 e IPv6 para incluir todas las direcciones IP válidas de su organización. Le recomendamos que actualice las políticas de su organización con los rangos de direcciones IPv6 además de los rangos de IPv4 que ya tiene para asegurarse de que las políticas seguirán funcionando a medida que realiza la transición a IPv6.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "someservice:*",
    "Resource": "*",
    "Condition": {
      "IpAddress": {
        "aws:SourceIp": [
          "203.0.113.0/24",
          "2001:DB8:1234:5678::/64"
        ]
      }
    }
  }
}
```

La clave de condición `aws:SourceIp` funciona solo en una política JSON si se llama a la API que se está probando directamente como usuario. Si en su lugar utiliza un servicio para llamar al servicio de destino en su nombre, el servicio de destino verá la dirección IP del servicio de llamada en lugar de la dirección IP del usuario de origen. Esto puede ocurrir, por ejemplo, si utiliza AWS CloudFormation para llamar a Amazon EC2 para crear instancias en su lugar. Actualmente no se puede transferir la dirección IP de origen mediante un servicio de llamada al servicio de destino para que lo evalúe una política de JSON. En este tipo de llamadas a la API de servicio, no utilice la clave de condición `aws:SourceIp`.

Operadores de condición de nombre de recurso de Amazon (ARN)

Los operadores de condición de nombre de recurso de Amazon (ARN) le permiten crear elementos `Condition` que restringen el acceso comparando una clave con un ARN. El ARN se considera una cadena.

Operador de condición	Descripción
`ArnEquals` , `ArnLike`	Coincidencia del ARN que distingue entre mayúsculas y minúsculas. Cada uno de los seis componentes del ARN delimitados por dos puntos se comprueba por separado y cada uno de ellos puede incluir comodines de coincidencia de varios caracteres (*) o comodines de coincidencia de un único carácter (?). Los operadores de condición `ArnEquals` y `ArnLike` se comportan de forma idéntica.
`ArnNotEquals` , `ArnNotLike`	Coincidencia negada del ARN. Los operadores de condición `ArnNotEquals` y `ArnNotLike` se comportan de forma idéntica.

Puede utilizar una variable de política con el operador de condición ARN.

En el siguiente ejemplo de política basada en recursos se muestra una política asociada a una cola de Amazon SQS a la que desea enviar mensajes SNS. Da a Amazon SNS permiso para enviar mensajes a la cola (o colas) de su elección, pero solo si el servicio envía los mensajes en nombre de un determinado tema (o temas) de Amazon SNS. La cola se especifica en el campo `Resource` y el tema de Amazon SNS se especifica como el valor de la clave `SourceArn`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "123456789012"},
    "Action": "SQS:SendMessage",
    "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
    "Condition": {"ArnEquals": {"aws:SourceArn":
 "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
  }
}
```

Si la clave especificada en una condición de política no está presente en el contexto de la solicitud, los valores no coincidirán. La clave `aws:SourceArn` está presente en el contexto de la solicitud solo si un recurso activa un servicio para llamar a otro servicio en nombre del propietario del recurso. Si un usuario de IAM intenta realizar esta operación directamente, la condición devuelve `false` y la instrucción deniega la solicitud implícitamente.

Operadores de condición …IfExists

Puede agregar `IfExists` al final de cualquier nombre de operador de condición, salvo la condición `Null` por ejemplo, `StringLikeIfExists`. El objetivo es decir lo siguiente: “Si la clave de la política está presente en el contexto de la solicitud, se debe procesar la clave según se indica en la política. Si la clave no está presente, el elemento de condición se evalúa en verdadero". Otros elementos de condición de la instrucción pueden seguir sin obtener una coincidencia, pero no una clave que falte cuando se comprueba con `...IfExists`. Si utiliza un elemento `"Effect": "Deny"` con un operador de condición negada como `StringNotEqualsIfExists`, la solicitud se sigue denegando aunque falte la etiqueta.

Ejemplo de uso de **`IfExists`**

Muchas claves de condición describen información sobre un determinado tipo de recurso y solo existen cuando se obtiene acceso a ese tipo de recurso. Estas claves de condición no están presentes en otros tipos de recursos. Esto no crea ningún problema cuando la instrucción de la política se aplica únicamente a un tipo de recurso. Sin embargo, en algunos casos una única instrucción se aplica a varios tipos de recursos, como, por ejemplo, cuando la instrucción de la política hace referencia a acciones de varios servicios o cuando una acción determinada de un servicio obtiene acceso a diferentes tipos de recursos en el mismo servicio. En estos casos, la inclusión de una clave de condición que se aplique únicamente a uno de los recursos de la instrucción de la política puede hacer que el elemento `Condition` de la instrucción dé un error y que su `"Effect"` no se aplique.

Por ejemplo, tomemos el siguiente ejemplo de política:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "THISPOLICYDOESNOTWORK",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {"StringLike": {"ec2:InstanceType": [
      "t1.*",
      "t2.*",
      "m3.*"
    ]}}
  }
}
```

El propósito de la política anterior es habilitar al usuario para que pueda lanzar cualquier instancia de tipo `t1`, `t2` o `m3`. Sin embargo, lanzar una instancia requiere poder obtener acceso a muchos recursos además de la instancia en sí; por ejemplo, imágenes, pares de claves, grupos de seguridad, entre otros. Toda la instrucción se evalúa con respecto a todos los recursos exigidos para lanzar la instancia. Estos recursos adicionales no tienen la clave de condición `ec2:InstanceType`, por lo que la comprobación `StringLike` da un error y no se concede al usuario la capacidad para lanzar cualquier tipo de instancia.

Para solucionar este problema, utilice en su lugar el operador de condición `StringLikeIfExists`. De esta forma, la prueba solo se realiza si la clave de condición existe. Podría leer la siguiente política como: “Si el recurso que se está comprobando tiene una clave de condición ‘`ec2:InstanceType`’, permita la acción solo si el valor de clave comienza por `t1.`, `t2.` o `m3.`. Si el recurso que se está comprobando no tiene esta clave de condición, no deberá tenerse en cuenta”. El asterisco (*) de los valores de la clave de condición, cuando se usa con el operador de condición `StringLikeIfExists`, se interpreta como un comodín para lograr coincidencias parciales de cadenas. La instrucción `DescribeActions` incluye las acciones requeridas para ver la instancia en la consola.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RunInstance",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*",
            "Condition": {
                "StringLikeIfExists": {
                    "ec2:InstanceType": [
                        "t1.*",
                        "t2.*",
                        "m3.*"
             ]}}
        },
        {
            "Sid": "DescribeActions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeVpcs",
```

```
                "ec2:DescribeKeyPairs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups"
            ],
            "Resource": "*"
        }]
}
```

Operador de condición para comprobar la existencia de claves de condición

Utilice un operador de condición `Null` para comprobar si una clave de condición está ausente en el momento de la autorización. En la instrucción de la política, utilice `true` (la clave no existe es nula) o `false` (la clave existe y su valor no es nulo).

No puede utilizar una variable de política con el operador de condición `Null`.

Por ejemplo, puede utilizar este operador de condición para determinar si un usuario utiliza sus propias credenciales para la operación o credenciales temporales. Si el usuario utiliza credenciales temporales, la clave `aws:TokenIssueTime` existe y tiene un valor. En el siguiente ejemplo se muestra una condición que establece que el usuario no debe utilizar credenciales temporales (la clave no debe existir) para utilizar la API de Amazon EC2.

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"true"}}
  }
}
```

Condiciones con varias claves de contexto o valores

Puede utilizar el elemento `Condition` de una política para probar varias claves de contexto o valores para una única clave de contexto en una solicitud. Cuando se realiza una solicitud a AWS, ya sea mediante programación o a través de la AWS Management Console, la solicitud incluye información sobre la entidad principal, la operación, las etiquetas y mucho más. Puede utilizar las claves de contexto para probar los valores de las claves de contexto coincidentes de la solicitud, con las claves de contexto especificadas en la condición de política. Para conocer la información y los datos incluidos en una solicitud, consulte El contexto de la solicitud.

Temas

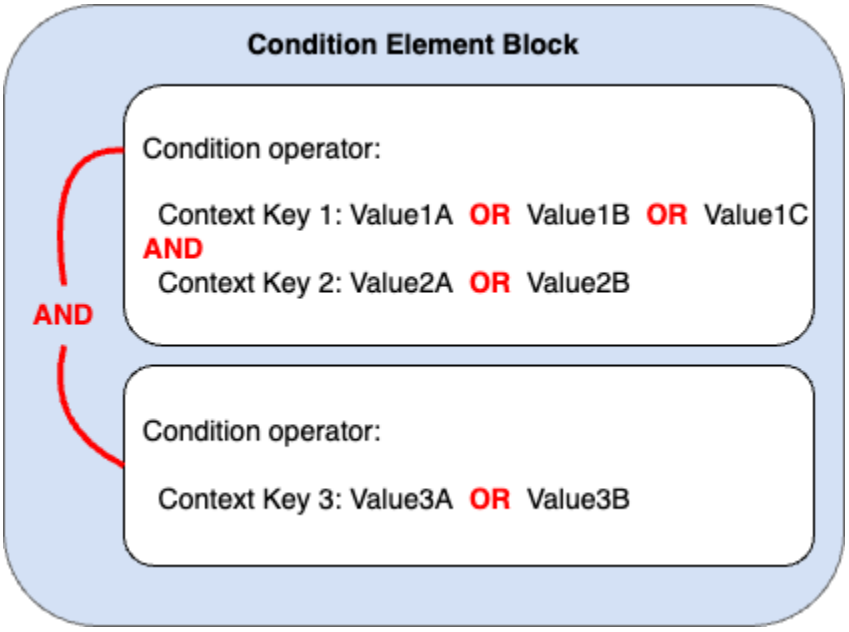
- Lógica de evaluación para varias claves de contexto o valores
- Lógica de evaluación de los operadores de condición con coincidencia negada

Lógica de evaluación para varias claves de contexto o valores

Un elemento `Condition` puede contener varios operadores de condición y cada uno de ellos puede contener, a su vez, varios pares de clave-valor de contexto. La mayoría de las claves de contexto admiten el uso de varios valores, a menos que se especifique lo contrario.

- Si su instrucción de política tiene varios operadores de condición, los operadores de condición se evalúan mediante un operador lógico `AND`.
- Si su instrucción de política tiene varias claves de contexto asociadas a un único operador de condición, las claves de contexto se evalúan mediante un operador lógico `AND`.
- Si un operador de condición contiene varios valores para una clave de contexto, esos valores se evalúan mediante un operador lógico `OR`.
- Si un operador de condición con coincidencia negada contiene varios valores para una clave de contexto, esos valores se evalúan mediante un operador lógico `NOR`.

Todas las claves de contexto de un bloque de elementos de condición deben resolverse como verdaderas para invocar el efecto `Allow` o `Deny` deseado. En la siguiente figura, se ilustra la lógica de evaluación de una condición con varios operadores de condición y pares clave-valor de contexto.



Por ejemplo, la siguiente política de bucket de S3 ilustra cómo se representa la figura anterior en una política. El bloque de condición incluye los operadores de condición `StringEquals` y `ArnLike` y las claves de contexto `aws:PrincipalTag` y `aws:PrincipalArn`. Para invocar el efecto `Allow` o `Deny` deseado, todas las claves de contexto en el bloque de condición deben resolverse como verdaderas. El usuario que realiza la solicitud debe tener ambas claves de etiqueta de las entidades principales, departamento y rol, que incluyen uno de los valores de clave de etiqueta especificados en la política. Además, el ARN de la entidad principal del usuario que realiza la solicitud debe coincidir con uno de los valores `aws:PrincipalArn` especificados en la política que se evaluarán como verdaderos.

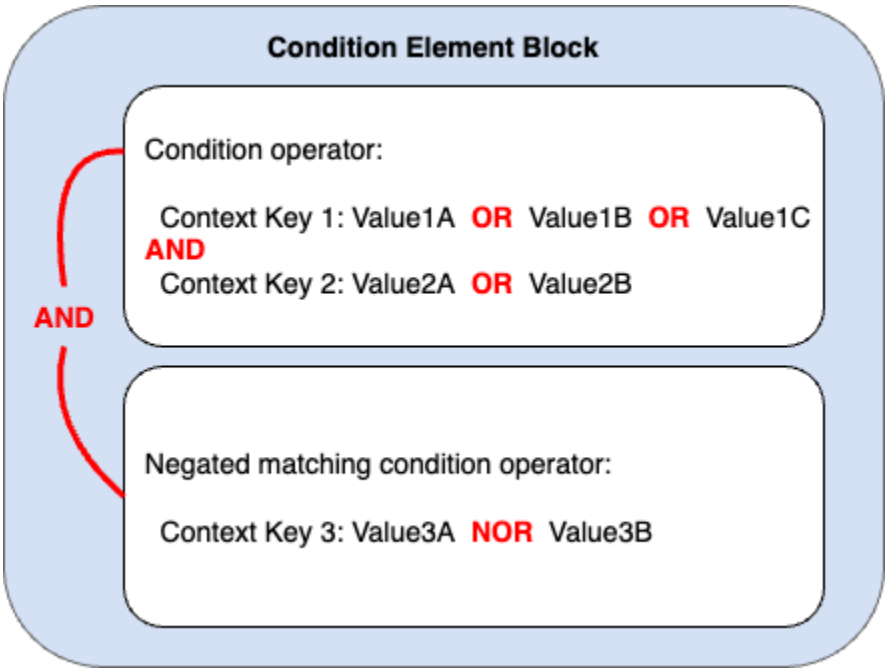
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
```

```
}
```

Lógica de evaluación de los operadores de condición con coincidencia negada

Algunos operadores de condición, como `StringNotEquals` o `ArnNotLike`, utilizan la coincidencia negada para comparar los pares clave-valor de contexto de su política con los pares clave-valor de contexto de una solicitud. Cuando se especifican varios valores para una clave de contexto única en una política con operadores de condición con coincidencia negada, los permisos vigentes funcionan como un operador lógico `NOR`. En la coincidencia negada, un operador lógico `NOR` o `NOT OR` devuelve verdadero solo si todos los valores se evalúan como falsos.

En la siguiente figura, se ilustra la lógica de evaluación de una condición con varios operadores de condición y pares clave-valor de contexto. La figura incluye un operador de condición con coincidencia negada para la clave de contexto 3.



Por ejemplo, la siguiente política de bucket de S3 ilustra cómo se representa la figura anterior en una política. El bloque de condición incluye los operadores de condición `StringEquals` y `ArnNotLike` y las claves de contexto `aws:PrincipalTag` y `aws:PrincipalArn`. Para invocar el efecto `Allow` o `Deny` deseado, todas las claves de contexto en el bloque de condición deben resolverse como verdaderas. El usuario que realiza la solicitud debe tener ambas claves de etiqueta de las entidades principales, departamento y rol, que incluyen uno de los valores de clave de etiqueta especificados en la política. Dado que el operador de condición `ArnNotLike` usa la coincidencia negada, el ARN de la entidad principal del usuario que realiza la solicitud no debe coincidir con

ninguno de los valores `aws:PrincipalArn` especificados en la política que se evaluarán como verdaderos.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": [
            "finance",
            "hr",
            "legal"
          ],
          "aws:PrincipalTag/role": [
            "audit",
            "security"
          ]
        },
        "ArnNotLike": {
          "aws:PrincipalArn": [
            "arn:aws:iam::222222222222:user/Ana",
            "arn:aws:iam::222222222222:user/Mary"
          ]
        }
      }
    }
  ]
}
```

Claves de contexto de valor único y multivalor

La diferencia entre las claves de contexto de valor único y multivalor depende del número de valores en el contexto de la solicitud, no de la cantidad de valores de la condición de política.

- Las claves de contexto de condición de valor único tienen como máximo un valor en el contexto de la solicitud. Por ejemplo, puede etiquetar recursos en AWS. Las etiquetas de recursos se almacenan como pares de valor de clave de etiqueta. Una clave de etiqueta de recurso puede tener un solo valor de etiqueta. Por lo tanto, the section called “ResourceTag” es una clave de contexto de un solo valor. No utilice un operador de conjunto de condiciones con una clave de contexto de un solo valor.
- Las claves de contexto de condición multivalor pueden tener varios valores en el contexto de la solicitud. Por ejemplo, se pueden etiquetar los recursos en AWS e incluir varios pares de valor de clave de etiqueta en una solicitud. Por lo tanto, the section called “TagKeys” es una clave de contexto multivalor. Las claves de contexto multivalor requieren un operador de conjunto de condiciones.

> **Important**
>
> Las claves de contexto multivalor requieren un operador de conjunto de condiciones. No utilice los operadores de conjunto de condiciones `ForAllValues` o `ForAnyValue` con claves de contexto de un solo valor. Para obtener más información sobre los operadores de conjuntos de condiciones, consulte Claves de contexto multivalor.

Las clasificaciones Valor único y Multivalor se incluyen en la descripción de cada clave de contexto de condición como Tipo de valor en el tema Claves de contexto de condición globales de AWS. En la Referencia de autorización de servicio, se utiliza una clasificación de tipos de valores diferente para las claves de contexto multivalor en el siguiente formato: un prefijo `ArrayOf` seguido del tipo de categoría del operador de condición. Por ejemplo, `ArrayOfString` o `ArrayOfARN`.

Por ejemplo, una solicitud puede originarse a lo sumo de un punto de conexión de VPC, por lo que the section called “SourceVpce” es una clave de contexto de valor único. Dado que un servicio puede tener más de un nombre de entidad principal de servicio que pertenece al servicio, aws:PrincipalServiceNamesList es una clave de contexto multivalor.

Puede utilizar cualquier clave de contexto de valor único disponible como variable de política. No se puede utilizar una clave de contexto multivalor como variable de política. Para obtener más información sobre las variables de las políticas, consulte Elementos de la política de IAM: variables y etiquetas.

Los operadores de conjuntos de condiciones `ForAllValues` o `ForAnyValue` son necesarios para las claves de contexto multivalor. Las claves de contexto que incluyen pares de clave-valor, tales como the section called “RequestTag” y the section called “ResourceTag”, pueden causar confusión porque puede haber múltiples valores de *`tag-key`*. Pero dado que cada *`tag-key`* solo puede tener un valor, `aws:RequestTag` y `aws:ResourceTag` son claves de contexto de valor único. El uso de operadores de conjuntos de condiciones con claves de contexto de valor único puede dar lugar a políticas demasiado permisivas.

Claves de contexto multivalor

Para comparar su clave de contexto de condición con un contexto de solicitud con varios valores, debe utilizar los operadores de conjunto `ForAllValues` o `ForAnyValue`. Estos operadores de conjunto se utilizan para comparar dos conjuntos de valores, como el conjunto de etiquetas en una solicitud y el conjunto de etiquetas en una condición de política.

Estos calificadores `ForAllValues` y `ForAnyValue` agregan la funcionalidad de operación de conjunto al operador de condición para que pueda probar claves de contexto con varios valores contra varios valores de clave de contexto en una condición de política. Además, si incluye una clave de contexto multivalor en su política con un comodín o una variable, también debe utilizar el operador de condición `StringLike`. Los valores de las claves de condición múltiples se deben escribir entre corchetes, como una matriz. Por ejemplo, `"Key2":["Value2A", "Value2B"]`.

- `ForAllValues`: este calificador prueba si el valor de cada miembro del conjunto de solicitudes es un subconjunto del conjunto de claves de contexto de condición. La condición devuelve verdadero si cada valor de clave de contexto de la solicitud coincide con al menos un valor de clave de contexto en la política. También devuelve verdadero si no hay claves de contexto en la solicitud o si los valores de clave de contexto se resuelven en un conjunto de datos nulo, como una cadena vacía. Para evitar que las claves de contexto faltantes o las claves de contexto con valores vacíos se evalúen como verdaderas, puede incluir el operador de condición `Null` en su política con un valor falso para comprobar si la clave de contexto existe y su valor no es nulo.

> **Important**
>
> Tenga cuidado si usa `ForAllValues` con un efecto `Allow`, porque puede resultar demasiado permisivo si la presencia de claves de contexto faltantes o claves de contexto con valores vacíos en el contexto de la solicitud es inesperada. Puede incluir el operador de condición `Null` en su política con un valor falso para comprobar si la clave de contexto

existe y su valor no es nulo. Para ver un ejemplo, consulte Control del acceso en función de las claves de etiqueta.

- `ForAnyValue`: este calificador prueba si al menos un miembro del conjunto de valores de clave de contexto de la solicitud coincide con al menos un miembro del conjunto de valores de la clave de contexto de su política. La clave de contexto devuelve verdadero si alguno de los valores de clave de contexto de la solicitud coincide con alguno de los valores de clave de contexto de la política. Si no hay una clave de contexto que coincida o si hay un conjunto de datos nulo, la condición devuelve falso.

> **Note**
>
> La diferencia entre las claves de contexto de valor único y multivalor depende del número de valores en el contexto de la solicitud, no de la cantidad de valores de la condición de política.

Ejemplos de políticas de condiciones

En las políticas de IAM, puede especificar varios valores para las claves de contexto de un solo valor y multivalor para compararlos con el contexto de la solicitud. El siguiente conjunto de ejemplos de políticas muestra condiciones de políticas con varias claves de contexto y valores.

> **Note**
>
> Si desea enviar una política para que se incluya en esta guía de referencia, utilice el botón Feedback (Comentarios) de la parte inferior de esta página. Para ver ejemplos de políticas basadas en identidad de IAM, consulte Ejemplos de políticas basadas en identidad de IAM.

Ejemplos de política de condición: claves de contexto con un solo valor

- Varios bloques de condiciones con claves de contexto de un solo valor. (Ver este ejemplo).
- Un bloque de condiciones con varias claves de contexto de un solo valor y varios valores. (Ver este ejemplo).

Ejemplos de política de condición: claves de contexto multivalor

- Política de denegación con el operador de conjunto de condiciones `ForAllValues`. (Ver este ejemplo).
- Política de denegación con el operador de conjunto de condiciones `ForAnyValue`. (Ver este ejemplo).

Ejemplos de clave de contexto multivalor

El siguiente conjunto de ejemplos de políticas muestra cómo crear condiciones de políticas con claves de contexto multivalor.

Ejemplo: política de denegación con el operador de conjunto de condiciones ForAllValues

El siguiente ejemplo de política basada en identidad niega el uso de acciones de etiquetado de IAM cuando se incluyen prefijos de clave de etiqueta específicos en la solicitud. Cada valor de la clave de contexto `aws:TagKeys` incluye un comodín (*) para la coincidencia parcial de cadenas. La política incluye el operador de conjunto `ForAllValues` con la clave de contexto `aws:TagKeys`, porque la clave de contexto de la solicitud puede incluir varios valores. Para que la clave de contexto `aws:TagKeys` devuelva verdadero, cada valor de la solicitud debe coincidir con al menos un valor de la política.

El operador de conjunto `ForAllValues` también devuelve verdadero si no hay claves de contexto en la solicitud o si los valores de clave de contexto se resuelven en un conjunto de datos nulo, como una cadena vacía. Para evitar que las claves de contexto faltantes o las claves de contexto con valores vacíos se evalúen como verdaderas, incluya el operador de condición `Null` en su política con un valor `false` para comprobar si la clave de contexto existe en la solicitud y su valor no es nulo.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

Ejemplo: política de denegación con el operador de conjunto de condiciones ForAnyValue

El siguiente ejemplo de política basada en identidad niega la creación de instantáneas de volúmenes de instancias de EC2 si alguna instantánea está etiquetada con una de las claves de etiqueta especificadas en la política, `environment` o `webserver`. La política incluye el operador de conjunto `ForAnyValue` con la clave de contexto `aws:TagKeys`, porque la clave de contexto de la solicitud puede incluir varios valores. Si su solicitud de etiquetado incluye alguno de los valores de clave de etiqueta especificados en la política, la clave de contexto `aws:TagKeys` devuelve verdadero e invoca el efecto de la política de denegación.

> **Important**
>
> Esta política no permite ninguna acción. Utilice esta política en combinación con otras políticas que permiten acciones específicas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "arn:aws:ec2:us-west-2::snapshot/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": ["environment", "webserver"]
                }
            }
        }
    ]
}
```

Ejemplos de políticas de claves de contexto con un solo valor

El siguiente conjunto de ejemplos de políticas muestra cómo crear condiciones de políticas con claves de contexto de valor único.

Ejemplo: varios bloques de condiciones con claves de contexto de un solo valor

Cuando un bloque de condiciones tiene varias condiciones, cada una con una única clave de contexto, todas las claves de contexto deben resolverse como verdaderas para que se invoque el efecto `Allow` o `Deny` deseado. Cuando se utilizan operadores de condición con coincidencia negada, la lógica de evaluación del valor de la condición se invierte.

El siguiente ejemplo permite a los usuarios crear volúmenes de EC2 y aplicar etiquetas a los volúmenes durante la creación de estos. El contexto de la solicitud debe incluir un valor para la clave de contexto `aws:RequestTag/project`, y el valor de la clave de contexto `aws:ResourceTag/environment` puede ser cualquiera excepto el de producción.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/environment": "production"
        }
      }
    }
  ]
}
```

El contexto de la solicitud debe incluir un valor de etiqueta del proyecto y no se puede crear para que un recurso de producción invoque el efecto `Allow`. El siguiente volumen de EC2 se creó correctamente porque el nombre del proyecto es `Feature3` con una etiqueta de recurso `QA`.

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=project,Value=Feature3},
{Key=environment,Value=QA}]'
```

Ejemplo: un bloque de condiciones con varias claves de contexto de un solo valor y varios valores

Cuando un bloque de condiciones contiene varias claves de contexto y cada clave de contexto tiene varios valores, cada clave de contexto debe resolverse como verdadera para que se invoque al menos un valor de clave para el efecto `Allow` o `Deny` deseado. Cuando se utilizan operadores

de condición con coincidencia negada, la lógica de evaluación del valor de la clave de contexto se invierte.

El siguiente ejemplo permite a los usuarios iniciar y ejecutar tareas en los clústeres de Amazon Elastic Container Service.

- El contexto de la solicitud debe incluir `production` O `pre-prod` para la clave de contexto `aws:RequestTag/environment` AND.
- La clave de contexto `ecs:cluster` asegura que las tareas se ejecuten en cualquiera de los clústeres de ECS con ARN `default1` O `default2`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:RunTask",
        "ecs:StartTask"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": [
            "production",
            "prod-backup"
          ]
        },
        "ArnEquals": {
          "ecs:cluster": [
            "arn:aws:ecs:us-east-1:111122223333:cluster/default1",
            "arn:aws:ecs:us-east-1:111122223333:cluster/default2"
          ]
        }
      }
    }
  ]
}
```

Elementos de la política de IAM: variables y etiquetas

Utilice las variables de las políticas de AWS Identity and Access Management (IAM) como marcadores de posición cuando no sepa el valor exacto de un recurso o una clave de condición al escribir la política.

> **Note**
>
> Si AWS no puede resolver una variable, esto puede provocar que toda la declaración no sea válida. Por ejemplo, si utiliza la variable `aws:TokenIssueTime`, la variable se resuelve a un valor solo cuando el solicitante se ha autenticado mediante credenciales temporales (un rol de IAM). Para evitar que las variables causen declaraciones no válidas, utilice el operador de condición IfExist.

Temas

Introducción

En las políticas de IAM, hay muchas acciones que le permiten asignar un nombre para los recursos concretos cuyo acceso desea controlar. Por ejemplo, la siguiente política permite a los usuarios enumerar y escribir objetos en el bucket de S3 `DOC-EXAMPLE-BUCKET` para proyectos de `marketing`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/marketing/*"]
    }
  ]
}
```

Puede darse el caso de que no sepa el nombre exacto del recurso cuando escribe la política. Es posible que le interese generalizar la política de modo que se adapte a muchos usuarios sin necesidad de realizar una única copia de la política para cada usuario. En lugar de crear una política independiente para cada usuario, le recomendamos que cree una única política de grupo que sirva para todos los usuarios de ese grupo.

Uso de variables en políticas

Puede definir valores dinámicos dentro de las políticas mediante variables de política que establezcan marcadores de posición en una política.

Las variables se marcan mediante un prefijo **$** seguido de un par de llaves (**{ }**) que incluyen el nombre de la variable del valor de la solicitud.

Cuando se evalúa una política, las variables de la política se sustituyen por valores procedentes de las claves de contexto condicionales pasadas en la solicitud. Las variables se pueden usar en políticas basadas en identidades, políticas de recursos, políticas de control de servicios, políticas de sesión y políticas de punto de conexión de VPC. Las políticas basadas en identidades que se utilizan como límites de permisos también admiten variables de política.

Las claves de contexto de condición global se pueden utilizar como variables en las solicitudes en los servicios de AWS. Las claves de condición específicas del servicio también se pueden usar como variables al interactuar con los recursos de AWS, pero solo están disponibles cuando se realizan solicitudes a los recursos que las admiten. Para obtener una lista de las claves de contexto disponibles para cada servicio y recurso de AWS, consulte la Referencia de autorización de servicios. En determinadas circunstancias, no puede rellenar las claves de contexto de condición

global con un valor. Para obtener más información sobre cada clave, consulte Claves de contexto de condición global de AWS.

> **Important**
>
> - Los nombres de clave no distinguen entre mayúsculas y minúsculas. Por ejemplo, `aws:CurrentTime` equivale a `AWS:currenttime`.
> - Puede utilizar cualquier clave de condición de valor único como variable. No se puede utilizar una clave de condición multivalor como variable.

En el siguiente ejemplo, se muestra una política para un usuario o rol de IAM que reemplaza un nombre de recurso específico con una variable de política. Puede reutilizar esta política si utiliza la clave de condición `aws:PrincipalTag`. Cuando se evalúa esta política, `${aws:PrincipalTag/team}` permite la acción solo si el nombre del bucket termina con un nombre de equipo de la etiqueta de entidad principal `team`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/team}/*"]
    }
  ]
}
```

La variable se marca con un prefijo $ seguido de un par de llaves ({ }). Dentro de los caracteres ${ }, se puede incluir el nombre del valor de la solicitud que quiere utilizar en la política. Los valores que puede utilizar se tratan más adelante en esta página.

Para obtener más información sobre esta clave de condición global, consulte aws:PrincipalTag/tag-key en la lista de claves de condición globales.

> **Note**
>
> Para poder utilizar las variables de políticas debe incluir el `Version` el elemento en una declaración, y la versión debe ser una que admita las variables de la política. Variables se introdujeron en la versión `2012-10-17`. Las versiones anteriores del lenguaje de políticas no son compatibles con las variables de políticas. Si no incluye el elemento `Version` ni lo establece en la fecha correspondiente a esta versión, las variables como `${aws:username}` se tratarán como cadenas literales en la política.
>
> El elemento de política `Version` es diferente de la versión de una política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, por otro lado, se crea al cambiar una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada. Para obtener más información sobre el elemento de política `Version`, consulte the section called “Version”. Para obtener más información sobre las versiones de política, consulte the section called “Control de versiones de políticas de IAM”.

Una política que permite a una entidad principal obtener objetos de la ruta /David de un bucket de S3 tiene el siguiente aspecto:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3::DOC-EXAMPLE-BUCKET/David/*"]
  }]
}
```

Si utiliza una política que se ha adjuntado al usuario `David`, dicho usuario obtiene objetos de su propio bucket de S3, pero usted tendrá que crear una política individual para cada usuario que incluya el nombre del usuario. Después, tendrá que asociar cada política a los usuarios individuales.

Si utiliza una variable de política, puede crear políticas reutilizables. La siguiente política permite al usuario obtener objetos de un bucket de Amazon S3 si el valor de la clave de etiqueta `aws:PrincipalTag` coincide con el valor de la clave de etiqueta `owner` introducido en la solicitud.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowUnlessOwnedBySomeoneElse",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["*"],
    "Condition": {
        "StringEquals": {
          "${s3:ExistingObjectTag/owner}": "${aws:PrincipalTag/owner}"
        }
      }
    }
  ]
}
```

Cuando utiliza una variable de política en lugar de un usuario de este tipo, no tiene que tener una política independiente para cada usuario individual. En el siguiente ejemplo, la política se adjunta a un rol de IAM que asumen los gerentes de producto mediante credenciales de seguridad temporales. Cuando un usuario solicita agregar un objeto de Amazon S3, IAM sustituye el valor de etiqueta `dept` de la solicitud actual por la variable `${aws:PrincipalTag}` y evalúa la política.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${aws:PrincipalTag/dept}/*"],
    }
  ]
}
```

Etiquetas como variables de la política

En algunos servicios de AWS puede asociar sus propios atributos personalizados a los recursos creados por dichos servicios. Por ejemplo, puede aplicar etiquetas a buckets de Amazon S3 o a

usuarios de IAM. Estas etiquetas son pares clave-valor. Puede definir el nombre de la clave de la etiqueta y el valor que es asociado a dicho nombre de clave. Podría, por ejemplo, crear una etiqueta con una clave **`department`** y un valor **`Human Resources`**. Para obtener más información sobre la etiquetación de entidades de IAM, consulte Etiquetado de recursos de IAM. Para obtener información acerca de cómo etiquetar recursos creados por otros servicios de AWS, consulte la documentación de dicho servicio. Para obtener más información acerca de cómo utilizar Tag Editor, consulte Uso de Tag Editor en la Guía del usuario de AWS Management Console.

Puede etiquetar recursos de IAM para simplificar la detección, la organización y el seguimiento de recursos de IAM. También puede etiquetar identidades de IAM para controlar el acceso a recursos o para el etiquetado en sí. Para obtener más información sobre cómo utilizar etiquetas de para controlar el acceso, consulte Control de acceso a usuarios y roles de IAM y para ellos mediante etiquetas.

Dónde puede utilizar variables de política

Puede utilizar variables de política en el elemento `Resource` y en la comparación de cadenas en el elemento `Condition`.

Elemento de recurso

Puede utilizar una variable de política en el elemento `Resource`, pero solo en la parte de recurso del ARN. Esta parte del ARN aparece después de los quintos dos puntos (:). No puede utilizar una variable para reemplazar partes del ARN antes de los quintos dos puntos, como el servicio o la cuenta. Para obtener más información sobre el formato del ARN, consulte ARN de IAM.

Para reemplazar parte de un ARN por un valor de etiqueta, incluya el prefijo y el nombre de clave entre `${ }`. Por ejemplo, el siguiente elemento de recurso se refiere únicamente a un bucket que tiene el mismo nombre que el valor en la solicitud de la etiqueta de departamento del usuario.

```
"Resource": ["arn:aws::s3:::bucket/${aws:PrincipalTag/department}"]
```

Muchos recursos de AWS utilizan ARN que contienen un nombre creado por el usuario. La siguiente política de IAM garantiza que solo los usuarios previstos con valores de etiquetas de acceso, proyecto, aplicación y entorno de acceso coincidentes puedan modificar sus recursos. Además, al utilizar coincidencias con caracteres comodín *, pueden permitir sufijos de nombres de recursos personalizados.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AllowAccessBasedOnArnMatching",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic"],
      "Resource": ["arn:aws:sns:*:*:${aws:PrincipalTag/access-project}-
${aws:PrincipalTag/access-application}-${aws:PrincipalTag/access-environment}-*"
      ]
    }
  ]
}
```

Elemento de condición

Puede utilizar una variable de política para valores de `Condition` en cualquier condición que involucre a los operadores de cadena o a los operadores ARN. Los operadores de cadena incluyen `StringEquals`, `StringLike` y `StringNotLike`. Los operadores ARN incluyen `ArnEquals` y `ArnLike`. No se puede utilizar una variable de política con otros operadores como operadores `Numeric`, `Date`, `Boolean`, `Binary`, `IP Address` o `Null`. Para obtener más información acerca de los operadores de condición, consulte Elementos de la política de JSON de IAM: operadores de condición.

Al hacer referencia a una etiqueta en una expresión del elemento `Condition`, utilice el prefijo y el nombre de la clave pertinentes como clave de condición. A continuación, utilice el valor que desea probar en el valor de condición.

Por ejemplo, el siguiente ejemplo de política permite el acceso completo a recursos, solo si la etiqueta `costCenter` se ha asociado al usuario. La etiqueta también debe tener un valor de `12345` o `67890`. Si la etiqueta no tiene ningún valor o tiene algún otro valor, se producirá un error en la solicitud.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
          "iam:*user*"
       ],
```

```
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:ResourceTag/costCenter": [ "12345", "67890" ]
        }
      }
    }
  ]
}
```

Variables de política sin valor

Cuando las variables de política hacen referencia a una clave de contexto de condición que no tiene ningún valor o no está presente en el contexto de autorización de una solicitud, el valor es efectivamente nulo. No hay un valor igual o similar. Es posible que las claves de contexto de condición no estén presentes en el contexto de autorización cuando:

- Utiliza claves de contexto de condición específicas del servicio en las solicitudes a recursos que no admiten esa clave de condición.
- Las etiquetas de las entidades principales, las sesiones, los recursos o las solicitudes de IAM no están presentes.
- Se presentan otras circunstancias, tal como se enumeran para cada contexto de condición global en Claves de contexto de condición globales de AWS.

Cuando utiliza una variable sin valor en el elemento de condición de una política de IAM, Elementos de la política de JSON de IAM: operadores de condición como `StringEquals` o `StringLike` no coinciden y la instrucción de política no entra en vigor.

Los operadores de condición invertida como `StringNotEquals` o `StringNotLike` coinciden con un valor nulo, ya que el valor de la clave de condición con la que están realizando la prueba no es igual o similar al valor realmente nulo.

En el siguiente ejemplo, `aws:principaltag/Team` debe ser igual a `s3:ExistingObjectTag/Team` para permitir el acceso. El acceso se deniega explícitamente cuando `aws:principaltag/Team` no está configurado. Si se utiliza una variable que no tiene ningún valor en el contexto de autorización como parte del elemento `Resource` o `NotResource` de una política, el recurso que incluye una variable de política sin valor no coincidirá con ningún recurso.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::/example-bucket/*",
    "Condition": {
      "StringNotEquals": {
        "s3:ExistingObjectTag/Team": "${aws:PrincipalTag/Team}"
       }
      }
    }
  ]
}
```

Información de la solicitud que puede utilizar para variables de políticas

Puede utilizar el elemento `Condition` de una política JSON para comparar las claves de la solicitud de contexto con los valores de claves que especifique en su política. Cuando se utiliza una variable de política, AWS sustituye un valor de la clave de contexto de solicitud en lugar de la variable de su política.

Valores clave principales

Los valores de `aws:username`, `aws:userid` y `aws:PrincipalType` dependen del tipo de principal que inició la solicitud. Por ejemplo, la solicitud podría realizarse con las credenciales de un usuario de IAM, un rol de IAM o el Usuario raíz de la cuenta de AWS. La lista siguiente muestra los valores de estas claves para diferentes tipos de entidades principales.

- Usuario raíz de la cuenta de AWS
  - `aws:username`: (ausente)
  - `aws:userid`: ID de Cuenta de AWS
  - `aws:PrincipalType`: `Account`
- Usuario de IAM
  - `aws:username`: *`nombre-usuario-IAM`*
  - `aws:userid`: ID único
  - `aws:PrincipalType`: `User`
- Usuario federado
  - `aws:username`: (ausente)

  - `aws:userid`: *`cuenta`*:*`nombre-especificado-por-intermediario`*
  - `aws:PrincipalType`: `FederatedUser`
- Usuario federado web y usuario federado SAML

> **Note**
>
> Para obtener más información acerca de las claves de política que están disponibles cuando usa la federación OIDC, consulte Identifique a los usuarios con la federación OIDC.

  - `aws:username`: (ausente)
  - `aws:userid`: (ausente)
  - `aws:PrincipalType`: `AssumedRole`
- Rol asumido
  - `aws:username`: (ausente)
  - `aws:userid`: *`Id-rol`*:*`nombre-de-rol-especificado-por-intermediario`*
  - `aws:PrincipalType`: `Assumed role`
- Rol asignado a una instancia de Amazon EC2
  - `aws:username`: (ausente)
  - `aws:userid`: *`id-rol`*:*`id-instancia-ec2`*
  - `aws:PrincipalType`: `Assumed role`
- Persona que llama de manera anónima (Amazon SQS, Amazon SNS y Amazon S3)
  - `aws:username`: (ausente)
  - `aws:userid`: (ausente)
  - `aws:PrincipalType`: `Anonymous`

Para los elementos de esta tabla lista , tenga en cuenta lo siguiente:

- ausente significa que el valor actual no está en la información de solicitud actual y que cualquier intento de asociarlo dará un error y hará que la declaración no sea válida.
- *`id-rol`* es un identificador exclusivo asignado a cada rol en el momento de su creación. Puede visualizar el ID de rol con el comando de AWS CLI: `aws iam get-role --role-name` *`rolename`*

- *nombre-especificado-por-intermediario* y *nombre-de-rol-especificado-por-intermediario* son nombres (como una aplicación o un servicio) que el proceso de llamada transmite cuando realiza una llamada para obtener credenciales temporales.
- *id-instancia-ec2* es un valor asignado a la instancia cuando esta se lanza y aparece en la página Instancias de la consola de Amazon EC2. También puede mostrar el ID de instancia ejecutando el comando de AWS CLI: `aws ec2 describe-instances`

Información disponible en las solicitudes de usuarios federados

Los usuarios federados son usuarios que se autentican mediante un sistema distinto de IAM. Por ejemplo, una empresa puede tener una aplicación para su uso interno que realiza llamadas a AWS. Puede ser poco práctico dar una identidad de IAM a todos los usuarios de la empresa que usen la aplicación. En lugar de ello, la empresa puede utilizar una aplicación proxy (nivel intermedio) que tenga una única identidad de IAM o la empresa puede utilizar un proveedor de identidades (IdP) SAML. La aplicación proxy o el proveedor de identidades SAML autentican a usuarios individuales que utilizan la red corporativa. Una aplicación proxy puede, pues, utilizar su identidad de IAM para obtener credenciales de seguridad temporales para usuarios individuales. Un proveedor de identidades SAML puede efectivamente intercambiar información de identidad para credenciales de seguridad temporales de AWS. En ese caso, las credenciales temporales pueden utilizarse para obtener acceso a los recursos de AWS.

Del mismo modo, puede crear una aplicación para un dispositivo móvil en el que la aplicación necesite obtener acceso a recursos de AWS. En ese caso, puede utilizar las federaciones OIDC, donde la aplicación autentifica al usuario mediante un proveedor de identidades popular como Inicio de sesión con Amazon, Amazon Cognito, Facebook o Google. Entonces, la aplicación puede utilizar la información de autenticación del usuario de estos proveedores para obtener credenciales de seguridad temporales para obtener acceso a recursos de AWS.

La manera recomendada de utilizar las federaciones OIDC consiste en aprovechar Amazon Cognito y los SDK para móviles de AWS. Para obtener más información, consulte lo siguiente:

- Guía del usuario de Amazon Cognito
- Escenarios habituales en las credenciales temporales

Caracteres especiales

Hay algunas variables de política predefinidas especiales que tienen valores fijos que le permiten representar caracteres que, de otro modo, tendrían un significado especial. Si estos caracteres

especiales forman parte de la cadena que intenta hacer corresponder y se insertan literalmente, es posible que se interpreten erróneamente. Por ejemplo, insertar un * asterisco en la cadena se interpretaría como un comodín que coincide con cualquier carácter, salvo un * literal. En estos casos puede utilizar las variables de política predefinidas siguientes:

- ${*}: debe utilizarse cuando se necesite un carácter * (asterisco).
- ${?}: debe utilizarse cuando se necesite carácter ? (signo de interrogación).
- ${$}: debe utilizarse cuando necesite un carácter $ (signo de dólar).

Estas variables de política predefinida se pueden utilizar en todas las cadenas donde pueda utilizar variables de política estándar.

Especificación de valores predeterminados

Para agregar un valor predeterminado a una variable, rodee el valor predeterminado entre comillas simples (`' '`), y separe el texto de la variable y el valor predeterminado con una coma y un espacio (`, `).

Por ejemplo, si una entidad principal está etiquetada con `team=yellow`, puede acceder al bucket de Amazon S3 de `ExampleCorp's` llamado *`DOC-EXAMPLE-BUCKET`*`-yellow`. Una política con este recurso permite a los miembros del equipo acceder al bucket su equipo, pero no al de otros equipos. Para los usuarios sin etiquetas de equipo, establece un valor predeterminado de `company-wide` para el nombre del bucket. Estos usuarios solo pueden acceder al bucket de *`DOC-EXAMPLE-BUCKET`*`-company-wide` en el que pueden ver información amplia, como instrucciones para unirse a un equipo.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Para obtener más información

Para obtener más información acerca de las políticas, consulte los siguientes temas:

- Políticas y permisos en IAM
- Ejemplos de políticas basadas en identidad de IAM
- Referencia de los elementos de las políticas de JSON de IAM
- Lógica de evaluación de políticas
- Federación OIDC

Elementos de la política de JSON de IAM: tipos de datos compatibles

En esta sección se presentan los tipos de datos que son compatibles al especificar valores en las políticas de JSON. El lenguaje de política no admite todos los tipos de elementos de política. Para obtener información sobre cada elemento, consulte las secciones anteriores.

- Strings
- Números (Ints y Floats)
- Booleano
- Null
- Lists
- Maps
- Structs (que son simplemente Maps anidados)

En la siguiente tabla se relaciona cada tipo de datos a su serialización. Tenga en cuenta que todas las políticas deben estar en UTF-8. Para obtener información sobre los tipos de datos JSON, diríjase a RFC 4627.

Tipo	JSON
Cadena	Cadena
Entero	Número
Float	Número
Booleano	true false
Null	null
Fecha	String que cumple con W3C Profile of ISO 8601
IpAddress	String que cumple con RFC 4632
Enumeración	Array (Matriz)
Objeto	Objeto

Lógica de evaluación de políticas

Cuando una entidad principal intenta utilizar la AWS Management Console, la API de AWS o la AWS CLI, la entidad principal envía una solicitud a AWS. Cuando un servicio de AWS recibe la solicitud, AWS lleva a cabo varios pasos para determinar si debe permitir o denegar la solicitud.

1. Autenticación – AWS primero autentica la entidad principal que realiza la solicitud, si fuera necesario. Este paso no es necesario para algunos servicios, como Amazon S3, que permiten algunas solicitudes de usuarios anónimos.
2. Procesamiento del contexto de la solicitud – AWS procesa la información recopilada en la solicitud para determinar las políticas que se aplican a esta.
3. Evaluación de políticas dentro de una misma cuenta – AWS evalúa todos los tipos de políticas, lo que afecta al orden en el que se evalúan las políticas.
4. Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta – AWS A continuación, procesa las políticas teniendo en cuenta el contexto de la solicitud para determinar si esta se permite o se deniega.

Procesamiento del contexto de la solicitud

AWS procesa la solicitud para recopilar la información siguiente en un contexto de solicitud:

- Acciones (u operaciones): las acciones u operaciones que la entidad principal desea realizar.
- Recursos: el objeto de recurso de AWS sobre el que se realizan las acciones u operaciones.
- Entidad principal: el usuario, el rol, el usuario federado o la aplicación que envió la solicitud. La información sobre la entidad principal incluye las políticas asociada a dicha entidad principal.
- Datos de entorno: información sobre la dirección IP, el agente de usuario, el estado de habilitación de SSL o la hora del día.
- Datos de recursos: datos relacionados con el recurso que se está solicitando. Esto puede incluir información como, por ejemplo, un nombre de tabla de DynamoDB o una etiqueta de una instancia Amazon EC2.

A continuación, AWS utiliza esta información para buscar políticas que se apliquen al contexto de la solicitud.

Evaluación de políticas dentro de una misma cuenta

El modo en que AWS evalúa las políticas depende de los tipos de las políticas aplicables al contexto de la solicitud. Los tipos de políticas siguientes, que se muestran por orden de frecuencia, están disponibles para su uso dentro de una misma Cuenta de AWS. Para obtener más información acerca de estos tipos de políticas, consulte Políticas y permisos en IAM. Para obtener información sobre cómo AWS evalúa las políticas para el acceso entre cuentas, consulte Lógica de evaluación de políticas entre cuentas.

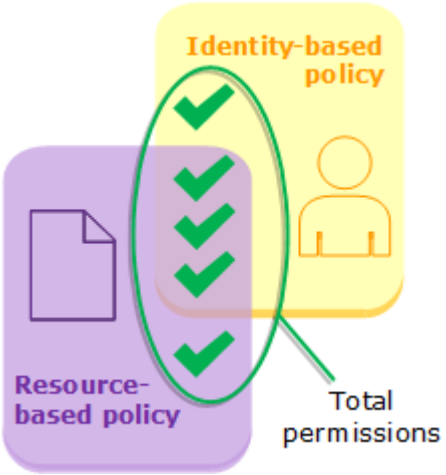
1. Políticas basadas en identidad - Las políticas basadas en identidad se asocian a una identidad de IAM (usuario, grupo de usuarios o rol) y conceden permisos a entidades de IAM (usuarios y roles). Cuando a una solicitud solo le son aplicables políticas basadas en identidad, AWS comprueba toda ellas para obtener al menos un permiso `Allow`.
2. Políticas basadas en recursos: las políticas basadas en recursos conceden permisos a la entidad principal (cuenta, usuario, rol y entidades principales como sesiones de rol y usuarios federados de IAM) especificada como entidad principal. Los permisos definen lo que la entidad principal puede hacer con el recurso al que está asociada la política. Cuando a una solicitud le son aplicables políticas basadas en recursos y también políticas basadas en identidad, AWS comprueba toda ellas para obtener al menos un permiso `Allow`. Cuando se evalúan las políticas basadas en recursos, el ARN de entidad principal especificado en la política determina si las denegaciones implícitas en otros tipos de políticas son aplicables a la decisión final.
3. Límites de permisos de IAM: los límites de permisos son una característica avanzada que le permite establecer los permisos máximos que una política basada en identidades puede conceder a una entidad de IAM (usuario o rol). Al establecer un límite de permisos para una entidad, esta solo puede realizar las acciones que le permitan tanto sus políticas basadas en identidad como sus límites de permisos. En algunos casos, una denegación implícita en un límite de permisos puede limitar los permisos concedidos por una política basada en recursos. Para obtener más información, consulte Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta más adelante en este tema.
4. AWS OrganizationsPolíticas de control de servicios (SCP): las SCP de Organizations especifican los permisos máximos para una organización o unidad organizativa (OU). El máximo de una SCP se aplica a las entidades principales de las cuentas miembro, incluido cada Usuario raíz de la cuenta de AWS. Si existe una SCP, las políticas basadas en identidad y en recursos solo concederán permisos a las entidades principales de las cuentas miembro si la SCP también permite la acción. Si existen a la vez un límite de permisos y una SCP, el límite, la SCP y la política basada en identidad deben permitir conjuntamente la acción.

5. Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetros cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Para crear una sesión de rol mediante programación, utilice una de las operaciones de API `AssumeRole*`. Al hacerlo y pasar las políticas de sesión, los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades de la entidad IAM y las políticas de la sesión. Para crear una sesión de un usuario federado, se usan las claves de acceso de un usuario de IAM para llamar mediante programación a la operación de API `GetFederationToken`. Una política basada en recursos tiene un efecto diferente en la evaluación de los permisos de política de sesión. La diferencia depende de si el usuario o ARN de la función o ARN de la sesión se muestran como el principal de la política basada en recursos. Para obtener más información, consulte Políticas de sesión.

Recuerde que una denegación explícita en cualquiera de estas políticas anulará el permiso.

Evaluación de políticas basadas en identidad con políticas basadas en recursos

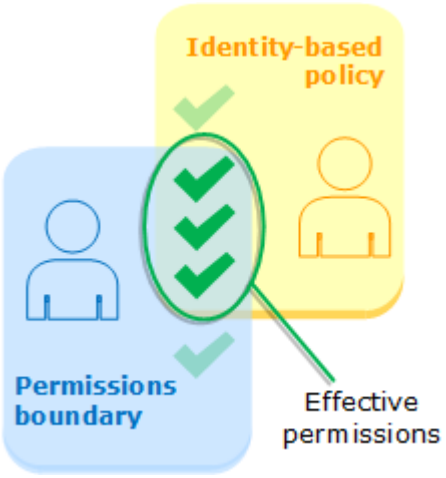
Las políticas basadas en identidad y las políticas basadas en recursos conceden permisos referidos a las identidades o recursos a los que están asociadas. Cuando una entidad de IAM (usuario o rol) solicita acceso a un recurso de la misma cuenta, AWS evalúa todos los permisos concedidos por las políticas basadas en identidad y las basadas en recursos. Los permisos resultantes son el total de aplicar los dos tipos. Si una política basada en identidad, una política basada en recursos, o ambas, permiten una acción, entonces AWS permite la acción. Una denegación explícita en una de estas políticas anulará el permiso.



Evaluación de políticas basadas en identidad con límites de permisos

Cuando AWS evalúa las políticas basadas en identidad y el límite de permisos para un usuario, los permisos resultantes son la intersección de las dos categorías. Esto significa que, cuando se

añade un límite de permisos a un usuario que ya tiene políticas de permisos basadas en identidad, es posible que se reduzca el número de acciones que puede realizar. Del mismo modo, al eliminar un límite de permisos de un usuario, es posible que aumente el número de acciones que este puede realizar. Una denegación explícita en una de estas políticas anulará el permiso. Para ver información acerca del modo en que se evalúan otros tipos de políticas con los límites de permisos, consulte Evaluación de los permisos efectivos cuando se usan límites.



Evaluación de políticas basadas en identidad con SCP de Organizations

Cuando un usuario pertenece a una cuenta que es miembro de una organización, los permisos resultantes son la intersección de las políticas del usuario y la SCP. Esto significa que tanto la política basada en identidad como la SCP deben permitir la acción. Una denegación explícita en una de estas políticas anulará el permiso.

Puede saber si su cuenta es miembro de una organización en AWS Organizations. Los miembros de la organización podrían verse afectados por una SCP. Para ver estos datos a través del comando AWS CLI u operación de la API de AWS, debe tener permisos para la acción

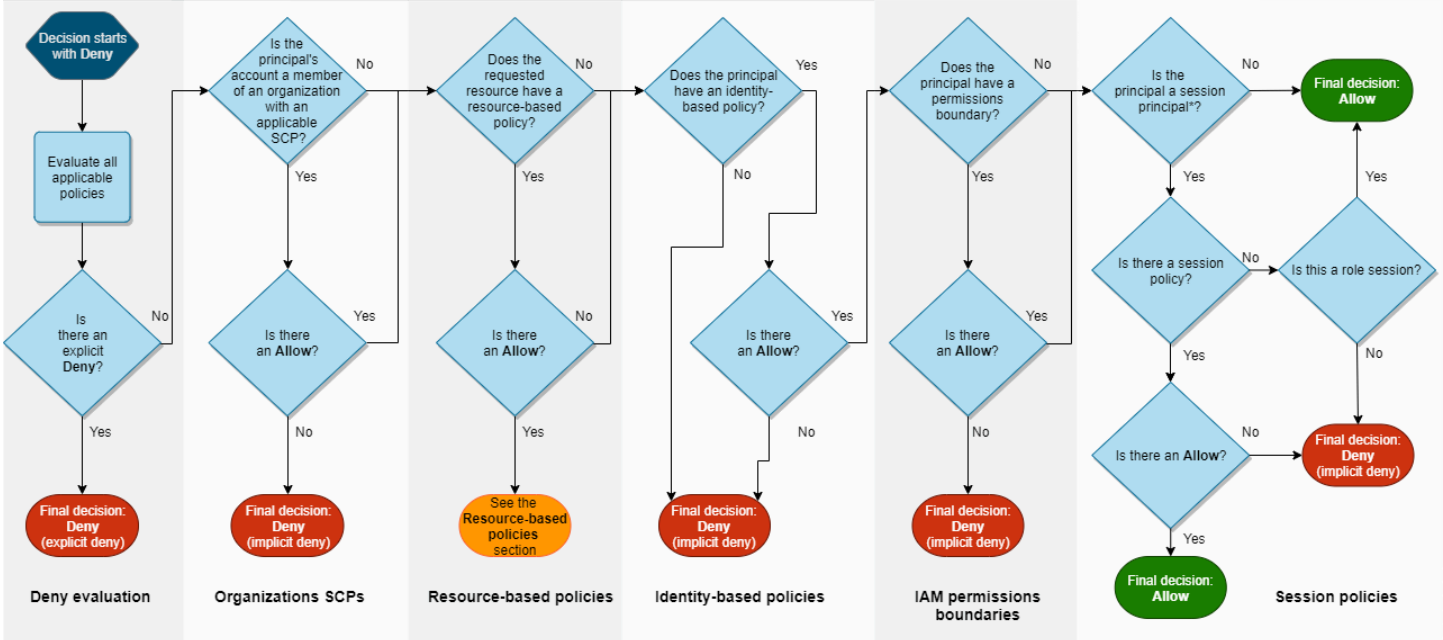
`organizations:DescribeOrganization` para su entidad de Organizations. Debe tener permisos adicionales para realizar la operación en la consola Organizations. Para saber si una SCP deniega el acceso a una solicitud específica o para cambiar los permisos efectivos, póngase en contacto con su administrador de AWS Organizations.

Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta

Supongamos que un principal envía una solicitud a AWS para acceder a un recurso en la misma cuenta que la entidad del principal. El código de aplicación de AWS decide si la solicitud debe autorizarse o denegarse. AWS evalúa todas las políticas que se aplican al contexto de la solicitud. A continuación, se proporciona un resumen general de la lógica de evaluación de AWS para las políticas dentro de una misma cuenta.

- De forma predeterminada, todas las solicitudes se deniegan de manera implícita a excepción de Usuario raíz de la cuenta de AWS, que tiene acceso completo.
- Un permiso explícito en una política basada en identidad o en recursos anula esta opción predeterminada.
- Si existe un límite de permisos, una SCP de Organizations o una política de sesión, es posible que anule el permiso con una denegación implícita.
- Una denegación explícita en cualquier política invalida cualquier permiso concedido.

El siguiente diagrama ofrece información detallada acerca de cómo se toma una decisión. Este diagrama de flujo no cubre el impacto de las políticas basadas en recursos y las denegaciones implícitas en otros tipos de políticas.



*A session principal is either a role session or an IAM federated user session.

1. Denegar la evaluación - De forma predeterminada, se deniegan todas las solicitudes. Esto se denomina denegación implícita. El código de aplicación del AWS evalúa todas las políticas de la cuenta aplicables a la solicitud. Esto incluye las SCP de AWS Organizations, las políticas basadas en recursos, las políticas basadas en identidad, los límites de permisos de IAM y las políticas de sesión. En todas estas políticas, el código de aplicación buscará una instrucción `Deny` que se aplique a la solicitud. Esto se denomina una denegación explícita. Si el código encuentra una sola denegación explícita aplicable, devuelve como decisión final Deny (Denegar). Si no hay ninguna denegación explícita, la evaluación del código de aplicación continúa.
2. Organizations SCP (Políticas de control de servicios de Organizations): A continuación, el código evalúa las SCP de AWS Organizations que se aplican a la solicitud. Las SCP se aplican a las entidades principales de la cuenta a la que se asocian las SCP. Si el código de aplicación no encuentra ninguna instrucción `Allow` aplicable en las SCP, la solicitud se deniega implícitamente. El código devuelve como decisión final Deny (Denegar). Si no hay ninguna SCP, o si la SCP permite la acción solicitada, la evaluación del código de aplicación continúa.
3. Políticas basadas en recursos: dentro de la misma cuenta, las políticas basadas en recursos tienen un impacto diferente en la evaluación de políticas según el tipo de entidad principal que acceda al recurso y que se permite en la política basada en recursos. Según el tipo de entidad principal, un `Allow` en una política basada en recursos puede dar lugar a una decisión final de `Allow`, incluso si existe una denegación implícita en una política basada en identidad, un límite de permisos o una política de sesión.

Para la mayoría de los recursos, solo necesita un permiso explícito para la entidad principal en una política basada en identidades o en una política basada en recursos para conceder acceso. Las políticas de confianza del rol de IAM y las políticas de claves de KMS son excepciones a esta lógica, porque deben permitir explícitamente el acceso a entidades principales.

La lógica de políticas basada en recursos difiere de otros tipos de políticas si la entidad principal especificada es un usuario de IAM, un rol de IAM o una entidad principal de sesión. Las entidades principales de sesión incluyen sesiones de rol de IAM o una sesión de usuario federado de IAM. Si una política basada en recursos concede permiso directamente al usuario de IAM o a la entidad principal de sesión que realiza la solicitud, una denegación implícita en una política basada en identidad, un límite de permisos o una política de sesión no afecta a la decisión final.

La tabla siguiente lo ayuda a comprender el impacto de las políticas basadas en recursos para los distintos tipos de entidades principales cuando hay denegaciones implícitas en las políticas basadas en identidad, los límites de permisos y las políticas de sesión basadas en identidad.

Políticas basadas en recursos y denegaciones implícitas en otros tipos de políticas (de la misma cuenta)

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
Rol de IAM	No aplicable	No aplicable	No aplicable	No aplicable	No aplicable	Un rol en sí no puede realizar una solicitud. Las solicitudes se realizan con la sesión de rol después de asumir un rol.

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
Sesión de rol de IAM	Permite el ARN de rol	Denegación implícita	Denegación implícita	Denegación implícita	DENY	El límite de permisos y la política de sesión se evalúan como parte de la decisión final. Una denegación implícita en cualquiera de las políticas da como resultado una decisión DENY.

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
Sesión de rol de IAM	Permite ARN de sesión de rol	Denegación implícita	Denegación implícita	Denegación implícita	PERMITIR	Los permisos se otorgan directamente a la sesión. Otros tipos de políticas no afectan a la decisión.
Usuario de IAM	Permite ARN de usuario de IAM	Denegación implícita	Denegación implícita	No aplicable	PERMITIR	Los permisos se conceden directamente al usuario. Otros tipos de políticas no afectan a la decisión.

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
Usuario federado de IAM (`GetFederationToken`)	Permite ARN de usuario de IAM	Denegación implícita	Denegación implícita	Denegación implícita	DENY	Una denegación implícita en el límite de permisos o en la política de sesión da como resultado un DENY.
Usuario federado de IAM (`GetFederationToken`)	Permite ARN de sesión de usuario federado de IAM	Denegación implícita	Denegación implícita	Denegación implícita	PERMITIR	Los permisos se otorgan directamente a la sesión. Otros tipos de políticas no afectan a la decisión.

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
Usuario raíz	Permite ARN de usuario raíz	No aplicable	No aplicable	No aplicable	PERMITIR	El usuario raíz ofrece acceso completo e ilimitado a todos los recursos de la Cuenta de AWS. Para obtener información sobre cómo controlar el acceso de los usuarios raíz a las cuentas de AWS Organizations, consulte Service control policies (SCPs)(Políticas de control de servicios

Entidad principal efectuando la solicitud	Política basada en recursos	Políticas basadas en identidad	Límite de permisos	Política de sesión	Resultado	Motivo
						[SCP]) en la Organizations User Guide (Guía del usuario de Organizations).
Principal del servicio de AWS	Permite una entidad principal del servicio de AWS	No aplicable	No aplicable	No aplicable	PERMITIR	Cuando una política basada en recursos concede permisos directamente a una entidad principal del servicio de AWS, otros tipos de políticas no afectan a la decisión.

- Rol de IAM: las políticas basadas en recursos que otorgan permisos a un ARN de rol de IAM están limitadas por una denegación implícita en un límite de permisos o una política de sesión.

  ARN de rol de ejemplo

  ```
  arn:aws:iam::111122223333:role/examplerole
  ```

- Sesión de rol de IAM: dentro de la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de sesión de rol de IAM otorgan permisos directamente a la sesión de rol asumida. Los permisos otorgados directamente a una sesión no están limitados por una denegación implícita en una política basada en la identidad, un límite de permisos ni una política de sesión. Cuando asume un rol y realiza una solicitud, la entidad principal que realiza la solicitud es el ARN de sesión de rol de IAM y no el ARN del rol en sí.

  Ejemplo ARN de sesión de rol

  ```
  arn:aws:sts::111122223333:assumed-role/examplerole/examplerolesessionname
  ```

- Usuario de IAM: dentro de la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de usuario de IAM (que no es una sesión de usuario federado) no están limitadas por una denegación implícita en una política basada en identidad o en un límite de permisos.

  Ejemplo de ARN de usuario de IAM

  ```
  arn:aws:iam::111122223333:user/exampleuser
  ```

- Sesiones de usuarios federados de IAM: una sesión de usuarios federados de IAM es una sesión creada mediante la llamada a `GetFederationToken`. Cuando un usuario federado realiza una solicitud, la entidad principal que realiza la solicitud es el ARN de usuario federado y no el ARN del usuario de IAM que se federó. En la misma cuenta, las políticas basadas en recursos que otorgan permisos a un ARN de usuario federado otorgan permisos directamente a la sesión. Los permisos otorgados directamente a una sesión no están limitados por una denegación implícita en una política basada en la identidad, un límite de permisos ni una política de sesión.

Sin embargo, si una política basada en recursos concede permiso al ARN del usuario de IAM que se federó, las solicitudes realizadas por el usuario federado durante la sesión están limitadas por una denegación implícita en un límite de permisos o una política de sesión.

Ejemplo de ARN de sesión de usuario federado de IAM

```
arn:aws:sts::111122223333:federated-user/exampleuser
```

4. Políticas basadas en identidad — A continuación, el código comprueba las políticas basadas en la identidad de la entidad principal. En el caso de un usuario de IAM, se trata de las políticas del usuario y las políticas de los grupos a los que el usuario pertenece. Si no hay políticas ni instrucciones basadas en la identidad que permitan la acción solicitada, la solicitud se deniega implícitamente y el código devuelve una decisión final de Deny. Si cualquier instrucción de cualquier política basada en identidad permite la acción solicitada, entonces el código continúa.
5. Límites de permisos de IAM: el código comprueba si la entidad de IAM utilizada por la entidad principal tiene un límite de permisos. Si la política empleada para establecer el límite de permisos no permite la acción solicitada, la solicitud se deniega implícitamente. El código devuelve como decisión final Deny (Denegar). Si no hay ningún límite de permisos, o si el límite de permisos permite la acción solicitada, el código continúa.
6. Políticas de sesión: a continuación, el código comprueba si la entidad principal es una entidad principal de sesión. Las entidades principales de sesión incluyen una sesión de rol de IAM o una sesión de usuario federado de IAM. Si la entidad principal no es una entidad principal de sesión, el código de ejecución devuelve una decisión final de Allow.

   Para las entidades principales de sesión, el código comprueba si se ha pasado una entidad principal de sesión en la solicitud. Puede pasar una política de sesión con la AWS CLI o la API de AWS a fin de obtener credenciales temporales para un rol o un usuario federado de IAM.

   - Si existe una política de sesión y no permite la acción solicitada, la solicitud se deniega implícitamente. El código devuelve como decisión final Deny (Denegar).
   - Si no hay ninguna política de sesión, el código comprueba si la entidad principal es una sesión de rol. Si la entidad principal es una sesión de rol, la solicitud se permite. De lo contrario, la solicitud se deniega implícitamente y el código devuelve una decisión final de Deny.
   - Si una política de sesión está presente y permite la acción solicitada, entonces el código de aplicación devuelve una decisión final de Allow.
7. Errores – Si el código de aplicación AWS encuentra un error en cualquier momento durante la evaluación, generará una excepción y se cerrará.

Ejemplo de evaluación de políticas basadas en identidad y políticas basadas en recursos

Los tipos de políticas más habituales son las políticas basadas en identidad y las políticas basadas en recursos. Cuando se solicita acceso a un recurso, AWS evalúa todos los permisos otorgados por las políticas para que haya al menos un permiso dentro de la misma cuenta. Una denegación explícita en cualquiera de las políticas anulará el permiso.

> **Important**
>
> Si la política basada en identidad o la política basada en recursos de la misma cuenta permite la solicitud y la otra no, la solicitud aún está permitida.

Supongamos que Carlos tiene el nombre de usuario `carlossalazar` y que intenta guardar un archivo en el bucket de Amazon S3 `carlossalazar-logs`.

Supongamos también que la política siguiente está asociada al usuario de IAM `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetAccountPublicAccessBlock",
                "s3:ListAccessPoints",
                "s3:ListAllMyBuckets"
            ],
            "Resource": "arn:aws:s3:::*"
        },
        {
            "Sid": "AllowS3Self",
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
```

```
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

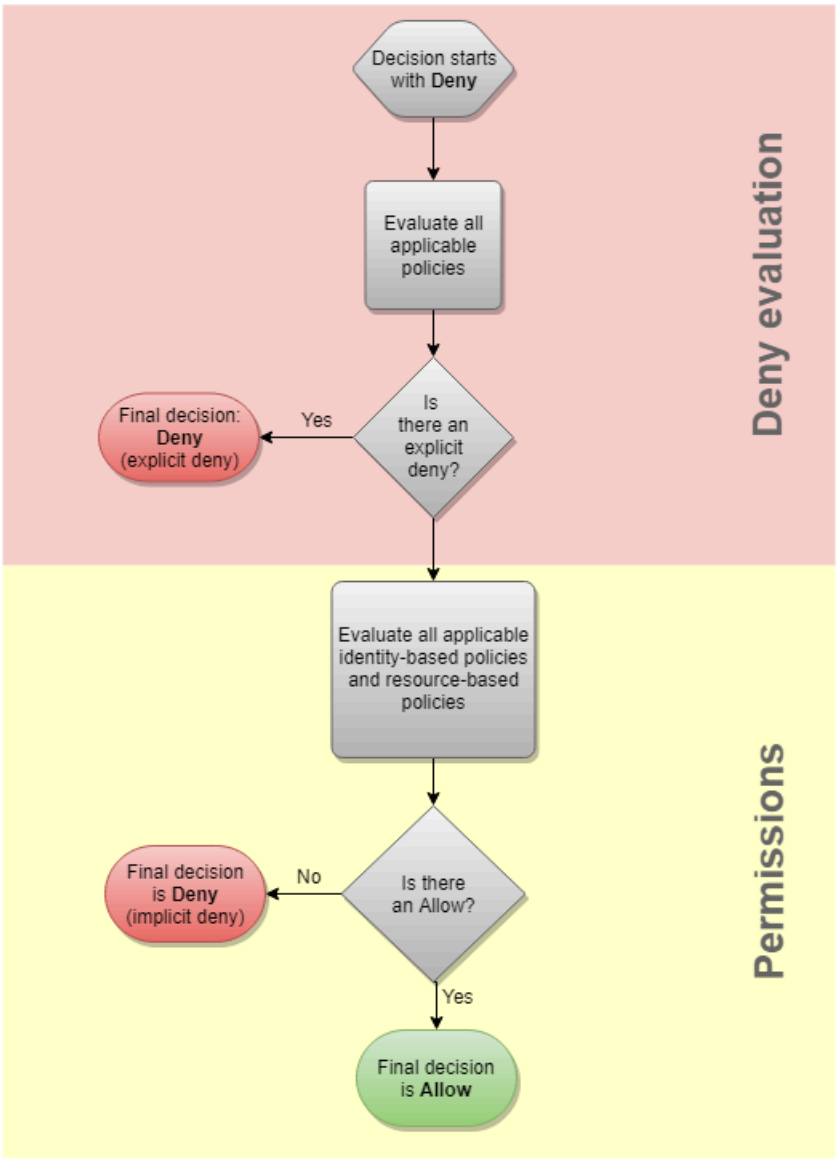
La instrucción `AllowS3ListRead` de esta política permite a Carlos ver una lista de todos los buckets de la cuenta. La instrucción `AllowS3Self` concede a Carlos acceso completo al bucket que tiene el mismo nombre que su nombre de usuario. La instrucción `DenyS3Logs` deniega a Carlos el acceso a los buckets de S3 que contengan `log` en el nombre.

Además, la siguiente política basada en recursos (denominada política de bucket) está asociada al bucket `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        }
    ]
}
```

Esta política especifica que únicamente el usuario `carlossalazar` puede obtener acceso al bucket `carlossalazar`.

Cuando Carlos solicita guardar un archivo en el bucket `carlossalazar-logs`, AWS determina qué políticas se aplican a la solicitud. En este caso, solo se aplican la política basada en identidad y la política basada en recursos. Ambas son políticas de permisos. Debido a que no se aplica ningún límite de permisos, la lógica de evaluación se reduce a lo siguiente.



AWS comprueba en primer lugar si existe una instrucción `Deny` que se aplique al contexto de la solicitud. Encuentra una, ya que la política basada en identidad deniega explícitamente a Carlos el acceso a los buckets de S3 que se usan para el registro. A Carlos se le deniega el acceso.

Supongamos que luego se da cuenta de su error e intenta guardar el archivo en el bucket `carlossalazar`. AWS comprueba si existe una instrucción `Deny` y no encuentra ninguna. A continuación, comprueba las políticas de permisos. Tanto la política basada en la identidad como la política basada en los recursos permiten la solicitud. Por lo tanto, AWS permite la solicitud. Si alguna de ellas denegase explícitamente la instrucción, la solicitud habría sido denegada. Si uno de los tipos de política permite la solicitud y el otro no, la solicitud sigue estando permitida.

Diferencia entre denegaciones implícitas y explícitas

Una solicitud da como resultado una denegación explícita si una política aplicable incluye una instrucción `Deny`. Si las políticas que se aplican a una solicitud incluyen una instrucción `Allow` y una instrucción `Deny`, la instrucción `Deny` prevalece sobre la instrucción `Allow`. La solicitud se deniega explícitamente.

Una denegación implícita se produce cuando no hay instrucciones `Deny` ni `Allow` aplicables. Dado que a los usuarios, roles o usuarios federados de IAM se les deniega el acceso de forma predeterminada, se les debe permitir explícitamente realizar una acción. De lo contrario, se les deniega implícitamente el acceso.

Al diseñar su estrategia de autorización, debe crear políticas con instrucciones `Allow` que permitan a las entidades principales realizar solicitudes sin problemas. Sin embargo, puede elegir cualquier combinación de denegaciones implícitas y explícitas.

Por ejemplo, puede crear la siguiente política que incluye acciones permitidas, acciones denegadas implícitamente y acciones denegadas explícitamente. La declaración `AllowGetList` permite acceso de solo lectura a acciones de IAM que empiezan por los prefijos `Get` y `List`. Todas las demás acciones de IAM, como `iam:CreatePolicy` se deniegan implícitamente. La declaración `DenyReports` deniega explícitamente el acceso a los informes de IAM al denegar el acceso a acciones que incluyen el sufijo `Report`, como `iam:GetOrganizationsAccessReport`. Si alguien agrega otra política a esta entidad principal para otorgarle acceso a los informes de IAM, como `iam:GenerateCredentialReport`, las solicitudes relacionadas con informes se siguen denegando debido a esta denegación explícita.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowGetList",
            "Effect": "Allow",
            "Action": [
                "iam:Get*",
                "iam:List*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenyReports",
```

```
            "Effect": "Deny",
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Lógica de evaluación de políticas entre cuentas

Puede permitir que un principal de una cuenta acceda a los recursos de una segunda cuenta. Esto se denomina acceso entre cuentas. Cuando permite el acceso entre cuentas, la cuenta donde se encuentra el principal se denomina cuenta de confianza. La cuenta donde se encuentra el recurso es la cuenta de confianza.

Para permitir el acceso entre cuentas, debe asociar una política basada en recursos al recurso que desea compartir. También debe asociar una política basada en identidad a la identidad que actúa como entidad principal en la solicitud. La política basada en recursos de la cuenta de confianza debe especificar la entidad principal de la cuenta de confianza que tendrá acceso al recurso. Puede especificar toda la cuenta o los usuarios de IAM, los usuarios federados, los roles de IAM o las sesiones de rol asumido. También puede especificar un servicio de AWS como principal. Para obtener más información, consulte Especificación de una entidad principal.

La política basada en la identidad del principal debe permitir el acceso solicitado al recurso en el servicio de confianza. Para ello puede especificar el ARN del recurso o permitir el acceso a todos los recursos (*).

En IAM, puede asociar una política basada en recursos a un rol de IAM para permitir que los principales de otras cuentas asuman ese rol. La política basada en recursos del rol se denomina política de confianza de rol. Después de asumir ese rol, los principales permitidos pueden utilizar las credenciales temporales resultantes para acceder a varios recursos de la cuenta. Este acceso se define en la política de permisos basada en la identidad del rol. Para saber la diferencia entre permitir el acceso entre cuentas mediante roles y permitir el acceso entre cuentas mediante otras políticas basadas en recursos, consulte Acceso a recursos entre cuentas en IAM.

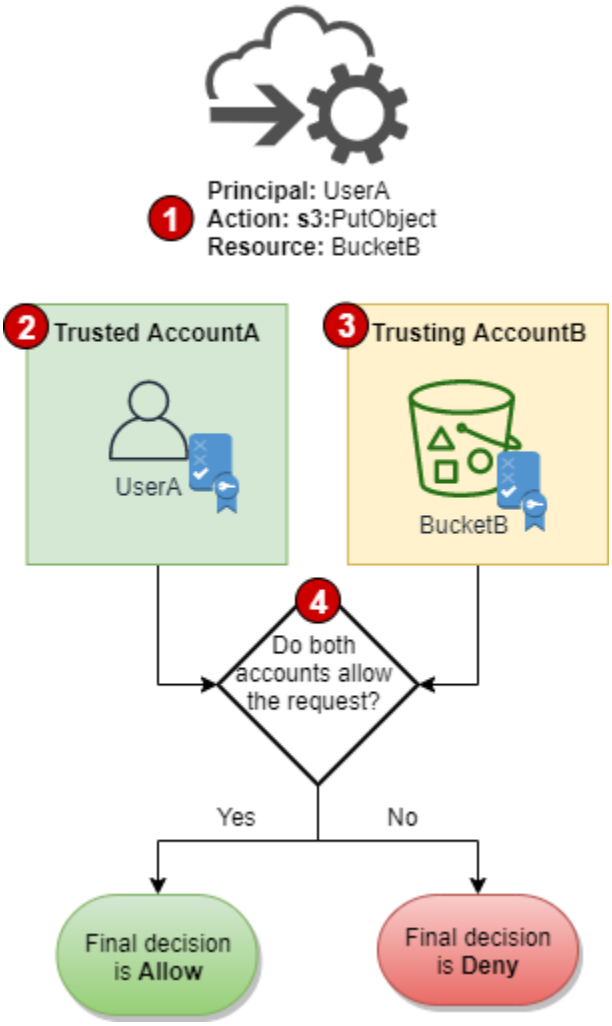
> **Important**
>
> Otros servicios pueden afectar a la lógica de evaluación de políticas. Por ejemplo, AWS Organizations admite políticas de control de servicios que se pueden aplicar a una o varias entidades principales. AWS Resource Access Manager admite fragmentos de políticas que

controlan las acciones que las entidades principales pueden realizar en los recursos que se comparten con ellas.

Determinación de si se permite una solicitud entre cuentas

Para las solicitudes entre cuentas, el solicitante en la `AccountA` de confianza debe tener una política basada en la identidad. Dicha política debe permitirles hacer una solicitud al recurso en la `AccountB` de confianza. Además, la política basada en recursos en `AccountB` debe permitir al solicitante en `AccountA` obtener acceso al recurso.

Cuando realiza una solicitud entre cuentas, AWS realiza dos evaluaciones. AWS evalúa la solicitud en la cuenta de confianza y en la cuenta en que se confía. Para obtener más información acerca de cómo se evalúa una solicitud dentro de una sola cuenta, consulte Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta. La solicitud solo se permite si ambas evaluaciones devuelven una decisión de tipo `Allow`.



1. Una solicitud entre cuentas es el proceso mediante el cual un principal de una cuenta realiza una solicitud para acceder a un recurso de otra cuenta.
2. El principal solicitante existe en la cuenta en la que se confía (`AccountA`). Cuando AWS evalúa esta cuenta, comprueba la política basada en la identidad y cualquier política que pueda limitar una política basada en la identidad. Para obtener más información, consulte Evaluación de políticas dentro de una misma cuenta.
3. El recurso solicitado existe en la cuenta de confianza (`AccountB`). Cuando AWS evalúa esta cuenta, comprueba la política basada en recursos que está asociada al recurso solicitado y cualquier política que pueda limitar una política basada en recursos. Para obtener más información, consulte Evaluación de políticas dentro de una misma cuenta.
4. AWS permite la solicitud solo si ambas evaluaciones de políticas de cuenta permiten la solicitud.

Ejemplo de evaluación de política entre cuentas

En el ejemplo siguiente se muestra una situación en la que una política basada en recursos de una cuenta concede permisos a un usuario de otra cuenta.

Suponga que Carlos es un desarrollador con un usuario de IAM denominado 111111111111 en la cuenta `carlossalazar`. Carlos quiere guardar un archivo en el bucket `Production-logs` Amazon S3 de la cuenta 222222222222.

Supongamos también que la política siguiente está asociada al usuario de IAM `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::Production/*"
        },
        {
            "Sid": "DenyS3Logs",
```

```
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::*log*",
                "arn:aws:s3:::*log*/*"
            ]
        }
    ]
}
```

La instrucción `AllowS3ListRead` de esta política permite a Carlos ver una lista de todos los buckets de Amazon S3. La instrucción `AllowS3ProductionObjectActions` concede a Carlos acceso completo al bucket `Production`. La instrucción `DenyS3Logs` deniega a Carlos el acceso a los buckets de S3 que contengan `log` en el nombre. También deniega el acceso a todos los objetos de esos buckets.

Además, la siguiente política basada en recursos (denominada política de bucket) está asociada al bucket `Production` de la cuenta 222222222222.

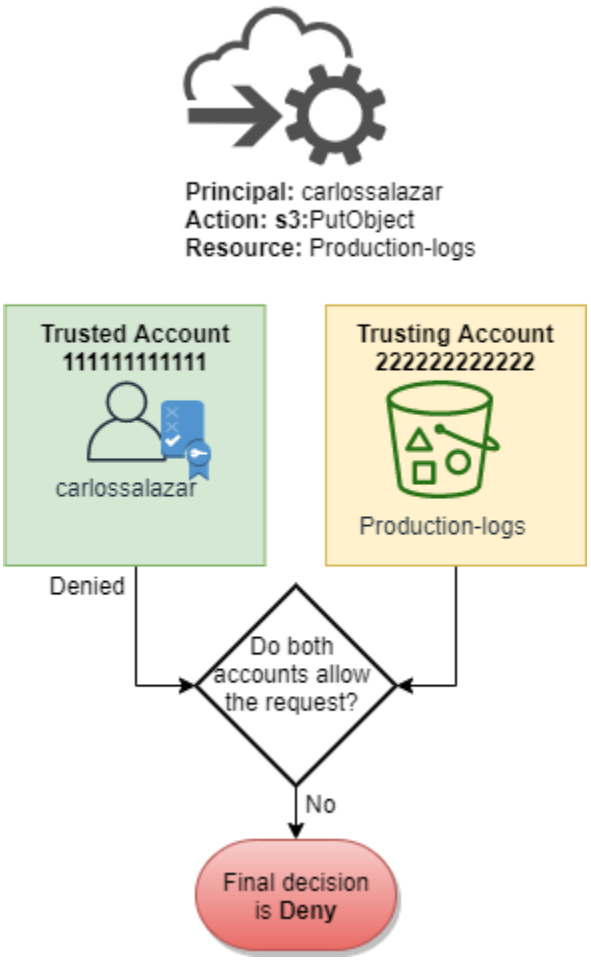
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::Production/*"
        }
    ]
}
```

Esta política permite al usuario `carlossalazar` obtener acceso a los objetos del bucket `Production`. Puede crear y editar, pero no eliminar los objetos del bucket. No puede administrar el propio bucket.

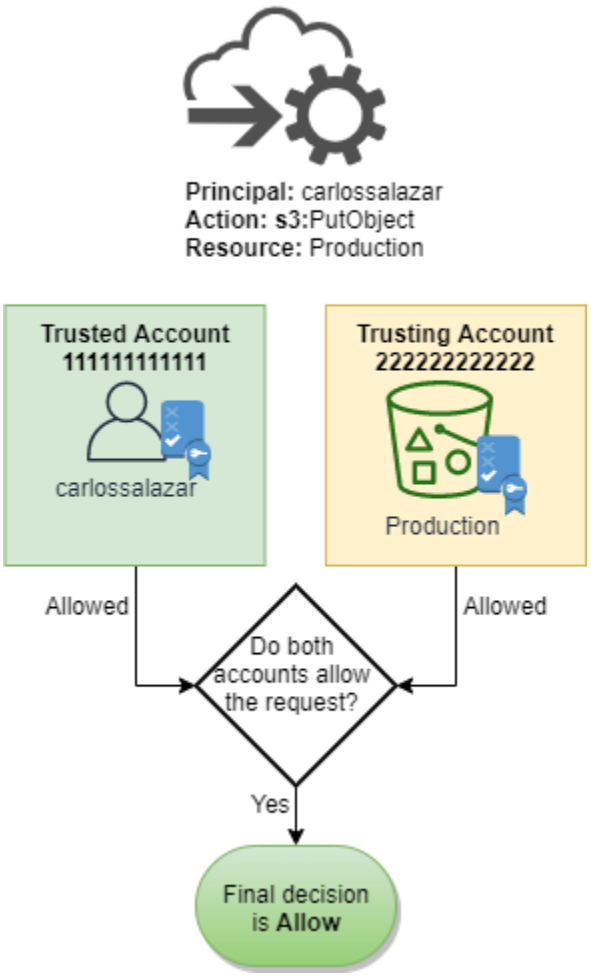
Cuando Carlos solicita guardar un archivo en el bucket `Production-logs`, AWS determina qué políticas se aplican a la solicitud. En este caso, la política basada en la identidad asociada al usuario

de `carlossalazar` es la única política que se aplica en la cuenta `111111111111`. En la cuenta `222222222222`, no hay ninguna política basada en recursos asociada al bucket `Production-logs`. Cuando AWS evalúa la cuenta `111111111111`, devuelve una decisión de tipo `Deny`. Esto se debe a que la instrucción `DenyS3Logs` de la política basada en la identidad deniega explícitamente el acceso a cualquier bucket de registro. Para obtener más información acerca de cómo se evalúa una solicitud dentro de una sola cuenta, consulte Cómo determinar si se una solicitud se permite o se deniega dentro de una cuenta.

Dado que la solicitud se deniega explícitamente dentro de una de las cuentas, la decisión final es denegar la solicitud.



Supongamos que Carlos se da cuenta de su error e intenta guardar el archivo en el bucket `Production`. AWS primero comprueba la cuenta `111111111111` para determinar si la solicitud está permitida. Solo se aplica la política basada en la identidad, que permite la solicitud. A continuación, AWS comprueba la cuenta `222222222222`. Solo se aplica la política basada en recursos asociada al bucket `Production`, que permite la solicitud. Dado que ambas cuentas permiten la solicitud, la decisión final es permitir la solicitud.



Gramática del lenguaje de la política JSON de IAM

En esta página se presenta una gramática formal del lenguaje que se utiliza para crear políticas JSON en IAM. Presentamos esta gramática para que pueda comprender cómo diseñar y validar políticas.

Para ver políticas de ejemplo, consulte los siguientes temas:

- Políticas y permisos en IAM
- Ejemplos de políticas basadas en identidad de IAM
- Ejemplos de políticas para trabajar en la consola de Amazon EC2 y ejemplos de políticas para trabajar con la CLI de AWS, la CLI de Amazon EC2 o un SDK de AWS en la Guía del usuario de Amazon EC2 para instancias de Linux.
- Ejemplos de política de bucket y Ejemplos de políticas de usuario en la Guía del usuario de Amazon Simple Storage Service.

Para ver ejemplos de políticas utilizadas en otros servicios de AWS, consulte la documentación de dichos servicios.

Temas

El lenguaje de la política y JSON

Las políticas se expresan en JSON. Cuando usted crea o edita una política JSON, IAM puede realizar la validación de políticas para ayudarle a crear una política eficaz. IAM identifica errores de sintaxis JSON, mientras que Analizador de acceso de IAM proporciona verificaciones de políticas adicionales con recomendaciones para ayudarle a perfeccionar aún más las políticas. Para obtener más información acerca la validación de políticas, consulte Validación de políticas de IAM. Para obtener más información acerca de las verificaciones de políticas de IAM Access Analyzer y las recomendaciones procesables, consulte Validación de políticas de IAM Access Analyzer.

En este documento, no podemos proporcionar una descripción completa de lo que constituye un código JSON válido. Sin embargo, presentamos algunas reglas de JSON básicas:

- Se permiten espacios en blanco entre entidades individuales.
- Los valores se encierran entre comillas. El uso de las comillas es opcional para los valores numéricos y booleanos.
- Muchos elementos (por ejemplo, `action_string_list` y `resource_string_list`) pueden utilizar una matriz JSON como valor. Las matrices pueden tener uno o varios valores. Si se incluye más de un valor, la matriz se encierra entre corchetes (`[` y `]`) y los valores se delimitan con comas, como en el siguiente ejemplo:

  ```
  "Action" : ["ec2:Describe*","ec2:List*"]
  ```
- Los tipos de datos JSON básicos (booleanos, número y cadena) se definen en RFC 7159.

Convenciones utilizadas en esta gramática

En esta gramática se utilizan las siguientes convenciones:

- Los siguientes caracteres son tokens de JSON y se incluyen en las políticas:

  ```
  { } [ ] " , :
  ```

- Los siguientes caracteres son caracteres especiales de la gramática y no se incluyen en las políticas:

  ```
  = < > ( ) |
  ```

- Si un elemento permite varios valores, se indicará mediante valores repetidos, un delimitador por comas y puntos suspensivos (`...`). Ejemplos:

  ```
  [<action_string>, <action_string>, ...]
  ```

  ```
  <principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }
  ```

  Si se permiten múltiples valores, también se podrá incluir un solo valor. En el caso de un solo valor, se omite la coma final. Si el elemento tiene una matriz (marcada con [y]), pero solo incluye un valor, los paréntesis son opcionales. Ejemplos:

  ```
  "Action": [<action_string>]
  ```

  ```
  "Action": <action_string>
  ```

- Un signo de interrogación (`?`) detrás de un elemento indica que dicho elemento es opcional. Ejemplo:

  ```
  <version_block?>
  ```

  Sin embargo, consulte las notas relativas a la gramática que se muestran a continuación para obtener más información acerca de los elementos opcionales.

- Una línea vertical (`|`) entre elementos indica alternativas. En la gramática, los paréntesis definen el alcance de las alternativas. Ejemplo:

  ```
  ("Principal" | "NotPrincipal")
  ```

- Los elementos que deben ser cadenas literales se encierran entre comillas dobles (`"`). Ejemplo:

  ```
  <version_block> = "Version" : ("2008-10-17" | "2012-10-17")
  ```

Para notas adicionales, consulte la sección Notas sobre la gramática de la política que se presenta tras la descripción de la gramática.

Gramática

En la siguiente lista se describe la gramática del lenguaje de la política. Para obtener más información sobre las convenciones utilizada aquí, consulte la sección anterior. Para obtener información adicional, consulte las siguientes notas.

> **Note**
>
> Esta gramática describe las políticas marcadas con una versión del `2008-10-17` y `2012-10-17`. El elemento de política `Version` es diferente de la versión de una política. El elemento de política `Version` se utiliza en una política y define la versión del lenguaje de la política. Una versión de política, por otro lado, se crea al realizar cambios en una política administrada por el cliente en IAM. La política modificada no anula la política existente. En cambio, IAM crea una nueva versión de la política administrada. Para obtener más información sobre el elemento de política `Version`, consulte Elementos de política JSON de IAM: Version. Para obtener más información sobre las versiones de política, consulte the section called “Control de versiones de políticas de IAM”.

```
policy  = {
     <version_block?>
     <id_block?>
     <statement_block>
}

<version_block> = "Version" : ("2008-10-17" | "2012-10-17")

<id_block> = "Id" : <policy_id_string>

<statement_block> = "Statement" : [ <statement>, <statement>, ... ]

<statement> = {
    <sid_block?>,
    <principal_block?>,
    <effect_block>,
    <action_block>,
    <resource_block>,
    <condition_block?>
}

<sid_block> = "Sid" : <sid_string>
```

```
<effect_block> = "Effect" : ("Allow" | "Deny")

<principal_block> = ("Principal" | "NotPrincipal") : ("*" | <principal_map>)

<principal_map> = { <principal_map_entry>, <principal_map_entry>, ... }

<principal_map_entry> = ("AWS" | "Federated" | "Service" | "CanonicalUser") :
    [<principal_id_string>, <principal_id_string>, ...]

<action_block> = ("Action" | "NotAction") :
    ("*" | [<action_string>, <action_string>, ...])

<resource_block> = ("Resource" | "NotResource") :
    : ("*" | <resource_string> | [<resource_string>, <resource_string>, ...])

<condition_block> = "Condition" : { <condition_map> }
<condition_map> = {
  <condition_type_string> : { <condition_key_string> : <condition_value_list> },
  <condition_type_string> : { <condition_key_string> : <condition_value_list> }, ...
}
<condition_value_list> = [<condition_value>, <condition_value>, ...]
<condition_value> = (<condition_value_string> | <condition_value_string> |
 <condition_value_string>)
```

Notas sobre la gramática de la política

- Una única política puede contener una matriz de instrucciones.
- Las políticas tienen un tamaño máximo de entre 2 048 caracteres y 10 240 caracteres, dependiendo de a qué entidad está asociada la política. Para obtener más información, consulte IAM y cuotas de AWS STS. Los cálculos del tamaño de la política no incluyen los espacios en blanco.
- Los elementos individuales no deben contener varias instancias de la misma clave. Por ejemplo, no puede incluir el bloque `Effect` dos veces en la misma instrucción.
- Los bloques pueden seguir cualquier orden. Por ejemplo, `version_block` puede ir detrás de `id_block` en una política. Del mismo modo, `effect_block`, `principal_block` y `action_block` pueden aparecer en cualquier orden dentro de una instrucción.
- El `id_block` es opcional en políticas basadas en recursos. No debe incluirse en políticas basadas en la identidad.

- El elemento `principal_block` es necesario en las políticas basadas en recursos (por ejemplo, en las políticas de bucket de Amazon S3) y en las políticas de confianza para los roles de IAM. No debe incluirse en políticas basadas en la identidad.
- El elemento `principal_map` de las políticas de bucket de Amazon S3 puede incluir el ID `CanonicalUser`. La mayoría de las políticas basadas en recursos no admiten este mapeo. Para obtener más información sobre el uso del ID de usuario canónico en una política de bucket, consulte Especificación de una entidad principal en una política en la Guía del usuario de Amazon Simple Storage Service.
- Cada valor de cadena (`policy_id_string`, `sid_string`, `principal_id_string`, `action_string`, `resource_string`, `condition_type_string`, `condition_key_string` y la versión de cadena de `condition_value`) puede tener sus propias restricciones de longitud mínima y máxima, permitir unos valores específicos o exigir un formato interno.

Notas acerca de los valores de cadena

En esta sección se proporciona información adicional sobre los valores de cadena que se utilizan en diferentes elementos de una política.

`action_string`

Consta de un espacio de nombres del servicio, dos puntos y el nombre de una acción. Los nombres de acción pueden incluir comodines. Ejemplos:

```
"Action":"ec2:StartInstances"

"Action":[
  "ec2:StartInstances",
  "ec2:StopInstances"
]

"Action":"cloudformation:*"

"Action":"*"

"Action":[
  "s3:Get*",
  "s3:List*"
]
```

`policy_id_string`

Ofrece una forma de incluir información acerca de la política como conjunto. Algunos servicios, como Amazon SQS y Amazon SNS, utilizan el elemento `Id` como reserva. A menos que lo limite un servicio individual, policy_id_string puede incluir espacios. Algunos servicios requieren este valor para ser exclusivos dentro de una cuenta de AWS.

> **Note**
>
> El `id_block` está permitido en políticas basadas en recursos, pero no en políticas basadas en la identidad.

No existe ningún límite respecto a la longitud, aunque esta cadena contribuye a la longitud total de la política, que está limitada.

```
"Id":"Admin_Policy"

"Id":"cd3ad3d9-2776-4ef1-a904-4c229d1642ee"
```

`sid_string`

Ofrece una forma de incluir información acerca de una instrucción individual. Para las políticas de IAM, los caracteres alfanuméricos básicos (A-Z, a-z, 0-9) son los únicos caracteres permitidos en el valor `Sid`. Otros servicios de AWS que admiten las políticas de recursos pueden tener otros requisitos para el valor `Sid`. Por ejemplo, algunos servicios requieren este valor para ser únicos dentro de una Cuenta de AWS y algunos servicios permiten caracteres adicionales como espacios en el valor `Sid`.

```
"Sid":"1"

"Sid": "ThisStatementProvidesPermissionsForConsoleAccess"
```

`principal_id_string`

Proporciona una forma de especificar una entidad principal utilizando el Nombre de recurso de Amazon (ARN) de la Cuenta de AWS, el usuario de IAM, el rol de IAM, el usuario federado o el usuario del rol asumido. Si se trata de una Cuenta de AWS, también puede utilizar el formulario abreviado `AWS:`*`accountnumber`* en lugar de todo el ARN. Para todas las opciones, incluidos los servicios de AWS, los roles asumidos, etc., consulte Especificación de una entidad principal.

Tenga en cuenta que puede utilizar * solo para especificar "todo el mundo/anónimo". No puede utilizarlo para especificar parte de un nombre o ARN.

`resource_string`

En la mayoría de los casos, se compone de un Nombre de recurso de Amazon (ARN).

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::examplebucket/*"
```

`condition_type_string`

Identifica el tipo de condición de prueba, como por ejemplo `StringEquals`, `StringLike`, `NumericLessThan`, `DateGreaterThanEquals`, `Bool`, `BinaryEquals`, `IpAddress`, `ArnEquals`, etc. Para obtener una lista completa de los tipos de condición, consulte Elementos de la política de JSON de IAM: operadores de condición.

```
"Condition": {
  "NumericLessThanEquals": {
    "s3:max-keys": "10"
  }
}

"Condition": {
  "Bool": {
    "aws:SecureTransport": "true"
  }
}

"Condition": {
  "StringEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}
```

`condition_key_string`

Identifica la clave de condición cuyo valor se prueba para determinar si se cumple la condición. AWS define un conjunto de claves de condición que están disponibles en servicios de AWS, incluyendo `aws:PrincipalType`, `aws:SecureTransport`, y `aws:userid`.

Para obtener una lista de las claves de condición de AWS, consulte Claves de contexto de condición globales de AWS. Para conocer las claves de condición que son específicas de un servicio, consulte la documentación correspondiente a dicho servicio, por ejemplo:

- Especificación de las condiciones de una política en la Guía del usuario de Amazon Simple Storage Service
- Políticas de IAM y Amazon EC2 en la Guía del usuario de Amazon EC2 para instancias de Linux.

```
"Condition":{
  "Bool": {
      "aws:SecureTransport": "true"
   }
}

"Condition": {
  "StringNotEquals": {
      "s3:x-amz-server-side-encryption": "AES256"
   }
}

"Condition": {
  "StringEquals": {
    "aws:ResourceTag/purpose": "test"
  }
}
```

`condition_value_string`

Identifica el valor de la cadena condition_key_string que determina si se cumple la condición. Para obtener una lista completa de los valores válidos para un tipo de condición, consulte Elementos de la política de JSON de IAM: operadores de condición.

```
"Condition":{
  "ForAnyValue:StringEquals": {
  "dynamodb:Attributes": [
   "ID",
   "PostDateTime"
         ]
  }
}
```

Managed Policies de AWS para funciones de trabajo

Recomendamos utilizar políticas que otorguen el menor privilegio, o que concedan solo los permisos necesarios para realizar una tarea. La forma más segura de conceder menos privilegios es escribir una política personalizada que tenga solamente los permisos necesarios para su equipo. Debe crear un proceso para permitir que su equipo solicite más permisos cuando sea necesario. Se necesita tiempo y experiencia para crear políticas administradas por el cliente de IAM que proporcionen a su equipo solo los permisos necesarios.

Para comenzar a agregar permisos a sus identidades de IAM (usuarios, grupos de usuarios y roles), puede utilizar Políticas administradas de AWS. Las políticas administradas de AWS cubren casos de uso comunes y están disponibles en su cuenta de Cuenta de AWS. Las políticas administradas de AWS no otorgan permisos de privilegios mínimos. Considere el riesgo de seguridad de conceder a sus entidades principales más permisos de los que necesitan para realizar su trabajo.

Puede adjuntar políticas administradas de AWS, incluidas las funciones de trabajo, a cualquier identidad de IAM. Para cambiar a permisos de privilegios mínimos, puede ejecutar AWS Identity and Access Management Access Analyzer para supervisar las entidades principales con las políticas administradas de AWS. Después de saber qué permisos están utilizando, puede escribir una política personalizada o generar una política con solo los permisos necesarios para su equipo. Esto es menos seguro, pero proporciona más flexibilidad a medida que aprende cómo usa su equipo AWS.

Las políticas administradas por AWS de funciones se han creado para estar en consonancia con las funciones comunes del sector de TI. Puede utilizar estas políticas para conceder los permisos necesarios para realizar las tareas que se esperan de alguien en una determinada función. Estas políticas agrupan permisos para numerosos servicios en una única política, lo que facilita el trabajo, ya que los permisos no están diseminados en varias políticas.

Uso de roles para combinar servicios

Algunas de las políticas utilizan los roles del servicio IAM de para ayudarle a sacar partido de las características de otros servicios de AWS. Estas políticas conceden acceso a `iam:passrole`, que permite a un usuario con la política transmitir un rol a un servicio de AWS. Este rol delega los permisos de IAM al servicio de AWS para llevar a cabo acciones en su nombre.

Debe crear los roles en función de sus necesidades. Por ejemplo, la política de administrador de red permite a un usuario con la política transmitir un rol denominado "flow-logs-vpc" al servicio de Amazon CloudWatch. CloudWatch utiliza dicho rol para registrar y capturar el tráfico IP de las VPC creadas por el usuario.

Para seguir las prácticas recomendadas de seguridad, las políticas de funciones incluyen filtros que limitan los nombres de roles válidos que pueden transmitirse. Esto evita la concesión de permisos innecesarios. Si los usuarios necesitan los roles de servicio opcionales, debe crear un rol que utilice la convención de nomenclatura especificada en la política. A continuación, conceda los permisos al rol. A continuación, el usuario podrá configurar el servicio para utilizar el rol, concediéndole cualquier permiso que el rol proporcione.

En las secciones siguientes, cada nombre de política es un enlace a la página de detalles de la política en la AWS Management Console. Ahí puede ver el documento de la política y revisar los permisos que concede.

Función de trabajo de administrador

Nombre de la política administrada por AWS: AdministratorAccess

Caso de uso: este usuario tiene acceso completo y puede delegar permisos a cada servicio y recurso de AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para todas las acciones de todos los servicios de AWS y todos los recursos de la cuenta. Para obtener más información sobre la política administrada, consulte AdministratorAccess en la AWS Guía de referencia de políticas administradas.

> **Note**
>
> Antes de que un usuario o rol de IAM pueda acceder a la consola de AWS Billing and Cost Management con los permisos de esta política, debe activar antes el acceso de usuarios y roles de IAM. Para ello, siga las instrucciones que se indican en el paso 1 del tutorial sobre cómo delegar el acceso a la consola de facturación.

Función de facturación de trabajo

Nombre de la política administrada por AWS: Billing

Caso de uso: este usuario necesita ver la información de facturación, configurar un pago y autorizarlo. El usuario puede monitorizar los costos acumulados de cada servicio de AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos completos para administrar la facturación, los costos, los métodos de pago, los presupuestos y los informes. Para ver otros ejemplos de políticas de administración de costes, consulte los ejemplos de políticas de AWS Billing en la Guía del usuario de AWS Billing and Cost Management. Para obtener más información sobre las políticas administradas, consulte Facturación en la Guía de referencia de políticas administradas de AWS.

> **Note**
>
> Antes de que un usuario o rol de IAM pueda acceder a la consola de AWS Billing and Cost Management con los permisos de esta política, debe activar antes el acceso de usuarios y roles de IAM. Para ello, siga las instrucciones que se indican en el paso 1 del tutorial sobre cómo delegar el acceso a la consola de facturación.

Función de trabajo de administrador de base de datos

Nombre de política administrada por AWS: DatabaseAdministrator

Caso de uso: este usuario configura y mantiene las bases de datos de la nube de AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para crear, configurar y mantener bases de datos. Incluye acceso a servicios de base de datos AWS, como Amazon DynamoDB, Amazon Relational Database Service (RDS) y Amazon Redshift. Vea la política para conocer la lista completa de servicios de base de datos que admite. Para obtener más información sobre la política gestionada, consulte DatabaseAdministrator en la Guía de referencia de políticas gestionadas AWS.

Esta política de función de trabajo permite transmitir roles a los servicios de AWS. Esta política permite la acción `iam:PassRole` únicamente para los roles indicados en la siguiente tabla. Para obtener más información, consulte Creación de roles y asociación de políticas (consola) más adelante en este tema.

Roles de servicio de IAM opcionales para el trabajo de función de administrador de base de datos

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Seleccionar esta política administrada por AWS
Permitir al usuario que monitorice las bases de datos de RDS	rds-monitoring-role	Rol de Amazon RDS para un monitoreo mejorado	AmazonRDS EnhancedMonitoring Role
Permitir que AWS Lambda monitorice la base de datos y obtenga acceso a las bases de datos externas	rdbms-lambda-access	Amazon EC2	AWSLambda_FullAccess
Permitir a Lambda cargar archivos en Amazon S3 y en clústeres de Amazon Redshift con DynamoDB	lambda_exec_role	AWS Lambda	Crear una nueva política administrada, tal y como se define en el blog de big data de AWS
Permitir que las funciones de Lambda actúen como activadores de las tablas de DynamoDB	lambda-dynamodb-*	AWS Lambda	AWSLambda DynamoDBExecutionRole
Permitir que las funciones de Lambda obtengan acceso a Amazon RDS en una VPC	lambda-vpc-execution-role	Crear un rol con una política de confianza, tal y como se define en la Guía del	AWSLambda VPCAccessExecution Role

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Seleccionar esta política administrada por AWS
		desarrollador de AWS Lambda	
Permitir que AWS Data Pipeline obtenga acceso a los recursos de AWS	DataPipelineDefaultRole	Crear un rol con una política de confianza, tal y como se define en la Guía del desarrollador de AWS Data Pipeline	La documentación AWS Data Pipeline enumera los permisos necesarios para este caso de uso. Consulte Roles de IAM para AWS Data Pipeline
Permitir que las aplicaciones que se ejecuten en instancias Amazon EC2 obtengan acceso a los recursos de AWS	DataPipelineDefaultResourceRole	Crear un rol con una política de confianza, tal y como se define en la Guía del desarrollador de AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Función del trabajo de científico de datos

Nombre de la política administrada por AWS: DataScientist

Caso de uso: este usuario ejecuta consultas y trabajos de Hadoop. El usuario también obtiene acceso a la información y la analiza para las tareas de análisis de datos e inteligencia empresarial.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para crear, administrar y ejecutar consultas en un clúster de Amazon EMR y realizar análisis de datos con herramientas tales como Amazon QuickSight. La política incluye el acceso a servicios de científicos de datos adicionales, como AWS Data Pipeline, Amazon EC2, Amazon Kinesis, Amazon Machine Learning y SageMaker. Vea la política para conocer la lista completa de servicios científicos de datos que admite. Para obtener más información sobre la política administrada, consulte DataScientist en Guía de referencia de políticas AWS administradas.

Esta política de función de trabajo permite transmitir roles a los servicios de AWS. Una instrucción permite pasar cualquier rol a SageMaker. Otra instrucción permite la acción `iam:PassRole` únicamente para los roles indicados en la siguiente tabla. Para obtener más información, consulte Creación de roles y asociación de políticas (consola) más adelante en este tema.

Roles de servicio de IAM opcionales para la función de científico de datos

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
Permitir que las instancias de Amazon EC2 obtengan acceso a servicios y recursos adecuados para clústeres	EMR-EC2_DefaultRole	Amazon EMR para EC2	AmazonElasticMapReduceforEC2Role
Permitir que Amazon EMR obtenga acceso a los servicios y recursos de Amazon EC2 para clústeres	EMR_DefaultRole	Amazon EMR	AmazonEMRServicePolicy_v2
Permita que Kinesis Managed Service for Apache Flink acceda a los orígenes de datos de streaming	kinesis-*	Crear un rol con una política de confianza, tal y como se define en el blog de big data de AWS.	Consulte el blog de big data de AWS, que define cuatro posibles opciones en función de su caso de uso

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
Permitir que AWS Data Pipeline obtenga acceso a los recursos de AWS	DataPipelineDefaultRole	Crear un rol con una política de confianza, tal y como se define en la Guía del desarrollador de AWS Data Pipeline	La documentación AWS Data Pipeline enumera los permisos necesarios para este caso de uso. Consulte Roles de IAM para AWS Data Pipeline
Permitir que las aplicaciones que se ejecuten en instancias Amazon EC2 obtengan acceso a los recursos de AWS	DataPipelineDefaultResourceRole	Crear un rol con una política de confianza, tal y como se define en la Guía del desarrollador de AWS Data Pipeline	AmazonEC2RoleforDataPipelineRole

Función de trabajo de usuario avanzado desarrollador

Nombre de la política administrada por AWS: PowerUserAccess

Caso de uso: este usuario realiza tareas de desarrollo de aplicaciones y puede crear y configurar recursos y servicios que respalden el desarrollo de aplicaciones compatibles con AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: la primera instrucción de esta política utiliza el elemento `NotAction` para permitir todas las acciones para todos los servicios de AWS y para todos los recursos, excepto AWS Identity and Access Management, AWS Organizations y AWS Account Management. La segunda instrucción concede permisos de IAM para crear un rol vinculado al servicio. Esto es

necesario en el caso de ciertos servicios que deben tener acceso a recursos de otro servicio, como un bucket de Amazon S3. También concede permisos de Organizations para ver información acerca de la organización del usuario, incluido el correo electrónico de la cuenta de administración y las limitaciones de la organización. Aunque esta política limita a IAM y Organizations, permite al usuario realizar todas las acciones de IAM Identity Center si está habilitado IAM Identity Center. También otorga permisos de administración de cuentas para ver qué regiones de AWS están habilitadas o deshabilitadas para la cuenta.

Función de trabajo del administrador de red

Nombre de la política administrada por AWS: NetworkAdministrator

Caso de uso: este usuario tiene la tarea de configurar y mantener los recursos de red de AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para crear y mantener recursos de red en Auto Scaling, Amazon EC2, AWS Direct Connect, Route 53, Amazon CloudFront, Elastic Load Balancing, AWS Elastic BeanstalkAmazon SNS, CloudWatch, CloudWatch Logs, Amazon S3, IAM y Amazon Virtual Private Cloud. Para obtener más información sobre la política administrada, consulte NetworkAdministrator en la Guía de referencia de políticas AWS administradas.

Esta función requiere poder transmitir roles a los servicios de AWS. Esta política concede permisos `iam:GetRole` y `iam:PassRole` únicamente para los roles indicados en la siguiente tabla. Para obtener más información, consulte Creación de roles y asociación de políticas (consola) más adelante en este tema.

Roles de servicio de IAM opcionales para la función de administrador de red

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
Permite que Amazon VPC cree y administre registros en CloudWatch Logs en nombre	flow-logs-*	Crear un rol con una política de confianza, tal y	Este caso de uso no tiene una política administr

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
del usuario para monitorear el tráfico IP entrante y saliente de la VPC		como se define en la Guía del usuario de Amazon VPC	ada por AWS existente, pero la documentación indica los permisos necesarios. Consulte la Guía del usuario de Amazon VPC

Acceso de solo lectura

Nombre de la política administrada por AWS: ReadOnlyAccess

Caso de uso: este usuario requiere acceso de solo lectura a todos los recursos de una cuenta de Cuenta de AWS.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para enumerar, obtener, describir y ver recursos y sus atributos de otro modo. No incluye funciones de mutación como crear o eliminar. Esta política incluye acceso de solo lectura a servicios de AWS relacionados con la seguridad, como AWS Identity and Access Management y AWS Billing and Cost Management. Vea la política para conocer la lista completa de servicios y acciones que admite esta política.

Función de trabajo de auditor de seguridad

Nombre de la política administrada por AWS: SecurityAudit

Caso de uso: este usuario monitoriza las cuentas para comprobar que cumplan con los requisitos de seguridad. Este usuario puede obtener acceso a los logs y eventos para investigar posibles infracciones de seguridad o actividades malintencionadas.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede permisos para ver los datos de configuración de muchos servicios de AWS y revisar sus registros. Para obtener más información sobre la política gestionada, consulte SecurityAudit en la Guía de referencia de políticas AWS gestionadas.

Función de trabajo de usuario de soporte

Nombre de la política administrada por AWS: SupportUser

Caso de uso: este usuario se pone en contacto con AWS Support, crea casos de soporte y consulta el estado de los casos existentes.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de política: esta política concede permisos para crear y actualizar casos de AWS Support. Para obtener información sobre las políticas administradas, consulte SupportUser en la Guía de referencia de políticas administradas de AWS.

Función de trabajo de administrador del sistema

Nombre de la política administrada por AWS: SystemAdministrator

Caso de uso: este usuario configura y mantiene los recursos para realizar operaciones de desarrollo.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: Esta política concede permisos para crear y mantener los recursos de una gran variedad de servicios de AWS, incluyendo AWS CloudTrail, Amazon CloudWatch, AWS CodeCommit, AWS CodeDeploy, AWS Config, AWS Directory Service, Amazon EC2, AWS Identity and Access Management, AWS Key Management Service, AWS Lambda, Amazon RDS, Route 53, Amazon S3, Amazon SES, Amazon SQS, AWS Trusted Advisor y Amazon VPC. Para obtener más

información sobre la política administrada, consulte SystemAdministrator en la Guía de referencia de políticas AWS administradas.

Esta función requiere poder transmitir roles a los servicios de AWS. Esta política concede permisos `iam:GetRole` y `iam:PassRole` únicamente para los roles indicados en la siguiente tabla. Para obtener más información, consulte Creación de roles y asociación de políticas (consola) más adelante en este tema. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Roles de servicio de IAM opcionales para el trabajo de administrador de sistemas

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
Permitir que las aplicaciones que se ejecutan en instancias EC2 de un clúster de Amazon ECS obtengan acceso a Amazon ECS	ecr-sysadmin-*	Rol de Amazon EC2 para EC2 Container Service	AmazonEC2ContainerServiceforEC2Role
Permitir a un usuario que monitorice las bases de datos	rds-monitoring-role	Rol de Amazon RDS para un monitoreo mejorado	AmazonRDSEnhancedMonitoringRole
Permitir que las aplicaciones que se ejecutan en instancias EC2 obtengan acceso a los recursos de AWS	ec2-sysadmin-*	Amazon EC2	Ejemplo de política para un rol que concede acceso a un bucket de S3, tal y como se muestra en la Guía del usuario de Amazon EC2 para instancias de Linux; personalizar según sea necesario

Caso de uso	Nombre de rol (* es un carácter comodín)	Tipo de rol de servicio que ha de seleccionarse	Política administrada por AWS que ha de seleccionarse
Permitir que Lambda lea DynamoDB Streams y escriba en los CloudWatch Logs	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

Función de trabajo de usuario de solo lectura

Nombre de la política administrada por AWS: ViewOnlyAccess

Caso de uso: este usuario puede ver en su cuenta una lista de metadatos básicos y recursos de AWS en los servicios. El usuario no puede leer contenido de recursos ni metadatos más allá de la información de cuotas y de listas correspondientes a los recursos.

Actualizaciones de políticas: AWS mantiene y actualiza esta política. Para obtener un historial de cambios para esta política, consulte la política en la consola de IAM y, a continuación, elija la pestaña Versiones de políticas. Para obtener más información acerca de las políticas de funciones de trabajo, consulte Actualizaciones de políticas administradas de AWS para funciones de trabajo.

Descripción de la política: esta política concede a `List*`, `Describe*`, `Get*`, `View*` y `Lookup*` acceso a los recursos de los servicios de AWS. Para ver qué acciones incluye esta política para cada servicio, consulte ViewOnlyAccess. Para obtener más información sobre la política gestionada, consulte ViewOnlyAccess en la Guía de referencia de políticas gestionadas AWS.

Actualizaciones de políticas administradas de AWS para funciones de trabajo

AWS mantiene todas estas políticas y las actualiza para incluir soporte de nuevos servicios y nuevas funciones a medida que AWS los agrega. Los clientes no pueden modificar estas políticas. Puede realizar una copia de la política y, a continuación, modificar la copia, pero que no se actualiza automáticamente como copia de AWS introduce nuevos servicios y operaciones del API.

Para una política de función de trabajo, puede ver el historial de versiones y la hora y fecha de cada actualización en la consola de IAM. Para hacer esto, utilice los vínculos de esta página para ver los detalles de la política. A continuación, elija la pestaña Versiones de políticas para ver las versiones.

Esta página muestra las últimas 25 versiones de una política. Para ver todas las versiones de una política, llame al comando get-policy-version de AWS CLI o la operación GetPolicyVersion de la API

> **Note**
>
> Puede tener hasta cinco versiones de una política administrada por el cliente, pero AWS conserva el historial de versiones completo de las políticas administradas de AWS.

Creación de roles y asociación de políticas (consola)

Varias de las políticas indicadas anteriormente hacen que pueda configurar servicios de AWS con roles que les permitan llevar a cabo operaciones por usted. Las políticas de funciones especifican los nombres exactos del rol que debe utilizar o al menos incluyen un prefijo que especifique la primera parte del nombre que puede utilizarse. Para crear uno de estos roles, siga los pasos que se indican en el siguiente procedimiento.

Cómo crear un rol para un Servicio de AWS (consola de IAM)

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
3. En Tipo de entidad de confianza, elija Servicio de AWS.
4. En Servicio o caso de uso, seleccione un servicio y, a continuación, el caso de uso. Los casos de uso son definidos por el servicio de modo tal que ya incluyen la política de confianza que el servicio mismo requiere.
5. Seleccione Siguiente.
6. Para las Políticas de permisos, las opciones dependen del caso de uso que haya seleccionado:
   - Si el servicio define los permisos para el rol, no puede seleccionar políticas de permisos.
   - Seleccione entre un conjunto limitado de políticas de permisos.
   - Seleccione una de todas las políticas de permisos.
   - No seleccione ninguna política de permisos en este momento. Después de crear el rol, genere las políticas y luego asócielas al rol.
7. (Opcional) Configure un límite de permisos. Se trata de una característica avanzada que está disponible para los roles de servicio, pero no para los roles vinculados a servicios.

a. Abra la sección Configurar límite de permisos y, a continuación, elija Utilizar un límite de permisos para controlar los permisos que puede tener el rol como máximo.

   IAM incluye una lista de las políticas administradas por AWS y de las políticas administradas por el cliente de cada cuenta.

b. Seleccione la política que desea utilizar para el límite de permisos.

8. Seleccione Siguiente.
9. Para Nombre del rol, las opciones varían según el servicio:

   - Si el servicio define el nombre del rol, no podrá editarlo.
   - Si el servicio define un prefijo para el nombre del rol, puede ingresar un sufijo opcional.
   - Si el servicio no define el nombre del rol, podrá nombrarlo usted mismo.

   > **Important**
   >
   > Cuando asigne un nombre a un rol, tenga en cuenta lo siguiente:
   >
   > - Los nombres de rol deben ser únicos dentro de su Cuenta de AWS, y no se pueden hacer únicos mediante mayúsculas y minúsculas.
   >
   >   Por ejemplo, no puede crear roles denominados tanto **`PRODROLE`** como **`prodrole`**. Cuando se utiliza un nombre de rol en una política o como parte de un ARN, el nombre de rol distingue entre mayúsculas y minúsculas, sin embargo, cuando un nombre de rol les aparece a los clientes en la consola, como por ejemplo durante el proceso de inicio de sesión, el nombre de rol no distingue entre mayúsculas y minúsculas.
   > - Dado que otras entidades podrían hacer referencia al rol, no es posible editar el nombre del rol una vez creado.

10. (Opcional) En Descripción, ingrese una descripción para el rol.
11. (Opcional) Para editar los casos de uso y los permisos de la función, en las secciones Paso 1: Seleccionar entidades confiables o en Paso 2: Agregar permisos, elija Editar.
12. (Opcional) Para ayudar a identificar, organizar o buscar el rol, agregue etiquetas como pares clave-valor. Para obtener más información sobre el uso de etiquetas en IAM, consulte Etiquetado de recursos de IAM en la Guía de usuario de IAM.
13. Revise el rol y, a continuación, elija Crear rol.

Ejemplo 1: configuración de un usuario como administrador de base de datos (consola)

Este ejemplo muestra los pasos necesarios para establecer a Alice, usuaria de IAM, como Administradora de base de datos. Use la información de la primera fila de la tabla de dicha sección y permita a la usuaria que habilite el monitoreo de Amazon RDS. Debe asociar la política DatabaseAdministrator al usuario de IAM de Alice para que pueda administrar los servicios de base de datos de Amazon. Esta política también permite que Alice transmita un rol denominado `rds-monitoring-role` al servicio de Amazon RDS que permite al servicio supervisar las bases de datos de Amazon RDS en su nombre.

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Políticas, escriba **`database`** en el cuadro de búsqueda y luego pulse Entrar.
3. Seleccione el botón de radio correspondiente a la política DatabaseAdministrator, luego Acciones y, por último Asociar.
4. En la lista de entidades, seleccione Alice y, a continuación, Asociar política. A partir de este momento Alice puede administrar las bases de datos de AWS. Sin embargo, debe configurar el rol de servicio para permitir a Alice monitorizar dichas bases de datos.
5. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
6. Elija el tipo de función de servicio de AWS y, a continuación, elija Amazon RDS.
7. Elija el caso de uso Rol de Amazon RDS para el monitoreo mejorado.
8. Amazon RDS define los permisos de su rol. Elija Siguiente: revisión para continuar.
9. El nombre del rol debe ser uno de los especificados en la política de DatabaseAdministrator que ahora tiene Alice. Uno de ellos es **`rds-monitoring-role`**. Ingréselo en Nombre del rol.
10. (Opcional) En Descripción del rol, introduzca una descripción para el nuevo rol.
11. Después de revisar los detalles, elija Crear rol.
12. Ahora Alice puede activar Monitoreo mejorado de RDS en la sección Monitoreo de la consola de Amazon RDS. Por ejemplo, puede hacerlo al crear una instancia de base de datos, crear una réplica de lectura o modificar una instancia de base de datos. Debe ingresar el nombre del rol creado (rds-monitoring-role) en el cuadro Rol de supervisión al establecer Habilitar supervisión mejorada en Sí.

Ejemplo 2: configuración de un usuario como administrador de red (consola)

Este ejemplo muestra los pasos necesarios para establecer a Jorge, usuario de IAM, como Administrador de red. Se usa la información de la tabla en esa sección para permitir a Jorge supervisar el tráfico IP que va hacia y desde una VPC. También permite a Jorge capturar dicha información en los registros de CloudWatch. Debe asociar la política NetworkAdministrator al usuario de IAM de Jorge para que pueda configurar los recursos de red de AWS. Esta política también permite a Jorge transmitir un rol cuyo nombre comience por `flow-logs*` a Amazon EC2 al crear un registro de flujo. En este caso, a diferencia del ejemplo 1, no existe un tipo de rol de servicio predefinido, de modo que debe realizar algunos pasos de forma distinta.

1. Inicie sesión en la AWS Management Console y abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Políticas y, a continuación, ingrese **`network`** en el cuadro de búsqueda y pulse Entrar.
3. Seleccione el botón de radio situado junto a la política NetworkAdministrator, luego Acciones y, por último, Asociar.
4. En la lista de usuarios, seleccione la casilla de verificación junto a Jorge y, a continuación, elija Asociar política. Jorge puede administrar ahora los recursos de red de AWS. Sin embargo, debe configurar el rol de servicio para permitir la monitorización del tráfico IP de la VPC.
5. Dado que el rol de servicio que necesita crear no tiene una política administrada predefinida, primero debe crearla. En el panel de navegación, seleccione Políticas y, a continuación, elija Crear política.
6. En la sección Editor de políticas, seleccione la opción JSON y copie el texto del siguiente documento de política de JSON. Pegue el texto en el cuadro de texto JSON.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Effect": "Allow",
```

```
      "Resource": "*"
    }
  ]
}
```

7. Resuelva las advertencias de seguridad, errores o advertencias generales generadas durante la validación de política y luego elija Siguiente.

> **Note**
>
> Puede alternar entre las opciones Visual y JSON del editor en todo momento. No obstante, si realiza cambios o selecciona Siguiente en la opción Visual del editor, es posible que IAM reestructure la política, con el fin de optimizarla para el editor visual. Para obtener más información, consulte Reestructuración de políticas.

8. En la página Revisar y crear, escriba **`vpc-flow-logs-policy-for-service-role`** como nombre de la política. Revise los Permisos definidos en esta política para ver los permisos que concede la política y, a continuación, seleccione Crear política para guardar su trabajo.

   La nueva política aparece en la lista de las políticas administradas y está lista para asociar.

9. En el panel de navegación de la consola de IAM, seleccione Roles y, a continuación, elija Crear rol.
10. Seleccione el tipo de rol Servicio de AWS y luego elija Amazon EC2.
11. Elija el caso de uso Amazon EC2
12. En la página Asociar políticas de permisos, seleccione la política que ha creado anteriormente, vpc-flow-logs-policy-for-service-role, a continuación, elija Siguiente: revisión.
13. El nombre del rol debe estar permitido por la política de NetworkAdministrator que Jorge tiene ahora. Los nombres que comiencen por `flow-logs-` están permitidos. En este ejemplo, ingrese **`flow-logs-for-jorge`** como Nombre de rol.
14. (Opcional) En Descripción del rol, introduzca una descripción para el nuevo rol.
15. Después de revisar los detalles, elija Crear rol.
16. Ahora puede configurar la política de confianza necesaria para este caso. En la página Roles, seleccione el rol flow-logs-for-jorge (el nombre, no la casilla de verificación). En la página de detalles del nuevo rol, elija la pestaña Relaciones de confianza y, a continuación, elija Editar relación de confianza.

17. Cambie la línea de "Service" para que se lea como sigue, sustituyendo la entrada por `ec2.amazonaws.com`:

```
        "Service": "vpc-flow-logs.amazonaws.com"
```

18. Jorge ahora puede crear registros de flujo para una VPC o subred en la consola de Amazon EC2. Al crear el registro de flujo, especifique el rol flow-logs-for-jorge. Este rol tiene los permisos para crear el log y escribir datos en él.

Claves de contexto de condición globales de AWS

Cuando una entidad principal realiza una solicitud a AWS, AWS recopila la información de la solicitud en un contexto de solicitud. Puede utilizar el elemento `Condition` de una política JSON para comparar las claves de la solicitud de contexto con los valores de claves que especifique en su política. La información de la solicitud proviene de diferentes orígenes, que abarcan la entidad principal que realiza la solicitud, el recurso con el que se realiza la solicitud y los metadatos sobre la solicitud en sí.

Las claves de condición globales se pueden usar en todos los servicios de AWS. Si bien estas claves de condición se pueden usar en todas las políticas, la clave no está disponible en todos los contextos de solicitud. Por ejemplo, la clave de condición `aws:SourceAccount` solo está disponible cuando la llamada a su recurso la realiza directamente una entidad principal de servicio de AWS. Para obtener más información sobre las circunstancias en las que se incluye una clave global en el contexto de la solicitud, consulte la información de Disponibilidad para cada clave.

Algunos servicios específicos crean sus propias claves de condición que están disponibles en el contexto de solicitud de otros servicios. Las claves de condición entre servicios son un tipo de clave de condición global que incluyen un prefijo que coincide con el nombre del servicio, como `ec2:` o `lambda:`, pero están disponibles en otros servicios.

Las claves de condición específicas del servicio se definen para su uso exclusivo con un servicio específico de AWS. Por ejemplo, Amazon S3 le permite escribir una política con la clave de condición `s3:VersionId` para limitar el acceso a una versión específica de un objeto de Amazon S3. Esta clave de condición es exclusiva del servicio, lo que significa que solo funciona con solicitudes al servicio Amazon S3. Para obtener información sobre las claves de condición específicas del servicio, consulte Acciones, recursos y claves de condición para los servicios de AWS y elija el servicio cuyas claves desee ver.

Note

Si utiliza claves de condición que solo están disponibles en algunas circunstancias, puede utilizar las versiones IfExists de los operadores de condición. Si faltan las claves de condición de un contexto de solicitud, la política podría generar un error en la evaluación. Por ejemplo, utilice la siguiente bloque de condiciones con `...IfExists` para satisfacer cuando una solicitud específica proviene de un rango de direcciones IP o de una VPC específica. Si no se incluye alguna o ambas claves en el contexto de la solicitud, la condición sigue devolviendo `true`. Los valores solo se comprueban si la clave especificada se incluye en el contexto de la solicitud.

```
"Condition": {
    "IpAddressIfExists": {"aws:SourceIp" : ["xxx"] },
    "StringEqualsIfExists" : {"aws:SourceVpc" : ["yyy"]}
}
```

Important

Para comparar su condición con un contexto de solicitud con varios valores de clave, debe utilizar `ForAllValues` u operadores de configuración `ForAnyValue`. Utilice operadores de conjunto solo con claves de condición de varios valores. No utilice operadores de conjunto con claves de condición de un solo valor. Para obtener más información, consulte Claves de contexto multivalor.

Propiedades de la entidad principal	Propiedades de una sesión de rol	Propiedades de la red	Propiedades del recurso	Propiedades de la solicitud
aws:PrincipalArn aws:PrincipalAccount aws:PrincipalOrgPaths	aws:FederatedProvider aws:TokenIssueTime	aws:SourceIp aws:SourceVpc aws:SourceVpce aws:VpcSourceIp	aws:ResourceAccount aws:ResourceOrgPaths	aws:CalledVia aws:CalledViaFirst aws:CalledViaLast

Propiedades de la entidad principal	Propiedades de una sesión de rol	Propiedades de la red	Propiedades del recurso	Propiedades de la solicitud
aws:PrincipalOrgID aws:PrincipalTag/tag-key aws:PrincipalIsAWSService aws:PrincipalServiceName aws:PrincipalServiceNamesList aws:PrincipalType aws:userid aws:username	aws:MultiFactorAuthAge aws:MultiFactorAuthPresent aws:Ec2InstanceSourceVpc aws:Ec2InstanceSourcePrivateIPv4 aws:SourceIdentity ec2:RoleDelivery ec2:SourceInstanceArn glue:RoleAssumedBy glue:CredentialIssuingService lambda:SourceFunctionArn ssm:SourceInstanceArn		aws:ResourceOrgID aws:ResourceTag/tag-key	aws:ViaAWSService aws:CurrentTime aws:EpochTime aws:referer aws:RequestedRegion aws:RequestTag/clave-etiqueta aws:TagKeys aws:SecureTransport aws:SourceArn aws:SourceAccount aws:SourceOrgPaths aws:SourceOrgID aws:UserAgent

Propiedades de la entidad principal	Propiedades de una sesión de rol	Propiedades de la red	Propiedades del recurso	Propiedades de la solicitud
	identitystore:UserId			

Propiedades de la entidad principal

Utilice las siguientes claves de condición para comparar los detalles de la entidad principal que realiza la solicitud con las propiedades de la entidad principal que especifique en la política. Para obtener una lista de las entidades principales que pueden realizar solicitudes, consulte Especificación de una entidad principal.

Contenido

- aws:PrincipalArn
- aws:PrincipalAccount
- aws:PrincipalOrgPaths
- aws:PrincipalOrgID
- aws:PrincipalTag/tag-key
- aws:PrincipalIsAWSService
- aws:PrincipalServiceName
- aws:PrincipalServiceNamesList
- aws:PrincipalType
- aws:userid
- aws:username

aws:PrincipalArn

Utilice esta clave para comparar el Nombre de recurso de Amazon (ARN) de la entidad principal que ha realizado la solicitud con el ARN que se especifique en la política. Para los roles de IAM, el contexto de solicitud devuelve el ARN del rol, no el ARN del usuario que asumió el rol.

- Disponibilidad: Esta clave se incluye en el contexto de solicitud de todas las solicitudes firmadas. Las solicitudes anónimas no incluyen esta clave. Puede especificar los siguientes tipos de entidades principales en esta clave de condición:
  - Rol de IAM
  - Usuario de IAM
  - Sesión de usuario federado de AWS STS
  - Usuario raíz de Cuenta de AWS
- Tipo de datos: ARN, cadena

  AWS recomienda utilizar operadores de ARN en lugar de operadores de cadenas al comparar los ARN.
- Tipo de valor: Valor único
- Valores de ejemplo: en la siguiente lista se muestra el valor de contexto de solicitud que se devuelve para los diferentes tipos de entidades principales que se pueden especificar en la clave de condición `aws:PrincipalArn`:
  - Rol de IAM: El contexto de la solicitud contiene el siguiente valor para la clave de condición `aws:PrincipalArn`. No especifique el ARN de la sesión de rol asumido como valor para esta clave de condición. Para obtener más información acerca de la entidad principal de la sesión de rol asumido, consulte Entidades principales de sesión de rol.

    ```
    arn:aws:iam::123456789012:role/role-name
    ```
  - Usuario de IAM: El contexto de la solicitud contiene el siguiente valor para la clave de condición `aws:PrincipalArn`.

    ```
    arn:aws:iam::123456789012:user/user-name
    ```
  - Sesiones de usuario federado de AWS STS: El contexto de la solicitud contiene el siguiente valor para la clave de condición `aws:PrincipalArn`.

    ```
    arn:aws:sts::123456789012:federated-user/user-name
    ```
  - Usuario raíz de la Cuenta de AWS: El contexto de la solicitud contiene el siguiente valor para la clave de condición `aws:PrincipalArn`. Cuando se especifica el ARN del usuario raíz como valor para la clave de condición `aws:PrincipalArn`, limita los permisos solo para el usuario raíz de la Cuenta de AWS. Esto es distinto de especificar el ARN del usuario raíz en el elemento principal de una política basada en recursos, que delega la autoridad en la Cuenta de AWS.

Para obtener más información sobre cómo especificar el ARN del usuario raíz en el elemento principal de una política basada en recursos, consulte Entidades principales de Cuenta de AWS.

```
arn:aws:iam::123456789012:root
```

Puede especificar el ARN del usuario raíz como un valor de la clave de condición `aws:PrincipalArn` en políticas de control de servicio (SCP) de AWS Organizations. Las SCP son un tipo de política de organización que se emplea para administrar permisos en una organización y solo afectan a las cuentas de miembros de la organización. Una SCP limita los permisos para los usuarios y roles de IAM en las cuentas miembro, incluido el usuario raíz de la cuenta de miembro. Para obtener más información sobre el efecto de las SCP sobre los permisos, consulte Efectos de las SCP en los permisos en la Guía del usuario de Organizations.

aws:PrincipalAccount

Utilice esta clave para comparar la cuenta a la que pertenece la entidad principal solicitante con el identificador de cuenta que especifique en la política. Para las solicitudes anónimas, el contexto de la solicitud devuelve `anonymous`.

- Disponibilidad: esta clave se incluye en el contexto de solicitud de todas las solicitudes, incluidas las anónimas.
- Tipo de datos: cadena
- Tipo de valor: Valor único

En el siguiente ejemplo, se deniega el acceso a todo excepto a las entidades principales con el número de cuenta `123456789012`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID:resource"
      ],
      "Condition": {
```

```
        "StringNotEquals": {
          "aws:PrincipalAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

aws:PrincipalOrgPaths

Utilice esta clave para comparar la ruta de acceso de AWS Organizations de la entidad principal que realiza la solicitud a la ruta de acceso en la política. La entidad principal puede ser un usuario de IAM, un rol de IAM, un usuario federado o un Usuario raíz de la cuenta de AWS. En una política, esta clave de condición garantiza que el solicitante es un miembro de la cuenta dentro de la raíz de la organización o unidades organizativas (OU) especificadas en AWS Organizations. Una ruta AWS Organizations es una representación de texto de la estructura de una entidad de Organizations. Para obtener más información sobre el uso y la comprensión de las rutas, consulte Comprender la ruta de la entidad AWS Organizations.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si la entidad principal es miembro de una organización. Las solicitudes anónimas no incluyen esta clave.
- Tipo de datos: cadena (lista)
- Tipo de valor: multivalor

> **Note**
>
> Los ID de organización son únicos globalmente, pero los ID de unidad organizativa y los ID de raíz solo son únicos dentro de una organización. Esto significa que no hay dos organizaciones que compartan el mismo ID de organización. Sin embargo, otra organización puede tener una unidad organizativa o raíz con el mismo ID que la suya. Le recomendamos que incluya siempre el ID de organización cuando especifique una unidad organizativa o raíz.

Por ejemplo, la siguiente condición devuelve `true` para las entidades principales en cuentas que están asociadas directamente a la unidad organizativa `ou-ab12-22222222`, pero no en sus unidades organizativas secundarias.

```
"Condition" : { "ForAnyValue:StringEquals" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

La siguiente condición devuelve `true` para las entidades principales de una cuenta que está asociada directamente a la unidad organizativa o a cualquiera de sus unidades organizativas secundarias. Cuando se incluye un comodín, se debe utilizar el operador de condición `StringLike`.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
*"]
}}
```

La siguiente condición devuelve `true` para las entidades principales de una cuenta que está adjunta directamente a cualquiera de las OU secundarias, pero no directamente a la OU primaria. La condición anterior es para la unidad organizativa o para cualquier unidad organizativa secundaria. La siguiente condición es solo para las secundarias (y las subsiguientes que le siguen).

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/
ou-*"]
}}
```

La siguiente condición permite el acceso a todas las entidades principales de la organización `o-a1b2c3d4e5`, independientemente de su unidad organizativa principal.

```
"Condition" : { "ForAnyValue:StringLike" : {
     "aws:PrincipalOrgPaths":["o-a1b2c3d4e5/*"]
}}
```

`aws:PrincipalOrgPaths` es una clave de condición multivalor. Las claves de condición multivalor pueden tener varios valores en el contexto de la solicitud. Cuando se utilizan varios valores con el operador de condición `ForAnyValue`, la ruta de acceso del principal debe coincidir con una de las rutas enumeradas en la política. Para obtener más información acerca de las claves de condición multivalor, consulte Claves de contexto multivalor.

```
    "Condition": {
        "ForAnyValue:StringLike": {
            "aws:PrincipalOrgPaths": [
                "o-a1b2c3d4e5/r-ab12/ou-ab12-33333333/*",
                "o-a1b2c3d4e5/r-ab12/ou-ab12-22222222/*"
            ]
        }
    }
```

aws:PrincipalOrgID

Utilice esta clave para comparar el identificador de la organización en AWS Organizations a la que pertenece la entidad principal solicitante con el identificador especificado en la política.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si la entidad principal es miembro de una organización. Las solicitudes anónimas no incluyen esta clave.
- Tipo de datos: cadena
- Tipo de valor: valor único

Esta clave global proporciona una alternativa a mostrar todos los ID de todas las cuentas de AWS de una organización. Puede utilizar esta clave de condición para simplificar la especificación del elemento `Principal` en una política basada en recursos. Puede especificar el ID de organización en el elemento de condición. Al añadir y quitar cuentas, las políticas que incluyen la clave `aws:PrincipalOrgID` incluyen automáticamente las cuentas correctas y no requieren una actualización manual.

Por ejemplo, la siguiente política de bucket de Amazon S3 permite a los miembros de cualquier cuenta de la organización `o-xxxxxxxxxxx` agregar un objeto al bucket `policy-ninja-dev`.

```
{
 "Version": "2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::policy-ninja-dev/*",
   "Condition": {"StringEquals":
     {"aws:PrincipalOrgID":"o-xxxxxxxxxxx"}
```

```
    }
  }
}
```

> **Note**
>
> Esta condición global también se aplica a la cuenta de administración de una organización AWS. Esta política impide que todos las entidades principales fuera de la organización especificada accedan al bucket de Amazon S3. Esto incluye cualquiera de los servicios de AWS que interactúan con sus recursos internos, tales como el envío de datos de registro de AWS CloudTrail a los buckets de Amazon S3. Para obtener información sobre cómo puede conceder acceso de forma segura para los servicios de AWS, consulte aws:PrincipalIsAWSService.

Para obtener más información sobre AWS Organizations, consulte ¿Qué es AWS Organizations? en la Guía del usuario de AWS Organizations.

aws:PrincipalTag/tag-key

Utilice esta clave para comparar la etiqueta asociada a la entidad principal que realiza la solicitud con la etiqueta que especifique en la política. Si la entidad principal tiene más de una etiqueta asociada, el contexto de la solicitud incluye una clave `aws:PrincipalTag` para cada clave de etiqueta asociada.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud si la entidad principal está usando un usuario de IAM con etiquetas asociadas. Se incluye para una entidad principal que utiliza un rol de IAM con etiquetas o etiquetas de sesión asociadas. Las solicitudes anónimas no incluyen esta clave.
- Tipo de datos: cadena
- Tipo de valor: valor único

Puede añadir atributos personalizados a un usuario o rol en forma de un par de clave-valor. Para obtener más información sobre las etiquetas en IAM, consulte Etiquetado de recursos de IAM. Puede utilizar `aws:PrincipalTag` para controlar el acceso para principales de AWS.

En este ejemplo se muestra cómo puede crear una política basada en identidad que permita a usuarios con la etiqueta **`department=hr`** administrar usuarios, grupos o roles de IAM. Para utilizar

esta política, sustituya el *texto en cursiva del marcador* de la política de ejemplo con su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalTag/department": "hr"
        }
      }
    }
  ]
}
```

aws:PrincipalIsAWSService

Utilice esta clave para comprobar si la llamada a su recurso se está realizando directamente por una entidad principal de servicio de AWS. Por ejemplo, AWS CloudTrail utiliza la entidad de servicio de `cloudtrail.amazonaws.com` para escribir registros en su bucket de Amazon S3. La clave de contexto de solicitud se establece en true cuando un servicio utiliza una entidad principal de servicio para realizar una acción directa en los recursos. La clave de contexto se configura como false si un servicio utiliza las credenciales de una entidad principal de IAM para realizar una solicitud en nombre de la entidad principal. También se configura en false si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal.

- Disponibilidad: esta clave está presente en el contexto de la solicitud de todas las solicitudes de API firmadas que utilizan credenciales de AWS. Las solicitudes anónimas no incluyen esta clave.
- Tipos de datos: booleano
- Tipo de valor: valor único

Puede utilizar esta clave de condición para limitar el acceso a las identidades de confianza y a las ubicaciones de red esperadas mientras concede acceso seguro a servicios de AWS.

En el siguiente ejemplo de política de bucket de Amazon S3, el acceso al bucket está restringido a menos que la solicitud se origine en `vpc-111bbb22` o provenga de una entidad principal de servicio, como CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWSLogs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22"
        },
        "BoolIfExists": {
          "aws:PrincipalIsAWSService": "false"
        }
      }
    }
  ]
}
```

En el siguiente video tiene más información acerca de cómo utilizar la clave de condición `aws:PrincipalIsAWSService` en una política.

Conceda acceso seguro a sus usuarios autorizados, ubicaciones de red esperadas y servicios de AWS juntos.

aws:PrincipalServiceName

Utilice esta clave para comparar el nombre de la entidad principal del servicio en la política con la entidad principal de servicio que está realizando solicitudes a sus recursos. Puede utilizar esta clave para comprobar si esta llamada es realizada por una entidad principal de servicio específica. Cuando una entidad principal de servicio realiza una solicitud directa a su recurso, la clave `aws:PrincipalServiceName` contiene el nombre de la entidad principal del servicio. Por ejemplo, el nombre de la entidad principal de servicio AWS CloudTrail es `cloudtrail.amazonaws.com`.

- Disponibilidad: esta clave está presente en la solicitud cuando a la llamada la realiza una entidad principal de servicio de AWS. Esta clave no está presente en ninguna otra situación, incluida la siguiente:
  - Si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal.
  - Si un servicio utiliza las credenciales de una entidad principal de IAM para realizar una solicitud en nombre de la entidad principal.
  - Si a la llamada la realiza directamente una entidad principal de IAM.
  - Si a la llamada la hace un solicitante anónimo.
- Tipo de datos: cadena
- Tipo de valor: valor único

Puede utilizar esta clave de condición para limitar el acceso a las identidades de confianza y a las ubicaciones de red esperadas mientras concede acceso seguro a un servicio de AWS.

En el siguiente ejemplo de política de bucket de Amazon S3, el acceso al bucket está restringido a menos que la solicitud se origine en `vpc-111bbb22` o provenga de una entidad principal de servicio, como CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/AWSLogs/AccountNumber/*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-111bbb22",
          "aws:PrincipalServiceName": "cloudtrail.amazonaws.com"
        }
      }
    }
  ]
}
```

aws:PrincipalServiceNamesList

Esta clave proporciona una lista de todas las entidades principales del servicio de nombres que pertenecen al servicio. Esta es una clave de condición avanzada. Puede utilizarla para restringir que el servicio acceda a su recurso solo desde una región específica. Algunos servicios pueden crear entidades de servicio regionales para indicar una instancia concreta del servicio dentro de una Región específica. Puede limitar el acceso a un recurso a una instancia concreta del servicio. Cuando una entidad de servicio realiza una solicitud directa a su recurso, el `aws:PrincipalServiceNamesList` contiene una lista desordenada de todos los nombres principales de servicio asociados a la instancia regional del servicio.

- Disponibilidad: esta clave está presente en la solicitud cuando a la llamada la realiza una entidad principal de servicio de AWS. Esta clave no está presente en ninguna otra situación, incluida la siguiente:
  - Si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal.
  - Si un servicio utiliza las credenciales de una entidad principal de IAM para realizar una solicitud en nombre de la entidad principal.
  - Si a la llamada la realiza directamente una entidad principal de IAM.
  - Si a la llamada la hace un solicitante anónimo.
- Tipo de datos: cadena (lista)
- Tipo de valor: multivalor

`aws:PrincipalServiceNamesList` es una clave de condición multivalor. Las claves de condición multivalor pueden tener varios valores en el contexto de la solicitud. Debe utilizar los operadores de servicio `ForAnyValue` o `ForAllValues` con los operadores de condición de cadena cuando utilice esta clave. Para obtener más información acerca de las claves de condición multivalor, consulte Claves de contexto multivalor.

aws:PrincipalType

Utilice esta clave para comparar el tipo de entidad principal que realiza la solicitud con el tipo de entidad principal que especifique en la política. Para obtener más información, consulte Especificación de una entidad principal. Para ver ejemplos específicos de valores de clave de `principal`, consulte Valores clave principales.

- Disponibilidad: esta clave se incluye en el contexto de solicitud de todas las solicitudes, incluidas las anónimas.
- Tipo de datos: cadena
- Tipo de valor: Valor único

aws:userid

Utilice esta clave para comparar el identificador principal del solicitante con el ID que especifique en la política. Para los usuarios de IAM, el valor del contexto de la solicitud es el ID de usuario. Para roles de IAM, este formato de valor puede variar. Para obtener más información sobre cómo aparece la información para diferentes entidades principales, consulte Especificación de una entidad principal. Para ver ejemplos específicos de valores de clave `principal`, consulte Valores clave principales.

- Disponibilidad: esta clave se incluye en el contexto de solicitud de todas las solicitudes, incluidas las anónimas.
- Tipo de datos: cadena
- Tipo de valor: valor único

aws:username

Utilice esta clave para comparar el nombre de usuario del solicitante con el nombre de usuario que especifique en la política. Para obtener más información sobre cómo aparece la información para diferentes entidades principales, consulte Especificación de una entidad principal. Para ver ejemplos específicos de valores de clave `principal`, consulte Valores clave principales.

- Disponibilidad: Esta clave siempre se incluye en el contexto de solicitud para los usuarios de IAM. Las solicitudes anónimas y las solicitudes que se realizan con los roles de IAM o Usuario raíz de la cuenta de AWS no incluyen esta clave. Las solicitudes realizadas con credenciales de IAM Identity Center no incluyen esta clave en el contexto.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Propiedades de una sesión de rol

Utilice las siguientes claves de condición para comparar las propiedades de la sesión de rol en el momento en que se generó. Estas claves de condición solo están disponibles cuando una entidad

principal con credenciales de usuario federado o de sesión de rol realiza una solicitud. Los valores de estas claves de condición están integrados en el token de sesión del rol.

Un rol es un tipo de entidad principal. También puede utilizar las claves de condición de la sección Propiedades de la entidad principal para evaluar las propiedades de un rol cuando este realiza una solicitud.

Contenido

- aws:FederatedProvider
- aws:TokenIssueTime
- aws:MultiFactorAuthAge
- aws:MultiFactorAuthPresent
- aws:Ec2InstanceSourceVpc
- aws:Ec2InstanceSourcePrivateIPv4
- aws:SourceIdentity
- ec2:RoleDelivery
- ec2:SourceInstanceArn
- glue:RoleAssumedBy
- glue:CredentialIssuingService
- lambda:SourceFunctionArn
- ssm:SourceInstanceArn
- identitystore:UserId

aws:FederatedProvider

Utilice esta clave para comparar el proveedor de identidad (IdP) emisor de la entidad principal con el IdP que especifica en la política. Esto significa que se asumió un rol de IAM mediante las operaciones de `AssumeRoleWithWebIdentity` AWS STS. Cuando se utilizan las credenciales temporales de la sesión de rol resultante para realizar una solicitud, el contexto de solicitud identifica el IdP que autenticó la identidad federada original.

- Disponibilidad: esta clave está presente cuando la entidad principal es una entidad principal de sesión de rol y esa sesión se ha emitido cuando se asumió un rol con `AssumeRoleWithWebIdentity`.

- Tipo de datos: cadena
- Tipo de valor: valor único

Por ejemplo, si el usuario se ha autenticado con Amazon Cognito, el contexto de la solicitud incluye el valor `cognito-identity.amazonaws.com`. Del mismo modo, si el usuario se ha autenticado con Login with Amazon, el contexto de la solicitud incluye el valor `www.amazon.com`.

Puede utilizar cualquier clave de condición de valor único como variable. El siguiente ejemplo de política basada en recursos utiliza la clave `aws:FederatedProvider` como una variable de la política en el ARN de un recurso. Esta política permite a cualquier entidad principal autenticada mediante un IdP obtener los objetos de un bucket de Amazon S3 con una ruta específica del proveedor de identidad emisor.

aws:TokenIssueTime

Utilice esta clave para comparar la fecha y la hora en que se emitieron las credenciales de seguridad temporales con la fecha y hora que especifique en la política.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo cuando la entidad principal utiliza credenciales temporales para realizar la solicitud. La clave no está presente en solicitudes de la AWS CLI, la API de AWS o el SDK de AWS realizadas con claves de acceso.
- Tipo de datos: fecha
- Tipo de valor: Valor único

Para obtener información sobre los servicios que admiten el uso de credenciales temporales, consulte Servicios de AWS que funcionan con IAM.

aws:MultiFactorAuthAge

Utilice esta clave para comparar el número de segundos desde que se autorizó a la entidad principal solicitante mediante MFA con el número especificado en la política. Para obtener más información acerca de MFA, consulte Uso de autenticación multifactor (MFA) en AWS.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si la entidad principal que hace la llamada se ha autenticado mediante MFA. La clave no estará presente si no se ha utilizado la MFA.
- Tipos de datos: numéricos
- Tipo de valor: Valor único

aws:MultiFactorAuthPresent

Utilice esta clave para comprobar si se utilizó la autenticación multifactor (MFA) para validar las credenciales de seguridad temporales que realizó la solicitud.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo cuando la entidad principal utiliza credenciales temporales para realizar la solicitud. La clave no está presente en la CLI de AWS, la API de AWS ni en las solicitudes del SDK de AWS realizadas con credenciales a largo plazo.
- Tipos de datos: booleano
- Tipo de valor: Valor único

Las credenciales temporales se utilizan para autenticar roles de IAM, usuarios federados, usuarios de IAM con tokens temporales de `sts:GetSessionToken` y usuarios de la AWS Management Console. Las claves de acceso de usuarios de IAM son credenciales a largo plazo, pero en algunos casos, AWS crea credenciales temporales en nombre de los usuarios de IAM para realizar operaciones. En estos casos, la clave `aws:MultiFactorAuthPresent` está presente en la solicitud y se establece en un valor de `false`. Hay dos casos frecuentes en los que puede suceder esto:

- Los usuarios de IAM de la AWS Management Console, sin saberlo, usan credenciales temporales. Los usuarios inician sesión en la consola con su nombre de usuario y contraseña, que son credenciales a largo plazo. Sin embargo, en segundo plano, la consola genera credenciales temporales en nombre del usuario.
- Si un usuario de IAM realiza una llamada a un servicio de AWS, el servicio vuelve a utilizar las credenciales del usuario para realizar otra solicitud a un servicio diferente. Por ejemplo, cuando se llama a Athena para obtener acceso a un bucket de Amazon S3 o cuando se usa AWS CloudFormation para crear una instancia de Amazon EC2. Para la próxima solicitud, AWS utiliza credenciales temporales.

Para obtener información sobre los servicios que admiten el uso de credenciales temporales, consulte Servicios de AWS que funcionan con IAM.

La clave `aws:MultiFactorAuthPresent` nunca está presente cuando se llama a una API o a un comando de la CLI con credenciales a largo plazo, como los pares de claves de acceso de usuario. Por lo tanto, le recomendamos que cuando compruebe esta clave utilice las versiones `...IfExists` de los operadores de condición.

Es importante entender que el siguiente elemento `Condition`no es una forma fiable de comprobar si una solicitud se ha autenticado mediante MFA:

```
#####   WARNING: NOT RECOMMENDED   #####
"Effect" : "Deny",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Esta combinación del efecto `Deny`, el elemento `Bool` y el valor `false` deniega las solicitudes que se pueden autenticar con MFA, pero que no han sido autenticadas de ese modo. Se aplica únicamente a las credenciales temporales que admiten el uso de MFA. Esta instrucción no deniega el acceso a las solicitudes realizadas con las credenciales a largo plazo ni a las solicitudes que se han autenticado con MFA. Utilice este ejemplo con precaución ya que su lógica es complicada y no comprueba si realmente se usó la autenticación tipo MFA.

Además no utilice la combinación del efecto `Deny`, el elemento `Null` y `true` ya que se comporta de la misma manera y la lógica es incluso más complicada.

Combinación recomendada

Le recomendamos que utilice el operador `BoolIfExists` para comprobar si se ha autenticado una solicitud con MFA.

```
"Effect" : "Deny",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Esta combinación de `Deny`, `BoolIfExists` y `false` deniega las solicitudes que no están autenticadas con MFA. En concreto, deniega las solicitudes de credenciales temporales que no incluyen MFA. También deniega las solicitudes realizadas con credenciales a largo plazo, como las operaciones de la AWS CLI o de la API de AWS realizadas con claves de acceso. El operador `*IfExists` comprueba la presencia de la clave `aws:MultiFactorAuthPresent` y si podría estar presente o no, como lo indica su existencia. Utilícelo cuando desee denegar solicitudes que no están autenticadas con MFA. Esto es más seguro, pero puede interrumpir cualquier código o script que utilice claves de acceso para obtener acceso a la AWS CLI o la API de AWS.

Combinaciones alternativas

También puede utilizar el operador `BoolIfExists` para permitir solicitudes autenticadas con MFA y solicitudes de la AWS CLI o la API de AWS realizadas con las credenciales a largo plazo.

```
"Effect" : "Allow",
"Condition" : { "BoolIfExists" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Esta condición coincide tanto si la clave existe y está presente como si la clave no existe. Esta combinación de `Allow`, `BoolIfExists` y `true` permite solicitudes autenticadas mediante MFA o solicitudes que no pueden autenticarse con MFA. Esto significa que las operaciones de la AWS CLI, la API de AWS y el SDK de AWS están permitidas cuando el solicitante utiliza sus claves de acceso a largo plazo. Esta combinación no permite solicitudes de credenciales temporales que podrían, pero no incluyen MFA.

Al crear una política con el editor visual de la consola de IAM y elegir MFA requerida, se aplica esta combinación. Esta configuración requiere MFA para acceso a la consola, pero permite el acceso mediante programación sin MFA.

También puede utilizar el operador `Bool` para permitir solicitudes programáticas y de consola solo cuando se autentique mediante MFA.

```
"Effect" : "Allow",
"Condition" : { "Bool" : { "aws:MultiFactorAuthPresent" : "true" } }
```

Esta combinación de `Allow`, `Bool` y `true` permite solo las solicitudes autenticadas con MFA. Se aplica únicamente a las credenciales temporales que admiten el uso de MFA. Esta instrucción no permite el acceso a las solicitudes realizadas con las claves de acceso a largo plazo ni a las solicitudes realizadas con credenciales temporales sin MFA.

No utilice una estructura de política similar a la siguiente para comprobar si la clave de MFA está presente:

```
#####   WARNING: USE WITH CAUTION   #####

"Effect" : "Allow",
"Condition" : { "Null" : { "aws:MultiFactorAuthPresent" : "false" } }
```

Esta combinación del efecto `Allow`, el elemento `Null` y el valor `false` permite únicamente las solicitudes que se pueden autenticar con MFA, independientemente de si la solicitud está, de hecho, autenticada. Esto permite todas las solicitudes que se realizan con credenciales temporales y deniega el acceso para las credenciales a largo plazo. Utilice este ejemplo con precaución ya que no comprueba si realmente se usó la autenticación tipo MFA.

aws:Ec2InstanceSourceVpc

Esta clave identifica la VPC a la que se entregaron las credenciales del rol de IAM de Amazon EC2. Puede usar esta clave en una política con la clave global `aws:SourceVPC` para comprobar si se realiza una llamada desde una VPC (`aws:SourceVPC`) que coincide con la VPC a la que se entregó la credencial (`aws:Ec2InstanceSourceVpc`).

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud cuando el solicitante firma las solicitudes con una credencial de rol de Amazon EC2. Se puede utilizar en políticas de IAM, políticas de control de servicios, políticas de punto de conexión de VPC y políticas de recursos.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Esta clave se puede usar con los valores del identificador de VPC, pero es más útil cuando se usa como una variable combinada con la clave de contexto `aws:SourceVpc`. La clave de contexto `aws:SourceVpc` se incluye en el contexto de la solicitud solo si el solicitante utiliza un punto de conexión de VPC para realizar la solicitud. Usar `aws:Ec2InstanceSourceVpc` con `aws:SourceVpc` permite un uso de `aws:Ec2InstanceSourceVpc` más amplio, ya que compara valores que normalmente cambian juntos.

> **Note**
>
> Esta clave de condición no está disponible en EC2-Classic.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RequireSameVPC",
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
            "aws:SourceVpc": "${aws:Ec2InstanceSourceVpc}"
        },
        "Null": {
          "ec2:SourceInstanceARN": "false"
```

```
        },
        "BoolIfExists": {
          "aws:ViaAWSService": "false"
        }
      }
    }
  ]
}
```

En el ejemplo anterior, se deniega el acceso si el valor `aws:SourceVpc` no es igual al valor `aws:Ec2InstanceSourceVpc`. La instrucción de política se limita únicamente a los roles que se utilizan como roles de instancia de Amazon EC2 al comprobar la existencia de la clave de condición `ec2:SourceInstanceARN`.

La política utiliza `aws:ViaAWSService` para permitir que AWS autorice solicitudes cuando las solicitudes se realizan en nombre de sus roles de instancia de Amazon EC2. Por ejemplo, cuando realiza una solicitud desde una instancia de Amazon EC2 a un bucket de Amazon S3 cifrado, Amazon S3 realiza una llamada a AWS KMS en su nombre. Algunas de las claves no están presentes cuando se hace la solicitud a AWS KMS.

aws:Ec2InstanceSourcePrivateIPv4

Esta clave identifica la dirección IPv4 privada de la interfaz de red elástica principal a la que se entregaron las credenciales del rol de IAM de Amazon EC2. Debe usar esta clave de condición con su clave complementaria `aws:Ec2InstanceSourceVpc` para garantizar que tiene una combinación única global de ID de VPC e IP privada de origen. Utilice esta clave con `aws:Ec2InstanceSourceVpc` para asegurarse de que la solicitud se haya realizado desde la misma dirección IP privada a la que se entregaron las credenciales.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud cuando el solicitante firma las solicitudes con una credencial de rol de Amazon EC2. Se puede utilizar en políticas de IAM, políticas de control de servicios, políticas de punto de conexión de VPC y políticas de recursos.
- Tipo de datos: dirección IP
- Tipo de valor: Valor único

> **Important**
>
> Esta clave no debe usarse sola en una instrucción `Allow`. Las direcciones IP privadas, por definición, no son únicas a nivel global. Debe utilizar

la clave `aws:Ec2InstanceSourceVpc` cada vez que utilice la clave `aws:Ec2InstanceSourcePrivateIPv4` para especificar la VPC desde la que se pueden usar las credenciales de la instancia de Amazon EC2.

> **Note**
>
> Esta clave de condición no está disponible en EC2-Classic.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourceVpc": "${aws:SourceVpc}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":  "*",
            "Resource": "*",
            "Condition": {
                "StringNotEquals": {
                    "aws:Ec2InstanceSourcePrivateIPv4": "${aws:VpcSourceIp}"
                },
                "Null": {
                    "ec2:SourceInstanceARN": "false"
                },
                "BoolIfExists": {
                    "aws:ViaAWSService": "false"
```

```
                }
            }
        }
    ]
}
```

aws:SourceIdentity

Utilice esta clave para comparar la identidad de origen establecida por la entidad principal con la identidad de origen que usted especifique en la política.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud después de establecer una identidad de origen cuando se asume un rol utilizando cualquier comando de la CLI de assume-role de AWS STS, o una operación de la API de `AssumeRole` de AWS STS.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Puede utilizar esta clave en una política para permitir acciones en AWS por entidades que han establecido una identidad de origen al asumir un rol. La actividad de la identidad de origen especificada del rol aparece en AWS CloudTrail. Esto facilita a los administradores determinar quién o qué ha realizado acciones con un rol en AWS.

A diferencia de `sts:RoleSessionName`, después de establecer la identidad de origen, el valor no se puede cambiar. Está presente en el contexto de la solicitud de todas las acciones tomadas por el rol. El valor persiste en sesiones de rol posteriores cuando se utilizan las credenciales de sesión para asumir otro rol. Asumir un rol de otro se llama encadenamiento de roles.

La clave `sts:SourceIdentity` está presente en la solicitud cuando la entidad principal establece inicialmente una identidad de origen mientras asume un rol utilizando cualquier comando assume-role CLI AWS STS, o una operación AWS STS de API `AssumeRole`. La clave `aws:SourceIdentity` presente en la solicitud de cualquier acción que se realice con una sesión de rol que tenga un conjunto de identidad de origen.

La siguiente política de confianza de rol de `CriticalRole` en la cuenta `111122223333` contiene una condición para `aws:SourceIdentity` que impide que una entidad principal sin una identidad de origen establecida para Saanvi o Diego asuma el rol.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "AssumeRoleIfSourceIdentity",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::123456789012:role/CriticalRole"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringLike": {
                    "aws:SourceIdentity": ["Saanvi","Diego"]
                }
            }
        }
    ]
}
```

Para obtener más información acerca del uso de información de identidad de origen, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

ec2:RoleDelivery

Utilice esta clave para comparar la versión del servicio de metadatos de instancia en la solicitud firmada con las credenciales del rol de IAM para Amazon EC2. El servicio de metadatos de la instancia distingue entre solicitudes IMDSv1 y IMDSv2 en función de si, en cualquier solicitud, los encabezados `PUT` o `GET`, que son exclusivos de IMDSv2, están presentes en dicha solicitud.

- Disponibilidad: esta clave forma parte del contexto de la solicitud cada vez que una instancia de Amazon EC2 crea la sesión de rol.
- Tipos de datos: numéricos
- Tipo de valor: Valor único
- Valores de ejemplo: 1.0, 2.0

Puede configurar el servicio de metadatos de instancia (IMDS) en cada instancia para que el código local o los usuarios deban usar IMDSv2. Si especifica que debe usarse IMDSv2, IMDSv1 dejará de funcionar.

- Servicio de metadatos de instancia, versión 1 (IMDSv1): un método de solicitud y respuesta
- Servicio de metadatos de instancia, versión 2 (IMDSv2): un método orientado a la sesión

Para obtener información sobre cómo configurar su instancia para usar IMDSv2, consulte Configurar las opciones de metadatos de la instancia.

En el siguiente ejemplo, se deniega el acceso si el valor ec2:RoleDelivery en el contexto de la solicitud es 1.0 (IMDSv1). Esta declaración de política puede aplicarse de manera general porque, si la solicitud no está firmada por las credenciales de rol de Amazon EC2, no tiene efecto.

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Para obtener más información, consulte Ejemplos de políticas para trabajar con metadatos de instancia.

ec2:SourceInstanceArn

Utilice esta clave para comparar el ARN de la instancia en la cual se generó la sesión del rol.

- Disponibilidad: esta clave forma parte del contexto de la solicitud cada vez que una instancia de Amazon EC2 crea la sesión de rol.
- Tipo de datos: ARN
- Tipo de valor: Valor único
- Valor de ejemplo: arn:aws:ec2:us-west-2:111111111111:instance/instance-id

Para ver ejemplos de políticas, consulte Permitir que una instancia específica vea recursos en otros servicios de AWS.

glue:RoleAssumedBy

El servicio de AWS Glue establece esta clave de condición para cada solicitud de API de AWS donde AWS Glue realiza una solicitud mediante un rol de servicio en nombre del cliente (no a través de una tarea o un punto de conexión de desarrollador, sino directamente a través del servicio de AWS Glue). Utilice esta clave para verificar si una llamada a un recurso de AWS proviene del servicio de AWS Glue.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud cuando AWS Glue realiza una solicitud mediante un rol de servicio en nombre del cliente.
- Tipo de datos: cadena
- Tipo de valor: Valor único
- Ejemplo de valor: esta clave siempre se establece como `glue.amazonaws.com`.

En el siguiente ejemplo se añade una condición para permitir que el servicio de AWS Glue obtenga un objeto de un bucket de Amazon S3.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:RoleAssumedBy": "glue.amazonaws.com"
        }
    }
}
```

glue:CredentialIssuingService

El servicio de AWS Glue establece esta clave para cada solicitud de API de AWS que utiliza un rol de servicio que proviene de una tarea o un punto de conexión de desarrollador. Utilice esta clave para verificar si una llamada a un recurso de AWS provino de una tarea de AWS Glue o de un punto de conexión de desarrollador.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud cuando AWS Glue realiza una solicitud que proviene de una tarea o un punto de conexión de desarrollador.
- Tipo de datos: cadena

- Tipo de valor: Valor único
- Ejemplo de valor: esta clave siempre se establece como `glue.amazonaws.com`.

En el siguiente ejemplo, se agrega una condición que está asociada a un rol de IAM que utiliza una tarea de AWS Glue. Esto garantiza que se permitan o denieguen determinadas acciones en función de si la sesión de rol se utiliza para el entorno de tiempo de ejecución de un trabajo de AWS Glue.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

lambda:SourceFunctionArn

Utilice esta clave para identificar el ARN de la función de Lambda a la que se entregaron las credenciales del rol de IAM. El servicio de Lambda establece esta clave para cada solicitud de API de AWS que provenga del entorno de ejecución de la función. Utilice esta clave para verificar si una llamada a un recurso de AWS proviene del código de una función de Lambda específica. Lambda también establece esta clave para algunas solicitudes que provienen de fuera del entorno de ejecución, como escribir registros en CloudWatch y enviar seguimientos a X-Ray.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud cuando se invoca el código de la función de Lambda.
- Tipo de datos: ARN
- Tipo de valor: Valor único
- Valor de ejemplo: arn:aws:lambda:us-east-1:123456789012:function:TestFunction

En el siguiente ejemplo se muestra cómo se permite que una función de Lambda específica tenga acceso `s3:PutObject` al bucket especificado.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition": {
                "ArnEquals": {
                    "lambda:SourceFunctionArn": "arn:aws:lambda:us-
east-1:123456789012:function:source_lambda"
                }
            }
        }
    ]
}
```

Para obtener más información, consulte Uso de las credenciales del entorno de ejecución de Lambda en la Guía para desarrolladores de AWS Lambda.

ssm:SourceInstanceArn

Utilice esta clave para identificar el ARN de la instancia administrada por AWS Systems Manager a la que se entregaron las credenciales del rol de IAM. Esta clave de condición no está presente cuando la solicitud proviene de una instancia administrada con un rol de IAM asociado a un perfil de instancia de Amazon EC2.

- Disponibilidad: esta clave aparece en el contexto de la solicitud siempre que se entregan las credenciales de rol a una instancia administrada por AWS Systems Manager.
- Tipo de datos: ARN
- Tipo de valor: Valor único
- Valor de ejemplo: arn:aws:ec2:us-west-2:111111111111:instance/instance-id

identitystore:UserId

Utilice esta clave para comparar la identidad del personal del IAM Identity Center en la solicitud firmada con la identidad especificada en la política.

- Disponibilidad: esta clave se incluye cuando la persona que llama a la solicitud es un usuario del IAM Identity Center.
- Tipo de datos: cadena

- Tipo de valor: Valor único
- Valor de ejemplo: 94482488-3041-7026-18f3-be45837cd0e4

Puede encontrar el ID de usuario de un usuario del IAM Identity Center al realizar una solicitud a la API GetUserID mediante la AWS CLI, la API de AWS o el SDK de AWS.

Propiedades de la red

Utilice las siguientes claves de condición para comparar los detalles de la red desde la que se originó o por la que pasó la solicitud con las propiedades de red que especifique en la política.

Contenido

- aws:SourceIp
- aws:SourceVpc
- aws:SourceVpce
- aws:VpcSourceIp

aws:SourceIp

Utilice esta clave para comparar la dirección IP del solicitante con la dirección IP que especifique en la política. La clave de condición `aws:SourceIp` solo puede utilizarse para rangos de direcciones IP públicas.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud, excepto cuando el solicitante utiliza un punto de enlace de la VPC para realizar la solicitud.
- Tipo de datos: dirección IP
- Tipo de valor: Valor único

La clave de condición `aws:SourceIp` se puede utilizar en una política para permitir que las entidades principales realicen solicitudes solo desde un rango de IP especificado.

> **Note**
>
> `aws:SourceIp` admite tanto las direcciones IPv4 como IPv6 o un rango de direcciones IP. Para obtener una lista de los Servicios de AWS que admiten IPv6, consulte los Servicios de AWS que admiten IPv6 en la Guía del usuario de Amazon VPC.

Por ejemplo, puede asociar la siguiente política basada en identidades a un rol de IAM. Esta política permite al usuario colocar objetos en el bucket de Amazon S3 *DOC-EXAMPLE-BUCKET3* si realiza la llamada desde el intervalo de dirección IPv4 especificado. Esta política también permite que un servicio de AWS que utilice Sesiones de acceso directo realice esta operación en su nombre.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "203.0.113.0/24"
                }
            }
        }
    ]
}
```

Si necesita restringir el acceso desde redes que admiten direcciones IPv4 e IPv6, puede incluir las direcciones IPv4 e IPv6 o los intervalos de direcciones IP en la condición de la política de IAM. La siguiente política basada en identidades permitirá al usuario colocar objetos en el bucket de Amazon S3 *DOC-EXAMPLE-BUCKET3* si el usuario realiza la llamada desde intervalos de direcciones IPv4 o IPv6 especificados. Antes de incluir los intervalos de direcciones IPv6 en su política de IAM, compruebe que el Servicio de AWS con el que está trabajando admita IPv6. Para obtener una lista de los Servicios de AWS que admiten IPv6, consulte los Servicios de AWS que admiten IPv6 en la Guía del usuario de Amazon VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET3/*",
            "Condition": {
                "IpAddress": {
```

```
                    "aws:SourceIp": [
                        "203.0.113.0/24",
                        "2001:DB8:1234:5678::/64"
                    ]
                }
            }
        }
    ]
}
```

Si la solicitud proviene de un host que utiliza un punto de enlace de Amazon VPC, entonces la clave `aws:SourceIp` no estará disponible. En su lugar, debe utilizar una clave específica de la VPC, como aws:VpcSourceIp. Para obtener más información acerca del uso de puntos de conexión de VPC, consulte Administración de identidades y accesos para puntos de conexión de VPC y servicios de punto de conexión de VPC en la Guía de AWS PrivateLink.

aws:SourceVpc

Utilice esta clave para comprobar si la solicitud pasa por la VPC a la que está asociado el punto de conexión de VPC. En una política, puede utilizar esta clave para permitir el acceso solo a una VPC específica. Para obtener más información, consulte Restricciones de acceso a una VPC específica en la Guía del usuario de Amazon Simple Storage Service.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si el solicitante utiliza un punto de conexión de VPC para realizar la solicitud.
- Tipo de datos: cadena
- Tipo de valor: Valor único

aws:SourceVpce

Utilice esta clave para comparar el identificador de punto de enlace de la VPC de la solicitud con el ID de punto de enlace que especifique en la política. En una política, puede utilizar esta clave para restringir el acceso a un punto de enlace de la VPC específico. Para obtener más información, consulte Restricciones de acceso a un punto de conexión de VPC específico en la Guía del usuario de Amazon Simple Storage Service.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si el solicitante utiliza un punto de conexión de VPC para realizar la solicitud.
- Tipo de datos: cadena

- Tipo de valor: Valor único

aws:VpcSourceIp

Utilice esta clave para comparar la dirección IP desde la que se realizó una solicitud con la dirección IP que ha especificado en la política. En una política, la clave coincide solo si la solicitud proviene de la dirección IP especificada y pasa a través de un punto de enlace de la VPC.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo si la solicitud se realiza mediante un punto de enlace de la VPC.
- Tipo de datos: dirección IP
- Tipo de valor: Valor único

Para obtener más información, consulte Controlar el acceso a servicios con puntos de conexión de VPC en la Guía del usuario de Amazon VPC.

> **Note**
>
> `aws:VpcSourceIp` admite tanto las direcciones IPv4 como IPv6 o un rango de direcciones IP. Para obtener una lista de los Servicios de AWS que admiten IPv6, consulte los Servicios de AWS que admiten IPv6 en la Guía del usuario de Amazon VPC.

Propiedades del recurso

Utilice las siguientes claves de condición para comparar detalles sobre el recurso que es el objetivo de la solicitud con las propiedades del recurso que especifique en la política.

Contenido

- aws:ResourceAccount
- aws:ResourceOrgPaths
- aws:ResourceOrgID
- aws:ResourceTag/tag-key

aws:ResourceAccount

Utilice esta clave para comparar el ID de la Cuenta de AWS del propietario del recurso solicitado con la cuenta del recurso de la política. A continuación, puede permitir o denegar el acceso a ese recurso en función de la cuenta propietaria del recurso.

- Disponibilidad: Esta clave siempre se incluye en el contexto de solicitud para la mayoría de servicios. Las siguientes acciones no admiten esta clave:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Elastic Block Store: Todas las acciones
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyFpgaImage`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
  - Amazon EventBridge
    - `events:PutEvents`: `PutEvents` de EventBridge llama a un bus de eventos de otra cuenta, si ese bus de eventos se configuró como un destino de EventBridge para varias cuentas antes del 2 de marzo de 2023. Para obtener más información, consulte Conceder permisos para permitir eventos de otras cuentas de AWS en la Guía del usuario de Amazon EventBridge.
  - Amazon GuardDuty

    - `guardduty:AcceptAdministratorInvitation`
  - Amazon Macie
    - `macie2:AcceptInvitation`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `securityhub:AcceptAdministratorInvitation`
  - Amazon WorkSpaces
    - `workspaces:DescribeWorkspaceImages`
- Tipo de datos: cadena
- Tipo de valor: Valor único

> **Note**
>
> Para obtener más información sobre las acciones no compatibles anteriores, consulte el repositorio de ejemplos de políticas de perímetro de datos.

Esta clave es igual al ID de Cuenta de AWS para la cuenta con los recursos evaluados en la solicitud.

Para la mayoría de los recursos de su cuenta, el ARN contiene el ID de cuenta del propietario de ese recurso. Para ciertos recursos, como los buckets de Amazon S3, el ARN del recurso no incluye el ID de cuenta. Los dos ejemplos siguientes muestran la diferencia entre un recurso con un ID de cuenta en el ARN y un ARN de Amazon S3 sin un ID de cuenta:

- `arn:aws:iam::123456789012:role/AWSExampleRole`: Rol de IAM creado y de propiedad dentro de la cuenta 123456789012.
- `arn:aws:s3:::DOC-EXAMPLE-BUCKET2`: Bucket de Amazon S3 creado y de propiedad dentro de la cuenta 111122223333, que no aparecen en el ARN.

Use la consola, la API o la CLI de AWS, para encontrar todos los recursos y los ARN correspondientes.

Tiene que escribir una política que deniega los permisos a los recursos en función del ID de cuenta del propietario del recurso. Por ejemplo, la siguiente política basada en identidad deniega el acceso al recurso especificado si este no pertenece a la cuenta especificada.

Para utilizar esta política, sustituya el texto del marcador de posición en cursiva por la información de la cuenta.

> **Important**
>
> Esta política no permite ninguna acción. En su lugar, utiliza el efecto `Deny` que deniega explícitamente el acceso a todos los recursos enumerados en la declaración que no pertenecen a la cuenta enumerada. Utilice esta política en combinación con otras políticas que permiten acceder a recursos específicos.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:account:*"
      ],
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "account"
          ]
        }
      }
    }
  ]
}
```

Esta política deniega el acceso a todos los recursos para un servicio específico de AWS a menos que la Cuenta de AWS especificada sea propietaria del recurso.

> **Note**
>
> Algunos Servicios de AWS requieren acceso a recursos que son propiedad de AWS que están alojados en otra Cuenta de AWS. El uso de `aws:ResourceAccount` en sus políticas basadas en identidad puede afectar a la capacidad de su identidad para acceder a estos recursos.

Ciertos servicios de AWS, como AWS Data Exchange, confían en el acceso a recursos fuera de sus Cuentas de AWS para operaciones normales. Si usa el elemento `aws:ResourceAccount` en sus políticas, incluya declaraciones adicionales para crear exenciones para esos servicios. Las políticas de ejemplo AWS: denegar el acceso a los recursos de Amazon S3 fuera de su cuenta, excepto AWS Data Exchange demuestran cómo denegar el acceso en función de la cuenta de recursos y, al mismo tiempo, definir excepciones para los recursos propiedad del servicio.

Utilice este ejemplo de políticas como plantillas para crear sus propias políticas personalizadas. Consulte la documentación de su servicio para obtener más información.

aws:ResourceOrgPaths

Utilice esta clave para comparar la ruta de AWS Organizations del recurso al que se accede en la política. En una política, esta clave de condición garantiza que el solicitante sea un miembro de la cuenta dentro de la raíz de la organización o unidades organizativas especificadas en AWS Organizations. Una ruta de AWS Organizations es una representación de texto de la estructura de una entidad de Organizations. Para obtener más información sobre el uso y la comprensión de las rutas, consulte Comprender la ruta de la entidad AWS Organizations

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si la cuenta propietaria del recurso es miembro de una organización. Esta clave de condición global no admite las siguientes acciones:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Elastic Block Store: Todas las acciones
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`

  - `ec2:CopyFpgaImage`
  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`: `PutEvents` de EventBridge llama a un bus de eventos de otra cuenta, si ese bus de eventos se configuró como un destino de EventBridge para varias cuentas antes del 2 de marzo de 2023. Para obtener más información, consulte Conceder permisos para permitir eventos de otras cuentas de AWS en la Guía del usuario de Amazon EventBridge.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`
- AWS Security Hub
  - `securityhub:AcceptAdministratorInvitation`
- Amazon WorkSpaces
  - `workspaces:DescribeWorkspaceImages`

- Tipo de datos: cadena (lista)
- Tipo de valor: Multivalor

> **Note**
>
> Para obtener más información sobre las acciones no compatibles anteriores, consulte el repositorio de ejemplos de políticas de perímetro de datos.

`aws:ResourceOrgPaths` es una clave de condición multivalor. Las claves de condición multivalor pueden tener varios valores en el contexto de la solicitud. Debe utilizar los operadores de servicio `ForAnyValue` o `ForAllValues` con los operadores de condición de cadena cuando utilice esta clave. Para obtener más información acerca de las claves de condición multivalor, consulte Claves de contexto multivalor.

Por ejemplo, la siguiente condición devuelve `True` para recursos que pertenecen a la organización `o-a1b2c3d4e5`. Cuando se incluye un comodín, se debe utilizar el operador de condición StringLike.

```
"Condition": {
      "ForAnyValue:StringLike": {
            "aws:ResourceOrgPaths":["o-a1b2c3d4e5/*"]
   }
}
```

Se devuelve la siguiente condición `True` para los recursos con el ID de la unidad organizativa `ou-ab12-11111111`. Coincidirá con los recursos que son propiedad de cuentas asociadas a la unidad organizativa ou-ab12-11111111 o a cualquiera de las unidades organizativas secundarias.

```
"Condition": { "ForAnyValue:StringLike" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/*"]
}}
```

Se devuelve la siguiente condición `True` para los recursos que son propiedad de las cuentas asociadas directamente al ID de la unidad organizativa `ou-ab12-22222222`, pero no a las unidades organizativas secundarias. El siguiente ejemplo utiliza el operador de condición StringEquals para especificar el requisito de coincidencia exacta para el ID de la unidad organizativa y no una coincidencia de comodín.

```
"Condition": { "ForAnyValue:StringEquals" : {
     "aws:ResourceOrgPaths":["o-a1b2c3d4e5/r-ab12/ou-ab12-11111111/ou-ab12-22222222/"]
}}
```

> **Note**
>
> Algunos Servicios de AWS requieren acceso a recursos que son propiedad de AWS que están alojados en otra Cuenta de AWS. El uso de `aws:ResourceOrgPaths` en sus políticas basadas en identidad puede afectar a la capacidad de su identidad para acceder a estos recursos.

Ciertos servicios de AWS, como AWS Data Exchange, confían en el acceso a recursos fuera de sus Cuentas de AWS para operaciones normales. Si usa la clave `aws:ResourceOrgPaths` en sus políticas, incluya declaraciones adicionales para crear exenciones para esos servicios. Las políticas de ejemplo AWS: denegar el acceso a los recursos de Amazon S3 fuera de su cuenta, excepto AWS Data Exchange demuestran cómo denegar el acceso en función de la cuenta de recursos y, al mismo tiempo, definir excepciones para los recursos propiedad del servicio. Puede crear una política similar para restringir el acceso a los recursos dentro de una unidad organizativa (OU) mediante la clave `aws:ResourceOrgPaths` y, al mismo tiempo, contabilizar los recursos propiedad del servicio.

Utilice este ejemplo de políticas como plantillas para crear sus propias políticas personalizadas. Consulte la documentación de su servicio para obtener más información..

aws:ResourceOrgID

Utilice esta clave para comparar el identificador de la organización de AWS Organizations a la que pertenece el recurso solicitado con el identificador especificado en la política.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo si la entidad principal es miembro de una organización. Esta clave de condición global no admite las siguientes acciones:
  - AWS Audit Manager
    - `auditmanager:UpdateAssessmentFrameworkShare`
  - Amazon Elastic Block Store: Todas las acciones
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`

  - `ec2:AcceptVpcPeeringConnection`
  - `ec2:CopyFpgaImage`
  - `ec2:CopyImage`
  - `ec2:CopySnapshot`
  - `ec2:CreateTransitGatewayPeeringAttachment`
  - `ec2:CreateVolume`
  - `ec2:CreateVpcEndpoint`
  - `ec2:CreateVpcPeeringConnection`
  - `ec2:DeleteTransitGatewayPeeringAttachment`
  - `ec2:DeleteVpcPeeringConnection`
  - `ec2:RejectTransitGatewayPeeringAttachment`
  - `ec2:RejectVpcEndpointConnections`
  - `ec2:RejectVpcPeeringConnection`
- Amazon EventBridge
  - `events:PutEvents`: `PutEvents` de EventBridge llama a un bus de eventos de otra cuenta, si ese bus de eventos se configuró como un destino de EventBridge para varias cuentas antes del 2 de marzo de 2023. Para obtener más información, consulte Conceder permisos para permitir eventos de otras cuentas de AWS en la Guía del usuario de Amazon EventBridge.
- Amazon GuardDuty
  - `guardduty:AcceptAdministratorInvitation`
- Amazon Macie
  - `macie2:AcceptInvitation`
- Amazon Route 53
  - `route53:AssociateVpcWithHostedZone`
  - `route53:CreateVPCAssociationAuthorization`
  - `route53:DeleteVPCAssociationAuthorization`
  - `route53:DisassociateVPCFromHostedZone`
  - `route53:ListHostedZonesByVPC`
- AWS Security Hub
  - `securityhub:AcceptAdministratorInvitation`
- Amazon WorkSpaces

    - `workspaces:DescribeWorkspaceImages`
- Tipo de datos: cadena
- Tipo de valor: Valor único

> **Note**
>
> Para obtener más información sobre las acciones no compatibles anteriores, consulte el repositorio de ejemplos de políticas de perímetro de datos.

Esta clave global devuelve el ID de organización de recursos de una solicitud determinada. Le permite crear reglas que se aplican a todos los recursos de una organización que se especifican en el elemento `Resource` de una política basada en identidad. Puede especificar el ID de organización en el elemento de condición. Al agregar y quitar cuentas, las políticas que incluyan la clave `aws:ResourceOrgID` tendrán automáticamente las cuentas correctas y no requerirán una actualización manual.

Por ejemplo, la siguiente política impide que la entidad principal agregue objetos al recurso `policy-genius-dev` a menos que el recurso de Amazon S3 pertenezca a la misma organización que la entidad principal que hace la solicitud.

> **Important**
>
> Esta política no permite ninguna acción. En su lugar, utiliza el efecto `Deny` que deniega explícitamente el acceso a todos los recursos enumerados en la declaración que no pertenecen a la cuenta enumerada. Utilice esta política en combinación con otras políticas que permiten acceder a recursos específicos.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
    "Resource": "arn:partition:s3:::policy-genius-dev/*",
    "Condition": {
```

```
      "StringNotEquals": {
        "aws:ResourceOrgID": "${aws:PrincipalOrgID}"
      }
    }
  }
}
```

> **Note**
>
> Algunos Servicios de AWS requieren acceso a recursos que son propiedad de AWS que están alojados en otra Cuenta de AWS. El uso de `aws:ResourceOrgID` en sus políticas basadas en identidad puede afectar a la capacidad de su identidad para acceder a estos recursos.

Ciertos servicios de AWS, como AWS Data Exchange, confían en el acceso a recursos fuera de sus Cuentas de AWS para operaciones normales. Si usa la clave `aws:ResourceOrgID` en sus políticas, incluya declaraciones adicionales para crear exenciones para esos servicios. Las políticas de ejemplo AWS: denegar el acceso a los recursos de Amazon S3 fuera de su cuenta, excepto AWS Data Exchange demuestran cómo denegar el acceso en función de la cuenta de recursos y, al mismo tiempo, definir excepciones para los recursos propiedad del servicio. Puede crear una política similar para restringir el acceso a los recursos dentro de una organización mediante la clave `aws:ResourceOrgID` y, al mismo tiempo, contabilizar los recursos propiedad del servicio.

Utilice este ejemplo de políticas como plantillas para crear sus propias políticas personalizadas. Consulte la documentación de su servicio para obtener más información..

En el siguiente video tiene más información acerca de cómo utilizar la clave de condición `aws:ResourceOrgID` en una política.

Asegúrese de que las identidades y las redes solo se puedan usar para acceder a recursos confiables.

aws:ResourceTag/tag-key

Utilice esta clave para comparar el par clave-valor de etiqueta que especifique en la política con el par clave-valor asociado al recurso. Por ejemplo, puede requerir que el acceso a un recurso solo se permita si el recurso tiene la clave de etiqueta `"Dept"` adjunta con el valor `"Marketing"`. Para obtener más información, consulte Control del acceso a los recursos de AWS.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud cuando el recurso solicitado ya tiene etiquetas asociadas o en solicitudes que crean un recurso con una etiqueta asociada. Esta clave se devuelve solo para los recursos que admiten autorización basada en etiquetas. Hay una clave de contexto para cada par clave-valor de etiqueta.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Esta clave de contexto tiene el formato `"aws:ResourceTag/tag-key":"tag-value"`, donde *`tag-key`* y *`tag-value`* son un par formado por una clave y un valor. Las claves y los valores de las etiquetas no distinguen entre mayúsculas y minúsculas. Esto significa que si especifica `"aws:ResourceTag/TagKey1": "Value1"` en el elemento de condición de su política, la condición coincidirá con una clave de etiqueta de recurso denominada `TagKey1` o `tagkey1`, pero no con ambas.

Para obtener ejemplos del uso de la clave de `aws:ResourceTag` para controlar el acceso a los recursos de IAM, consulte Control del acceso a los recursos de AWS.

Para obtener ejemplos del uso de la clave de `aws:ResourceTag` para controlar el acceso a otros recursos de AWS, consulte Control de acceso a los recursos de AWS mediante etiquetas.

Para obtener un tutorial sobre el uso de la clave de condición de `aws:ResourceTag` para el control de acceso basado en atributos (ABAC), consulte Tutorial de IAM: definición de permisos para acceder a los recursos de AWS en función de etiquetas.

Propiedades de la solicitud

Utilice las siguientes claves de condición para comparar los detalles sobre la solicitud y su contenido con las propiedades de solicitud especificadas en la política.

Contenido

- aws:CalledVia
- aws:CalledViaFirst
- aws:CalledViaLast
- aws:ViaAWSService
- aws:CurrentTime
- aws:EpochTime
- aws:referer

- aws:RequestedRegion
- aws:RequestTag/clave-etiqueta
- aws:TagKeys
- aws:SecureTransport
- aws:SourceArn
- aws:SourceAccount
- aws:SourceOrgPaths
- aws:SourceOrgID
- aws:UserAgent

aws:CalledVia

Utilice esta clave para comparar los servicios de la política con los servicios que realizaron solicitudes en nombre de la entidad principal de IAM (usuario o rol). Cuando una entidad principal realiza una solicitud a un servicio de AWS, ese servicio puede utilizar las credenciales de la entidad principal para realizar solicitudes posteriores a otros servicios. La clave `aws:CalledVia` contiene una lista ordenada de cada servicio de la cadena que realizó solicitudes en nombre de la entidad principal.

Por ejemplo, puede utilizar AWS CloudFormation para leer y escribir desde una tabla de Amazon DynamoDB. DynamoDB utiliza el cifrado suministrado por AWS Key Management Service (AWS KMS).

- Disponibilidad: Esta clave está presente en la solicitud cuando un servicio que admite `aws:CalledVia` utiliza las credenciales de una entidad principal de IAM para realizar una solicitud a otro servicio. Esta clave no está presente si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal. Esta clave tampoco está presente cuando la entidad principal realiza la llamada directamente.
- Tipo de datos: cadena (lista)
- Tipo de valor: Multivalor

Para utilizar la clave de condición `aws:CalledVia` en una política, debe proporcionar las entidades principales del servicio para permitir o denegar las solicitudes de servicio de AWS. AWS admite el uso de las siguientes entidades principales del servicio con `aws:CalledVia`.

Entidad principal de servicio
aoss.amazonaws.com
athena.amazonaws.com
backup.amazonaws.com
cloud9.amazonaws.com
cloudformation.amazonaws.com
databrew.amazonaws.com
dataexchange.amazonaws.com
dynamodb.amazonaws.com
imagebuilder.amazonaws.com
kms.amazonaws.com
mgn.amazonaws.com
nimble.amazonaws.com
omics.amazonaws.com
ram.amazonaws.com
robomaker.amazonaws.com
servicecatalog-appregistry.amazonaws.com
sqlworkbench.amazonaws.com
ssm-guiconnect.amazonaws.com

Para permitir o denegar el acceso cuando cualquier servicio realiza una solicitud utilizando las credenciales de la entidad principal, utilice la clave de condición `aws:ViaAWSService`. Esa clave de condición admite los servicios de AWS.

La clave `aws:CalledVia` es una clave multivalor. Sin embargo, no se puede aplicar el orden utilizando esta clave en una condición. En el ejemplo anterior, User 1 realiza una solicitud a AWS CloudFormation, que llama a DynamoDB, que a su vez llama a AWS KMS. Se trata de tres solicitudes distintas. La llamada final a AWS KMS la realiza User 1 a través de AWS CloudFormation y después a través de DynamoDB.

En este caso, la clave `aws:CalledVia` en el contexto de solicitud incluye `cloudformation.amazonaws.com` y `dynamodb.amazonaws.com`, en ese orden. Si solo le preocupa que la llamada se realice a través de DynamoDB en algún lugar de la cadena de solicitudes, puede utilizar esta clave de condición en su política.

Por ejemplo, la siguiente política permite administrar la clave AWS KMS denominada `my-example-key`, pero solo si DynamoDB es uno de los servicios que realiza la solicitud. El operador de la condición `ForAnyValue:StringEquals` garantiza que DynamoDB sea uno de los servicios que realiza la llamada. Si la entidad principal realiza la llamada a AWS KMS directamente, la condición devuelve `false` y la política no permite la solicitud.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": ["dynamodb.amazonaws.com"]
                }
            }
        }
    ]
}
```

Si desea especificar el servicio que realiza la primera o la última llamada de la cadena, puede utilizar las claves `aws:CalledViaFirst` y `aws:CalledViaLast`. Por ejemplo, la siguiente política permite administrar la clave denominada `my-example-key` en AWS KMS. Estas operaciones de AWS KMS solo se permiten si se incluyeron varias solicitudes en la cadena. La primera solicitud debe hacerse a través de AWS CloudFormation y la última, a través de DynamoDB. Si otros servicios realizan solicitudes en el centro de la cadena, la operación sigue estando permitida.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region:111122223333:key/my-example-key",
            "Condition": {
                "StringEquals": {
                    "aws:CalledViaFirst": "cloudformation.amazonaws.com",
                    "aws:CalledViaLast": "dynamodb.amazonaws.com"
                }
            }
        }
    ]
}
```

Las claves `aws:CalledViaFirst` y `aws:CalledViaLast` están presentes en la solicitud cuando un servicio utiliza las credenciales de una entidad principal de IAM para llamar a otro servicio. Indican los primeros y últimos servicios que realizaron llamadas en la cadena de solicitudes. Suponga, por ejemplo, que AWS CloudFormation llama a otro servicio denominado `X Service`, que llama a DynamoDB y que luego llama a AWS KMS. La llamada final a AWS KMS la realiza `User 1` a través de AWS CloudFormation, luego a través de `X Service` y finalmente a través de DynamoDB. Primero se llamó a través de AWS CloudFormation y la última llamada se realizó a través de DynamoDB.

aws:CalledViaFirst

Utilice esta clave para comparar los servicios de la política con el primer servicio que realizó una solicitud en nombre de la entidad principal de IAM (usuario o rol). Para obtener más información, consulte `aws:CalledVia`.

- Disponibilidad: esta clave está presente en la solicitud cuando un servicio utiliza las credenciales de una entidad principal de IAM para realizar al menos otra solicitud a un servicio diferente. Esta clave no está presente si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal. Esta clave tampoco está presente cuando la entidad principal realiza la llamada directamente.
- Tipo de datos: cadena
- Tipo de valor: Valor único

aws:CalledViaLast

Utilice esta clave para comparar los servicios de la política con el último servicio que realizó una solicitud en nombre de la entidad principal de IAM (usuario o rol). Para obtener más información, consulte `aws:CalledVia`.

- Disponibilidad: esta clave está presente en la solicitud cuando un servicio utiliza las credenciales de una entidad principal de IAM para realizar al menos otra solicitud a un servicio diferente. Esta clave no está presente si el servicio utiliza un rol de servicio o un rol vinculado al servicio para realizar una llamada en nombre de la entidad principal. Esta clave tampoco está presente cuando la entidad principal realiza la llamada directamente.
- Tipo de datos: cadena
- Tipo de valor: valor único

aws:ViaAWSService

Utilice esta clave para comprobar si un servicio de AWS realiza una solicitud a otro servicio en su nombre.

La clave de contexto de la solicitud devuelve `true` cuando un servicio utiliza las credenciales de una entidad principal de IAM para realizar una solicitud en nombre de la entidad principal. La clave de contexto devuelve `false` si el servicio utiliza un rol de servicio o un rol vinculado al servicio para

realizar una llamada en nombre de la entidad principal. La clave de contexto de la solicitud también devuelve `false` cuando la entidad principal realiza la llamada directamente.

- Disponibilidad: esta clave siempre se incluye en el contexto de la solicitud.
- Tipos de datos: booleano
- Tipo de valor: Valor único

Puede utilizar esta clave de condición para permitir o denegar el acceso en función de si un servicio realizó la solicitud.

aws:CurrentTime

Utilice esta clave para comparar la fecha y la hora de la solicitud con la fecha y la hora que especifique en la política. Para ver una política de ejemplo que utilice este clave de condición, consulte AWS: permite el acceso en función de la fecha y la hora.

- Disponibilidad: Esta clave siempre se incluye en el contexto de la solicitud.
- Tipo de datos: fecha
- Tipo de valor: valor único

aws:EpochTime

Utilice esta clave para comparar la fecha y hora de la solicitud en formato de tiempo epoch o Unix con el valor que especifique en la política. Esta clave también acepta el número de segundos desde el 1 de enero de 1970.

- Disponibilidad: esta clave siempre se incluye en el contexto de la solicitud.
- Tipo de datos: fecha, numérico
- Tipo de valor: valor único

aws:referer

Utilice esta clave para comparar quién hizo referencia a la solicitud en el navegador cliente con el remitente que especificó en la política. El valor de contexto de la solicitud `aws:referer` lo proporciona el intermediario en un encabezado HTTP. El encabezado de `Referer` se incluye en una solicitud de navegador web cuando se selecciona un enlace en una página web. El encabezado de `Referer` contiene la dirección URL de la página web donde se seleccionó el enlace.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo si la solicitud al recurso de AWS se invocó mediante un enlace desde una URL de página web en el navegador. Esta clave no se incluye para las solicitudes programáticas porque no se utiliza un enlace del navegador para tener acceso al recurso de AWS.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Por ejemplo, puede acceder a un objeto de Amazon S3 directamente mediante una URL o mediante la invocación directa de la API. Para obtener más información, consulte Operaciones de la API de Amazon S3 utilizando directamente un navegador web. Cuando se accede a un objeto de Amazon S3 desde una dirección URL que existe en una página web, la dirección URL de la página web de origen se utiliza en `aws:referer`. Cuando se accede a un objeto de Amazon S3 escribiendo la dirección URL en el navegador, el `aws:referer` no está presente. Cuando se invoca la API directamente, el `aws:referer` tampoco está presente. Puede utilizar la clave de condición de `aws:referer` de una política para permitir las solicitudes realizadas desde un referente específico, como un enlace en una página web del dominio de su empresa.

> **Warning**
>
> Esta clave debe utilizarse con cuidado. Es peligroso incluir un valor de encabezado de referencia conocido públicamente. Las partes no autorizadas podrían utilizar navegadores personalizados o modificados para proporcionar cualquier valor `aws:referer` que eligieran. En consecuencia, `aws:referer` no debe utilizarse para impedir que entidades no autorizadas realicen solicitudes a AWS de forma directa. Se ofrece únicamente para que los clientes puedan proteger su contenido digital, como el contenido almacenado en Amazon S3, para evitar las referencias en sitios de terceros no autorizados.

aws:RequestedRegion

Utilice esta clave para comparar la región de AWS a la que se llamó en la solicitud con la región que ha especificado en la política. Puede utilizar esta clave de condición global para controlar qué regiones se pueden solicitar. Para ver las regiones de AWS de cada servicio, consulte Puntos de enlace y cuotas de servicio en la Referencia general de Amazon Web Services.

- Disponibilidad: Esta clave siempre se incluye en el contexto de la solicitud.
- Tipo de datos: cadena

- Tipo de valor: Valor único

Algunos servicios globales, como IAM, tienen un único punto de enlace. Como este punto de enlace se encuentra físicamente en la región EE. UU. Este (Norte de Virginia), las llamadas de IAM siempre se realizan a la región us-east-1. Por ejemplo, si crea una política que deniegue el acceso a todos los servicios cuando la región solicitada no sea us-west-2, las llamadas a IAM siempre generan un error. Para ver un ejemplo de cómo evitar este problema, consulte NotAction con Deny.

> **Note**
>
> La clave de condición `aws:RequestedRegion` le permite controlar qué punto de enlace de un servicio se invoca, pero no controlar el impacto de la operación. Algunos servicios afectan a varias regiones.
> Por ejemplo, Amazon S3 tiene operaciones de la API que se extienden a todas las regiones.
>
> - Puede invocar `s3:PutBucketReplication` en una región (lo que se consigue con la clave de condición `aws:RequestedRegion`), pero afectar también a otras regiones en función de las opciones de configuración de replicaciones.
> - Puede invocar `s3:CreateBucket` para crear un bucket en otra región y utilizar la clave de condición `s3:LocationConstraint` para controlar las regiones aplicables.

Puede utilizar esta clave de contexto para limitar el acceso a los servicios de AWS a un conjunto de regiones determinado. Por ejemplo, la política siguiente permite a un usuarios ver todas las instancias Amazon EC2 en la AWS Management Console. Sin embargo, solo permite realizar cambios en las instancias de Irlanda (eu-west-1), Londres (eu-west-2) o París (eu-west-3).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "InstanceConsoleReadOnly",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Export*",
                "ec2:Get*",
                "ec2:Search*"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "InstanceWriteRegionRestricted",
            "Effect": "Allow",
            "Action": [
                "ec2:Associate*",
                "ec2:Import*",
                "ec2:Modify*",
                "ec2:Monitor*",
                "ec2:Reset*",
                "ec2:Run*",
                "ec2:Start*",
                "ec2:Stop*",
                "ec2:Terminate*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": [
                        "eu-west-1",
                        "eu-west-2",
                        "eu-west-3"
                    ]
                }
            }
        }
    ]
}
```

aws:RequestTag/clave-etiqueta

Utilice esta clave para comparar el par clave-valor de etiqueta que se transfirió en la solicitud con el par de etiquetas especificado en la política. Por ejemplo, podría comprobar si la solicitud incluya la clave de etiqueta `"Dept"` y que tenga el valor `"Accounting"`. Para obtener más información, consulte Control del acceso durante solicitudes de AWS.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud cuando se transfieren pares de valor de clave en la solicitud. Cuando se transfieren varias etiquetas en la solicitud, hay una clave de contexto para cada par clave-valor de etiqueta.
- Tipo de datos: cadena
- Tipo de valor: Valor único

Esta clave de contexto tiene el formato `"aws:RequestTag/tag-key":"tag-value"`, donde *`tag-key`* y *`tag-value`* son un par formado por una clave y un valor. Las claves y los valores de las etiquetas no distinguen entre mayúsculas y minúsculas. Esto significa que si especifica `"aws:RequestTag/TagKey1": "Value1"` en el elemento de condición de su política, la condición coincidirá con una clave de etiqueta de solicitud denominada `TagKey1` o `tagkey1`, pero no con ambas.

En este ejemplo se muestra que, si bien la clave tiene un solo valor, puede seguir utilizando varios pares de valor de clave en una solicitud si las claves son diferentes.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:::instance/*",
    "Condition": {
      "StringEquals": {
        "aws:RequestTag/environment": [
          "preprod",
          "production"
        ],
        "aws:RequestTag/team": [
          "engineering"
        ]
      }
    }
  }
}
```

aws:TagKeys

Utilice esta clave para comparar las claves de etiqueta de una solicitud con las claves que especifique en la política. Como práctica recomendada cuando utilice políticas para controlar el acceso mediante etiquetas, utilice la clave de condición `aws:TagKeys` para definir lo que se permite realizar a las claves de etiqueta. Para obtener más información y políticas de ejemplo, consulte the section called “Control del acceso en función de las claves de etiqueta”.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo si la operación permite pasar etiquetas en la solicitud.
- Tipo de datos: cadena (lista)

- Tipo de valor: multivalor

Esta clave de contexto tiene el formato `"aws:TagKeys":"tag-key"`, donde *`tag-key`* es una lista de claves de etiqueta sin valores (por ejemplo, `["Dept","Cost-Center"]`).

Dado que puede incluir varios pares de clave-valor de etiqueta en una solicitud, el contenido de la solicitud podría ser una solicitud con varios valores. En este caso, debe utilizar la `ForAllValues` o los operadores de establecimiento `ForAnyValue`. Para obtener más información, consulte Claves de contexto multivalor.

Algunos servicios admiten el etiquetado con las operaciones del recurso, como, por ejemplo, crear, modificar o eliminar un recurso. Para permitir el etiquetado y las operaciones como una sola llamada, debe crear una política que incluya tanto la acción de etiquetado como la acción de modificación del recurso. A continuación, puede utilizar la clave de condición `aws:TagKeys` para imponer el uso de claves de etiqueta específicas en la solicitud. Por ejemplo, para limitar las etiquetas cuando alguien crea una instantánea de Amazon EC2, debe incluir la acción de creación `ec2:CreateSnapshot` y la acción de etiquetado `ec2:CreateTags` en la política. Para ver una política para este escenario que utiliza `aws:TagKeys`, consulte Crear una instantánea con etiquetas en la Guía del usuario de Amazon EC2 para instancias de Linux.

aws:SecureTransport

Utilice esta clave para comprobar si la solicitud se envió mediante SSL. El contexto de la solicitud devuelve `true` o `false`. En una política, solo puede permitir acciones específicas si la solicitud se envía mediante SSL.

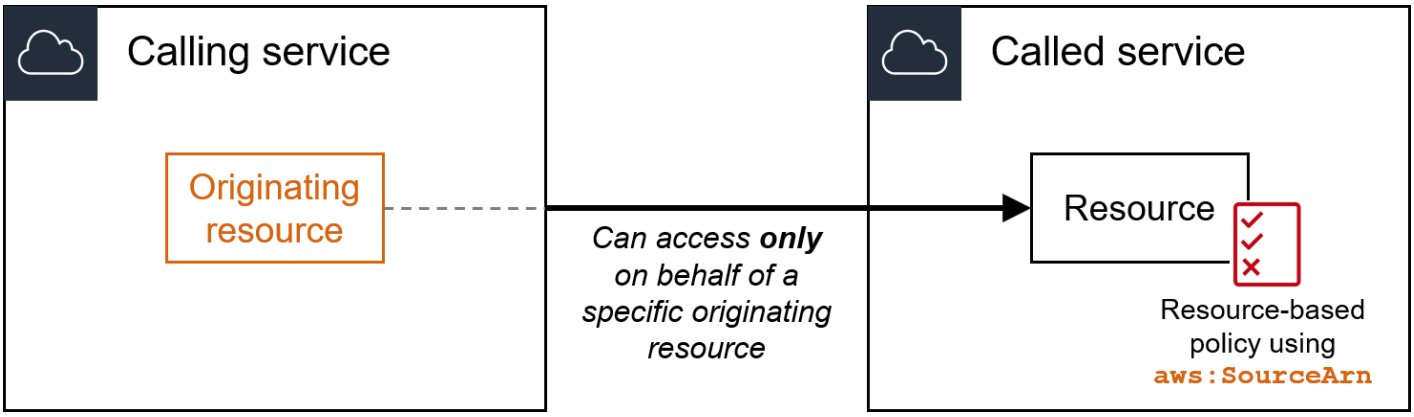
- Disponibilidad: esta clave siempre se incluye en el contexto de la solicitud.
- Tipos de datos: booleano
- Tipo de valor: Valor único

aws:SourceArn

Utilice esta clave para comparar el Nombre de recurso de Amazon (ARN) del recurso que realiza una solicitud de servicio a servicio con el ARN que especifique en la política, pero solamente cuando la solicitud la realiza una entidad principal de servicio de AWS. Cuando el ARN del origen incluye el ID de cuenta, no es necesario utilizar `aws:SourceAccount` con `aws:SourceArn`.

Esta clave no funciona con el ARN de la entidad principal que realiza la solicitud. En su lugar, utilice `aws:PrincipalArn`.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo cuando la llamada a su recurso la realiza directamente una entidad principal de servicio de AWS en nombre de un recurso para el que la configuración activó la solicitud de servicio a servicio. El servicio que realiza la llamada pasa el ARN del recurso original al servicio llamado.



Las siguientes integraciones de servicios no admiten esta clave de condición global:

Servicio de llamadas (entidad principal del servicio)	Servicio llamado (política basada en recursos)	Descripción
logdelivery.elb.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3

> **Note**
>
> No se admiten todas las integraciones de servicios con AWS Security Token Service (AWS STS) y AWS Key Management Service (AWS KMS). Para obtener más información, consulte la documentación del servicio de llamadas. El uso de `aws:SourceArn` en

políticas de clave de KMS para las claves utilizadas por los Servicios de AWS mediante la concesión de claves de KMS puede provocar un comportamiento inesperado.

- Tipo de datos: ARN, cadena

  AWS recomienda utilizar operadores de ARN en lugar de operadores de cadenas al comparar los ARN.

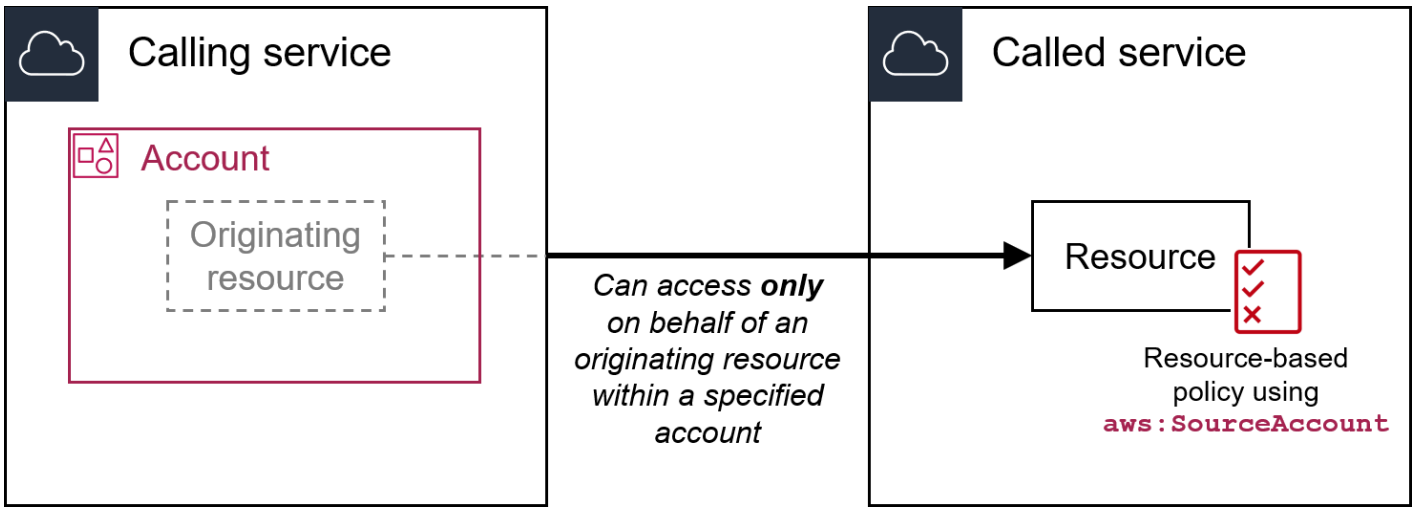
- Tipo de valor: valor único

Puede utilizar esta clave de condición para evitar que se utilice un servicio de AWS como un sustituto confuso durante las transacciones entre servicios. Utilice esta clave solo en las políticas basadas en recursos en las que `Principal` sea una entidad principal de un Servicio de AWS. Establezca el valor de esta clave de condición en el ARN del recurso en la solicitud. Por ejemplo, cuando una actualización de bucket de Amazon S3 activa la publicación de un tema de Amazon SNS, el servicio Amazon S3 invoca la operación `sns:Publish` de la API. En la política de tema que permite la operación `sns:Publish`, establezca el valor de la clave de condición en el ARN del bucket de Amazon S3. Para obtener información acerca de cómo y cuándo se recomienda esta clave de condición, consulte la documentación de los servicios de AWS que está utilizando.

aws:SourceAccount

Utilice esta clave para comparar el ID de cuenta del recurso que realiza una solicitud de servicio a servicio con el ID de cuenta que especifique en la política, pero solamente cuando la solicitud la realice una entidad principal de servicio de AWS.

- Disponibilidad: esta clave se incluye en el contexto de la solicitud solo cuando la llamada a su recurso la realiza directamente una entidad principal de servicio de AWS en nombre de un recurso para el que la configuración activó la solicitud de servicio a servicio. El servicio que realiza la llamada debe pasar el ID de cuenta del recurso original al servicio llamado.



Las siguientes integraciones de servicios no admiten esta clave de condición global:

Servicio de llamadas (entidad principal del servicio)	Servicio llamado (política basada en recursos)	Descripción
logdelivery.elb.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3

> **Note**
>
> No se admiten todas las integraciones de servicios con AWS Security Token Service (AWS STS) y AWS Key Management Service (AWS KMS). Para obtener más información, consulte la documentación del servicio de llamadas. El uso de `aws:SourceAccount` en políticas de clave de KMS para las claves utilizadas por los Servicios de AWS mediante la concesión de claves de KMS puede provocar un comportamiento inesperado.

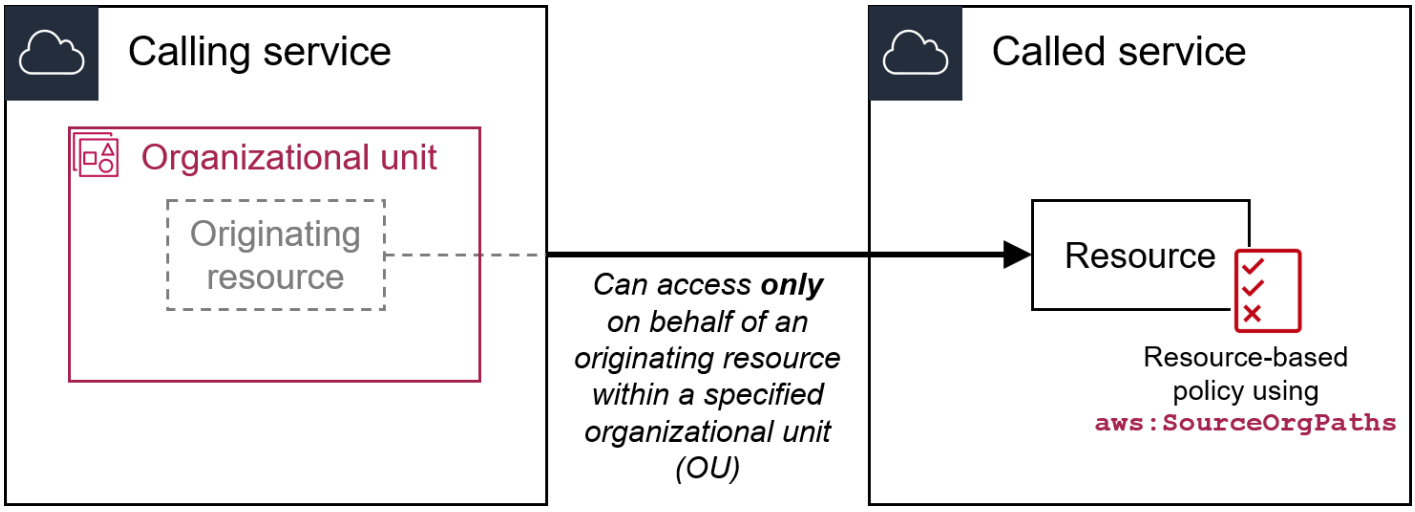
- Tipo de datos: cadena
- Tipo de valor: valor único

Puede utilizar esta clave de condición para evitar que se utilice un servicio de AWS como un sustituto confuso durante las transacciones entre servicios. Utilice esta clave solo en las políticas basadas en recursos en las que `Principal` sea una entidad principal de un Servicio de AWS. Establezca el valor de esta clave de condición en el ID de la cuenta del recurso en la solicitud. Por ejemplo, cuando una actualización de bucket de Amazon S3 activa la publicación de un tema de Amazon SNS, el servicio Amazon S3 invoca la operación `sns:Publish` de la API. En la política de tema que permite la operación de `sns:Publish`, establezca el valor de la clave de condición en el ID de cuenta del bucket de Amazon S3. Para obtener información acerca de cómo y cuándo se recomienda esta clave de condición, consulte la documentación de los servicios de AWS que está utilizando.

aws:SourceOrgPaths

Utilice esta clave para comparar la ruta de AWS Organizations del recurso que realiza una solicitud de servicio a servicio con el ID de cuenta que especifique en la política, pero solamente cuando la solicitud la realiza una entidad principal de servicio de AWS. Una ruta de Organizations es una representación de texto de la estructura de una entidad de Organizations. Para obtener más información acerca de las rutas de acceso, consulte Comprender la ruta de la entidad de AWS Organizations.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo cuando la llamada a su recurso la realiza directamente un responsable de la entidad principal del servicio de AWS en nombre de un recurso propiedad de una cuenta que es miembro de una organización. El servicio que realiza la llamada debe pasar la ruta de la organización del recurso original al servicio llamado.



Las siguientes integraciones de servicios no admiten esta clave de condición global:

Servicio de llamadas (entidad principal del servicio)	Servicio llamado (política basada en recursos)	Descripción
logdelivery.elb.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3
Todas las entidades principales del servicio	Bot Amazon Lex	Permitir que Servicios de AWS use el bot Amazon Lex

> **Note**
>
> No se admiten todas las integraciones de servicios con AWS Security Token Service (AWS STS) y AWS Key Management Service (AWS KMS). Para obtener más información, consulte la documentación del servicio de llamadas. El uso de `aws:SourceOrgPaths` en políticas clave de KMS para las claves utilizadas por los Servicios de AWS mediante la concesión de claves de KMS puede provocar un comportamiento inesperado.

- Tipo de datos: cadena (lista)
- Tipo de valor: Multivalor

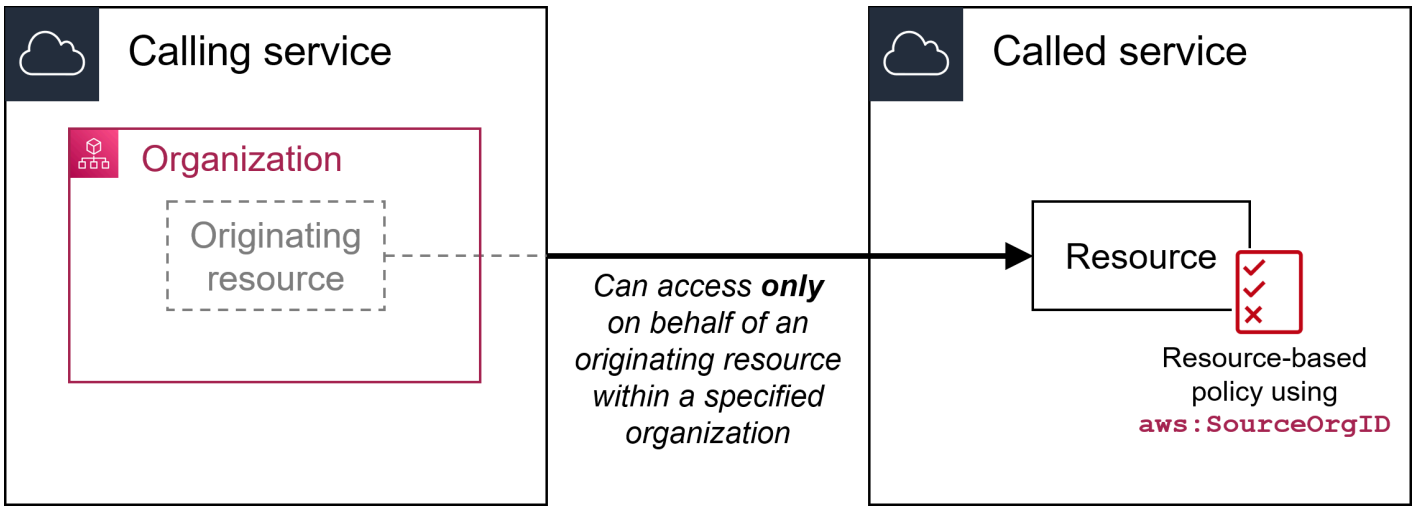
Puede utilizar esta clave de condición para evitar que se utilice un servicio de AWS como un sustituto confuso durante las transacciones entre servicios. Utilice esta clave solo en las políticas basadas en recursos en las que `Principal` sea una entidad principal de Servicio de AWS. Establezca el valor de esta clave de condición en la ruta de la organización del recurso en la solicitud. Por ejemplo, cuando una actualización de bucket de Amazon S3 activa la publicación de un tema de Amazon SNS, el servicio Amazon S3 invoca la operación `sns:Publish` de la API. En la política de temas que permite la operación de `sns:Publish`, establezca el valor de la clave de condición en la ruta de organización del bucket de Amazon S3. Para obtener información acerca de cómo y cuándo se recomienda esta clave de condición, consulte la documentación de los servicios de AWS que está utilizando.

`aws:SourceOrgPaths` es una clave de condición multivalor. Las claves de condición multivalor pueden tener varios valores en el contexto de la solicitud. Debe utilizar los operadores de servicio `ForAnyValue` o `ForAllValues` con los operadores de condición de cadena cuando utilice esta clave. Para obtener más información acerca de las claves de condición multivalor, consulte Claves de contexto multivalor.

aws:SourceOrgID

Utilice esta clave para comparar el ID de la organización del recurso que realiza una solicitud de servicio a servicio con el ID de cuenta que especifique en la política, pero solamente cuando la solicitud la realiza una entidad principal de servicio de AWS. Al agregar y quitar cuentas a una organización en AWS Organizations, las políticas que incluyan la clave `aws:SourceOrgID` tendrán automáticamente las cuentas correctas y no requerirán una actualización manual.

- Disponibilidad: Esta clave se incluye en el contexto de la solicitud solo cuando la llamada a su recurso la realiza directamente un responsable de la entidad principal del servicio de AWS en nombre de un recurso propiedad de una cuenta que es miembro de una organización. El servicio que realiza la llamada pasa el ID de la organización del recurso original al servicio llamado.



Las siguientes integraciones de servicios no admiten esta clave de condición global:

Servicio de llamadas (entidad principal del servicio)	Servicio llamado (política basada en recursos)	Descripción
logdelivery.elb.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3

Servicio de llamadas (entidad principal del servicio)	Servicio llamado (política basada en recursos)	Descripción
logdelivery.elasticloadbalancing.amazonaws.com	Bucket de Amazon S3	Habilite el registro de acceso de Elastic Load Balancing en el bucket de Amazon S3
Todas las entidades principales del servicio	Bot Amazon Lex	Permitir que Servicios de AWS use el bot Amazon Lex

> **Note**
>
> No se admiten todas las integraciones de servicios con AWS Security Token Service (AWS STS) y AWS Key Management Service (AWS KMS). Para obtener más información, consulte la documentación del servicio de llamadas. El uso de `aws:SourceOrgID` en políticas de clave de KMS para las claves utilizadas por los Servicios de AWS mediante la concesión de claves de KMS puede provocar un comportamiento inesperado.

- Tipo de datos: cadena
- Tipo de valor: valor único

Puede utilizar esta clave de condición para evitar que se utilice un servicio de AWS como un sustituto confuso durante las transacciones entre servicios. Utilice esta clave solo en las políticas basadas en recursos en las que `Principal` sea una entidad principal de un Servicio de AWS. Establezca el valor de esta clave de condición en la ID de la organización del recurso en la solicitud. Por ejemplo, cuando una actualización de bucket de Amazon S3 activa la publicación de un tema de Amazon SNS, el servicio Amazon S3 invoca la operación `sns:Publish` de la API. En la política de temas que permite la operación de `sns:Publish`, establezca el valor de la clave de condición en el ID de organización del bucket de Amazon S3. Para obtener información acerca de cómo y cuándo se recomienda esta clave de condición, consulte la documentación de los servicios de AWS que está utilizando.

aws:UserAgent

Utilice esta clave para comparar la aplicación cliente del solicitante con la aplicación que especifique en la política.

- Disponibilidad: esta clave siempre se incluye en el contexto de la solicitud.
- Tipo de datos: cadena
- Tipo de valor: Valor único

> **Warning**
>
> Esta clave debe utilizarse con cuidado. Puesto que el valor `aws:UserAgent` lo proporciona el intermediario en un encabezado HTTP, partes no autorizadas podrían utilizar navegadores personalizados o modificados para proporcionar cualquier valor `aws:UserAgent` que eligieran. En consecuencia, `aws:UserAgent` no debe utilizarse para impedir que entidades no autorizadas realicen solicitudes a AWS de forma directa. Puede utilizarlo para permitir únicamente aplicaciones cliente específicas y solo después de probar su política.

Otras claves de condición entre servicios

AWS STS admite claves de condición de federación basada en SAML y claves de condición entre servicios para la federación de OIDC. Estas claves están disponibles cuando un usuario federado mediante SAML realiza operaciones de AWS en otros servicios.

Claves de contexto de condición de IAM y AWS STS

Puede utilizar el elemento `Condition` en una política JSON para probar el valor de las claves que se incluyen en el contexto de solicitud de todas las solicitudes de AWS. Estas claves proporcionan información sobre la propia solicitud o sobre los recursos a los que se refiere. Puede comprobar que las claves han especificado valores antes de permitir la acción solicitada por el usuario. De este modo, dispondrá de control detallado sobre cuándo las instrucciones de la política de JSON coinciden o no coinciden con una solicitud entrante. Para obtener información sobre cómo utilizar el elemento `Condition` en una política de JSON, consulte Elementos de política JSON de IAM: Condition.

En este tema se describen las claves definidas y proporcionadas por el servicio de IAM (con el prefijo `iam:`) y el servicio AWS Security Token Service (AWS STS) (con un prefijo `sts:`). Existen otros servicios de AWS que también proporcionan claves específicas del servicio relevantes para las acciones y los recursos definidos por dicho servicio. Para obtener más información, consulte Acciones, recursos y claves de condición para servicios de AWS. La documentación de un servicio que admite las claves de condición a menudo dispone de información adicional. Por ejemplo, para

obtener información sobre las claves que puede utilizar en las políticas de recursos de Amazon S3, consulte Claves de política de Amazon S3 en la Guía del usuario de Amazon Simple Storage Service.

Temas

Claves disponibles para IAM

Puede utilizar las siguientes claves de condición en políticas que controlan el acceso a los recursos de IAM:

iam:AssociatedResourceArn

Funciona con operadores ARN.

Especifica el ARN del recurso al que se asociará este rol en el servicio de destino. El recurso generalmente pertenece al servicio al que la entidad principal pasa el rol. A veces, el recurso puede pertenecer a un tercer servicio. Por ejemplo, puede pasar un rol a Amazon EC2 Auto Scaling que utilice en una instancia de Amazon EC2. En este caso, la condición coincidiría con el ARN de la instancia de Amazon EC2.

Esta clave de condición solo se aplica a la acción PassRole de una política. No se puede utilizar para limitar cualquier otra acción.

Utilice esta clave de condición en una política para permitir que una entidad pase un rol, pero solo si ese rol está asociado con el recurso especificado. Puede utilizar caracteres comodín (*) para permitir operaciones realizadas en un tipo específico de recurso sin restringir la región o el ID de recurso. Por ejemplo, puede permitir que un usuario o rol de IAM pasen cualquier rol al servicio de Amazon EC2 para que se utilice con instancias en la región `us-east-1` o `us-west-1`. No se permitiría al usuario o rol de IAM pasar roles a otros servicios. Además, no permite que Amazon EC2 utilice el rol con instancias de otras regiones.

```
{
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-west-1:111122223333:instance/*"
            ]
        }
    }
}
```

> **Note**
>
> Los servicios de AWS que admiten iam:PassedToService admiten también esta clave de condición.

iam: AWSServiceName

Funciona con operadores de cadena.

Especifica el servicio de AWS al que está asociado este rol.

En este ejemplo, permite que una entidad cree un rol vinculado al servicio si el nombre del servicio es access-analyzer.amazonaws.com.

```
{
   "Version": "2012-10-17",
   "Statement": [{
       "Effect": "Allow",
       "Action": "iam:CreateServiceLinkedRole",
       "Resource": "*",
       "Condition": {
         "StringLike": {
           "iam:AWSServiceName": "access-analyzer.amazonaws.com"
         }
       }
```

```
    }]
}
```

iam: Certificación FIDO

Funciona con operadores de cadena.

Comprueba el nivel de certificación FIDO del dispositivo MFA al registrar una clave de seguridad FIDO. La certificación del dispositivo se obtiene del Servicio de metadatos (MDS) de FIDO Alliance. Si el estado o el nivel de certificación de su clave de seguridad FIDO cambia, no se actualizará a menos que el dispositivo no esté registrado y se haya registrado de nuevo para obtener la información de certificación actualizada.

Valores posibles de L1, L1plus, L2, L2plus, L3, L3plus

En este ejemplo, registra una clave de seguridad y recupera la certificación FIDO de nivel 1 plus para tu dispositivo.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-certification": "L1plus"
                }
            }
        }
    ]
```

```
 }
```

iam: Certificación FIDO-FIPS-140-2

Funciona con operadores de cadena.

Comprueba el nivel de certificación de validación FIPS-140-2 del dispositivo MFA al registrar una clave de seguridad FIDO. La certificación del dispositivo se obtiene del Servicio de metadatos (MDS) de FIDO Alliance. Si el estado o el nivel de certificación de su clave de seguridad FIDO cambia, no se actualizará a menos que el dispositivo no esté registrado y se haya registrado de nuevo para obtener la información de certificación actualizada.

Valores posibles de L1, L2, L3, L4

En este ejemplo, registra una clave de seguridad y recupera la certificación FIPS-140-2 de nivel 2 para su dispositivo.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-2-certification": "L2"
                }
            }
        }
    ]
```

```
}
```

iam: Certificación FIDO-FIPS-140-3

Funciona con operadores de cadena.

Comprueba el nivel de certificación de validación FIPS-140-3 del dispositivo MFA al registrar una clave de seguridad FIDO. La certificación del dispositivo se obtiene del Servicio de metadatos (MDS) de FIDO Alliance. Si el estado o el nivel de certificación de su clave de seguridad FIDO cambia, no se actualizará a menos que el dispositivo no esté registrado y se haya registrado de nuevo para obtener la información de certificación actualizada.

Valores posibles de L1, L2, L3, L4

En este ejemplo, registra una clave de seguridad y recupera la certificación FIPS-140-3 de nivel 3 para su dispositivo.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L3"
                }
            }
        }
    ]

}
```

iam: Registrar clave de seguridad

Funciona con operadores de cadena.

Comprueba el estado actual de la activación del dispositivo MFA.

Valores posibles de `Create` o `Activate`.

En este ejemplo, registra una clave de seguridad y recupera la certificación FIPS-140-3 de nivel 1 para su dispositivo.

```
{
      "Version": "2012-10-17",
      "Statement": [{
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Create"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:RegisterSecurityKey" : "Activate",
                    "iam:FIDO-FIPS-140-3-certification": "L1"
                }
            }
        }
    ]

 }
```

iam:OrganizationsPolicyId

Funciona con operadores de cadena.

Comprueba que la política con el ID de AWS Organizations especificado coincide con la política que se utiliza en la solicitud. Para ver una política de ejemplo de IAM que utilice esta clave de

condición, consulte IAM: ver la información del último acceso al servicio para una política de Organizaciones.

iam: PassedToService

Funciona con operadores de cadena.

Especifica el principal del servicio al que puede pasarse un rol. Esta clave de condición solo se aplica a la acción PassRole de una política. No se puede utilizar para limitar cualquier otra acción.

Cuando utilice esta clave de condición en una política, especifique el servicio mediante una entidad principal del servicio. El principal de un servicio es un nombre de servicio que puede especificarse en el elemento `Principal` de una política. El formato habitual es: `SERVICE_NAME_URL.amazonaws.com`.

Puede utilizar `iam:PassedToService` para que los usuarios solo puedan transferir roles a servicios específicos. Por ejemplo, un usuario puede crear una función de servicio que confíe en CloudWatch para escribir en su nombre datos de registro en un bucket de Amazon S3. A continuación, el usuario debe asociar una política de permisos y una política de confianza al nuevo rol de servicio. En este caso, la política de confianza debe especificar `cloudwatch.amazonaws.com` en el elemento `Principal`. Para ver una política que permita al usuario pasar el rol a CloudWatch, consulte IAM: pasar una función de IAM a un servicio de AWS específico.

Con esta clave de condición puede asegurar que los usuarios solo crearán roles de servicio para los servicios que especifique. Por ejemplo, si un usuario con la política anterior intenta crear una función de servicio para Amazon EC2, la operación fallará. El error se produce porque el usuario no tiene permiso para pasar el rol a Amazon EC2.

A veces se pasa un rol a un servicio que, a continuación, pasa el rol a otro servicio. `iam:PassedToService` incluye solo el servicio final que asume el rol, no el servicio intermedio que pasa el rol.

> **Note**
>
> Algunos servicios no admiten esta clave de condición.

iam:PermissionsBoundary

Funciona con operadores ARN.

Comprueba que la política especificada se asocia como límite de permisos por el recursos principal de IAM. Para obtener más información, consulte Límites de permisos para las entidades de IAM

iam:PolicyARN

Funciona con operadores ARN.

Comprueba el nombre de recurso de Amazon (ARN) de una política administrada en las solicitudes que impliquen una política administrada. Para obtener más información, consulte Control del acceso a políticas.

iam:ResourceTag/***key-name***

Funciona con operadores de cadena.

Comprueba que la etiqueta asociada al recurso de identidad (usuario o rol) coincida con el valor y el nombre de la clave especificada.

> **Note**
>
> IAM y AWS STS admiten tanto la clave de condición `iam:ResourceTag` de IAM como la clave de condición global `aws:ResourceTag`.

Puede agregar atributos personalizados a recursos de IAM en forma de un par de valor de clave. Para obtener más información sobre el etiquetado de recursos de IAM, consulte the section called "Etiquetado de recursos de IAM". Puede utilizar `ResourceTag` para controlar el acceso a los recursos de AWS, incluidos los recursos de IAM. No obstante, debido a que IAM no es compatible con etiquetas de grupos, no puede utilizar etiquetas para controlar el acceso a grupos.

En este ejemplo se muestra cómo podría crear una política basada en identidad que permita eliminar usuarios con la etiqueta **`status=terminated`**. Para utilizar esta política, sustituya el *`texto en cursiva del marcador`* de la política de ejemplo con su propia información. A continuación, siga las instrucciones en Crear una política o Editar una política.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
```

```
        "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
    }]
}
```

Claves disponibles para las federaciones de identidades AWS de OIDC

Puede utilizar la federación de OIDC para otorgar credenciales de seguridad temporales a los usuarios autenticados a través de un proveedor de OpenID (OP) compatible con OpenID Connect a un proveedor de identidad IAM OpenID Connect (OIDC) de su cuenta de AWS. Algunos ejemplos de este tipo de proveedores incluyen Login with Amazon, Amazon Cognito, Google o Facebook. Se pueden utilizar los identificadores de identidad (id_tokens) de su propio OP de OpenID, así como los id_tokens emitidos a las cuentas de servicio de los clústeres de Amazon Elastic Kubernetes Service. En ese caso habrá claves de condición adicionales disponibles cuando se utilicen las credenciales de seguridad temporales al realizar una solicitud. Puede utilizar esas claves para crear políticas que limita el acceso de los usuarios federados a los recursos asociados a un proveedor, aplicación o usuario específico. Estas claves suelen utilizarse en la política de confianza de un rol. Defina las claves de condición utilizando el nombre del proveedor de OIDC seguido de la reclamación (`:aud`, `:azp`, `:amr`, `:sub`). En el caso de los roles utilizados por Amazon Cognito, las claves se definen usando `cognito-identity.amazonaws.com` seguidas de la afirmación.

amr

Funciona con operadores de cadena.

Ejemplo: `cognito-identity.amazonaws.com:amr`

Si utiliza Amazon Cognito para la federación de OIDC, la clave `cognito-identity.amazonaws.com:amr` (Authentication Methods Reference) incluye la información de inicio de sesión sobre el usuario. La clave comporta muchos valores, lo que significa que el usuario la prueba en una política utilizando operadores de definición de condición. La clave puede contener los siguientes valores:

- Si el usuario no está autenticado, la clave contiene únicamente `unauthenticated`.
- Si el usuario está autenticado, la clave contiene el valor `authenticated` y el nombre del proveedor de inicio de sesión utilizado en la llamada (`graph.facebook.com`, `accounts.google.com` o `www.amazon.com`).

A modo de ejemplo, la siguiente condición en la política de confianza de un rol de Amazon Cognito prueba si el usuario no está autenticado:

```
"Condition": {
  "StringEquals":
    { "cognito-identity.amazonaws.com:aud": "us-east-2:identity-pool-id" },
  "ForAnyValue:StringLike":
    { "cognito-identity.amazonaws.com:amr": "unauthenticated" }
}
```

aud

Funciona con operadores de cadena.

Utilice la clave de condición de aud para verificar que el ID de cliente de Google o el ID del grupo de identidades de Amazon Cognito coincide con el que ha especificado en la política. Puede utilizar la clave aud con la clave sub del mismo proveedor de identidad.

Ejemplos:

- graph.facebook.com:app_id
- accounts.google.com:aud
- cognito-identity.amazonaws.com:aud

El campo graph.facebook.com:app_id proporciona el contexto de público que se corresponde con el campo aud utilizado por otros proveedores de identidades.

La clave de condición accounts.google.com:aud coincide con los siguientes campos Token de ID de Google.

- aud para los ID de cliente de Google OAuth 2.0 de su aplicación, cuando el campo azp no está configurado. Cuando se establece el campo azp, el campo aud coincide con la clave de condición accounts.google.com:oaud.
- azp cuando se establece el campo azp. Esto puede suceder en aplicaciones híbridas donde una aplicación web y una aplicación de Android tienen un ID de cliente de Google OAuth 2.0 diferente pero comparten el mismo proyecto de API de Google.

Para obtener más información sobre los campos aud y azp de Google, consulte la Guía de OpenID Connect de la plataforma de identidad de Google.

Si escribe una política con la clave de condición accounts.google.com:aud, debe saber si la aplicación es una aplicación híbrida que establece el campo azp.

azp Campo no definido

La siguiente política de ejemplo funciona para las aplicaciones no híbridas que no establecen el campo `azp`. En este caso, el valor del campo `aud` del token de ID de Google coincide con los valores de la clave de condición `accounts.google.com:aud` y `accounts.google.com:oaud`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "aud-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
                }
            }
        }
    ]
}
```

`azp` Especificación de campos

La siguiente política de ejemplo funciona para las aplicaciones híbridas que no establecen el campo `azp`. En este caso, el valor del campo `aud` del token de ID de Google solo coincide con el valor de la clave de condición `accounts.google.com:oaud`. El valor del campo `azp` coincide con el valor de la clave de condición `accounts.google.com:aud`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {"Federated": "accounts.google.com"},
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "accounts.google.com:aud": "azp-value",
                    "accounts.google.com:oaud": "aud-value",
                    "accounts.google.com:sub": "sub-value"
```

```
                }
            }
        }
    ]
}
```

id

Funciona con operadores de cadena.

Ejemplos:

- `graph.facebook.com:id`
- `www.amazon.com:app_id`
- `www.amazon.com:user_id`

Utilice estas claves para verificar que el ID de aplicación (o sitio) o el ID de usuario coincide con el que ha especificado en la política. Esto funciona para Facebook o Login with Amazon. Puede utilizar la clave `app_id` con la clave `id` del mismo proveedor de identidad.

oaud

Funciona con operadores de cadena.

Ejemplo: `accounts.google.com:oaud`

Si utiliza Google para la federación de OIDC, esta clave especifica la audiencia de Google (`aud`) para la que está prevista este token de ID. Debe ser uno de los ID de cliente de OAuth 2.0 de su aplicación.

sub

Funciona con operadores de cadena.

Ejemplos:

- `accounts.google.com:sub`
- `cognito-identity.amazonaws.com:sub`

Utilice estas claves para verificar que el ID de usuario coincide con el que ha especificado en la política. Puede utilizar la clave `sub` con la clave `aud` del mismo proveedor de identidad.

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [
       "Condition": {
             "StringEquals": {
                 "oidc.eks.us-east-1.amazonaws.com/id/111122223333:aud":
"sts.amazonaws.com",
                 "oidc.eks.us-east-1.amazonaws.com/id/111122223333:sub":
"system:serviceaccount:default:assumer"
              }
           }
   ]
     }
```

Más información acerca de la federación OIDC

Para obtener más información sobre la federación OIDC, consulte lo siguiente:

- Guía del usuario de Amazon Cognito.
- Federación OIDC

Claves de contexto de federación de OIDC AWS multiservicios

Algunas claves de condición de federación de OIDC se pueden utilizar en políticas de confianza de roles para definir a qué pueden acceder los usuarios en otros servicios de AWS. Las siguientes son las claves de condición que se pueden utilizar en políticas de confianza de roles cuando las entidades principales federadas asumen otro rol, así como en políticas de recursos de otros servicios de AWS para autorizar el acceso a los recursos por parte de las entidades principales federadas. Si utiliza Amazon Cognito para la federación de OIDC, estas claves están disponibles cuando el usuario se autentifica.

Seleccione una clave de condición para ver la descripción.

- amr
- aud
- id
- sub

> **Note**
>
> No está disponible ninguna otra clave de condición de federación basada en identidad web para su uso después de la autenticación y autorización del proveedor de identidades (IdP) externo para la operación AssumeRoleWithWebIdentity inicial.

Claves disponibles para la federación AWS STS basada en SAML

Si trabaja con federación basada en SAML utilizando AWS Security Token Service (AWS STS), puede incluir claves de condición adicionales en la política.

Políticas de confianza de roles de SAML

En la política de confianza de un rol, puede incluir las siguientes claves, que le ayudarán a determinar si el intermediario puede asumir el rol. Excepto `saml:doc`, todos los valores se derivan de la aserción de SAML. Todos los elementos de la lista están disponibles en el editor visual de la consola IAM al crear o editar una política con condiciones. Los elementos marcados con `[]` pueden tener un valor que sea una lista del tipo especificado.

saml:aud

Funciona con operadores de cadena.

Es una dirección URL de punto de enlace a la que se presentan las aserciones de SAML. El valor de esta clave proviene del campo `SAML Recipient` de la aserción, no del campo `Audience`.

saml:commonName[]

Funciona con operadores de cadena.

Se trata de un atributo `commonName`.

saml:cn[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:doc

Funciona con operadores de cadena.

Representa al principal que se utilizó para asumir el rol. El formato es *ID-cuenta*/*nombre-fácil-de-recordar-del-proveedor*, como `123456789012/SAMLProviderName`. El valor ID-cuenta hace referencia a la cuenta que posee el proveedor SAML.

saml:edupersonaffiliation[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonassurance[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonentitlement[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonnickname[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonorgdn

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonorgunitdn[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonprimaryaffiliation

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonprimaryorgunitdn

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonprincipalname

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersonscopedaffiliation[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:edupersontargetedid[]

Funciona con operadores de cadena.

Es un atributo `eduPerson`.

saml:eduorghomepageuri[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:eduorgidentityauthnpolicyuri[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:eduorglegalname[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:eduorgsuperioruri[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:eduorgwhitepagesuri[]

Funciona con operadores de cadena.

Es un atributo `eduOrg`.

saml:givenName[]

Funciona con operadores de cadena.

Se trata de un atributo `givenName`.

saml:iss

Funciona con operadores de cadena.

Se trata del emisor representado por un URN.

saml:mail[]

Funciona con operadores de cadena.

Se trata de un atributo `mail`.

saml:name[]

Funciona con operadores de cadena.

Se trata de un atributo `name`.

saml:namequalifier

Funciona con operadores de cadena.

Un valor hash basado en el nombre descriptivo del proveedor SAML. El valor es la concatenación de los siguientes valores, en orden y separados por un carácter '/':

1. El valor de respuesta `Issuer` (`saml:iss`)
2. El ID de la cuenta de `AWS`.
3. El nombre descriptivo (la última parte del ARN) del proveedor SAML en IAM

La concatenación del ID de cuenta y del nombre fácil de recordar del proveedor SAML está disponible para las políticas de IAM como clave `saml:doc`. Para obtener más información, consulte Identificación única de los usuarios en la federación basada en SAML.

saml:organizationStatus[]

Funciona con operadores de cadena.

Es un atributo `organizationStatus`.

saml:primaryGroupSID[]

Funciona con operadores de cadena.

Se trata de un atributo `primaryGroupSID`.

saml:sub

Funciona con operadores de cadena.

Se trata del asunto de la demanda, que incluye un valor que identifica de forma unívoca a un usuario individual dentro de una organización (por ejemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

saml:sub_type

Funciona con operadores de cadena.

Esta clave puede tener el valor `persistent`, `transient` o ser el URI `Format` completo de los elementos `Subject` y `NameID` utilizados en la aserción de SAML. El valor `persistent` indica que el valor de `saml:sub` es el mismo para un usuario entre sesiones. Si el valor es `transient`, el usuario tendrá un valor `saml:sub` diferente para cada sesión. Para obtener información sobre el atributo `NameID` del elemento `Format`, consulte Configure aserciones SAML para la respuesta de autenticación.

saml:surname[]

Funciona con operadores de cadena.

Se trata de un atributo `surnameuid`.

saml:uid[]

Funciona con operadores de cadena.

Se trata de un atributo `uid`.

saml:x500UniqueIdentifier[]

Funciona con operadores de cadena.

Es un atributo `x500UniqueIdentifier`.

Para obtener información general sobre los atributos `eduPerson` y `eduOrg`, consulte el sitio web REFEDS Wiki. Para ver una lista de `eduPerson` atributos, consulte eduPerson Object Class Specification (201602).

Las claves de condición cuyo tipo es una lista pueden incluir múltiples valores. Para crear condiciones en la política para los valores de lista, puede utilizar operadores de definición (`ForAllValues`, `ForAnyValue`). Por ejemplo, a fin de permitir que todos los usuarios cuya afiliación sea “profesorado” o “personal” (pero no “estudiante”), puede utilizar una condición como la siguiente:

```
"Condition": {
   "ForAllValues:StringLike": {
     "saml:edupersonaffiliation":[ "faculty", "staff"]
   }
}
```

Claves de contexto de federación de AWS STS basadas en SAML multiservicios

Algunas claves de condición de federación basadas en SAML se pueden utilizar en solicitudes posteriores para autorizar operaciones de AWS en otros servicios y llamadas `AssumeRole`. Las siguientes son las claves de condición que se pueden utilizar en políticas de confianza de roles cuando las entidades principales federadas asumen otro rol, así como en políticas de recursos de otros servicios de AWS para autorizar el acceso a los recursos por parte de las entidades principales federadas. Para obtener más información sobre el uso de estas claves, consulte Acerca de la federación basada en SAML 2.0.

Seleccione una clave de condición para ver la descripción.

- saml:namequalifier
- saml:sub
- saml:sub_type

> **Note**
>
> No está disponible ninguna otra clave de condición de federación basada en SAML para su uso después de la respuesta de autenticación del proveedor de identidades (IdP) externo inicial.

Claves disponibles para AWS STS

Puede utilizar las siguientes claves de condición en políticas de confianza de rol de IAM para los roles que se asumen utilizando operaciones de AWS Security Token Service (AWS STS).

saml:sub

Funciona con operadores de cadena.

Se trata del asunto de la demanda, que incluye un valor que identifica de forma unívoca a un usuario individual dentro de una organización (por ejemplo, `_cbb88bf52c2510eabe00c1642d4643f41430fe25e3`).

sts:AWSServiceName

Funciona con operadores de cadena.

Utilice esta clave para especificar un servicio donde se puede utilizar un token de portador. Cuando utilice esta clave de condición en una política, especifique el servicio mediante una entidad principal del servicio. El principal de un servicio es un nombre de servicio que puede especificarse en el elemento `Principal` de una política. Por ejemplo, `codeartifact.amazonaws.com` es la entidad principal de servicio AWS CodeArtifact.

Algunos servicios de AWS requieren que tenga permiso para obtener un token al portador del servicio AWS STS para poder acceder a sus recursos mediante programación. Por ejemplo, AWS CodeArtifact requiere que las entidades principales usen tokens portador para realizar algunas operaciones. El comando `aws codeartifact get-authorization-token` devuelve un token portador. A continuación, puede utilizar el token portador para realizar AWS CodeArtifact operaciones. Para obtener más información acerca de los tokens al portador, consulte Uso de tokens al portador.

Disponibilidad - Esta clave está presente en las solicitudes que reciben un token de portador. No puedes hacer una llamada directa a AWS STS para obtener un token. Cuando realiza algunas operaciones en otros servicios, el servicio solicita el token de portador en su nombre.

Puede utilizar esta clave de condición para permitir a los principales obtener un token de portador para usarlo con un servicio específico.

sts:DurationSeconds

Funciona con operadores numéricos.

Utilice esta clave para especificar la duración (en segundos) que un principal puede utilizar al obtener un token portador AWS STS.

Algunos servicios de AWS requieren que tenga permiso para obtener un token al portador del servicio AWS STS para poder acceder a sus recursos mediante programación. Por ejemplo, AWS CodeArtifact requiere que las entidades principales usen tokens portador para realizar algunas operaciones. El comando `aws codeartifact get-authorization-token` devuelve un token portador. A continuación, puede utilizar el token portador para realizar AWS CodeArtifact operaciones. Para obtener más información acerca de los tokens al portador, consulte Uso de tokens al portador.

Disponibilidad - Esta clave está presente en las solicitudes que reciben un token de portador. No puedes hacer una llamada directa a AWS STS para obtener un token. Cuando realiza algunas operaciones en otros servicios, el servicio solicita el token de portador en su nombre. La clave no está presente para las AWS STS operaciones assume-rol.

sts:ExternalId

Funciona con operadores de cadena.

Utilice esta clave para exigir que una entidad principal proporcione un identificador específico al asumir un rol de IAM.

Disponibilidad - Esta clave está presente en la solicitud cuando el principal proporciona un ID externo mientras asume un rol utilizando AWS CLI o API de AWS.

Un identificador único que podría ser necesario al asumir un rol en otra cuenta. Si el administrador de la cuenta a la que pertenece el rol le ha proporcionado un ID externo, entonces proporcione dicho valor en el parámetro `ExternalId`. Este valor puede ser cualquier cadena como, por ejemplo, una frase de contraseña o un número de cuenta. La función principal del ID externo es abordar y prevenir el problema del suplente confuso. Para obtener más información acerca del ID externo y el problema del suplente confuso, consulte Cómo utilizar un ID externo al otorgar acceso a los recursos de AWS a terceros.

El valor `ExternalId` debe tener 2 caracteres como mínimo y 1224 como máximo. El valor debe ser alfanumérico sin espacio en blanco. También puede incluir los símbolos siguientes: más (+), igual (=), coma (,), punto (.), arroba (@), dos puntos (:), barra inclinada (/) y guion (-).

sts:RequestContext/context-key

Funciona con operadores de cadena.

Utilice esta clave para comparar los pares clave-valor del contexto de la sesión que están integrados en la afirmación de contexto firmada por el emisor del token de confianza pasada en la solicitud con los valores-clave del contexto especificados en la política de confianza del rol.

Disponibilidad: Esta clave está presente en la solicitud cuando se proporciona una afirmación de contexto en el parámetro de la solicitud de `ProvidedContexts` mientras asume un rol con la operación de API de AWS STS `AssumeRole`.

Esta clave de contexto tiene el formato `"sts:RequestContext/context-key":"context-value"` en donde `context-key` y `context-value` son un par clave-valor de contexto. Cuando se integran varias claves de contexto en la afirmación de contexto firmada pasada en la solicitud, hay una clave de contexto para cada par clave-valor. Debe conceder permiso para la acción `sts:SetContext` en la política de confianza de roles a fin de permitir que una entidad principal establezca las claves de contexto en el token de sesión resultante.

Puede utilizar esta clave en una política de confianza de rol para aplicar un control de acceso detallado basado en el usuario o en sus atributos cuando asuman un rol. Por ejemplo, puede configurar Amazon Redshift como una aplicación del centro de identidad de IAM para acceder a los recursos de Amazon S3 en nombre de sus empleados o de sus identidades federadas.

La siguiente política de confianza de rol permite a la entidad principal de servicio de Amazon Redshift asumir un rol en la cuenta `111122223333`. También otorga permiso a la entidad principal del servicio Amazon Redshift para establecer las claves de contexto en la solicitud, siempre que se establezca el valor de la `identitystore:UserId` clave de contexto. `1111-22-3333-44-5555` Una vez asumido el rol, la actividad aparece en los registros de AWS CloudTrail dentro del elemento de `AdditionalEventData`, que contienen los pares clave-valor del contexto de la sesión que estableció el proveedor del contexto en la solicitud de asumir el rol. Esto hace que sea más fácil para los administradores diferenciar entre sesiones de rol cuando un rol es utilizado por diferentes entidades principales. El proveedor de contexto especificado establece los pares clave-valor, no por AWS CloudTrail ni AWS STS. Esto le da al proveedor del contexto el control sobre el contexto que se incluye en los registros y la información de sesión de CloudTrail.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "redshift.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Condition": {
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "sts:RequestContext/identitystore:UserId":
 "1111-22-3333-44-5555"
                }
            }
        }
    ]
}
```

sts:RequestContextProviders

Funciona con operadores ARN.

Utilice esta clave para comparar el ARN del proveedor de contexto de la solicitud con el ARN del proveedor de contexto especificado en la política de confianza del rol.

Disponibilidad: Esta clave está presente en la solicitud cuando se proporciona una afirmación de contexto en el parámetro de la solicitud de `ProvidedContexts` mientras asume un rol con la operación de API de AWS STS `AssumeRole`.

La siguiente condición de ejemplo comprueba que el ARN del proveedor de contexto pasado en la solicitud coincide con el ARN especificado en la condición de política de confianza del rol.

```
      "Condition": {
        "ForAllValues:ArnEquals": {
          "sts:RequestContextProviders": [
            "arn:aws:iam::aws:contextProvider/IdentityCenter"
          ]
        }
      }
```

sts:RoleSessionName

Funciona con operadores de cadena.

Utilice esta clave para comparar el nombre de sesión que especifica una entidad principal al asumir un rol con el valor especificado en la política.

Disponibilidad - Esta clave está presente en la solicitud cuando la entidad principal asume el rol mediante AWS Management Console, cualquier comando assume-role CLI o cualquier operación AWS STSde API `AssumeRole`.

Puede utilizar esta clave en una política de confianza de rol para exigir que los usuarios proporcionen un nombre de sesión específico cuando asuman un rol. Por ejemplo, puede requerir que los usuarios de IAM especifiquen su propio nombre de usuario como nombre de sesión. Después de que el usuario de IAM asuma el rol, la actividad aparece en los registros de AWS CloudTrail con el nombre de sesión que coincide con su nombre de usuario. Esto hace que sea más fácil para los administradores diferenciar entre sesiones de rol cuando un rol es utilizado por diferentes entidades principales.

La siguiente política de confianza de rol requiere que los usuarios de IAM de la cuenta `111122223333` proporcionen su nombre de usuario de IAM como nombre de sesión cuando asuman el rol. Este requisito se aplica utilizando lavariable de condición `aws:username` en la clave de condición. Esta política permite a los usuarios de IAM asumir el rol al que está asociada la política. Esta política no permite a nadie que utilice credenciales temporales asumir el rol porque la variable `username` solo está presente para los usuarios de IAM.

> **Important**
>
> Puede utilizar cualquier clave de condición de valor único como variable. No se puede utilizar una clave de condición multivalor como variable.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
```

```
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "${aws:username}"}
            }
        }
    ]
}
```

Cuando un administrador ve el registro de AWS CloudTrail de una acción, puede comparar el nombre de la sesión con los nombres de usuario en su cuenta. En el ejemplo siguiente, el usuario denominado `matjac` realizó la operación utilizando el rol denominado `MateoRole`. El administrador puede entonces ponerse en contacto con Mateo Jackson, quien tiene el nombre del usuario `matjac`.

```
    "assumedRoleUser": {
        "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:matjac",
        "arn": "arn:aws:sts::111122223333:assumed-role/MateoRole/matjac"
    }
```

Si permite el acceso entre cuentas mediante roles, los usuarios de una cuenta pueden asumir un rol en otra cuenta. El ARN del usuario de rol asumido indicado en CloudTrail incluye la cuenta donde existe el rol. No incluye la cuenta del usuario que asumió el rol. Los usuarios son únicos solo dentro de una cuenta. Por lo tanto, se recomienda utilizar este método para comprobar los registros de CloudTrail solo para los roles que asumen los usuarios en las cuentas que administra. Los usuarios pueden utilizar el mismo nombre de usuario en varias cuentas.

sts:SourceIdentity

Funciona con operadores de cadena.

Utilice esta clave para comparar la identidad de origen que especifica una entidad principal al asumir un rol con el valor especificado en la política.

Disponibilidad – Esta clave está presente en la solicitud cuando la entidad principal establece inicialmente una identidad de origen mientras asume un rol utilizando cualquier comando assume-rol CLI de AWS STS, o una operación AWS STS de API `AssumeRole`.

Puede utilizar esta clave en una política de confianza de rol para requerir que sus usuarios establezcan una identidad de origen específica cuando asuman un rol. Por ejemplo, puede requerir que su personal o identidades federadas especifiquen un valor para la identidad de origen. Puede configurar su proveedor de identidades (IdP) para que utilice uno de los atributos

asociados a los usuarios, como un nombre de usuario o un correo electrónico como identidad de origen. A continuación, el IdP pasa la identidad de origen como un atributo en las afirmaciones que envía a AWS. El valor del atributo de identidad de origen identifica al usuario o aplicación que está asumiendo el rol.

Después de que el usuario asuma el rol, la actividad aparece en registros de AWS CloudTrail con el valor de identidad de origen que se ha establecido. Esto facilita a los administradores determinar quién o qué ha realizado acciones con un rol en AWS. Debe conceder permisos para la acción `sts:SetSourceIdentity` para permitir que una identidad establezca una identidad de origen.

A diferencia de `sts:RoleSessionName`, después de establecer la identidad de origen, el valor no se puede cambiar. Está presente en el contexto de solicitud para todas las acciones realizadas con el rol por la identidad de origen. El valor persiste en sesiones de rol posteriores cuando se utilizan las credenciales de sesión para asumir otro rol. Asumir un rol de otro se llama encadenamiento de roles.

Puede utilizar la condición de clave global `aws:SourceIdentity` para controlar aun más el acceso a los recursos de AWS basados en el valor de la identidad de origen en solicitudes posteriores.

La siguiente política de confianza de rol permite al usuario de IAM `AdminUser` asumir un rol en la cuenta `111122223333`. También concede permiso a `AdminUser` para establecer una identidad de origen, siempre y cuando el conjunto de identidades de origen sea `DiegoRamirez`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowAdminUserAssumeRole",
            "Effect": "Allow",
            "Principal": {"AWS": " arn:aws:iam::111122223333:user/AdminUser"},
            "Action": [
                "sts:AssumeRole",
                "sts:SetSourceIdentity"
            ],
            "Condition": {
                "StringEquals": {"sts:SourceIdentity": "DiegoRamirez"}
            }
        }
    ]
```

```
}
```

Para obtener más información acerca del uso de información de identidad de origen, consulte Monitorear y controlar las acciones realizadas con roles asumidos.

sts:TransitiveTagKeys

Funciona con operadores de cadena.

Utilice esta clave para comparar las claves de etiqueta de sesión transitiva de la solicitud con las especificadas en la política.

Disponibilidad - Esta clave está presente en la solicitud cuando se realiza una solicitud con credenciales de seguridad temporales. Estas incluyen credenciales creadas mediante cualquier operación assume-role o la `GetFederationToken` operación.

Cuando realiza una solicitud con credenciales de seguridad temporales, el contexto de solicitud incluye la clave de contexto `aws:PrincipalTag`. Esta clave incluye una lista de etiquetas de sesión, etiquetas de sesión transitivas y etiquetas de rol. Las etiquetas de sesión transitivas son etiquetas que persisten en todas las sesiones posteriores cuando se utilizan las credenciales de sesión para asumir otro rol. Asumir un rol de otro se llama encadenamiento de roles.

Puede utilizar esta clave de condición en una política para requerir que se establezcan etiquetas de sesión específicas como transitivas al asumir un rol o federar un usuario.

Acciones, recursos y claves de condición de los servicios de AWS

Cada servicio de AWS puede definir acciones, recursos y claves de contexto de condición para utilizarse en las políticas de IAM. Para obtener una lista de los servicios de AWS, sus acciones, recursos y claves de contexto de condiciones, consulte Acciones, recursos y claves de condiciones en la referencia de autorizaciones de servicio.

Recursos para obtener más información sobre IAM.

IAM es un producto completo y encontrará muchos recursos que le ayudarán a obtener más información sobre cómo IAM puede ayudarle a proteger sus recursos y Cuenta de AWS.

Temas

Identidades

Consulte estos recursos para crear, administrar y utilizar identidades.

- Manage identities in IAM Identity Center (Administración de identidades en IAM Identity Center): información de procedimiento sobre la creación de usuarios y grupos en IAM Identity Center.
- Identidades de IAM (usuarios, grupos de usuarios y roles): una descripción más detallada de los usuarios, grupos y roles.

Credenciales (contraseñas, claves de acceso y dispositivos MFA)

Revise las siguientes guías para administrar contraseñas, claves de acceso y dispositivos MFA para su Cuenta de AWS y para los usuarios de IAM.

- Administración de las contraseñas de usuarios en AWS: describe las opciones para administrar las contraseñas de los usuarios de IAM en su cuenta.
- Administración de las claves de acceso de los usuarios de IAM: describe cómo funcionan las claves de acceso y cómo puede utilizarlas para hacer llamadas programáticas a AWS. Existen otras alternativas más seguras a las claves de acceso que le recomendamos que tenga en cuenta

en primer lugar. Para más información, consulte Consideraciones y alternativas para las claves de acceso a largo plazo en la Guía de Referencia general de AWS.

- Uso de autenticación multifactor (MFA) en AWS: describe cómo configurar su cuenta y usuarios de IAM para exigir una contraseña y un código de un solo uso que se genera en un dispositivo antes de permitir el inicio de sesión. (esto se denomina a veces autenticación de dos factores).

Para obtener información general acerca de los tipos de credenciales que se utilizan para acceder a Amazon Web Services, consulte Credenciales de seguridad de AWS en la Guía de Referencia general de AWS.

Permisos y políticas

Descubra los mecanismos internos de las políticas de IAM y obtenga consejos sobre las mejores formas de conceder permisos:

- Políticas y permisos en IAM: presenta el lenguaje de política que se utiliza para definir los permisos. Describe cómo pueden asociarse los permisos a usuarios o grupos o, para algunos productos de AWS, a recursos.
- Referencia de los elementos de las políticas de JSON de IAM: ofrece descripciones y ejemplos de cada elemento de lenguaje de política.
- Validación de políticas de IAM: busque recursos para la validación de políticas JSON.
- Ejemplos de políticas basadas en identidad de IAM: muestra ejemplos de políticas para tareas comunes en varios productos de AWS.
- Generador de políticas de AWS: permite crear políticas personalizadas eligiendo productos y acciones de una lista.
- Simulador de políticas de IAM: prueba si una política permitiría o denegaría una solicitud específica a AWS.

Federación y delegación

Puede conceder acceso a los recursos de su Cuenta de AWS a los usuarios que se autentican (inician sesión) en otro lugar. Estos pueden ser usuarios de IAM de otra Cuenta de AWS (lo que se conoce como delegación), usuarios que se autentican con el proceso de inicio de sesión de la organización o usuarios de un proveedor de identidad a través de Internet, como Login with Amazon, Facebook, Google u otro proveedor de identidad compatible con OpenID Connect (OIDC). En estos

casos, los usuarios obtienen las credenciales de seguridad temporales para obtener acceso a los recursos de AWS.

- Tutorial de IAM: delegación del acceso entre cuentas de AWS mediante roles de IAM: le indica cómo conceder acceso entre cuentas a un usuario de IAM de otra Cuenta de AWS.
- Escenarios habituales en las credenciales temporales: describe formas en las que los usuarios pueden federarse en AWS después de autenticarse fuera de AWS.

IAM y otros productos de AWS

La mayoría de los productos de AWS se integran con IAM para que puedan utilizar las características de IAM con el fin de ayudar a proteger el acceso a los recursos de dichos productos. En los siguientes recursos se explica IAM y la seguridad de algunos de los productos de AWS más populares. Para obtener una lista completa de productos que funcionan con IAM, incluidos enlaces para obtener más información sobre cada uno de ellos, consulte Servicios de AWS que funcionan con IAM.

Uso de IAM con Amazon EC2

- Control del acceso a los recursos de Amazon EC2: describe cómo utilizar las características de IAM para permitir a los usuarios administrar instancias de Amazon EC2, volúmenes, etc.
- Uso de perfiles de instancia: describe cómo utilizar roles de IAM para proporcionar de forma segura credenciales para las aplicaciones que se ejecutan en instancias Amazon EC2 y que necesitan acceso a otros productos de AWS.

Uso de IAM con Amazon S3

- Administración de los permisos de acceso a sus recursos de Amazon S3 - Analiza el modelo de seguridad de Amazon S3 para los buckets y los objetos, que incluye las políticas de IAM.
- Escritura de políticas de IAM: conceder acceso a carpetas específicas de usuario en un bucket de Amazon S3: describe cómo permitir a los usuarios proteger sus propias carpetas en Amazon S3. (para consultar más entradas sobre Amazon S3 y IAM, elija la etiqueta S3 ubicada debajo del título de la entrada del blog).

Uso de IAM con Amazon RDS

- Uso de AWS Identity and Access Management (IAM) para administrar el acceso a los recursos de Amazon RDS - Describe cómo utilizar IAM para controlar el acceso a las instancias de la base de datos, las instantáneas de la base de datos y más.
- Una introducción a los permisos de nivel de recursos de RDS: describe cómo utilizar IAM para controlar el acceso a determinadas instancias de Amazon RDS.

Uso de IAM con Amazon DynamoDB

- Uso de IAM para controlar el acceso a los recursos de DynamoDB: describe cómo utilizar IAM para permitir a los usuarios administrar las tablas e índices de DynamoDB.
- En el siguiente vídeo (8:55) se explica cómo proporcionar control de acceso a atributos o elementos individuales (o ambos) de bases de datos de DynamoDB.

  Primeros pasos con el control de acceso de grano fino para DynamoDB

Prácticas de seguridad generales

Obtenga instrucciones y consejos de expertos sobre las mejores formas de proteger sus recursos y Cuenta de AWS:

- Best Practices for Security, Identity, &, Compliance (Prácticas recomendadas de seguridad, identidad y cumplimiento): encuentre recursos sobre cómo administrar la seguridad en los productos y Cuentas de AWS que incluyen sugerencias para la arquitectura de seguridad, el uso de IAM, el cifrado y la seguridad de los datos, etc.
- Administración de identidades y accesos: el marco AWS Well-Architected Framework ayuda a comprender los conceptos clave, los principios de diseño y las prácticas recomendadas en cuanto a arquitectura para diseñar y ejecutar cargas de trabajo en la nube.
- Prácticas recomendadas de seguridad en IAM: ofrece recomendaciones sobre formas de utilizar IAM con el fin de proteger sus recursos y Cuenta de AWS.
- Guía del usuario de AWS CloudTrail— Utilice AWS CloudTrail para realizar un seguimiento de un historial de llamadas a la API realizadas a AWS y almacenar esa información en archivos de registro. Esto le ayuda a determinar los usuarios y las cuentas que obtuvieron acceso a los recursos de su cuenta, cuándo se realizaron las llamadas, qué acciones se solicitaron, etc.

Recursos generales de

Examine estos recursos para obtener más información sobre IAM y AWS.

- Información de producto de IAM: información general sobre el producto de AWS Identity and Access Management.
- AWS re:Post para AWS Identity and Access Management: visite AWS re:Post para abordar preguntas técnicas relacionadas con IAM con la comunidad de AWS.

- Clases y talleres: enlaces a cursos basados en roles y especializados, además de laboratorios autoguiados para ayudarlo a desarrollar sus conocimientos sobre AWS y obtener experiencia práctica.
- Centro para desarrolladores de AWS: explore los tutoriales, descargue herramientas y obtenga información sobre los eventos para desarrolladores de AWS.
- Herramientas para desarrolladores de AWS: enlaces a herramientas para desarrolladores, SDK, conjuntos de herramientas de IDE y herramientas de línea de comandos para desarrollar y administrar aplicaciones de AWS.
- Centro de recursos de introducción: aprenda a configurar su Cuenta de AWS, únase a la comunidad de AWS y lance su primera aplicación.
- Tutoriales prácticos: comience con tutoriales paso a paso antes de lanzar su primera aplicación en AWS.
- Documentos técnicos de AWS: enlaces a una lista completa de documentos técnicos de AWS que tratan una gran variedad de temas técnicos, como arquitecturas, seguridad y economía de la nube, escritos por arquitectos de soluciones de AWS o expertos técnicos.
- AWS SupportCentro de : punto para crear y administrar los casos de AWS Support. También incluye enlaces a otros recursos útiles como foros, preguntas técnicas frecuentes, estado de los servicios y de AWS Trusted Advisor.
- AWS Support: la página web principal para obtener información acerca de AWS Support, un canal de soporte individualizado y de respuesta rápida que le ayudará a crear y ejecutar aplicaciones en la nube.
- Contacte con nosotros – Un punto central de contacto para las consultas relacionadas con la facturación AWS, cuentas, eventos, abuso y demás problemas.
- AWSTérminos del sitio de : información detallada sobre nuestros derechos de autor y marca comercial, su cuenta, licencia y acceso al sitio, entre otros temas.

Llamar a la API de IAM mediante solicitudes de consulta HTTP

Contenido

Puede acceder a los servicios de IAM y AWS STS mediante programación con la API de consulta. Las solicitudes de la API de consultas son solicitudes HTTPS que deben contener un parámetro `Action` que indique la acción que se va a realizar. IAM y AWS STS admiten solicitudes GET y POST para todas las acciones. Es decir, la API no requiere que utilice GET para algunas acciones y POST para otras. Sin embargo, las solicitudes GET están sujetas al tamaño de limitación de una URL, aunque este límite depende del navegador, que suele ser un límite de 2048 bytes. Por lo tanto, para las solicitudes de la API de consultas que requieran tamaños más grandes, debe utilizar una solicitud POST.

La respuesta es un documento XML. Para obtener más información acerca de la respuesta, consulte las páginas de cada acción en la Referencia de API de IAM o en la Referencia de API AWS Security Token Service.

> **Tip**
>
> En lugar de hacer llamadas directas a las operaciones de la API de IAM o AWS STS, puede utilizar uno de los SDK de AWS. Los SDK de AWS constan de bibliotecas y código de muestra para diversos lenguajes de programación y plataformas (Java, Ruby, .NET, iOS, Android, etc.). Los SDK proporcionan una forma cómoda de crear acceso mediante programación a IAM y AWS. Por ejemplo, los SDK se encargan de tareas como firmar solicitudes criptográficamente (véase más abajo), administrar los errores y reintentar las solicitudes de forma automática. Para obtener información sobre los SDK de AWS, incluido cómo descargarlos e instalarlos, consulte la página Herramientas para Amazon Web Services.

Para obtener más información acerca de las acciones y los errores de la API, consulte la Referencia de la API de IAM o la Referencia de la API de AWS Security Token Service.

Puntos de conexión

IAM y AWS STS tienen cada uno un único punto de enlace global:

- (IAM) https://iam.amazonaws.com
- (AWS STS) https://sts.amazonaws.com

> **Note**
>
> AWS STS también admite el envío de solicitudes a puntos de enlace regionales, además del punto de enlace global. Para poder utilizar AWS STS en una región, primero debe activar STS en esa región para su Cuenta de AWS. Para obtener más información sobre cómo activar regiones adicionales de AWS STS, consulte Administrar AWS STS en una Región de AWS.

Para obtener más información acerca de los puntos de enlace de AWS y las regiones de todos los servicios, consulte Puntos de enlace y cuotas de servicio en la Referencia general de AWS.

HTTPS obligatorio

Dado que la API de consultas devuelve información confidencial como, por ejemplo, credenciales de seguridad, debe utilizar HTTPS con todas las solicitudes de la API.

Firma de solicitudes de la API de IAM

Las solicitudes deben firmarse con un ID de clave de acceso y una clave de acceso secreta. Es absolutamente recomendable que no utilice las credenciales de su usuario Usuario raíz de la cuenta de AWS para el trabajo diario con IAM. Puede utilizar las credenciales de un usuario de IAM o utilizar AWS STS para generar credenciales de seguridad temporales.

Para firmar las solicitudes de la API, le recomendamos que utilice la versión 4 Signature de AWS. Para obtener más información cómo utilizar Signature Version 4, vaya a Proceso de firma de Signature Version 4 en la Referencia general de AWS.

Si necesita utilizar Signature Version 2, encontrará información sobre este proceso de firma en la Referencia general de AWS.

Para obtener más información, consulte lo siguiente:

- Credenciales de seguridad de AWS. Ofrece información general sobre los tipos de credenciales que se utiliza para obtener acceso a AWS.
- Prácticas recomendadas de seguridad en IAM. Presenta una lista de sugerencias para utilizar el servicio de IAM para ayudar a proteger sus recursos de AWS.
- Credenciales de seguridad temporales en IAM. Describe cómo crear y utilizar credenciales de seguridad temporales.

Historial de revisión de IAM

En la tabla siguiente se describen las actualizaciones principales de la documentación de IAM.

Cambio	Descripción	Fecha
AccessAnalyzerServiceRolePolicy : permisos agregados	El analizador de acceso de IAM agregó compatibilidad con permisos para recuperar el estado actual del bloqueo de acceso público para instantáneas de Amazon EC2 a los permisos a nivel de servicio de AccessAnalyzerServiceRolePolicy.	23 de enero de 2024
AccessAnalyzerServiceRolePolicy : permisos agregados	El analizador de acceso de IAM agregó flujos y tablas de DynamoDB a los permisos a nivel de servicio de AccessAnalyzerServiceRolePolicy.	11 de enero de 2024
AccessAnalyzerServiceRolePolicy : permisos agregados	El analizador de acceso de IAM agregó buckets de directorio de Amazon S3 a los permisos de nivel de servicio de AccessAnalyzerServiceRolePolicy.	1 de diciembre de 2023
IAMAccessAnalyzerReadOnlyAccess : permisos agregados	El Analizador de acceso de IAM agregó permisos a IAMAccessAnalyzerReadOnlyAccess para que pueda comprobar si las actualizaciones de sus	26 de noviembre de 2023

	políticas otorgan acceso adicional. El Analizador de acceso de IAM requiere este permiso para realizar comprobaciones de políticas en sus políticas.	
El Analizador de acceso de IAM agregó analizadores de acceso no utilizados	El Analizador de acceso de IAM simplifica la inspección del acceso no utilizado para orientarle hacia el nivel de privilegio mínimo. El Analizador de acceso de IAM analiza continuamente sus cuentas para identificar el acceso no utilizado y crea un panel centralizado con los resultados.	26 de noviembre de 2023
El Analizador de acceso de IAM agregó verificaciones de políticas personalizadas	El Analizador de acceso de IAM ahora proporciona verificaciones de políticas personalizadas para validar que las políticas de IAM cumplen sus estándares de seguridad antes de las implementaciones.	26 de noviembre de 2023

AccessAnalyzerServiceRolePolicy : permisos agregados	El Analizador de acceso de IAM agregó acciones de IAM a los permisos de nivel de servicio de AccessAnalyzerServiceRolePolicy para admitir las siguientes acciones: • Listado de las entidades de una política • Generación de información sobre los últimos accesos al servicio • Listado de Información de clave de acceso	26 de noviembre de 2023
Última acción a la que se ha accedido y asistencia en la elaboración de políticas para más de 60 servicios y acciones adicionales	IAM ahora admite información sobre el último acceso a una acción y genera políticas con información relativa al nivel de acción para más de 60 servicios adicionales, junto con una lista de las acciones para las que se dispone de información sobre el último acceso a una acción.	1 de noviembre de 2023
Asistencia a la última acción a la que se ha accedido para 140 servicios	IAM proporciona ahora información sobre la última acción a la que se ha accedido para más de 140 servicios , junto con una lista de las acciones para las que se dispone de información sobre la última acción a la que se ha accedido.	14 de septiembre de 2023

Compatibilidad con varios dispositivos de autenticación multifactor (MFA) para usuarios raíz y usuarios de IAM	Ahora puede agregar hasta ocho dispositivos MFA por usuario, incluidas claves de seguridad FIDO, contraseña temporal de un solo uso (TOTP) con aplicaciones de autenticador virtual o tokens TOTP de hardware.	16 de noviembre de 2022
Compatibilidad del Analizador de acceso de IAM con nuevos tipos de recursos	El Analizador de acceso de IAM agregó compatibilidad con los siguientes tipos de recursos: • Instantáneas de volúmenes de Amazon EBS • Repositorios de Amazon ECR • Sistemas de archivos de Amazon EFS • Instantáneas de base de datos de Amazon RDS • Instantáneas de clúster de base de datos de Amazon RDS • Temas de Amazon SNS	25 de octubre de 2022
Obsolescencia de U2F y actualización de WebAuthn y FIDO	Se eliminaron las menciones a U2F como opción de MFA y se agregó información sobre las claves de seguridad WebAuthn, FIDO2 y FIDO.	31 de mayo de 2022

Actualizaciones de la resiliencia en IAM	Se agregó información sobre cómo mantener el acceso a las credenciales de IAM cuando un evento interrumpe la comunicación entre Regiones de AWS.	16 de mayo de 2022
Nuevas claves de condición global para los recursos	Ahora puede controlar el acceso a los recursos en función de la cuenta, la unidad organizativa o la organización en AWS Organizations que contiene sus recursos. Puede utilizar las claves de condición global `aws:ResourceAccount`, `aws:ResourceOrgID` y `aws:ResourceOrgPaths` en una política de IAM.	27 de abril de 2022
Ejemplos de código de IAM con SDK de AWS	Se agregaron ejemplos de código que muestran cómo utilizar IAM con un kit de desarrollo de software (SDK) de AWS. Los ejemplos están divididos en extractos de código que muestran cómo llamar a funciones de servicio individuales y ejemplos que muestran cómo hacer una tarea específica llamando a múltiples funciones dentro del mismo servicio.	7 de abril de 2022

Actualizaciones del diagrama de flujo lógico de evaluación de políticas	Actualizaciones del diagrama de flujo lógico de evaluación de políticas y texto relacionado en la sección Cómo determinar si una solicitud se permite o se deniega en una cuenta.	17 de noviembre de 2021
Actualizaciones de prácticas recomendadas de seguridad	Se agregó información sobre la creación de usuarios administradores de IAM en lugar de utilizar credenciales de usuario raíz, se eliminó la práctica recomendada de utilizar grupos de usuarios para asignar permisos a los usuarios de IAM y se aclaró cuándo utilizar políticas administradas en lugar de políticas insertadas.	5 de octubre de 2021
Actualizaciones del tema lógico de evaluación de políticas para políticas basadas en recursos	Se agregó información sobre el impacto de las políticas basadas en recursos y los distintos tipos de entidades principales en la misma cuenta.	5 de octubre de 2021
Actualizaciones de claves de condición de valor único y multivalor	Las diferencias entre las claves de condición de valor único y multivalor se explican ahora con más detalle. El tipo de valor se agregó a cada clave de contexto de condición global de AWS.	30 de septiembre de 2021

El Analizador de acceso de IAM admite puntos de acceso de varias regiones de Amazon S3	El Analizador de acceso de IAM identifica los buckets de Amazon S3 que permiten el acceso público y entre cuentas, incluidos los que utilizan los Puntos de acceso de varias regiones de Amazon S3.	2 de septiembre de 2021
Actualizaciones de política administrada de AWS: actualización de una política existente	El Analizador de acceso de IAM actualizó una política administrada de AWS.	2 de septiembre de 2021
Más servicios admitidos para la generación de políticas a nivel de acción	El Analizador de acceso de IAM puede generar políticas de IAM con información de actividad de acceso de nivel de acción para servicios de AWS adicionales.	24 de agosto de 2021
Generación de políticas de IAM para seguimientos entre cuentas	Ahora puede utilizar el Analizador de acceso de IAM para generar políticas detalladas basadas en su actividad de acceso mediante un seguimiento de AWS CloudTrail en una cuenta diferente, por ejemplo, un seguimiento AWS Organizations centralizado.	18 de agosto de 2021

Verificaciones adicionales de políticas del Analizador de acceso de IAM	El Analizador de acceso de IAM amplió la validación de políticas mediante la adición de nuevas verificaciones de políticas que validan las condiciones incluidas en las políticas de IAM. Estas verificaciones analizan el bloque de condición en la declaración de política e informan de advertencias de seguridad, errores y sugerencias junto con recomendaciones procesables. El Analizador de acceso de IAM agregó las siguientes verificaciones de políticas: • Error: formato principal de servicio no válido • Error: falta la clave de etiqueta en la condición • Advertencia de seguridad : denegar NotAction con clave de condición de etiqueta no compatible para el servicio • Advertencia de seguridad : denegar con clave de condición de etiqueta no admitida para el servicio • Advertencia de seguridad : faltan las claves de condición emparejadas	29 de junio de 2021

	• Sugerencia: permitir NotAction con clave de condición de etiqueta no compatible para el servicio • Sugerencia: permitir con clave de condición de etiqueta no compatible para el servicio	
Soporte de la última acción a la que se ha accedido por última vez para más servicios	Ahora puede ver la información de la última acción a la que se ha accedido en la consola de IAM sobre la última vez que una entidad principal de IAM utilizó una acción para los siguientes servicios: acciones de administración de Amazon EC2, IAM, Lambda y Amazon S3. También puede utilizar la AWS CLI o la API de AWS para recuperar los datos de informes. Puede utilizar esta información para identificar permisos innecesarios, de modo que pueda perfeccionar sus políticas de IAM para que cumplan mejor con el principio de privilegio mínimo.	19 de abril de 2021

Monitorear y controlar las acciones realizadas con roles asumidos	Los administradores pueden configurar los roles de IAM para requerir que las identidad es pasen una identidad de fuente, que se ha registrad o en AWS CloudTrail. La revisión de la información de identidad de fuente ayuda a los administradores a determinar quién o qué realizó acciones con sesiones de rol asumidas.	13 de abril de 2021
Generar políticas de IAM basadas en la actividad de acceso	Ahora puede utilizar el Analizador de acceso de IAM para generar políticas detalladas basadas en su actividad de acceso que se encuentra en su AWS CloudTrail.	7 de abril de 2021
Verificaciones de políticas del Analizador de acceso de IAM	El Analizador de acceso de IAM ofrece ahora más de 100 verificaciones de políticas con recomendaciones procesabl es durante la creación de políticas.	16 de marzo de 2021
Opciones de validación de políticas ampliadas	Validación de políticas ampliadas disponible en la consola de IAM, API de AWS, y AWS CLI mediante verificaciones de políticas en el Analizador de acceso de IAM para ayudarle a crear políticas JSON seguras y funcionales.	15 de marzo de 2021

Etiquetado de recursos de IAM	Ahora puede etiquetar recursos de IAM adicionales utilizando un par clave-valor de etiqueta.	11 de febrero de 2021
Política de contraseñas predeterminada para los usuarios de IAM	Si no establece una política de contraseñas personalizada para su cuenta de Cuenta de AWS, las contraseñas de los usuarios de IAM deben cumplir ahora la política de contraseñas predeterminada de AWS.	18 de noviembre de 2020
Las páginas de las acciones, los recursos y las claves de condición para los servicios de AWS se movieron	Cada servicio de AWS puede definir acciones, recursos y claves de contexto de condición para utilizarse en las políticas de IAM. Ahora puede encontrar la lista de los servicios de AWS, sus acciones, recursos y claves de contexto de condiciones en la referencia de autorizaciones de servicio.	16 de noviembre de 2020

Mayor duración de la sesión de rol de los usuarios de IAM	Los usuarios de IAM ahora pueden tener una duración de sesión de rol más larga al cambiar roles en el AWS Management Console, lo que reduce las interrupciones debidas al vencimiento de la sesión. A los usuarios se les concede la duración máxima de sesión establecida para el rol, o el tiempo restante en la sesión del usuario de IAM, lo que sea menor.	24 de julio de 2020
Utilice Service Quotas para solicitar aumentos rápidos para entidades de IAM	Puede solicitar aumentos de cuota para cuotas de IAM ajustables mediante la consola Service Quotas. Ahora, algunos aumentos se aprueban automáticamente en Service Quotas y están disponibles en su cuenta en pocos minutos. Las solicitud es más grandes se envían a AWS Support.	25 de junio de 2020

La información de acceso reciente de IAM ahora incluye acciones de administración de Amazon S3	Además de los datos sobre los últimos accesos al servicio, ahora puede ver en la consola de IAM información sobre la última vez que una entidad principal de IAM utilizó una acción de Amazon S3. También puede utilizar la AWS CLI o la API de AWS para recuperar los informes de datos. El informe contiene detalles sobre las acciones y los servicios permitido s a los que las entidades principales intentaron acceder la última vez y cuándo lo hicieron. Puede utilizar esta información para identificar permisos innecesarios, de modo que pueda perfeccionar sus políticas de IAM para que cumplan mejor con el principio de privilegio mínimo.	3 de junio de 2020
Adición de capítulos de seguridad	El capítulo de seguridad le ayuda a comprender cómo configurar IAM y AWS STS para cumplir sus objetivos de seguridad y cumplimie nto de normas. También puede aprender a utilizar otros servicios de AWS que lo ayuden a monitorear y proteger los recursos de IAM.	29 de abril de 2020

sts:RoleSessionName	Ahora puede escribir una política que conceda permisos en función del nombre de sesión que especifique una entidad principal al asumir un rol.	21 de abril de 2020
Actualización de la página de inicio de sesión de AWS	Cuando inicia sesión en la página principal de inicio de sesión de AWS, no puede elegir iniciar sesión como el Usuario raíz de la cuenta de AWS o como un usuario de IAM. Cuando lo haga, la etiqueta de la página indica si debe proporcionar la dirección de correo electrónico de su usuario raíz o la informaci ón de su cuenta de usuario de IAM. Esta documentación incluye capturas de pantalla actualizadas para ayudarlo a comprender las páginas de inicio de sesión de AWS.	4 de marzo de 2020

Claves de condición aws:ViaAWSService y aws:CalledVia	Ahora puede escribir una política para limitar si los servicios pueden realizar solicitudes en nombre de una entidad principal de IAM (usuario o rol). Cuando una entidad principal realiza una solicitud a un servicio de AWS, ese servicio puede utilizar las credenciales de la entidad principal para realizar solicitudes posteriores a otros servicios. Utilice la clave de condición `aws:ViaAWSService` para determinar si algún servicio realiza una solicitud utilizando las credenciales de una entidad principal. Utilice las claves de condición `aws:CalledVia` para determinar si servicios específicos realizan una solicitud con las credenciales de una entidad principal.	20 de febrero de 2020
El simulador de políticas agrega compatibilidad con límites de permisos	Ahora puede probar el efecto de los límites de permisos en las entidades de IAM con el simulador de políticas de IAM.	23 de enero de 2020

Evaluación de políticas entre cuentas	Ahora puede saber cómo evalúa AWS las políticas para el acceso entre cuentas. Esto ocurre cuando un recurso de una cuenta de confianza incluye una política basada en recursos que permite que una entidad principal de otra cuenta tenga acceso al recurso. La solicitud debe estar permitida en ambas cuentas.	2 de enero de 2020
Etiquetas de sesión	Ahora puede incluir etiquetas al asumir un rol o federar un usuario en AWS STS. Al realizar las operaciones `AssumeRole` o `GetFederationToken`, puede pasar las etiquetas de sesión como atributos. Cuando realiza las operaciones `AssumeRoleWithSAML` o `AssumeRoleWithWebIdentity`, puede pasar atributos de sus identidades corporativas a AWS.	22 de noviembre de 2019

Controlar el acceso para grupos de cuentas de Cuentas de AWS en AWS Organizations	Ahora puede hacer referencia a unidades organizativas (OU) desde AWS Organizations en políticas de IAM. Si utiliza Organizations para organizar sus cuentas en unidades organizativas, puede exigir que las entidades principales pertenezcan a una unidad organizativa específica antes de concederles acceso a sus recursos. Las entidades principales incluyen Usuario raíz de la cuenta de AWS, usuarios de IAM y roles de IAM. Para ello, especifique la ruta de acceso de la unidad organizativa en la clave de condición `aws:PrincipalOrgPaths` de las políticas.	20 de noviembre de 2019
Último uso del rol	Ahora puede ver la fecha, la hora y la región en la que se utilizó un rol por última vez. Esta información también le ayuda a identificar los roles no utilizados en su cuenta. Puede utilizar la AWS Management Console, AWS CLI y la API de AWS para ver información acerca de cuándo se utilizó un rol por última vez.	19 de noviembre de 2019

Actualización a la página de claves de contexto de condición global	Ahora puede saber cuándo se incluye cada una de las claves de condición globales en el contexto de una solicitud . También puede navegar a cada clave con mayor facilidad utilizando la tabla de contenido (TOC) de página. La informaci ón de la página le ayuda a escribir políticas más precisas. Por ejemplo, si los empleados utilizan la federación con roles de IAM, debe utilizar la clave `aws:userId` y no la clave `aws:userName` . La clave `aws:userName` se aplica únicamente a los usuarios de IAM y no a los roles.	6 de octubre de 2019

ABAC en AWS	Descubra cómo funciona el control de acceso basado en atributos (ABAC) en AWS utilizando etiquetas y cómo se compara con el modelo de autorización tradicional de AWS. Utilice el tutorial de ABAC para obtener informaci ón sobre cómo crear y probar una política que permita a roles de IAM con etiquetas principales obtener acceso a los recursos de con etiquetas coincidentes. Esta estrategi a permite a las personas ver o editar solo los recursos de AWS necesarios para sus trabajos.	3 de octubre de 2019
Operación GetAccessKeyInfo de AWS STS	Puede revisar las claves de acceso de AWS en su código para determinar si las claves proceden de una cuenta de su propiedad. Puede transferi r un ID de clave de acceso mediante el comando de la AWS CLI `aws sts get-access-key-info` o la operación de la API de AWS `GetAccessKeyInfo` .	24 de julio de 2019

Visualización de la información sobre los últimos accesos al servicio de Organizations en IAM	Ahora puede consultar la información de una entidad o política de AWS Organizations sobre los últimos accesos a un servicio en la sección AWS Organizations de la consola de IAM. También puede utilizar la AWS CLI o la API de AWS para recuperar los informes de datos. Estos datos incluyen información sobre la última vez que las entidades principales de una cuenta de Organizations intentaron obtener acceso a los servicios permitidos y cuándo. Puede utilizar esta información para identificar permisos innecesarios, de modo que pueda perfeccionar sus políticas de Organizations para que cumplan mejor con el principio de privilegio mínimo.	20 de junio de 2019
Uso de una política administrada como una política de sesión	Ahora puede pasar hasta 10 ARN de política administrada cuando asume una función. Esto le permite limitar los permisos de las credenciales temporales del rol.	7 de mayo de 2019

Compatibilidad de la región de AWS STS de los tokens de sesión para el punto de enlace global	Ahora puede elegir si desea utilizar la versión 1 o la versión 2 de tokens de punto de conexión global. Los tokens de la versión 1 son válidos únicamente en las regiones de AWS que están disponibles de forma predeterminada. Estos tokens no funcionarán en regiones habilitadas manualmente, como, por ejemplo, Asia Pacífico (Hong Kong). Los tokens de la versión 2 son válidos en todas las regiones. Sin embargo, los tokens de versión 2 son más largos y podrían afectar a los sistemas en los que almacena tokens temporalmente.	26 de abril de 2019
Permitir la habilitación y desactivación de regiones de AWS	Ahora puede crear una política que permite a un administrador habilitar y deshabilitar la región Asia Pacífico (Hong Kong) (ap-east-1).	24 de abril de 2019

Página Mis credenciales de seguridad del usuario de IAM	Los usuarios de IAM ahora pueden administrar sus propias credenciales en la página Mis credenciales de seguridad. En esta página de la AWS Management Console, se muestra información de la cuenta, como el ID de cuenta y el ID de usuario canónico. Los usuarios también pueden ver y editar sus propias contraseñas, claves de acceso, certificados X.509, claves SSH y credenciales de Git.	24 de enero de 2019
API del Asesor de acceso	Ahora puede utilizar la AWS CLI y la API de AWS para ver información sobre los últimos accesos a un servicio.	7 de diciembre de 2018
Etiquetado de usuarios y roles de IAM	Ahora puede utilizar etiquetas de IAM para agregar atributos personalizados a una identidad (rol o usuario de IAM) mediante un par de clave-valor de etiqueta. También puede utilizar etiquetas para controlar un acceso de identidad a recursos o para controlar qué etiquetas se pueden asociar a una identidad.	14 de noviembre de 2018

Claves de seguridad U2F	A partir de ahora, puede utilizar claves de seguridad U2F como opción de autenticación multifactor (MFA) para iniciar sesión en la AWS Management Console.	25 de septiembre de 2018
Compatibilidad con puntos de conexión de Amazon VPC	Ahora puede establecer una conexión privada entre su VPC y AWS STS en la región EE. UU. Oeste (Oregón).	31 de julio de 2018
Límites de permisos	Con esta nueva característica, resulta más fácil conceder a los empleados de confianza la posibilidad de administrar los permisos de IAM sin concederles también acceso administrativo pleno a IAM.	12 de julio de 2018
aws:PrincipalOrgID	Una nueva clave de condición que proporciona una forma más sencilla de controlar el acceso a los recursos de AWS especificando la organización de AWS de las entidades principales de IAM.	17 de mayo de 2018
aws:RequestedRegion	Una nueva clave de condición proporciona una forma más sencilla de utilizar las políticas de IAM para controlar el acceso a las regiones de AWS.	25 de abril de 2018

Mayor duración de la sesión para los roles de IAM	Un rol de IAM ahora puede tener una duración de la sesión de 12 horas.	28 de marzo de 2018
Flujo de trabajo de creación de roles actualizado	Un nuevo flujo de trabajo mejora el proceso de creación de relaciones de confianza y de asociación de permisos a los roles.	8 de septiembre de 2017
Proceso de inicio de sesión en su cuenta de Cuenta de AWS	Se actualizó la experiencia de inicio de sesión de AWS para permitir al usuario raíz y a los usuarios de IAM utilizar el enlace Iniciar sesión en la consola de la página de inicio de la AWS Management Console.	25 de agosto de 2017
Ejemplos de políticas de IAM	La actualización de la documentación contiene más de 30 políticas de ejemplo.	2 de agosto de 2017
Prácticas recomendadas de IAM	La información añadida a la sección Usuarios de la consola de IAM permite seguir las prácticas recomendadas de IAM con mayor facilidad.	5 de julio de 2017
Recursos de Auto Scaling	Los permisos de nivel de recursos pueden controlar el acceso y los permisos para los recursos del escalado automático.	16 de mayo de 2017

Bases de datos de Amazon RDS para MySQL y Amazon Aurora	Los administradores de base de datos pueden asociar usuarios de base de datos a usuarios y roles de IAM y, por lo tanto, administrar el acceso de los usuarios a todos los recursos de AWS desde una única ubicación.	24 de abril de 2017
Roles vinculados al servicio	Los roles vinculados a servicios proporcionan una forma más fácil y segura de delegar permisos a los servicios de AWS.	19 de abril de 2017
Resúmenes de políticas	Los nuevos resúmenes de políticas hacen que sea más fácil entender los permisos en las políticas de IAM.	23 de marzo de 2017