aws

Guía del usuario

AWS AppConfig

AWS AppConfig: Guía del usuario

Table of Contents

¿Qué es AWS AppConfig?

Las marcas de características y las configuraciones dinámicas de AWS AppConfig ayudan a los creadores de software a ajustar de forma rápida y segura el comportamiento de las aplicaciones en los entornos de producción sin implementaciones de código completas. AWS AppConfig acelera la frecuencia de publicación del software, mejora la resiliencia de las aplicaciones y ayuda a abordar los problemas emergentes con mayor rapidez. Con las marcas de características, puede lanzar gradualmente nuevas capacidades para los usuarios y medir el impacto de esos cambios antes de implementar completamente las nuevas capacidades para todos los usuarios. Con las marcas operativas y las configuraciones dinámicas, puede actualizar las listas de bloqueados, las listas de permitidos, los límites de limitación, la verbosidad de los registros y realizar otros ajustes operativos para responder rápidamente a los problemas en los entornos de producción.

> **Note**
>
> AWS AppConfig es una capacidad de AWS Systems Manager.

Mejore la eficiencia y publique los cambios con mayor rapidez

El uso de marcas de características con nuevas capacidades acelera el proceso de publicación de cambios en los entornos de producción. En lugar de confiar en ramas de desarrollo de larga duración que requieren complicadas fusiones antes de una publicación, las marcas de características permiten escribir software mediante un desarrollo basado en troncos. Las marcas de características permiten distribuir de forma segura el código previo a la publicación en una canalización de CI/CD que está oculta a los usuarios. Cuando esté listo para publicar los cambios, puede actualizar la marca de características sin necesidad de implementar código nuevo. Una vez finalizado el lanzamiento, la marca puede seguir funcionando como un interruptor de bloqueo para deshabilitar una nueva característica o capacidad sin necesidad de revertir la implementación del código.

Evite cambios o fallos no intencionados con las características de seguridad integradas

AWS AppConfig ofrece las siguientes características de seguridad para evitar que se activen las marcas de características o se actualicen los datos de configuración, lo que podría provocar errores en las aplicaciones.

- Validadores: un validador garantiza que los datos de configuración sean correctos sintáctica y semánticamente antes de implementar los cambios en los entornos de producción.

- Estrategias de implementación: una estrategia de implementación le permite publicar lentamente los cambios en los entornos de producción en el plazo de minutos u horas.
- Supervisión y reversión automática: AWS AppConfig se integra con Amazon CloudWatch para supervisar los cambios en sus aplicaciones. Si la aplicación deja de funcionar debido a un cambio de configuración incorrecto y ese cambio activa una alarma en CloudWatch, AWS AppConfig revierte automáticamente el cambio para minimizar el impacto en los usuarios de la aplicación.

Implementaciones de marcas de características seguras y escalables

AWS AppConfig se integra con AWS Identity and Access Management (IAM) para proporcionar un acceso detallado y basado en roles al servicio. AWS AppConfig también se integra con AWS Key Management Service (AWS KMS) para el cifrado y AWS CloudTrail para la auditoría. Antes de distribuirlos a clientes externos, todos AWS AppConfig los controles de seguridad de fueron desarrollados y validados inicialmente por clientes internos que utilizan el servicio a gran escala.

Casos de uso de AWS AppConfig

A pesar de que el contenido de la configuración de una aplicación puede variar considerablemente de una aplicación a otra, AWS AppConfig admite los siguientes casos de uso, que abarcan una amplia gama de necesidades de los clientes:

- Marcas de características y conmutadores: ofrezca nuevas capacidades de forma segura a sus clientes en un entorno controlado. Si tiene algún problema, deshaga los cambios al instante.
- Ajuste de la aplicación: introduzca cuidadosamente los cambios en la aplicación y, al mismo tiempo, pruebe el impacto de esos cambios con los usuarios de los entornos de producción.
- Lista de permitidos o lista de bloqueados: controle el acceso a características premium o bloquee instantáneamente a usuarios específicos sin necesidad de implementar código nuevo.
- Almacenamiento de configuración centralizado: mantenga sus datos de configuración organizados y coherentes en todas sus cargas de trabajo. Puede utilizar AWS AppConfig para implementar los datos de configuración almacenados en el almacén de configuración alojado de AWS AppConfig, AWS Secrets Manager, en el almacén de parámetros de Systems Manager o en Amazon S3.

Beneficios de utilizar AWS AppConfig

AWS AppConfig ofrece los siguientes beneficios a su organización:

- Reducir el tiempo de inactividad inesperado para sus clientes

  AWS AppConfig reduce el tiempo de inactividad de las aplicaciones al permitirle crear reglas para validar la configuración. Las configuraciones que no son válidas no se pueden implementar. AWS AppConfig proporciona dos opciones para validar configuraciones.

  - Para la validación sintáctica, puede usar un esquema JSON. AWS AppConfig valida la configuración mediante el esquema JSON para asegurarse de que los cambios de configuración cumplen los requisitos de la aplicación.
  - Para la validación semántica, AWS AppConfig puede llamar a una función de AWS Lambda de su propiedad para validar los datos de su configuración.
- Implementar rápidamente los cambios en un conjunto de destinos

  AWS AppConfig simplifica la administración de aplicaciones a escala mediante la implementación de cambios de configuración desde una ubicación central. AWS AppConfig admite configuraciones almacenadas en el almacén de configuración alojado en AWS AppConfig, parámetros del almacén de parámetros de Systems Manager (SSM), documentos de SSM, y Amazon S3. Puede utilizar AWS AppConfig con aplicaciones alojadas en instancias EC2, AWS Lambda, contenedores, aplicaciones móviles o dispositivos IoT.

  Los destinos no tienen que configurarse con el Agente de SSM ni con el perfil de instancia de IAM que requieren otras capacidades de Systems Manager. Esto significa que AWS AppConfig funciona con instancias no administradas.
- Actualizar aplicaciones sin interrupciones

  AWS AppConfig implementa cambios de configuración en sus destinos en tiempo de ejecución sin un proceso de compilación pesado ni sin retirar sus destinos de servicio.
- Controlar la implementación de cambios en toda la aplicación

  Al implementar cambios de configuración en sus destinos, AWS AppConfig le permite minimizar el riesgo mediante una estrategia de implementación. Las estrategias de implementación le permiten implementar lentamente los cambios de configuración en su flota. Si tiene algún problema durante la implementación, puede revertir el cambio de configuración antes de que llegue a la mayoría de sus hosts.

Cómo funciona AWS AppConfig

En esta sección se proporciona una descripción general de cómo funciona AWS AppConfig y cómo empezar.

1. Identifique los valores de configuración del código que desee administrar en la nube

   Antes de empezar a crear artefactos de AWS AppConfig, le recomendamos que identifique los datos de configuración de su código que desee utilizar para gestionarlos de forma dinámica utilizando AWS AppConfig. Algunos buenos ejemplos son las marcas de características o conmutadores, las listas de permitidos y bloqueados, la verbosidad del registro, los límites de servicio y las reglas de limitación, por mencionar algunos ejemplos.

   Si sus datos de configuración ya están en la nube, puede aprovechar las características de validación, implementación y extensión de AWS AppConfig para agilizar aún más la administración de los datos de configuración.

2. Crear un espacio de nombres de aplicaciones

   Para crear un espacio de nombres, se crea un artefacto de AWS AppConfig denominado aplicación. Una aplicación es simplemente una estructura organizativa, como una carpeta.

3. Crear entornos

   Se definen uno o más entornos para cada aplicación AWS AppConfig. Un entorno es una agrupación lógica de destinos, como aplicaciones en un entorno `Beta` o de `Production`, funciones de AWS Lambda o contenedores. También puede definir entornos para subcomponentes de aplicaciones como, por ejemplo, los componentes `Web`, `Mobile` y `Back-end` para la aplicación.

   Puede configurar alarmas de Amazon CloudWatch para cada entorno. El sistema supervisa las alarmas durante la implementación de la configuración. Si se activa una alarma, el sistema deshace la configuración.

4. Creación de un perfil de configuración

   Un perfil de configuración incluye, entre otras cosas, un URI que permite a AWS AppConfig localizar los datos de configuración en la ubicación almacenada y un tipo de perfil. AWS AppConfig admite dos tipos de perfiles de configuración: marcas de características y configuraciones de formato libre. Los perfiles de configuración de las marcas de características almacenan sus datos en el AWS AppConfig almacén de configuración alojado de y el URI

es simplemente `hosted`. En el caso de los perfiles de configuración de formato libre, puede almacenar los datos en el almacén de configuración alojado de AWS AppConfig o en cualquier servicio de AWS con el que se integre con AWS AppConfig, tal y como se describe en Crear un perfil de configuración de formato libre en AWS AppConfig.

Un perfil de configuración también puede incluir validadores opcionales para garantizar que los datos de configuración sean correctos sintáctica y semánticamente. AWS AppConfig realiza una comprobación mediante los validadores al iniciar una implementación. Si se detecta algún error, la implementación se revierte a los datos de configuración anteriores.

5. Implementar datos de configuración

   Al crear una nueva implementación, puede especificar las siguientes opciones:

   - ID de la aplicación
   - ID del perfil de configuración
   - Una versión de configuración
   - Un ID de entorno en el que desea implementar los datos de configuración
   - Un ID de estrategia de implementación que define la rapidez con la que desea que se apliquen los cambios

   Al llamar a la acción de la API StartDeployment, AWS AppConfig lleva a cabo las siguientes tareas:

   1. Recupera los datos de configuración del almacén de datos subyacente mediante el URI de ubicación del perfil de configuración.
   2. Comprueba que los datos de configuración sean correctos sintáctica y semánticamente utilizando los validadores que especificó al crear su perfil de configuración.
   3. Guarda en caché una copia de los datos para que la aplicación pueda recuperarlos. Esta copia en caché se denomina datos implementados.

6. Recupera la configuración

   Puede configurar el agente de AWS AppConfig como un host local y hacer que el agente busque en AWS AppConfig actualizaciones de configuración. El agente llama a las acciones de API StartConfigurationSession y GetLatestConfiguration y almacena en caché los datos de configuración de forma local. Para recuperar los datos, la aplicación realiza una llamada HTTP al servidor localhost. AWS AppConfig El agente admite varios casos de uso, como se describe en Métodos de recuperación simplificados.

Si su caso de uso no admite el agente de AWS AppConfig, puede configurar la aplicación para que busque en AWS AppConfig actualizaciones de configuración mediante una llamada directa a las acciones de API StartConfigurationSession y GetLatestConfiguration.

Introducción a AWS AppConfig

Los siguientes recursos pueden serle de ayuda cuando trabaje directamente con AWS AppConfig.

Vea más videos de AWS en el Canal de YouTube de Amazon Web Services.

Los siguientes blogs pueden ayudarle a obtener más información acerca de AWS AppConfig y sus capacidades:

- Uso de AWS AppConfig marcas de características
- Prácticas recomendadas para validar las marcas de características y los datos de configuración de AWS AppConfig

SDK

Para obtener información acerca de los SDK de AWS AppConfig de lenguajes específicos, consulte los siguientes recursos:

- AWS Command Line Interface
- SDK de AWS para .NET
- AWS SDK para C++
- AWS SDK para Go
- AWS SDK para Java V2
- AWS SDK para JavaScript
- SDK de AWS para PHP V3
- SDK de AWS para Python
- SDK de AWS para Ruby V3

Precios de AWS AppConfig

El precio AWS AppConfig es de pago por uso en función de los datos de configuración y de la recuperación de marcas de características. Recomendamos utilizar el agente de AWS AppConfig para ayudar a optimizar los costos. Para obtener más información, consulte Precios de AWS Systems Manager.

Cuotas de AWS AppConfig

La información sobre los puntos de conexión y las Service Quotas de AWS AppConfig, junto con otras cuotas de Systems Manager, se encuentra en la Referencia general de Amazon Web Services.

> **Note**
>
> Para obtener información acerca de las cuotas para los servicios que almacenan configuraciones de AWS AppConfig, consulte Acerca de las cuotas y limitaciones de los almacenes de configuraciones.

Con AWS AppConfig figuración

Si aún no lo ha hecho, regístrese para obtener un usuario administrativo Cuenta de AWS y cree uno.

Inscríbase en un Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirse a 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.

   Cuando te registras en una Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea una. 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 ejecutar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado 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.

Cómo crear un usuario administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión AWS Management Consolecomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. 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 Cuenta de AWS raíz (consola) en la Guía del usuario de IAM.

Crear un usuario administrativo

1. Activar IAM Identity Center

   Consulte las instrucciones en Enabling 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 su uso Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Inicio de sesión como usuario administrativo

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

Conceder acceso programático

Los usuarios necesitan acceso programático si quieren interactuar con personas AWS ajenas a. AWS Management Console La forma de conceder el acceso programático depende del tipo de usuario que acceda. AWS

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

¿Qué usuario necesita acceso programático?	Para	De
Identidad del personal (Usuarios administrados en el Centro de identidades de IAM)	Usa credenciales temporales para firmar las solicitudes programáticas a los AWS CLI AWS SDK o las API. AWS	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Configuración del uso AWS IAM Identity Center en AWS CLI la Guía del AWS Command Line Interface usuario. • Para ver AWS los SDK, las herramientas y las AWS API, consulte la autenticación del IAM Identity Center en la Guía de referencia de AWS los SDK y las herramientas.
IAM	Utilice credenciales temporales para firmar las solicitudes programáticas a los AWS SDK o las AWS CLI API. AWS	Siga las instrucciones de Uso de credenciales temporales con AWS recursos de la Guía del usuario de IAM.
IAM	(No recomendado) Utilice credenciales de larga duración para firmar las solicitudes programáticas a los AWS CLI AWS SDK o las API. AWS	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Autenticación con credenciales de usuario de IAM en la Guía del usuario.AWS Command Line Interface • Para obtener información AWS sobre los SDK y las herramientas, consulte Autenticarse con credencia

¿Qué usuario necesita acceso programático?	Para	De
		les de larga duración en la Guía de referencia de los AWS SDK y las herramien tas. • Para ver AWS las API, consulte Administrar las claves de acceso para los usuarios de IAM en la Guía del usuario de IAM.

(Opcional) Configure los permisos para la reversión en función de las alarmas CloudWatch

Puedes configurarlo AWS AppConfig para volver a una versión anterior de una configuración en respuesta a una o más CloudWatch alarmas de Amazon. Al configurar una implementación para responder a CloudWatch las alarmas, se especifica un rol AWS Identity and Access Management (de IAM). AWS AppConfig requiere este rol para poder monitorear CloudWatch las alarmas.

> **Note**
>
> El rol de IAM debe pertenecer a la cuenta vigente. De forma predeterminada, solo AWS AppConfig puede monitorear las alarmas propiedad de la cuenta corriente. Si desea configurarlo AWS AppConfig para revertir las implementaciones en respuesta a las métricas de una cuenta diferente, debe configurar las alarmas entre cuentas. Para obtener más información, consulta la CloudWatch consola multicuentas entre regiones en la Guía CloudWatch del usuario de Amazon.

Utilice los siguientes procedimientos para crear una función de IAM que permita la reversión en función AWS AppConfig de las alarmas. CloudWatch Esta sección contiene los procedimientos siguientes.

1. Paso 1: Cree la política de permisos para la reversión en función de las alarmas CloudWatch

2. Paso 2: Cree la función de IAM para la reversión en función de las alarmas CloudWatch
3. Paso 3: Añadir una relación de confianza

Paso 1: Cree la política de permisos para la reversión en función de las alarmas CloudWatch

Utilice el siguiente procedimiento para crear una política de IAM que dé AWS AppConfig permiso para llamar a la acción de la `DescribeAlarms` API.

Para crear una política de permisos de IAM para la reversión en función de las alarmas CloudWatch

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas) y, a continuación, seleccione Create policy (Crear política).
3. En la página Crear política, elija la pestaña JSON.
4. Sustituya el contenido predeterminado de la pestaña JSON con la siguiente política de permisos y, a continuación, elija Siguiente: Etiquetas.

> **Note**
>
> Para obtener información sobre las alarmas CloudWatch compuestas, se deben asignar * permisos a la operación de la DescribeAlarmsAPI, como se muestra aquí. No puede devolver información sobre las alarmas compuestas si `DescribeAlarms` su alcance es más limitado.

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "cloudwatch:DescribeAlarms"
                ],
                "Resource": "*"
            }
        ]
```

```
    }
```

5. Escriba etiquetas para este rol y, a continuación, elija Next: Review (Siguiente: Revisar).
6. En la página Revisar, introduzca **`SSMCloudWatchAlarmDiscoveryPolicy`** en el campo Nombre.
7. Elija Crear política. El sistema le devuelve a la página Policies (Políticas).

Paso 2: Cree la función de IAM para la reversión en función de las alarmas CloudWatch

Utilice el siguiente procedimiento para crear un rol de IAM y asignarle la política que creó en el procedimiento anterior.

Para crear una función de IAM para la reversión en función de las alarmas CloudWatch

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y luego seleccione Create role.
3. En Select type of trusted entity (Seleccionar tipo de entidad de confianza), elija AWS service (Servicio de AWS).
4. Justo debajo de Elegir el servicio que utilizará este rol, elija EC2: Permite que las instancias EC2 llamen a los servicios de AWS en su nombre y, a continuación, elija Siguiente: Permisos.
5. En la página de política de permisos adjunta, busque SSM. CloudWatchAlarmDiscoveryPolicy
6. Elija esta política y, a continuación, elija Siguiente: Etiquetas.
7. Escriba etiquetas para este rol y, a continuación, elija Next: Review (Siguiente: Revisar).
8. En la página Crear rol escriba **`SSMCloudWatchAlarmDiscoveryRole`** en el campo Nombre de rol y, a continuación, elija Crear rol.
9. En la página Roles, seleccione el rol que acaba de crear. Se abre la página Resumen.

Paso 3: Añadir una relación de confianza

Utilice el siguiente procedimiento para configurar el rol de que acaba de crear para confiar en AWS AppConfig.

Para añadir una relación de confianza para AWS AppConfig

1. En la página Summary del rol que acaba de crear, elija la pestaña Trust Relationships y, después, seleccione Edit Trust Relationship.
2. Edite la política para incluir solo "`appconfig.amazonaws.com`", tal y como se muestra en el siguiente ejemplo:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "appconfig.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

3. Elija Actualizar política de confianza.

Creación de indicadores de características y datos de configuración de formato libre en AWS AppConfig

Los temas de esta sección le ayudan a realizar las siguientes tareas en AWS AppConfig. Estas tareas crean artefactos importantes a la hora de implementar los datos de configuración.

1. Crear un espacio de nombres de aplicaciones

   Para crear un espacio de nombres de aplicaciones, debe crear un AWS AppConfig artefacto denominado aplicación. Una aplicación es simplemente una estructura organizativa, como una carpeta.

2. Crear entornos

   Para cada AWS AppConfig aplicación, debe definir uno o más entornos. Un entorno es un grupo de AWS AppConfig objetivos de despliegue lógico, como las aplicaciones de un `Production` entorno `Beta` O. También puede definir entornos para subcomponentes de aplicaciones como, por ejemplo, los componentes `AWS Lambda functions`, `Containers`, `Web`, `Mobile`, y `Back-end`.

   Puede configurar CloudWatch las alarmas de Amazon para cada entorno para revertir automáticamente los cambios de configuración problemáticos. El sistema supervisa las alarmas durante la implementación de la configuración. Si se activa una alarma, el sistema deshace la configuración.

3. Creación de un perfil de configuración

   Un perfil de configuración incluye, entre otras cosas, un URI que permite AWS AppConfig localizar los datos de configuración en la ubicación almacenada y en un tipo de perfil. AWS AppConfig admite dos tipos de perfiles de configuración: indicadores de características y configuraciones de formato libre. Los perfiles de configuración de los indicadores de función almacenan sus datos en el almacén de configuración AWS AppConfig alojado y el URI es simple`hosted`. En el caso de los perfiles de configuración de formato libre, puede almacenar los datos en el almacén de configuración AWS AppConfig hospedado o en otra función o AWS servicio de Systems Manager con el que se integre AWS AppConfig, tal y como se describe enCrear un perfil de configuración de formato libre en AWS AppConfig.

   Un perfil de configuración también puede incluir validadores opcionales para garantizar que los datos de configuración sean correctos desde el punto de vista sintáctico y semántico. AWS

AppConfig realiza una comprobación mediante los validadores al iniciar una implementación. Si se detecta algún error, la implementación se detiene antes de realizar cambios en los destinos de la configuración.

> **Note**
>
> A menos que tenga necesidades específicas para almacenar información confidencial AWS Secrets Manager o gestionar datos en Amazon Simple Storage Service (Amazon S3), le recomendamos alojar los datos de configuración en AWS AppConfig el almacén de configuración hospedado, ya que es el que ofrece la mayoría de las funciones y mejoras.

Temas

Configuraciones de ejemplo

Utilice AWS AppConfiguna capacidad para crear AWS Systems Manager, administrar e implementar rápidamente configuraciones de aplicaciones. Una configuración es un conjunto de opciones que influyen en el comportamiento de la aplicación. Estos son algunos ejemplos.

Configuración de marcas de características

La siguiente configuración de marcas de características habilita o deshabilita los pagos móviles y los pagos predeterminados por región.

JSON

```
{
  "allow_mobile_payments": {
    "enabled": false
```

```
  },
  "default_payments_per_region": {
    "enabled": true
  }
}
```

YAML

```
---
allow_mobile_payments:
  enabled: false
default_payments_per_region:
  enabled: true
```

Configuración de operación

La siguiente configuración de formato libre impone límites a la forma en que una aplicación procesa las solicitudes.

JSON

```
{
  "throttle-limits": {
    "enabled": "true",
    "throttles": [
      {
        "simultaneous_connections": 12
      },
      {
        "tps_maximum": 5000
      }
    ],
    "limit-background-tasks": [
      true
    ]
  }
}
```

YAML

```
---
throttle-limits:
```

```
  enabled: 'true'
  throttles:
  - simultaneous_connections: 12
  - tps_maximum: 5000
  limit-background-tasks:
  - true
```

Configuración de lista de control de acceso

La siguiente configuración de formato libre de la lista de control de acceso especifica qué usuarios o grupos pueden acceder a una aplicación.

JSON

```
{
  "allow-list": {
    "enabled": "true",
    "cohorts": [
      {
        "internal_employees": true
      },
      {
        "beta_group": false
      },
      {
        "recent_new_customers": false
      },
      {
        "user_name": "Jane_Doe"
      },
      {
        "user_name": "John_Doe"
      }
    ]
  }
}
```

YAML

```
---
allow-list:
  enabled: 'true'
```

```
 cohorts:
 - internal_employees: true
 - beta_group: false
 - recent_new_customers: false
 - user_name: Jane_Doe
 - user_name: Ashok_Kumar
```

Acerca del rol de IAM del perfil de configuración

Puede crear la función de IAM que proporciona acceso a los datos de configuración mediante AWS AppConfig. O puede crear el rol de IAM usted mismo. Si crea el rol mediante AWS AppConfig, el sistema lo crea y especifica una de las siguientes políticas de permisos, según el tipo de fuente de configuración que elija.

El origen de configuración es un secreto en Secrets Manager

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
             ],
            "Resource": [
                "arn:aws:secretsmanager:Región de AWS:account_ID:secret:secret_name-
a1b2c3"
            ]
        }
    ]
}
```

El origen de configuración es un parámetro en Parameter Store

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter"
```

```
            ],
            "Resource": [
                "arn:aws:ssm:Región de AWS:account_ID:parameter/parameter_name"
            ]
        }
    ]
    }
```

El origen de configuración es un documento de SSM

```
{

    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:Región de AWS:account_ID:document/document_name"
            ]
        }
    ]
}
```

Si crea el rol mediante AWS AppConfig, el sistema también crea la siguiente relación de confianza para el rol.

```
{

  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "appconfig.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Creación de un espacio de nombres para su aplicación en AWS AppConfig

Los procedimientos de esta sección le ayudan a crear un AWS AppConfig artefacto denominado aplicación. Una aplicación es simplemente una estructura organizativa, como una carpeta, que identifica el espacio de nombres de su aplicación. Esta construcción organizativa tiene una relación con alguna unidad de código que se puede poner en marcha. Por ejemplo, puede crear una aplicación llamada MyMobileApp para organizar y administrar los datos de configuración de una aplicación móvil instalada por sus usuarios. Debe crear estos artefactos antes de poder utilizarlos AWS AppConfig para implementar y recuperar indicadores de características o datos de configuración de formato libre.

> **Note**
>
> Puede utilizarlos AWS CloudFormation para crear AWS AppConfig artefactos, como aplicaciones, entornos, perfiles de configuración, despliegues, estrategias de despliegue y versiones de configuración alojadas. Para obtener más información, consulte Referencia de tipos de recursos de AWS AppConfig en la Guía del usuario de AWS CloudFormation .

Temas

Crear una AWS AppConfig aplicación (consola)

Utilice el siguiente procedimiento para crear una AWS AppConfig aplicación mediante la AWS Systems Manager consola.

Para crear una aplicación

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En la pestaña Applications (Aplicaciones), elija Create application (Crear aplicación).
3. En Name (Nombre), escriba un nombre para la aplicación.
4. En Description (Descripción), escriba información acerca de la aplicación.

5. (Opcional) En la sección Extensiones, elija una extensión de la lista. Para obtener más información, consulte Acerca de AWS AppConfig las extensiones.
6. (Opcional) En la sección Etiquetas, introduzca una clave y un valor opcional. Puede especificar un máximo de 50 etiquetas para un recurso.
7. Elija Crear aplicación.

AWS AppConfig crea la aplicación y, a continuación, muestra la pestaña Entornos. Continúe en Creación de entornos para su aplicación en AWS AppConfig.

Creación de una AWS AppConfig aplicación (línea de comandos)

El siguiente procedimiento describe cómo utilizar AWS CLI (en Linux o Windows) o cómo AWS Tools for PowerShell crear una AWS AppConfig aplicación.

Para crear una aplicación paso a paso

1. Abra el AWS CLI.
2. Ejecute el siguiente comando para crear una aplicación.

   Linux

   ```
   aws appconfig create-application \
     --name A_name_for_the_application \
     --description A_description_of_the_application \
     --tags User_defined_key_value_pair_metadata_for_the_application
   ```

   Windows

   ```
   aws appconfig create-application ^
     --name A_name_for_the_application ^
     --description A_description_of_the_application ^
     --tags User_defined_key_value_pair_metadata_for_the_application
   ```

   PowerShell

   ```
   New-APPCApplication `
     -Name Name_for_the_application `
     -Description Description_of_the_application `
     -Tag Hashtable_type_user_defined_key_value_pair_metadata_for_the_application
   ```

El sistema devuelve información similar a la siguiente.

Linux

```
{
   "Id": "Application ID",
   "Name": "Application name",
   "Description": "Description of the application"
}
```

Windows

```
{
   "Id": "Application ID",
   "Name": "Application name",
   "Description": "Description of the application"
}
```

PowerShell

```
ContentLength    : Runtime of the command
Description      : Description of the application
HttpStatusCode   : HTTP Status of the runtime
Id               : Application ID
Name             : Application name
ResponseMetadata : Runtime Metadata
```

Creación de entornos para su aplicación en AWS AppConfig

Para cada AWS AppConfig aplicación, defina uno o más entornos. Un entorno es un grupo de AppConfig objetivos de despliegue lógico, como aplicaciones en un `Beta` `Production` entorno, AWS Lambda funciones o contenedores. También puede definir entornos para subcomponentes de aplicaciones como, por ejemplo, los componentes `Web`, `Mobile` y `Back-end` para la aplicación. Puede configurar CloudWatch las alarmas de Amazon para cada entorno. El sistema supervisa las alarmas durante la implementación de la configuración. Si se activa una alarma, el sistema deshace la configuración.

Antes de empezar

Si quiere habilitar la AWS AppConfig reversión de una configuración en respuesta a una CloudWatch alarma, debe configurar un rol AWS Identity and Access Management (IAM) con permisos que le permitan responder AWS AppConfig a CloudWatch las alarmas. Elija este rol en el siguiente procedimiento. Para obtener más información, consulte (Opcional) Configure los permisos para la reversión en función de las alarmas CloudWatch .

Temas

- Crear un AWS AppConfig entorno (consola)
- Creación de un AWS AppConfig entorno (línea de comandos)

Crear un AWS AppConfig entorno (consola)

Utilice el siguiente procedimiento para crear un AWS AppConfig entorno mediante la AWS Systems Manager consola.

Para crear una de entorno

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En la pestaña Aplicaciones, elija el nombre de una aplicación para abrir la página de detalles.
3. Seleccione la pestaña Entornos y, a continuación, elija Crear entorno.
4. En Name (Nombre), introduzca un nombre para el entorno.
5. En Description (Descripción), escriba información acerca del entorno.
6. (Opcional) En la sección Monitores, elija el campo de rol de IAM y, a continuación, elija un rol de IAM con permiso para revertir una configuración si se activa una alarma.
7. En la lista de CloudWatch alarmas, elija una o más alarmas para monitorizarlas. AWS AppConfig revierte el despliegue de la configuración si una de estas alarmas pasa a un estado de alarma.
8. (Opcional) En la sección Asociar extensiones, elija una extensión de la lista. Para obtener más información, consulte Acerca de AWS AppConfig las extensiones.
9. (Opcional) En la sección Etiquetas, introduzca una clave y un valor opcional. Puede especificar un máximo de 50 etiquetas para un recurso.
10. Seleccione Crear entorno.

AWS AppConfig crea el entorno y, a continuación, muestra la página de detalles del entorno. Continúe en Creación de un perfil de configuración en AWS AppConfig.

Creación de un AWS AppConfig entorno (línea de comandos)

El siguiente procedimiento describe cómo utilizar el AWS CLI (en Linux o Windows) o AWS Tools for PowerShell cómo crear un AWS AppConfig entorno.

Para crear un entorno paso a paso

1. Abra el AWS CLI.
2. Ejecute el siguiente comando para crear un entorno.

   Linux

   ```
   aws appconfig create-environment \
     --application-id The_application_ID \
     --name A_name_for_the_environment \
     --description A_description_of_the_environment \
     --monitors
    "AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
    role_for_AWS AppConfig_to_monitor_AlarmArn" \
     --tags User_defined_key_value_pair_metadata_of_the_environment
   ```

   Windows

   ```
   aws appconfig create-environment ^
     --application-id The_application_ID ^
     --name A_name_for_the_environment ^
     --description A_description_of_the_environment ^
     --monitors
    "AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
    role_for_AWS AppConfig_to_monitor_AlarmArn" ^
     --tags User_defined_key_value_pair_metadata_of_the_environment
   ```

   PowerShell

   ```
   New-APPCEnvironment `
     -Name Name_for_the_environment `
     -ApplicationId The_application_ID
     -Description Description_of_the_environment `
     -Monitors
    @{"AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
    role_for_AWS AppConfig_to_monitor_AlarmArn"} `
   ```

```
  -Tag Hashtable_type_user_defined_key_value_pair_metadata_of_the_environment
```

El sistema devuelve información similar a la siguiente.

Linux

```
{
   "ApplicationId": "The application ID",
   "Id": "The_environment ID",
   "Name": "Name of the environment",
   "State": "The state of the environment",
   "Description": "Description of the environment",

   "Monitors": [
      {
         "AlarmArn": "ARN of the Amazon CloudWatch alarm",
         "AlarmRoleArn": "ARN of the IAM role for AppConfig to monitor AlarmArn"
      }
   ]
}
```

Windows

```
{
   "ApplicationId": "The application ID",
   "Id": "The environment ID",
   "Name": "Name of the environment",
   "State": "The state of the environment"
   "Description": "Description of the environment",

   "Monitors": [
      {
         "AlarmArn": "ARN of the Amazon CloudWatch alarm",
         "AlarmRoleArn": "ARN of the IAM role for AppConfig to monitor AlarmArn"
      }
   ]
}
```

PowerShell

```
ApplicationId     : The application ID
```

```
ContentLength     : Runtime of the command
Description       : Description of the environment
HttpStatusCode    : HTTP Status of the runtime
Id                : The environment ID
Monitors          : {ARN of the Amazon CloudWatch alarm, ARN of the IAM role for
 AppConfig to monitor AlarmArn}
Name              : Name of the environment
Response Metadata : Runtime Metadata
State             : State of the environment
```

Continúe en Creación de un perfil de configuración en AWS AppConfig.

Creación de un perfil de configuración en AWS AppConfig

Un perfil de configuración incluye, entre otras cosas, un URI que permite AWS AppConfig localizar los datos de configuración en su ubicación almacenada y un tipo de configuración. AWS AppConfig admite dos tipos de perfiles de configuración: indicadores de características y configuraciones de formato libre. La configuración de un indicador de función almacena los datos en el almacén de configuración AWS AppConfig alojado y el URI es simple`hosted`. Una configuración de formato libre puede almacenar datos en el almacén de configuración AWS AppConfig hospedado, en diversas funciones de Systems Manager o en un AWS servicio que se integre con. AWS AppConfig Para obtener más información, consulte Crear un perfil de configuración de formato libre en AWS AppConfig.

Un perfil de configuración también puede incluir validadores opcionales para garantizar que los datos de configuración sean correctos desde el punto de vista sintáctico y semántico. AWS AppConfig realiza una comprobación mediante los validadores al iniciar una implementación. Si se detecta algún error, la implementación se detiene antes de realizar cambios en los destinos de la configuración.

> **Note**
>
> Si es posible, le recomendamos que aloje los datos de configuración en el almacén de configuración AWS AppConfig alojado, ya que es el que ofrece la mayoría de las funciones y mejoras.

Temas

- Acerca de los validadores

- Crear un perfil de configuración de indicadores de características en AWS AppConfig
- Crear un perfil de configuración de formato libre en AWS AppConfig

Acerca de los validadores

Cuando crea un perfil de configuración, puede especificar hasta dos validadores. Un validador garantiza que los datos de configuración sean sintáctica y semánticamente correctos. Si planea usar un validador, debe crearlo antes de crear el perfil de configuración. AWS AppConfig admite los siguientes tipos de validadores:

- AWS Lambda funciones: compatible con indicadores de características y configuraciones de formato libre.
- Esquema JSON: compatible con configuraciones de formulario libre. (valida AWS AppConfig automáticamente los indicadores de funciones comparándolos con un esquema JSON).

Temas

- Validadores de funciones AWS Lambda
- Validadores de esquemas JSON

Validadores de funciones AWS Lambda

Los validadores de funciones de Lambda deben configurarse con el siguiente esquema de eventos. AWS AppConfig utiliza este esquema para invocar la función de Lambda. El contenido es una cadena codificada en base64 y el URI es una cadena.

```
{
    "applicationId": "The application ID of the configuration profile being
 validated",
    "configurationProfileId": "The ID of the configuration profile being validated",
    "configurationVersion": "The version of the configuration profile being validated",
    "content": "Base64EncodedByteString",
    "uri": "The configuration uri"
}
```

AWS AppConfig verifica que el encabezado `X-Amz-Function-Error` Lambda esté establecido en la respuesta. Lambda establece este encabezado si la función arroja una excepción. Para obtener

más información acerca de `X-Amz-Function-Error`, consulte Tratamiento de errores y reintentos automáticos en AWS Lambda en la Guía del desarrollador de AWS Lambda .

Aquí se incluye un ejemplo simple de un código de respuesta de Lambda para una validación satisfactoria.

```
import json

def handler(event, context):
     #Add your validation logic here
     print("We passed!")
```

Aquí se incluye un ejemplo simple de un código de respuesta de Lambda para una validación incorrecta.

```
def handler(event, context):
     #Add your validation logic here
     raise Exception("Failure!")
```

Aquí hay otro ejemplo que se valida solo si el parámetro de configuración es un número primo.

```
function isPrime(value) {
    if (value < 2) {
        return false;
    }

    for (i = 2; i < value; i++) {
        if (value % i === 0) {
            return false;
        }
    }

    return true;
}

exports.handler = async function(event, context) {
    console.log('EVENT: ' + JSON.stringify(event, null, 2));
    const input = parseInt(Buffer.from(event.content, 'base64').toString('ascii'));
    const prime = isPrime(input);
    console.log('RESULT: ' + input + (prime ? ' is' : ' is not') + ' prime');
    if (!prime) {
```

```
        throw input + "is not prime";
    }
}
```

AWS AppConfig llama a su Lambda de validación al llamar a las operaciones `StartDeployment` y `ValidateConfigurationActivity` API. Debe proporcionar permisos de `appconfig.amazonaws.com` para invocar su Lambda. Para obtener más información, consulte Cómo conceder acceso a las funciones a los AWS servicios. AWS AppConfig limita el tiempo de ejecución de la validación de Lambda a 15 segundos, incluida la latencia de inicio.

Validadores de esquemas JSON

Si crea una configuración en un documento SSM, debe especificar o crear un esquema JSON para esa configuración. Un esquema JSON define las propiedades permitidas para cada ajuste de configuración de la aplicación. Este esquema JSON funciona como un conjunto de reglas para garantizar que los ajustes de configuración nuevos o actualizados se ajusten a las prácticas recomendadas requeridas por su aplicación. A continuación se muestra un ejemplo.

```
    {
      "$schema": "http://json-schema.org/draft-04/schema#",
      "title": "$id$",
      "description": "BasicFeatureToggle-1",
      "type": "object",
      "additionalProperties": false,
      "patternProperties": {
          "[^\\s]+$": {
              "type": "boolean"
          }
      },
      "minProperties": 1
    }
```

Al crear una configuración a partir de un documento SSM, el sistema verifica automáticamente que la configuración se ajusta a los requisitos del esquema. Si no es así, AWS AppConfig devuelve un error de validación.

> **Important**
>
> Tenga en cuenta la siguiente información importante sobre los validadores de esquemas JSON:

- Los datos de configuración almacenados en documentos de SSM deben validarse con un esquema JSON asociado para poder agregar la configuración al sistema. Los parámetros SSM no requieren un método de validación, pero le recomendamos que cree una comprobación de validación para las configuraciones de parámetros SSM nuevas o actualizadas mediante este método. AWS Lambda
- La configuración de un documento SSM utiliza el tipo de documento `ApplicationConfiguration`. El esquema JSON correspondiente utiliza el tipo de documento `ApplicationConfigurationSchema`.
- AWS AppConfig es compatible con la versión 4.X de JSON Schema para esquemas en línea. Si la configuración de la aplicación requiere una versión diferente de JSON Schema, debe crear un validador de Lambda.

Crear un perfil de configuración de indicadores de características en AWS AppConfig

Puede utilizar los indicadores de características para activar o desactivar las funciones de sus aplicaciones o para configurar diferentes características de las funciones de las aplicaciones mediante los atributos de los indicadores. AWS AppConfig almacena las configuraciones de las marcas de características en el almacén de configuraciones AWS AppConfig hospedado en un formato de marcas de características que contiene datos y metadatos sobre las marcas y los atributos de las marcas. Para obtener más información sobre el almacén de configuración AWS AppConfig hospedado, consulte Acerca del almacén de configuración AWS AppConfig hospedado la sección.

Temas

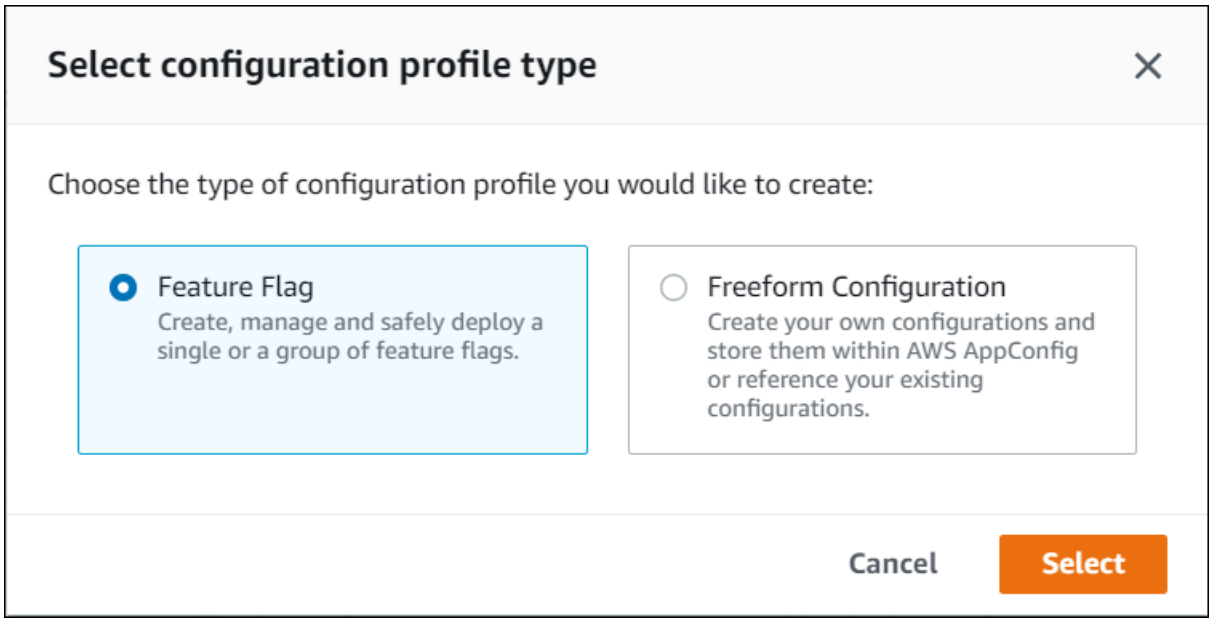
- Creación de una marca de características y un perfil de configuración de marca de características (consola)
- Creación de una marca de características y un perfil de configuración de una marca de características (línea de comandos)
- Escriba la referencia para AWS.AppConfig.FeatureFlags

Creación de una marca de características y un perfil de configuración de marca de características (consola)

Utilice el siguiente procedimiento para crear un AWS AppConfig perfil de configuración de indicadores de características y una configuración de indicadores de características mediante la AWS AppConfig consola.

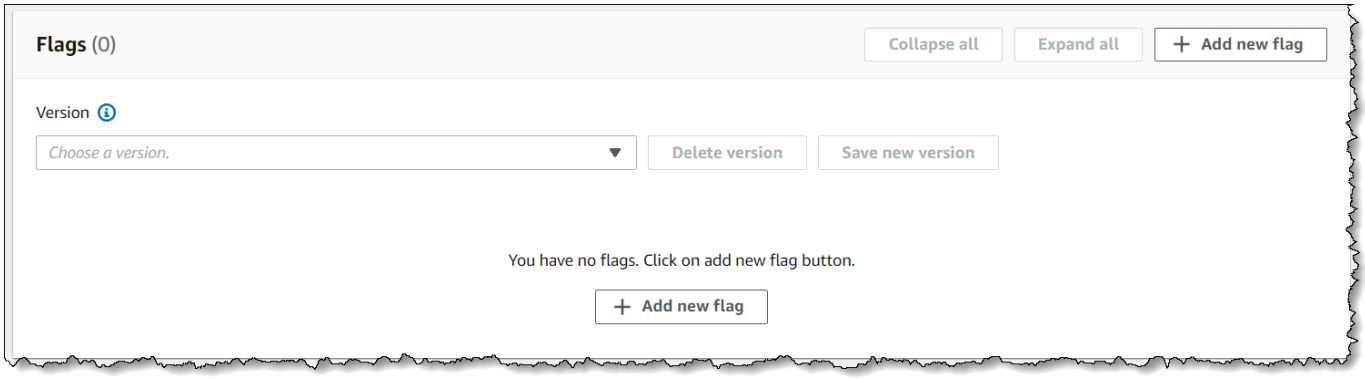
Para crear un perfil de configuración

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En la pestaña Aplicaciones, elija la aplicación que creó en Crear una AWS AppConfig configuración y, a continuación, elija la pestaña Perfiles de configuración e indicadores de características.
3. Seleccione Crear.
4. Elija una marca de características.

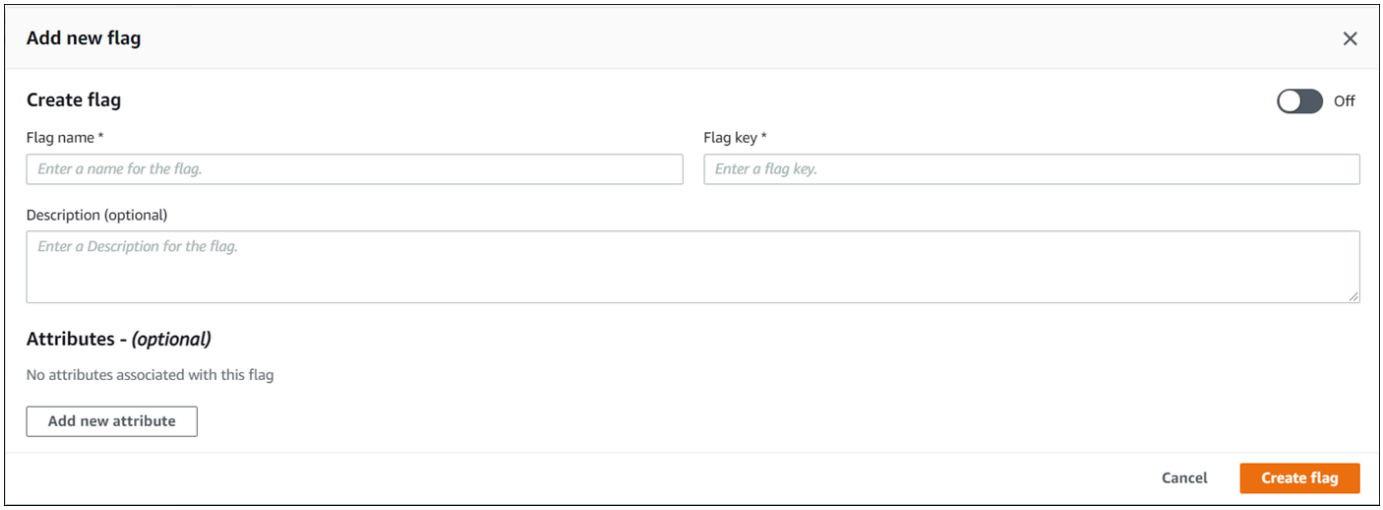


Para crear una marca de características

1. En la configuración que creó, elija Agregar nueva marca.



2. Proporcione un nombre de marca y una descripción (opcional). La clave de marca se rellena automáticamente al reemplazar los espacios con guiones bajos en el nombre que proporcionó. Puede editar la clave de marca si desea un valor o formato diferente. Una vez creada la marca, puede editar el nombre de la marca, pero no su clave.



3. Especifique si la marca de características está habilitada o deshabilitada mediante el botón de conmutación.
4. (Opcional) Añada atributos y restricciones de atributos a la marca de características. Los atributos le permiten proporcionar valores adicionales dentro de su marca. Si lo desea, puede validar los valores de los atributos según las restricciones especificadas. Las restricciones garantizan que no se implementen valores inesperados en la aplicación.

   AWS AppConfig Los indicadores de características admiten los siguientes tipos de atributos y sus correspondientes restricciones.

Tipo	Constraint	Descripción
Cadena	Expresión regular	Patrón de expresión regular para la cadena
	Enum	Lista de valores aceptables para la cadena
Número	Mínimo	Valor numérico mínimo para el atributo
	Máximo	Valor numérico máximo para el atributo
Booleano	Ninguna	Ninguna
Matriz de cadenas	Expresión regular	Patrón de expresión regular para los elementos de la matriz
	Enum	Lista de valores aceptable s para los elementos de la matriz
Matriz de números	Mínimo	Valor numérico mínimo para los elementos de la matriz
	Máximo	Valor numérico máximo para los elementos de la matriz

Note

Observe la siguiente información.

- Para los nombres de atributos, la palabra “habilitado” es una palabra reservada. No se puede crear un atributo de marca de características denominado “habilitado”. No hay otras palabras reservadas.

- Los atributos de una marca de características solo se incluyen en la respuesta `GetLatestConfiguration` si dicha marca está habilitada.
- Seleccione Valor obligatorio para especificar si se requiere un valor de atributo.

5. Seleccione Guardar nueva versión.

Continúe en Implementación de marcas de características y datos de configuración en AWS AppConfig.

Creación de una marca de características y un perfil de configuración de una marca de características (línea de comandos)

El siguiente procedimiento describe cómo utilizar AWS Command Line Interface (en Linux o Windows) o las Herramientas para Windows PowerShell para crear un perfil de configuración de indicadores de AWS AppConfig características. Si lo prefiere, puede utilizar AWS CloudShell para ejecutar los comandos que se indican a continuación. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell .

Para crear una configuración de una marca de características paso a paso

1. Abra el AWS CLI.
2. Cree un perfil de configuración de la marca de características especificando su tipo como `AWS.AppConfig.FeatureFlags`. El perfil de configuración debe usar `hosted` como URI de ubicación.

   Linux

   ```
   aws appconfig create-configuration-profile \
     --application-id The_application_ID \
     --name A_name_for_the_configuration_profile \
     --location-uri hosted \
     --type AWS.AppConfig.FeatureFlags
   ```

   Windows

   ```
   aws appconfig create-configuration-profile ^
     --application-id The_application_ID ^
     --name A_name_for_the_configuration_profile ^
     --location-uri hosted ^
   ```

```
  --type AWS.AppConfig.FeatureFlags
```

PowerShell

```
New-APPCConfigurationProfile `
  -Name A_name_for_the_configuration_profile `
  -ApplicationId The_application_ID `
  -LocationUri hosted `
  -Type AWS.AppConfig.FeatureFlags
```

3. Cree los datos de configuración de su marca de características. Los datos deben estar en formato JSON y ajustarse al esquema JSON de `AWS.AppConfig.FeatureFlags`. Para obtener más información acerca del nuevo esquema, consulte Escriba la referencia para AWS.AppConfig.FeatureFlags.
4. Utilice la API de `CreateHostedConfigurationVersion` para guardar los datos de configuración de su marca de características en AWS AppConfig.

   Linux

   ```
   aws appconfig create-hosted-configuration-version \
     --application-id The_application_ID \
     --configuration-profile-id The_configuration_profile_id \
     --content-type "application/json" \
     --content file://path/to/feature_flag_configuration_data \
     file_name_for_system_to_store_configuration_data
   ```

   Windows

   ```
   aws appconfig create-hosted-configuration-version ^
     --application-id The_application_ID ^
     --configuration-profile-id The_configuration_profile_id ^
     --content-type "application/json" ^
     --content file://path/to/feature_flag_configuration_data ^
     file_name_for_system_to_store_configuration_data
   ```

   PowerShell

   ```
   New-APPCHostedConfigurationVersion `
   ```

```
    -ApplicationId The_application_ID `
    -ConfigurationProfileId The_configuration_profile_id `
    -ContentType "application/json" `
    -Content file://path/to/feature_flag_configuration_data `
    file_name_for_system_to_store_configuration_data
```

Este es un ejemplo de comando de Linux.

```
aws appconfig create-hosted-configuration-version \
  --application-id 1a2b3cTestApp \
  --configuration-profile-id 4d5e6fTestConfigProfile \
  --content-type "application/json" \
  --content Base64Content
```

El parámetro `content` utiliza los siguientes datos codificados `base64`.

```
{
  "flags": {
    "flagkey": {
      "name": "WinterSpecialBanner"
    }
  },
  "values": {
    "flagkey": {
      "enabled": true
    }
  },
  "version": "1"
}
```

El sistema devuelve información similar a la siguiente.

Linux

```
{
   "ApplicationId"          : "1a2b3cTestApp",
   "ConfigurationProfileId" : "4d5e6fTestConfigProfile",
   "VersionNumber"          : "1",
   "ContentType"            : "application/json"
}
```

Windows

```
{
   "ApplicationId"          : "1a2b3cTestApp",
   "ConfigurationProfileId" : "4d5e6fTestConfigProfile",
   "VersionNumber"          : "1",
   "ContentType"            : "application/json"
}
```

PowerShell

```
ApplicationId          : 1a2b3cTestApp
ConfigurationProfileId : 4d5e6fTestConfigProfile
VersionNumber          : 1
ContentType            : application/json
```

`service_returned_content_file`Contiene sus datos de configuración, que incluyen algunos metadatos AWS AppConfig generados.

> **Note**
>
> Al crear la versión de configuración alojada, AWS AppConfig verifica que los datos se ajusten al esquema `AWS.AppConfig.FeatureFlags` JSON. AWS AppConfig además, valida que cada atributo del indicador de entidad de sus datos satisfaga las restricciones que ha definido para esos atributos.

Escriba la referencia para AWS.AppConfig.FeatureFlags

Utilice el esquema JSON de `AWS.AppConfig.FeatureFlags` como referencia para crear los datos de configuración de su marca de características.

```
{
      "$schema": "http://json-schema.org/draft-07/schema#",
      "definitions": {
        "flagSetDefinition": {
          "type": "object",
          "properties": {
            "version": {
```

```
            "$ref": "#/definitions/flagSchemaVersions"
          },
          "flags": {
            "$ref": "#/definitions/flagDefinitions"
          },
          "values": {
            "$ref": "#/definitions/flagValues"
          }
        },
        "required": ["version", "flags"],
        "additionalProperties": false
      },
      "flagDefinitions": {
        "type": "object",
        "patternProperties": {
          "^[a-z][a-zA-Z\\d-]{0,63}$": {
            "$ref": "#/definitions/flagDefinition"
          }
        },
        "maxProperties": 100,
        "additionalProperties": false
      },
      "flagDefinition": {
        "type": "object",
        "properties": {
          "name": {
            "$ref": "#/definitions/customerDefinedName"
          },
          "description": {
            "$ref": "#/definitions/customerDefinedDescription"
          },
          "_createdAt": {
            "type": "string"
          },
          "_updatedAt": {
            "type": "string"
          },
          "_deprecation": {
            "type": "object",
            "properties": {
              "status": {
                "type": "string",
                "enum": ["planned"]
              }
```

```
            },
            "additionalProperties": false
          },
          "attributes": {
            "$ref": "#/definitions/attributeDefinitions"
          }
        },
        "additionalProperties": false
      },
      "attributeDefinitions": {
        "type": "object",
        "patternProperties": {
          "^[a-z][a-zA-Z\\d-_]{0,63}$": {
            "$ref": "#/definitions/attributeDefinition"
          }
        },
        "maxProperties": 25,
        "additionalProperties": false
      },
      "attributeDefinition": {
        "type": "object",
        "properties": {
          "description": {
            "$ref": "#/definitions/customerDefinedDescription"
          },
          "constraints": {
            "oneOf": [
              { "$ref": "#/definitions/numberConstraints" },
              { "$ref": "#/definitions/stringConstraints" },
              { "$ref": "#/definitions/arrayConstraints" },
              { "$ref": "#/definitions/boolConstraints" }
            ]
          }
        },
        "additionalProperties": false
      },
      "flagValues": {
        "type": "object",
        "patternProperties": {
          "^[a-z][a-zA-Z\\d-_]{0,63}$": {
            "$ref": "#/definitions/flagValue"
          }
        },
        "maxProperties": 100,
```

```
      "additionalProperties": false
    },
    "flagValue": {
      "type": "object",
      "properties": {
        "enabled": {
          "type": "boolean"
        },
        "_createdAt": {
          "type": "string"
        },
        "_updatedAt": {
          "type": "string"
        }
      },
      "patternProperties": {
        "^[a-z][a-zA-Z\\d-_]{0,63}$": {
          "$ref": "#/definitions/attributeValue",
          "maxProperties": 25
        }
      },
      "required": ["enabled"],
      "additionalProperties": false
    },
    "attributeValue": {
      "oneOf": [
        { "type": "string", "maxLength": 1024 },
        { "type": "number" },
        { "type": "boolean" },
        {
          "type": "array",
          "oneOf": [
            {
              "items": {
                "type": "string",
                "maxLength": 1024
              }
            },
            {
              "items": {
                "type": "number"
              }
            }
          ]
```

```
            }
          ],
          "additionalProperties": false
        },
        "stringConstraints": {
          "type": "object",
          "properties": {
            "type": {
              "type": "string",
              "enum": ["string"]
      },
            "required": {
              "type": "boolean"
            },
            "pattern": {
              "type": "string",
              "maxLength": 1024
            },
            "enum": {
              "type": "array",
              "maxLength": 100,
              "items": {
                "oneOf": [
                  {
                    "type": "string",
                    "maxLength": 1024
                  },
                  {
                    "type": "integer"
                  }
                ]
              }
            }
          },
          "required": ["type"],
          "not": {
            "required": ["pattern", "enum"]
          },
          "additionalProperties": false
        },
        "numberConstraints": {
          "type": "object",
          "properties": {
            "type": {
```

```
            "type": "string",
            "enum": ["number"]
      },
            "required": {
              "type": "boolean"
            },
            "minimum": {
              "type": "integer"
            },
            "maximum": {
              "type": "integer"
            }
          },
          "required": ["type"],
          "additionalProperties": false
        },
        "arrayConstraints": {
          "type": "object",
          "properties": {
            "type": {
              "type": "string",
              "enum": ["array"]
      },
            "required": {
              "type": "boolean"
            },
            "elements": {
              "$ref": "#/definitions/elementConstraints"
      }
          },
          "required": ["type"],
          "additionalProperties": false
        },
        "boolConstraints": {
          "type": "object",
          "properties": {
            "type": {
              "type": "string",
              "enum": ["boolean"]
      },
            "required": {
              "type": "boolean"
            }
          },
```

```
          "required": ["type"],
          "additionalProperties": false
        },
        "elementConstraints": {
          "oneOf": [
            { "$ref": "#/definitions/numberConstraints" },
            { "$ref": "#/definitions/stringConstraints" }
          ]
        },
        "customerDefinedName": {
          "type": "string",
          "pattern": "^[^\\n]{1,64}$"
        },
        "customerDefinedDescription": {
          "type": "string",
          "maxLength": 1024
        },
        "flagSchemaVersions": {
          "type": "string",
          "enum": ["1"]
        }
      },
      "type": "object",
      "$ref": "#/definitions/flagSetDefinition",
      "additionalProperties": false
    }
```

Important

Para recuperar los datos de configuración de las marcas de características, su aplicación debe llamar a la API `GetLatestConfiguration`. No puede recuperar los datos de configuración de las marcas de características mediante una llamada a `GetConfiguration`, lo cual está obsoleto. Para obtener más información, consulte la referencia GetLatestConfigurationde la AWS AppConfig API.

Cuando la aplicación llama GetLatestConfigurationy recibe una configuración recién implementada, se elimina la información que define las marcas y los atributos de las funciones. El JSON simplificado contiene un mapa de claves que coinciden con cada una de las claves de marca que especificó. El JSON simplificado también contiene valores asignados de `true` o `false` para el atributo `enabled`.

Si una marca establece `enabled` en `true`, también estará presente cualquier atributo de la marca. El siguiente esquema JSON describe el formato de la salida JSON.

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "patternProperties": {
    "^[a-z][a-zA-Z\\d-_]{0,63}$": {
      "$ref": "#/definitions/attributeValuesMap"
    }
  },
  "maxProperties": 100,
  "additionalProperties": false,
  "definitions": {
    "attributeValuesMap": {
      "type": "object",
      "properties": {
        "enabled": {
          "type": "boolean"
        }
      },
      "required": ["enabled"],
      "patternProperties": {
        "^[a-z][a-zA-Z\\d-_]{0,63}$": {
          "$ref": "#/definitions/attributeValue"
        }
      },
      "maxProperties": 25,
      "additionalProperties": false
    },
    "attributeValue": {
      "oneOf": [
        { "type": "string","maxLength": 1024 },
        { "type": "number" },
        { "type": "boolean" },
        {
          "type": "array",
          "oneOf": [
            {
              "items": {
                "oneOf": [
                  {
                    "type": "string",
```

```
                    "maxLength": 1024
                  }
                ]
              }
            },
            {
              "items": {
                "oneOf": [
                  {
                    "type": "number"
                  }
                ]
              }
            }
          ]
        }
      ],
      "additionalProperties": false
    }
  }
}
```

Crear un perfil de configuración de formato libre en AWS AppConfig

Un perfil de configuración incluye, entre otras cosas, un URI que permite AWS AppConfig localizar los datos de configuración en la ubicación almacenada y en un tipo de perfil. AWS AppConfig admite dos tipos de perfiles de configuración: indicadores de características y configuraciones de formato libre. Los perfiles de configuración de los indicadores de función almacenan sus datos en el almacén de configuración AWS AppConfig alojado y el URI es simple`hosted`. Para los perfiles de configuración de formato libre, puede almacenar los datos en el almacén de configuración AWS AppConfig hospedado o en cualquiera de los siguientes AWS servicios y capacidades de Systems Manager:

Ubicación	Tipos de archivo admitidos
AWS AppConfig almacén de configuración hospedado	YAML, JSON y texto si se agregan con. AWS Management Console Cualquier tipo de archivo si se añade mediante la acción de la AWS AppConfig CreateHostedConfigurationVersionAPI.

Ubicación	Tipos de archivo admitidos
Amazon Simple Storage Service (Amazon S3)	Cualquiera
AWS CodePipeline	Canalización (según la define el servicio)
AWS Secrets Manager	Secreto (según lo define el servicio)
AWS Systems Manager Parameter Store	Parámetros de cadena estándar y segura (tal como los define almacén de parámetros)
AWS Systems Manager almacén de documentos (documentos SSM)	YAML, JSON, texto

Un perfil de configuración también puede incluir validadores opcionales para garantizar que los datos de configuración sean correctos desde el punto de vista sintáctico y semántico. AWS AppConfig realiza una comprobación mediante los validadores al iniciar una implementación. Si se detecta algún error, la implementación se detiene antes de realizar cambios en los destinos de la configuración.

> **Note**
>
> Si es posible, le recomendamos que aloje los datos de configuración en el almacén de configuración AWS AppConfig alojado, ya que es el que ofrece la mayoría de las funciones y mejoras.

Para las configuraciones de formato libre almacenadas en el almacén de configuraciones AWS AppConfig hospedado o en documentos SSM, puede crear la configuración de formato libre mediante la consola de Systems Manager al crear un perfil de configuración. El proceso se describe más adelante en este tema.

Para las configuraciones de formato libre almacenadas en el almacén de parámetros, Secrets Manager o Amazon S3, primero debe crear el parámetro, el secreto o el objeto y almacenarlo en el almacén de configuración correspondiente. Después de almacenar los datos de configuración, utilice el procedimiento de este tema para crear el perfil de configuración.

Temas

- Acerca de las cuotas y limitaciones de los almacenes de configuraciones

- Acerca del almacén de configuración AWS AppConfig hospedado
- Acerca de las configuraciones almacenadas en Amazon S3
- Creación de una configuración y un perfil de configuración de formato libre

Acerca de las cuotas y limitaciones de los almacenes de configuraciones

Los almacenes de configuración compatibles AWS AppConfig tienen las siguientes cuotas y limitaciones.

	AWS AppConfig almacén de configuración hospedado	Amazon S3	Almacén de parámetros de Systems Manager	AWS Secrets Manager	Almacén de documentos de Systems Manager	AWS CodePipeline
Límite de tamaño de la configuración	2 MB por defecto, 4 MB como máximo	2 MB Impuesto por S3 AWS AppConfig, no	4 KB (capa gratuita) /8 KB (parámetros avanzados)	64 KB	64 KB	2 MB Impuesto por AWS AppConfig, no CodePipeline
Límite de almacenamiento de recursos	1 GB	Sin límite	10 000 parámetros (capa gratuita) /100 000 parámetros (parámetros avanzados)	500.000	500 documentos	Limitado por el número de perfiles de configuración por aplicación (100 perfiles por aplicación)

	AWS AppConfig almacén de configuración hospedado	Amazon S3	Almacén de parámetros de Systems Manager	AWS Secrets Manager	Almacén de documentos de Systems Manager	AWS CodePipeline
Cifrado en el servidor	Sí	SSE-S3, SSE-KMS	Sí	Sí	No	Sí
AWS CloudFormation apoyo	Sí	No para crear ni actualizar datos	Sí	Sí	No	Sí
Precios	Free	Consulte Precios de Amazon S3	Consulte Precios de AWS Systems Manager	Consulte Precios de AWS Secrets Manager	Free	Consulte Precios de AWS CodePipeline

Acerca del almacén de configuración AWS AppConfig hospedado

AWS AppConfig incluye un almacén de configuración interno o alojado. Las configuraciones deben ser de 2 MB o menos. El almacén de configuración AWS AppConfig hospedado ofrece las siguientes ventajas en comparación con otras opciones de almacenamiento de configuración.

- No es necesario configurar y configurar otros servicios, como Amazon Simple Storage Service (Amazon S3) o Parameter Store.
- No necesita configurar los permisos AWS Identity and Access Management (IAM) para usar el almacén de configuración.
- Puede almacenar configuraciones en YAML, JSON o como documentos de texto.
- Usar el almacén no supone costo alguno.
- Puede crear una configuración y agregarla al almacén cuando cree un perfil de configuración.

Acerca de las configuraciones almacenadas en Amazon S3

Puede almacenar configuraciones en un bucket de Amazon Simple Storage Service (Amazon S3). Cuando se crea el perfil de configuración, se especifica el URI de un objeto único de S3 que se encuentra en un bucket. También debe especificar el nombre de recurso de Amazon (ARN) de un rol AWS Identity and Access Management (IAM) que da AWS AppConfig permiso para obtener el objeto. Antes de crear un perfil de configuración para un objeto de Amazon S3, tenga en cuenta las siguientes restricciones.

Restricción	Detalles
Tamaño	Las configuraciones almacenadas como objetos de S3 pueden tener un tamaño máximo de 1 MB.
Cifrado del objeto	Un perfil de configuración puede dirigirse a objetos cifrados en SSE-S3 y SSE-KMS.
Clases de almacenamiento	AWS AppConfig admite las siguientes clases de almacenamiento de S3:`STANDARD`, `INTELLIGENT_TIERING` `REDUCED_REDUNDANCY` `STANDARD_IA` , y. `ONEZONE_IA` No se admiten las siguientes clases: todas las clases Glacier de S3 (`DEEP_ARCHIVE` y `GLACIER`).
Control de versiones	AWS AppConfig requiere que el objeto S3 utilice el control de versiones.

Configuración de permisos para una configuración almacenada como un objeto de Amazon S3

Al crear un perfil de configuración para una configuración almacenada como un objeto S3, debe especificar un ARN para una función de IAM que dé AWS AppConfig permiso para obtener el objeto. El rol debe incluir los permisos siguientes:

Permisos para acceder al objeto de S3

- s3: GetObject

- s3: GetObjectVersion

Permisos para mostrar los buckets de S3

s3: ListAllMyBuckets

Permisos para acceder al bucket de S3 donde se almacena el objeto

- s3: GetBucketLocation
- s3: GetBucketVersioning
- s3: ListBucket
- s3: ListBucketVersions

Complete el siguiente procedimiento para crear un rol que AWS AppConfig permita almacenar una configuración en un objeto S3.

Creación de la política de IAM para acceder a un objeto de S3

Utilice el siguiente procedimiento para crear una política de IAM que AWS AppConfig permita almacenar una configuración en un objeto de S3.

Para crear una política de IAM que permita acceder a un objeto de S3

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Policies (Políticas) y, a continuación, seleccione Create policy (Crear política).
3. En la página Crear política, elija la pestaña JSON.
4. Actualice la siguiente política de ejemplo con información sobre el bucket de S3 y el objeto de configuración. A continuación, pegue la política en el campo de texto de la pestaña JSON .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
```

```
      ],
      "Resource": "arn:aws:s3:::my-bucket/my-configurations/my-configuration.json"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:GetBucketVersioning",
        "s3:ListBucketVersions",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::my-bucket"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

5. Elija Revisar política.
6. En la página Review policy (Revisar política), escriba un nombre en el cuadro Name (Nombre) y, a continuación, escriba una descripción.
7. Elija Crear política. El sistema le devuelve a la página Roles.

Creación del rol de IAM para acceder a un objeto de S3

Utilice el siguiente procedimiento para crear un rol de IAM que permita AWS AppConfig almacenar una configuración en un objeto de S3.

Para crear un rol de IAM para acceder a Athena y Amazon S3

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y luego seleccione Crear rol.
3. En la sección Seleccionar tipo de entidad de confianza, elija servicio de AWS .
4. En la sección Choose a use case (Elegir un caso de uso) en Common use cases (Casos de uso comunes), elija EC2, y, a continuación, elija Next: Permissions (Siguiente: Permisos).

5. En la página Attach permissions policy (Asociar política de permisos) en el cuadro de búsqueda, escriba el nombre de la política que creó en el procedimiento anterior.
6. Elija la política y, a continuación, elija Siguiente: Etiquetas.
7. En la página Agregar etiquetas (opcional) escriba una clave y un valor opcional y, a continuación, elija Siguiente: Revisar.
8. En la página Review (Revisar), escriba un nombre en el campo Role name (Nombre de rol) y, a continuación, escriba una descripción.
9. Elija Create role. El sistema le devuelve a la página Roles.
10. En la página Roles, elija el rol que acaba de crear para abrir la página Summary (Resumen). Anote los valores de Role Name (Nombre de rol) y Role ARN (ARN de rol). Especifique el ARN de rol cuando cree el perfil de configuración más adelante en este tema.

Creación de una relación de confianza

Utilice el siguiente procedimiento para configurar el rol de que acaba de crear para confiar en AWS AppConfig.

Para añadir una relación de confianza

1. En la página Summary del rol que acaba de crear, elija la pestaña Trust Relationships y, después, seleccione Edit Trust Relationship.
2. Elimine `"ec2.amazonaws.com"` y agregue `"appconfig.amazonaws.com"`, como se muestra en el ejemplo siguiente.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "appconfig.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

3. Elija Actualizar política de confianza.

Creación de una configuración y un perfil de configuración de formato libre

En esta sección se describe cómo crear una configuración de formato libre y un perfil de configuración. Antes de comenzar, anote la siguiente información.

- El siguiente procedimiento requiere que especifique un rol de servicio de IAM para que AWS AppConfig pueda acceder a los datos de configuración en el almacén de configuración que elija. Esta función no es necesaria si utiliza el almacén de configuración AWS AppConfig hospedado. Si elige S3, el almacén de parámetros o el almacén de documentos de Systems Manager, debe elegir o bien un rol de IAM existente o elegir la opción para que el sistema cree automáticamente el rol. Para obtener más información acerca de este rol, consulte Acerca del rol de IAM del perfil de configuración.
- Si desea crear un perfil de configuración para configuraciones almacenadas en S3, debe configurar permisos. Para obtener más información acerca de los permisos y otros requisitos para usar S3 como almacén de configuración, consulte Acerca de las configuraciones almacenadas en Amazon S3.
- Si desea utilizar validadores, revise los detalles y requisitos para usarlos. Para obtener más información, consulte Acerca de los validadores.

Temas

- Crear un perfil de configuración de AWS AppConfig formato libre (consola)
- Creación de un perfil de configuración de AWS AppConfig formato libre (línea de comandos)

Crear un perfil de configuración de AWS AppConfig formato libre (consola)

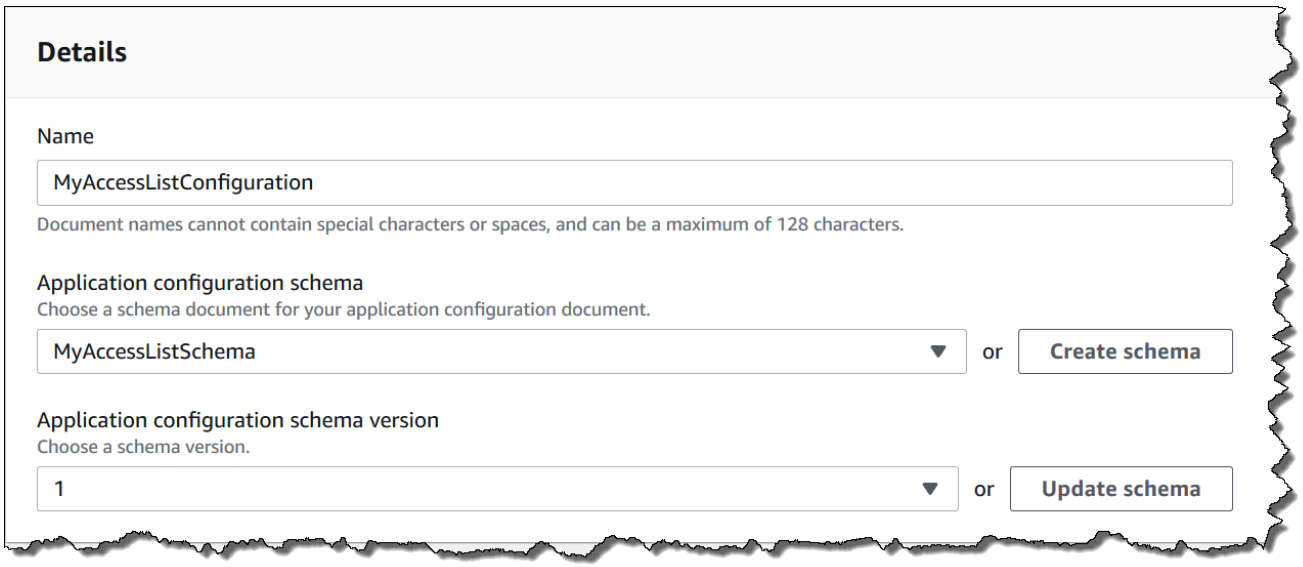
Utilice el siguiente procedimiento para crear un perfil de configuración de AWS AppConfig formato libre y (opcionalmente) una configuración de formato libre mediante la consola. AWS Systems Manager

Para crear un perfil de configuración

1. Abra la consola en https://console.aws.amazon.com/systems-manager/appconfig/ AWS Systems Manager .
2. En la pestaña Aplicaciones, elija una aplicación y, a continuación, elija la pestaña Perfiles de configuración e indicadores de características.
3. Seleccione Crear.

4. Elija la configuración Freeform y, a continuación, seleccione Seleccionar.
5. En Name (Nombre), introduzca un nombre para el perfil de configuración.
6. En Description (Descripción), introduzca información sobre el perfil de configuración.
7. En la sección Fuente de origen de la configuración, elija una opción.
8.
   - Si seleccionó la configuración AWS AppConfig alojada, elija Texto, JSON o YAML e introduzca la configuración en el campo. Seleccione Siguiente y vaya al Paso 10 de este procedimiento.
   - Si seleccionó el objeto Amazon S3, introduzca el URI del objeto en el campo de origen del objeto S3 y, a continuación, seleccione Siguiente.
   - Si seleccionó un parámetros de AWS Systems Manager , elija el nombre del parámetro de la lista. Elija Siguiente.
   - Si seleccionó Secreto del Secrets Manager, introduzca el nombre del secreto. Elija Siguiente.
   - Si seleccionó AWS CodePipeline, elija Siguiente y vaya al Paso 10 de este procedimiento.
   - Si seleccionó un documento de AWS Systems Manager , lleve a cabo los siguientes pasos.

   a. En la sección Document source (Origen del documento), elija Saved document (Documento guardado) o New document (Nuevo documento).
   b. Si eligió Documento guardado, elija el documento de SSM de la lista. Si elige New document (Nuevo documento), aparecerán las secciones Details (Detalles) y Content (Contenido).
   c. En la sección Details (Detalles), escriba un nombre para la nueva configuración de la aplicación.
   d. En la sección Application configuration schema (Esquema de configuración de la aplicación) elija el esquema JSON en la lista o elija Create schema (Crear esquema). Si eligió Crear esquema, Systems Manager abre la página Crear esquema. Escriba los detalles del esquema en la sección Content (Contenido) y, a continuación, elija Create schema (Crear esquema).



e. En Application configuration schema version (Versión del esquema de configuración de la aplicación), elija la versión de la lista o elija Update schema (Actualizar esquema) para editar el esquema y crear una nueva versión.

f. En la sección Content (Contenido) elija YAML o JSON y, a continuación, especifique los datos de configuración en el campo.

g. Elija Siguiente.

9. En la sección Función de servicio, elija Nueva función de servicio para AWS AppConfig crear la función de IAM que proporciona acceso a los datos de configuración. AWS AppConfig rellena automáticamente el campo del nombre del rol en función del nombre que ingresó anteriormente. O bien, para elegir un rol que ya existe en IAM, elija Rol de servicio existente. Elija el rol mediante la lista de Role ARN (ARN de rol).

10. En la página Add validators (Agregar validadores), elija JSON Schema (Esquema JSON) o AWS Lambda. Si elige JSON Schema (Esquema JSON), introduzca el esquema JSON en el campo. Si elige AWS Lambda, elija el nombre de recurso de Amazon (ARN) y la versión de la función de la lista.

> **Important**
>
> Los datos de configuración almacenados en documentos de SSM deben validarse con un esquema JSON asociado para poder agregar la configuración al sistema. Los parámetros SSM no requieren un método de validación, pero le recomendamos que cree una comprobación de validación para las configuraciones de parámetros SSM nuevas o actualizadas mediante. AWS Lambda

11. (Opcional) En la sección Etiquetas, introduzca una clave y un valor opcional. Puede especificar un máximo de 50 etiquetas para un recurso.
12. Elija Create configuration profile (Crear perfil de configuración).

> **Important**
>
> Si ha creado un perfil de configuración para AWS CodePipeline, después de crear una estrategia de despliegue, como se describe en la siguiente sección, debe crear una canalización CodePipeline que especifique AWS AppConfig como proveedor de despliegue. Para obtener información sobre cómo crear una canalización que se especifique AWS AppConfig como proveedor de despliegue, consulte el tutorial sobre cómo crear un canal que se utilice AWS AppConfig como proveedor de despliegue en la Guía del AWS CodePipeline usuario.

Continúe en Implementación de marcas de características y datos de configuración en AWS AppConfig.

Creación de un perfil de configuración de AWS AppConfig formato libre (línea de comandos)

El siguiente procedimiento describe cómo utilizar el perfil de configuración de formato libre AWS CLI (en Linux o Windows) o cómo AWS Tools for PowerShell crear un perfil de configuración de AWS AppConfig formato libre. Si lo prefiere, puede utilizar AWS CloudShell para ejecutar los comandos que se indican a continuación. Para obtener más información, consulte ¿Qué es AWS CloudShell? en la Guía del usuario de AWS CloudShell .

> **Note**
>
> Para las configuraciones de formato libre alojadas en el almacén de configuraciones AWS AppConfig hospedado, especifique `hosted` el URI de ubicación.

Crear un perfil de configuración paso a paso

1. Abra el AWS CLI.
2. Ejecute el siguiente comando para crear un perfil de configuración de formato libre.

Linux

```
aws appconfig create-configuration-profile \
  --application-id The_application_ID \
  --name A_name_for_the_configuration_profile \
  --description A_description_of_the_configuration_profile \
  --location-uri A_URI_to_locate_the_configuration or hosted \
  --retrieval-role-
arn The_ARN_of_the_IAM_role_with_permission_to_access_the_configuration_at_the_specified
 \
  --tags User_defined_key_value_pair_metadata_of_the_configuration_profile \
  --validators "Content=JSON_Schema_content_or_the_ARN_of_an_AWS
 Lambda_function,Type=JSON_SCHEMA or LAMBDA"
```

Windows

```
aws appconfig create-configuration-profile ^
  --application-id The_application_ID ^
  --name A_name_for_the_configuration_profile ^
  --description A_description_of_the_configuration_profile ^
  --location-uri A_URI_to_locate_the_configuration or hosted  ^
  --retrieval-role-
arn The_ARN_of_the_IAM_role_with_permission_to_access_the_configuration_at_the_specified
 ^
  --tags User_defined_key_value_pair_metadata_of_the_configuration_profile ^
  --validators "Content=JSON_Schema_content_or_the_ARN_of_an_AWS
 Lambda_function,Type=JSON_SCHEMA or LAMBDA"
```

PowerShell

```
New-APPCConfigurationProfile `
  -Name A_name_for_the_configuration_profile `
  -ApplicationId The_application_ID `
  -Description Description_of_the_configuration_profile `
  -LocationUri A_URI_to_locate_the_configuration or hosted `
  -
RetrievalRoleArn The_ARN_of_the_IAM_role_with_permission_to_access_the_configuration_at_
 `
  -
Tag Hashtable_type_user_defined_key_value_pair_metadata_of_the_configuration_profile
 `
```

```
  -Validators "Content=JSON_Schema_content_or_the_ARN_of_an_AWS
Lambda_function,Type=JSON_SCHEMA or LAMBDA"
```

> **Note**
>
> Si creó una configuración en el almacén de configuraciones AWS AppConfig hospedado, puede crear nuevas versiones de la configuración mediante las operaciones de la CreateHostedConfigurationVersionAPI. Para ver AWS CLI los detalles y ejemplos de comandos de esta operación de API, consulte create-hosted-configuration-versionla Referencia de AWS CLI comandos.

Otros orígenes de datos de configuración

En este tema se incluye información sobre otros AWS servicios que se integran con AWS AppConfig.

AWS AppConfig integración con AWS Secrets Manager

Secrets Manager ayuda a cifrar, almacenar y recuperar de forma segura las credenciales de sus bases de datos y otros servicios. En lugar de codificar las credenciales de sus aplicaciones, puede hacer llamadas a Secrets Manager para recuperar sus credenciales siempre que las necesite. Secrets Manager le ayuda a proteger el acceso a sus recursos y datos de TI al permitirle rotar y administrar el acceso a sus secretos.

Al crear un perfil de configuración de formato libre, puede elegir Secrets Manager como origen de los datos de configuración. Debe incorporarse a Secrets Manager y crear un secreto antes de crear el perfil de configuración. Para obtener más información sobre Secrets Manager, consulte ¿Qué es AWS Secrets Manager? en la Guía AWS Secrets Manager del usuario. Para obtener información sobre la creación de un perfil de configuración que utilice Secrets Manager, consulte Creación de indicadores de características y datos de configuración de formato libre en AWS AppConfig.

Implementación de marcas de características y datos de configuración en AWS AppConfig

Tras crear los artefactos necesarios para trabajar con marcas de características y datos de configuración de formato libre, puede crear una nueva implementación. Al crear una nueva implementación, puede especificar las siguientes opciones:

- ID de la aplicación
- ID del perfil de configuración
- Una versión de configuración
- Un ID de entorno en el que desea implementar los datos de configuración
- Un ID de estrategia de implementación que define la rapidez con la que desea que se apliquen los cambios
- Un identificador de clave de AWS Key Management Service (AWS KMS) para cifrar los datos mediante una clave administrada por el cliente.

Al llamar a la acción de la API StartDeployment, AWS AppConfig lleva a cabo las siguientes tareas:

1. Recupera los datos de configuración del almacén de datos subyacente mediante el URI de ubicación del perfil de configuración.
2. Comprueba que los datos de configuración sean correctos sintáctica y semánticamente utilizando los validadores que especificó al crear su perfil de configuración.
3. Guarda en caché una copia de los datos para que la aplicación pueda recuperarlos. Esta copia en caché se denomina datos implementados.

AWS AppConfig se integra con Amazon CloudWatch para supervisar las implementaciones. Si una implementación activa una alarma en CloudWatch, AWS AppConfig la revierte automáticamente para minimizar el impacto en los usuarios de la aplicación.

Temas

- Uso de estrategias de implementación
- Implementar una configuración
- Integración de la implementación de AWS AppConfig con CodePipeline

Uso de estrategias de implementación

Una estrategia de implementación le permite publicar lentamente los cambios en los entornos de producción en el plazo de minutos u horas. Una estrategia de implementación de AWS AWS AppConfig define los siguientes aspectos importantes de una implementación de configuración.

<table>
<tr><th>Opción</th><th>Descripción</th></tr>
<tr><td>Tipo de implementación</td><td>El tipo de implementación define cómo se implementa o despliegala configuración. AWS AppConfig admite los tipos de implementación Lineal y Exponencial.
Lineal: para este tipo, AWS AppConfig procesa la implementación mediante incrementos del factor de crecimiento distribuido uniformemente a lo largo del tiempo de implementación. A continuación, se muestra un ejemplo de cronograma para una implementación de 10 horas que utiliza un crecimiento lineal del 20%:<table><tr><th>Tiempo transcurrido</th><th>Implementación en curso</th></tr><tr><td>0 horas</td><td>0%</td></tr><tr><td>2 horas</td><td>20%</td></tr><tr><td>4 horas</td><td>40%</td></tr><tr><td>6 horas</td><td>60%</td></tr><tr><td>8 horas</td><td>80%</td></tr><tr><td>10 horas</td><td>100%</td></tr></table>
Exponencial: para este tipo, AWS AppConfig procesa la implementación exponenci</td></tr>
</table>

Opción	Descripción

	almente utilizando la siguiente fórmula: `G*(2^N)`. En esta fórmula, `G` es el porcentaj e de pasos especificado por el usuario y `N` es el número de pasos hasta que la configura ción se implementa en todos los destinos. Por ejemplo, si especifica un factor de crecimiento de 2, el sistema implementa la configuración de la siguiente manera:
	`2*(2^0)`
	`2*(2^1)`
	`2*(2^2)`
	Expresada numéricamente, la implement ación se despliega de la siguiente manera: 2 % de los destinos, 4 % de los destinos, 8 % de los destinos, y así sucesivamente hasta que la configuración se haya implementado en todos los destinos.

Porcentaje de pasos (factor de crecimiento)	Esta configuración especifica el porcentaje de intermediarios objetivo durante cada paso de la implementación. **Note** En el SDK y la Referencia de la API de AWS AppConfig, `step percentage` se llama `growth factor`.

Opción	Descripción
Tiempo de implementación	Esta configuración especifica una cantidad de tiempo durante el cual AWS AppConfig se implementa en los hosts. No se trata de un valor de tiempo de espera. Se trata de una ventana de tiempo durante la cual la implement ación se procesa en intervalos.
Tiempo procesamiento	Esta configuración especifica la cantidad de tiempo que AWS AppConfig supervisa las alarmas de Amazon CloudWatch después de que la configuración se haya implementado en el 100 % de sus objetivos, antes de considera r que la implementación está completa. Si se activa una alarma durante este tiempo, AWS AppConfig revierte la implementación. Debe configurar permisos para que AWS AppConfig pueda realizar restauraciones en función de las alarmas de CloudWatch. Para obtener más información, consulte (Opcional) Configure los permisos para la reversión en función de las alarmas CloudWatch .

Puede elegir una de las estrategias predefinidas que incluye AWS AppConfig o crear la suya propia.

Temas

- Estrategias de implementación predefinidas
- Creación de una estrategia de implementación

Estrategias de implementación predefinidas

AWS AppConfig incluye estrategias de implementación predefinidas para ayudarle a implementar rápidamente una configuración. En lugar de crear sus propias estrategias, puede elegir una de las siguientes opciones al implementar una configuración.

Estrategia de implementación	Descripción
AppConfig.Linear20PercentEvery6Minutes	AWSrecomendado: Esta estrategia implementa la configuración en el 20% de todos los objetivos cada seis minutos para una implementación de un 30 minutos. El sistema monitoriza las alarmas de Amazon CloudWatch durante 30 minutos. Si no se reciben alarmas en este momento, se completa la implementación. Si se activa una alarma durante este tiempo, AWS AppConfig revierte la implementación. Recomendamos usar esta estrategia para las implementaciones de producción porque se ajusta a las prácticas recomendadas de AWS y presta especial atención a la seguridad de la implementación debido a su larga duración y tiempo de incorporación.
AppConfig.Canary10Percent20Minutes	AWSrecomendado: Esta estrategia procesa la implementación exponencialmente utilizando un factor de crecimiento del 10 % durante 20 minutos. El sistema monitoriza las alarmas de Amazon CloudWatch durante 10 minutos. Si no se reciben alarmas en este momento, se completa la implementación. Si se activa una alarma durante este tiempo, AWS AppConfig revierte la implementación. Recomendamos utilizar esta estrategia para implementaciones de producción, ya que se ajusta a las prácticas recomendadas de AWS para implementaciones de configuración.

Estrategia de implementación	Descripción
AppConfig.AllAtOnce	Quick (Inmediata): Esta estrategia implementa la configuración en todos los destinos inmediatamente. El sistema monitoriza las alarmas de Amazon CloudWatch durante 10 minutos. Si no se reciben alarmas en este momento, se completa la implementación. Si se activa una alarma durante este tiempo, AWS AppConfig revierte la implementación.
AppConfig.Linear50PercentEvery30Seconds	Prueba/demostración: Esta estrategia implementa la configuración en la mitad de todos los objetivos cada 30 segundos para una implementación de un minuto. El sistema monitoriza las alarmas de Amazon CloudWatch durante 1 minuto. Si no se reciben alarmas en este momento, se completa la implementación. Si se activa una alarma durante este tiempo, AWS AppConfig revierte la implementación. Recomendamos utilizar esta estrategia solo con fines de prueba o demostración, ya que tiene una duración y tiempo de cocción cortos.

Creación de una estrategia de implementación

Puede crear un máximo de 20 estrategias de implementación. Al implementar una configuración, puede elegir la estrategia de implementación que funcione mejor para la aplicación y el entorno.

Creación de una estrategia de implementación de AWS AppConfig (consola)

Utilice el siguiente procedimiento para crear una estrategia de implementación de AWS AppConfig mediante la consola de AWS Systems Manager.

Para crear una estrategia de implementación

1. Abra la consola de AWS Systems Manager en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. Elija la pestaña Deployment Strategies (Estrategias de implementación) y, a continuación, elija Create deployment strategy (Crear estrategia de implementación).
4. En Name (Nombre), escriba un nombre para la estrategia de implementación.
5. En Description (Descripción), introduzca información sobre la estrategia de implementación.
6. En Deployment type (Tipo de implementación), elija un tipo.
7. En Step percentage (Porcentaje de pasos), elija el porcentaje de intermediarios objetivo durante cada paso de la implementación.
8. En Deployment time (Tiempo de implementación), especifique la duración total de la implementación en minutos u horas.
9. En el Tiempo de incorporación, introduzca el tiempo total, en minutos u horas, durante el que se monitorizarán las alarmas de Amazon CloudWatch antes de continuar con el siguiente paso de una implementación o antes de considerar que la implementación se ha completado.
10. En la sección Tags (Etiquetas) introduzca una clave y un valor opcional. Puede especificar un máximo de 50 etiquetas para un recurso.
11. Elija Create deployment strategy (Crear estrategia de implementación).

> **Important**
>
> Si ha creado un perfil de configuración para AWS CodePipeline, debe crear una canalización en CodePipeline que especifique AWS AppConfig como proveedor de implementación. No tiene que realizar Implementar una configuración. Sin embargo, debe configurar un cliente para recibir las actualizaciones de la configuración de la aplicación, tal y como se describe en Recuperación de configuraciones mediante una llamada directa a las API. Para obtener información sobre cómo crear una canalización que especifique AWS AppConfig como proveedor de implementación, consulte Tutorial: Creación de una canalización que utiliza AWS AppConfig como proveedor de implementación en la Guía del usuario de AWS CodePipeline.

Continúe en Implementar una configuración.

Creación de una estrategia de implementación de AWS AppConfig (línea de comandos)

En el siguiente procedimiento se describe cómo utilizar la AWS CLI (en Linux o Windows) o AWS Tools for PowerShell para crear una estrategia de implementaciónAWS AppConfig.

Crear una estrategia de implementación paso a paso

1. Abra la AWS CLI.
2. Ejecute el siguiente comando para crear una estrategia de implementación.

   Linux

```
aws appconfig create-deployment-strategy \
  --name A_name_for_the_deployment_strategy \
  --description A_description_of_the_deployment_strategy \
  --deployment-duration-in-minutes Total_amount_of_time_for_a_deployment_to_last
 \
  --final-bake-time-in-minutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 \
  --growth-
factor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_interva
 \
  --growth-
type The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_time
 \
  --replicate-
to To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document \
  --tags User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

   Windows

```
aws appconfig create-deployment-strategy ^
  --name A_name_for_the_deployment_strategy ^
  --description A_description_of_the_deployment_strategy ^
  --deployment-duration-in-minutes Total_amount_of_time_for_a_deployment_to_last
 ^
  --final-bake-time-in-minutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 ^
```

```
  --growth-
factor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_interva
 ^
  --growth-
type The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_time
 ^
  --name A_name_for_the_deployment_strategy ^
  --replicate-
to To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document ^
  --tags User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

PowerShell

```
New-APPCDeploymentStrategy `
  --Name A_name_for_the_deployment_strategy `
  --Description A_description_of_the_deployment_strategy `
  --DeploymentDurationInMinutes Total_amount_of_time_for_a_deployment_to_last `
  --FinalBakeTimeInMinutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 `
  --
GrowthFactor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_i
 `
  --
GrowthType The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_
 `
  --
ReplicateTo To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document
 `
  --
Tag Hashtable_type_User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

El sistema devuelve información similar a la siguiente.

Linux

```
{
   "Id": "Id of the deployment strategy",
   "Name": "Name of the deployment strategy",
   "Description": "Description of the deployment strategy",
   "DeploymentDurationInMinutes": "Total amount of time the deployment lasted",
```

```
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": "The percentage of targets that received a deployed
 configuration during each interval",
   "FinalBakeTimeInMinutes": "The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete",
   "ReplicateTo": "The Systems Manager (SSM) document where the deployment
 strategy is saved"
}
```

Windows

```
{
   "Id": "Id of the deployment strategy",
   "Name": "Name of the deployment strategy",
   "Description": "Description of the deployment strategy",
   "DeploymentDurationInMinutes": "Total amount of time the deployment lasted",
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": "The percentage of targets that received a deployed
 configuration during each interval",
   "FinalBakeTimeInMinutes": "The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete",
   "ReplicateTo": "The Systems Manager (SSM) document where the deployment
 strategy is saved"
}
```

PowerShell

```
ContentLength               : Runtime of the command
DeploymentDurationInMinutes : Total amount of time the deployment lasted
Description                 : Description of the deployment strategy
FinalBakeTimeInMinutes      : The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete
GrowthFactor                : The percentage of targets that received a deployed
 configuration during each interval
GrowthType                  : The linear or exponential algorithm used to define
 how percentage grew over time
HttpStatusCode              : HTTP Status of the runtime
Id                          : The deployment strategy ID
Name                        : Name of the deployment strategy
ReplicateTo                 : The Systems Manager (SSM) document where the
 deployment strategy is saved
```

```
ResponseMetadata            : Runtime Metadata
```

Implementar una configuración

Tras crear los artefactos necesarios para trabajar con marcas de características y datos de configuración de formato libre, puede crear una nueva implementación mediante la AWS Management Console, la AWS CLI o el SDK. Al iniciar una implementación en AWS AppConfig se llama a la acción de la API StartDeployment. Esta llamada incluye los ID de una aplicación de AWS AppConfig, entorno, perfil de configuración y (opcionalmente) la versión de datos de la configuración que se va a implementar. La llamada también incluye el ID de la estrategia de implementación que se va a utilizar, que determina cómo se implementan los datos de configuración.

Si implementan secretos almacenados en AWS Secrets Manager, objetos de Amazon Simple Storage Service (Amazon S3) cifrados con una clave administrada por el cliente o parámetros de cadena segura almacenados en el AWS Systems Manager almacén de parámetros cifrados con una clave administrada por el cliente, debe especificar un valor para el parámetro `KmsKeyIdentifier`. Si la configuración no está cifrada o lo está con una Clave administrada de AWS, no es necesario especificar un valor para el parámetro `KmsKeyIdentifier`.

> **Note**
>
> El valor que se especifica para `KmsKeyIdentifier` debe ser una clave administrada por el cliente. No tiene que ser la misma clave que utilizó para cifrar la configuración.
> Al iniciar una implementación con una `KmsKeyIdentifier`, la política de permisos adjunta a su entidad principal de AWS Identity and Access Management (IAM) debe permitir la operación `kms:GenerateDataKey`.

AWS AppConfig supervisa la distribución a todos los hosts y notifica el estado. Si se produce un error en una distribución, AWS AppConfig revierte la configuración.

> **Note**
>
> Solo puede implementar una única configuración a la vez en un entorno. Sin embargo, puede implementar una configuración por entorno en diferentes entornos al mismo tiempo.

Implementación de una configuración (consola)

Use el siguiente procedimiento para implementar una configuración de AWS AppConfig mediante la consola de AWS Systems Manager.

Para implementar una configuración mediante la consola

1. Abra la consola de AWS Systems Manager en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. En la pestaña Applications (Aplicaciones), elija la aplicación y, a continuación, elija View details (Ver detalles).
4. En la pestaña Environments (Entornos), elija un entorno y, a continuación, elija View details (Ver detalles).
5. Elija Start deployment (Iniciar la implementación).
6. En Configuration (Configuración), seleccione una configuración de la lista.
7. Dependiendo del origen de la configuración, utilice la lista Document version (Versión de documento) o Parameter version (Versión de parámetro) para elegir la versión que desea implementar.
8. En Deployment strategy (Estrategia de implementación), elija una estrategia de la lista.
9. En Deployment description (Descripción de la implementación), introduzca una descripción.
10. En la sección Tags (Etiquetas) introduzca una clave y un valor opcional. Puede especificar un máximo de 50 etiquetas para un recurso.
11. Elija Start deployment (Iniciar la implementación).

Implementación de una configuración (línea de comandos)

En el siguiente procedimiento se describe cómo utilizar la AWS CLI (en Linux o Windows) o AWS Tools for PowerShell para desplegar una estrategia de implementaciónAWS AppConfig.

Para implementar una configuración paso a paso

1. Abra la AWS CLI.
2. Ejecute el siguiente comando para desplegar una configuración.

Linux

```
aws appconfig start-deployment \
  --application-id The_application_ID \
  --environment-id The_environment_ID \
  --deployment-strategy-id The_deployment_strategy_ID \
  --configuration-profile-id The_configuration_profile_ID \
  --configuration-version The_configuration_version_to_deploy \
  --description A_description_of_the_deployment \
  --tags User_defined_key_value_pair_metadata_of_the_deployment
```

Windows

```
aws appconfig start-deployment ^
  --application-id The_application_ID ^
  --environment-id The_environment_ID ^
  --deployment-strategy-id The_deployment_strategy_ID ^
  --configuration-profile-id The_configuration_profile_ID ^
  --configuration-version The_configuration_version_to_deploy ^
  --description A_description_of_the_deployment ^
  --tags User_defined_key_value_pair_metadata_of_the_deployment
```

PowerShell

```
Start-APPCDeployment `
  -ApplicationId The_application_ID `
  -ConfigurationProfileId The_configuration_profile_ID `
  -ConfigurationVersion The_configuration_version_to_deploy `
  -DeploymentStrategyId The_deployment_strategy_ID `
  -Description A_description_of_the_deployment `
  -EnvironmentId The_environment_ID `
  -Tag Hashtable_type_user_defined_key_value_pair_metadata_of_the_deployment
```

El sistema devuelve información similar a la siguiente.

Linux

```
{
   "ApplicationId": "The ID of the application that was deployed",
   "EnvironmentId" : "The ID of the environment",
```

```
  "DeploymentStrategyId": "The ID of the deployment strategy that was
 deployed",
  "ConfigurationProfileId": "The ID of the configuration profile that was
 deployed",
  "DeploymentNumber": The sequence number of the deployment,
  "ConfigurationName": "The name of the configuration",
  "ConfigurationLocationUri": "Information about the source location of the
 configuration",
  "ConfigurationVersion": "The configuration version that was deployed",
  "Description": "The description of the deployment",
  "DeploymentDurationInMinutes": Total amount of time the deployment lasted,
  "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
  "GrowthFactor": The percentage of targets to receive a deployed configuration
 during each interval,
  "FinalBakeTimeInMinutes": Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete,
  "State": "The state of the deployment",

  "EventLog": [
     {
        "Description": "A description of the deployment event",
        "EventType": "The type of deployment event",
        "OccurredAt": The date and time the event occurred,
        "TriggeredBy": "The entity that triggered the deployment event"
     }
  ],

  "PercentageComplete": The percentage of targets for which the deployment is
 available,
  "StartedAt": The time the deployment started,
  "CompletedAt": The time the deployment completed
}
```

Windows

```
{
   "ApplicationId": "The ID of the application that was deployed",
   "EnvironmentId" : "The ID of the environment",
   "DeploymentStrategyId": "The ID of the deployment strategy that was
 deployed",
   "ConfigurationProfileId": "The ID of the configuration profile that was
 deployed",
```

```
  "DeploymentNumber": The sequence number of the deployment,
  "ConfigurationName": "The name of the configuration",
  "ConfigurationLocationUri": "Information about the source location of the
 configuration",
  "ConfigurationVersion": "The configuration version that was deployed",
  "Description": "The description of the deployment",
  "DeploymentDurationInMinutes": Total amount of time the deployment lasted,
  "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
  "GrowthFactor": The percentage of targets to receive a deployed configuration
 during each interval,
  "FinalBakeTimeInMinutes": Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete,
  "State": "The state of the deployment",

  "EventLog": [
     {
        "Description": "A description of the deployment event",
        "EventType": "The type of deployment event",
        "OccurredAt": The date and time the event occurred,
        "TriggeredBy": "The entity that triggered the deployment event"
     }
  ],

  "PercentageComplete": The percentage of targets for which the deployment is
 available,
  "StartedAt": The time the deployment started,
  "CompletedAt": The time the deployment completed
}
```

PowerShell

```
ApplicationId               : The ID of the application that was deployed
CompletedAt                 : The time the deployment completed
ConfigurationLocationUri    : Information about the source location of the
 configuration
ConfigurationName           : The name of the configuration
ConfigurationProfileId      : The ID of the configuration profile that was
 deployed
ConfigurationVersion        : The configuration version that was deployed
ContentLength               : Runtime of the deployment
DeploymentDurationInMinutes : Total amount of time the deployment lasted
DeploymentNumber            : The sequence number of the deployment
```

```
DeploymentStrategyId        : The ID of the deployment strategy that was
 deployed
Description                 : The description of the deployment
EnvironmentId               : The ID of the environment that was deployed
EventLog                    : {Description : A description of the deployment
 event, EventType : The type of deployment event, OccurredAt : The date and time
 the event occurred,
         TriggeredBy : The entity that triggered the deployment event}
FinalBakeTimeInMinutes      : Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete
GrowthFactor                : The percentage of targets to receive a deployed
 configuration during each interval
GrowthType                  : The linear or exponential algorithm used to define
 how percentage grew over time
HttpStatusCode              : HTTP Status of the runtime
PercentageComplete          : The percentage of targets for which the deployment
 is available
ResponseMetadata            : Runtime Metadata
StartedAt                   : The time the deployment started
State                       : The state of the deployment
```

Integración de la implementación de AWS AppConfig con CodePipeline

AWS AppConfig es una acción de implementación integrada para AWS CodePipeline (CodePipeline). CodePipeline es un servicio de entrega continua completamente administrado que lo ayuda a automatizar sus canales de lanzamiento para actualizaciones de infraestructura y aplicaciones rápidas y confiables. CodePipeline automatiza las fases de creación, prueba e implementación de su proceso de publicación cada vez que se produce un cambio de código, de acuerdo con el modelo de publicación que defina. Para obtener más información, consulte ¿Qué es AWS CodePipeline?

La integración de AWS AppConfig con CodePipeline ofrece los siguientes beneficios:

- Los clientes que utilizan CodePipeline para gestionar la organización ahora tienen un medio liviano de implementar cambios de configuración en sus aplicaciones sin tener que implementar toda su base de código.
- Los clientes que deseen utilizar AWS AppConfig para administrar implementaciones de configuración, pero están limitados porque AWS AppConfig no admite su código actual o almacén

de configuración, ahora tienen opciones adicionales. CodePipeline admite AWS CodeCommit, GitHub y BitBucket (por nombrar algunos).

> **Note**
>
> La integración de AWS AppConfig con CodePipeline solo se admite en Regiones de AWS donde CodePipeline esté disponible.

Cómo funciona la integración

Para empezar, hay que instalar y configurar CodePipeline, lo que incluye añadir la configuración a un almacén de código compatible con CodePipeline. A continuación, debe configurar su entorno de AWS AppConfig llevando a cabo las siguientes tareas:

- Creación de un espacio de nombres y un perfil de configuración
- Elija una estrategia de implementación predefinida o cree la suya propia

Tras completar estas tareas, debe crear una canalización en CodePipeline donde se especifique AWS AppConfig como el proveedor de implementación. A continuación, puede realizar un cambio en la configuración y cargarlo en el almacén de código de CodePipeline. Al cargar la nueva configuración, se inicia automáticamente una nueva implementación en CodePipeline. Tras completar la implementación, puede verificar los cambios. Para obtener información sobre cómo crear una canalización que especifique AWS AppConfig como proveedor de implementación, consulte Tutorial: Creación de una canalización que utiliza AWS AppConfig como proveedor de implementación en la Guía del usuario de AWS CodePipeline.

Recuperación de marcas de características y datos de configuración en AWS AppConfig

La aplicación recupera los indicadores de funciones y los datos de configuración de formato libre mediante el establecimiento de una sesión de configuración mediante el servicio de AWS AppConfig datos. Si utiliza uno de los métodos de recuperación simplificados que se describen en esta sección, la extensión AWS AppConfig Agent Lambda AWS AppConfig o el agente administra una serie de llamadas a la API y tokens de sesión en su nombre. Configura el AWS AppConfig agente como un host local y hace que el agente consulte si hay actualizaciones AWS AppConfig de configuración. El agente llama a las acciones StartConfigurationSessiony a la GetLatestConfigurationAPI y almacena en caché los datos de configuración de forma local. Para recuperar los datos, la aplicación realiza una llamada HTTP al servidor localhost. AWS AppConfig El agente admite varios casos de uso, como se describe enMétodos de recuperación simplificados.

Si lo prefiere, puede llamar manualmente a estas acciones de la API para recuperar una configuración. El proceso de la API funciona de la siguiente manera:

La aplicación establece una sesión de configuración mediante la acción de API `StartConfigurationSession`. A continuación, el cliente de la sesión realiza llamadas periódicas a `GetLatestConfiguration` para comprobar y recuperar los datos más recientes disponibles.

Al llamar`StartConfigurationSession`, el código envía los identificadores (ID o nombre) de una AWS AppConfig aplicación, un entorno y un perfil de configuración que la sesión rastrea.

Como respuesta, AWS AppConfig proporciona una `InitialConfigurationToken` que se proporciona al cliente de la sesión y que se utiliza la primera vez que llama a `GetLatestConfiguration` esa sesión.

Al llamar a `GetLatestConfiguration`, su código de cliente envía el valor más reciente de `ConfigurationToken` del que dispone y recibe como respuesta:

- `NextPollConfigurationToken`: el valor de `ConfigurationToken` que se utilizará en la siguiente llamada a `GetLatestConfiguration`.
- Configuración: los datos más recientes destinados a la sesión. Puede estar vacía si el cliente ya tiene la versión más reciente de la configuración.

Esta sección incluye la siguiente información.

Contenidos

Acerca del servicio AWS AppConfig de plano de datos

El 18 de noviembre de 2021, AWS AppConfig lanzó un nuevo servicio de plano de datos. Este servicio reemplaza el proceso anterior de recuperación de datos de configuración mediante la acción de API `GetConfiguration`. El servicio de plano de datos utiliza dos nuevas acciones de API, StartConfigurationSessiony GetLatestConfiguration. El servicio de plano de datos también utiliza nuevos puntos de conexión.

Si empezaste a usarlo AWS AppConfig antes del 28 de enero de 2022, es posible que el servicio esté llamando a la acción de `GetConfiguration` API directamente o que esté utilizando un cliente proporcionado por AWS, como la extensión AWS AppConfig Agent Lambda, para llamar a esta acción de API. Si llama directamente a la acción de API `GetConfiguration`, tome las medidas necesarias para utilizar las acciones de API `StartConfigurationSession` y `GetLatestConfiguration`. Si utiliza la extensión AWS AppConfig Agent Lambda, consulte la sección titulada Cómo afecta este cambio a la extensión Agent AWS AppConfig Lambda más adelante en este tema.

Las nuevas acciones de API del plano de datos ofrecen las siguientes ventajas en comparación con la acción de API `GetConfiguration`, que ahora ha quedado obsoleta.

1. No es necesario administrar un parámetro `ClientID`. Con el servicio de plano de datos, `ClientID` se administra internamente mediante el token de sesión creado por `StartConfigurationSession`.
2. Ya no es necesario incluir `ClientConfigurationVersion` para indicar la versión en caché de los datos de configuración. Con el servicio de plano de datos, `ClientConfigurationVersion` se administra internamente mediante el token de sesión creado por `StartConfigurationSession`.
3. El nuevo punto de conexión dedicado a las llamadas a la API del plano de datos mejora la estructura del código al separar las llamadas al plano de control y al plano de datos.

4. El nuevo servicio de plano de datos mejora la extensibilidad futura de las operaciones del plano de datos. Al utilizar una sesión de configuración que gestiona la recuperación de los datos de configuración, el equipo de AWS AppConfig puede crear mejoras más potentes en el futuro.

Migración desde **`GetConfiguration`** a **`GetLatestConfiguration`**

Para empezar a utilizar el nuevo servicio de plano de datos, debe actualizar el código que llama a la acción de API `GetConfiguration`. Inicie una sesión de configuración utilizando la acción de API `StartConfigurationSession` y, a continuación, llame a la acción de API `GetLatestConfiguration` para recuperar los datos de configuración. Para mejorar el rendimiento, le recomendamos que guarde en caché los datos de configuración de manera local. Para obtener más información, consulte Recuperación de configuraciones mediante una llamada directa a las API.

Cómo afecta este cambio a la extensión AWS AppConfig Agent Lambda

Este cambio no afecta directamente al funcionamiento de la extensión AWS AppConfig Agent Lambda. Las versiones anteriores de la extensión AWS AppConfig Agent Lambda llamaban a la acción de la `GetConfiguration` API en su nombre. Las versiones más recientes llaman a las acciones de API del plano de datos. Si utiliza la extensión de Lambda de AWS AppConfig , le recomendamos que actualice su extensión al nombre de recurso de Amazon (ARN) más reciente y actualice los permisos para las nuevas llamadas a la API. Para obtener más información, consulte Recuperación de datos de configuración mediante la extensión AWS AppConfig Agent Lambda.

Métodos de recuperación simplificados

AWS AppConfig ofrece varios métodos simplificados para recuperar los datos de configuración. Si utiliza indicadores de AWS AppConfig características o datos de configuración de formato libre en una AWS Lambda función, puede utilizar la extensión AWS AppConfig Agent Lambda para recuperar las configuraciones. Si tiene aplicaciones que se ejecutan en instancias de Amazon EC2, puede usar el AWS AppConfig agente para recuperar las configuraciones. AWS AppConfig El agente también admite aplicaciones que se ejecutan en imágenes de contenedor de Amazon Elastic Kubernetes Service (Amazon EKS) o Amazon Elastic Container Service (Amazon ECS).

Tras completar la configuración inicial, estos métodos de recuperación de los datos de configuración son más sencillos que llamar directamente a las API. AWS AppConfig Implementan automáticamente las mejores prácticas y pueden reducir el costo de uso, AWS AppConfig ya que se requieren menos llamadas a la API para recuperar las configuraciones.

Temas

Recuperación de datos de configuración mediante la extensión AWS AppConfig Agent Lambda

Una AWS Lambda extensión es un proceso complementario que aumenta las capacidades de una función Lambda. Una extensión puede iniciarse antes de que se invoque una función, ejecutarse en paralelo con una función y continuar ejecutándose después de que se procese la invocación de una función. Una extensión de Lambda es como un cliente que se ejecuta en paralelo a una invocación de Lambda. Este cliente paralelo puede interactuar con su función en cualquier momento de su ciclo de vida.

Si utiliza indicadores de AWS AppConfig características u otros datos de configuración dinámica en una función de Lambda, le recomendamos que añada la extensión Agent AWS AppConfig Lambda como capa a la función de Lambda. Esto simplifica la llamada a los indicadores de funciones, y la propia extensión incluye prácticas recomendadas que simplifican su uso y, al mismo AWS AppConfig tiempo, reducen los costes. La reducción de los costos se debe a la reducción de las llamadas a la API al AWS AppConfig servicio y a la reducción de los tiempos de procesamiento de las funciones de Lambda. Para obtener más información sobre las extensiones de Lambda, consulte Extensiones de Lambda en la Guía para desarrolladores de AWS Lambda .

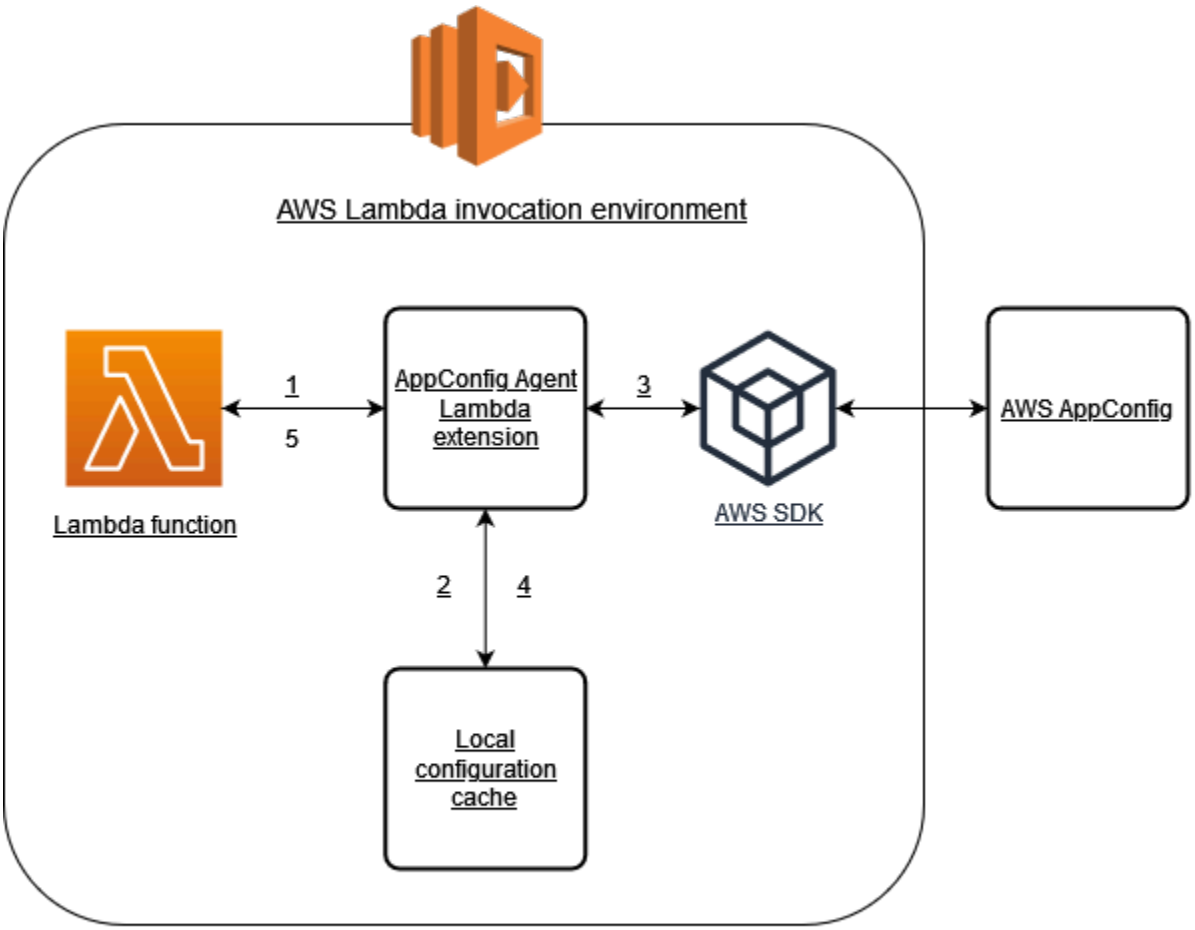
> **Note**
>
> AWS AppConfig los precios se basan en la cantidad de veces que se llama y se recibe una configuración. Sus costos aumentan si su Lambda realiza varios arranques en frío y recupera nuevos datos de configuración con frecuencia.

En este tema se incluye información sobre la extensión AWS AppConfig Agent Lambda y el procedimiento para configurar la extensión para que funcione con la función Lambda.

Cómo funcionan

Si se utiliza AWS AppConfig para gestionar las configuraciones de una función de Lambda sin extensiones de Lambda, debe configurar la función de Lambda para que reciba actualizaciones de configuración mediante la integración con las acciones de la API y de la API. StartConfigurationSessionGetLatestConfiguration

La integración de la extensión AWS AppConfig Agent Lambda con la función Lambda simplifica este proceso. La extensión se encarga de llamar al AWS AppConfig servicio, administrar una caché local de los datos recuperados, rastrear los identificadores de configuración necesarios para las próximas llamadas al servicio y comprobar periódicamente si hay actualizaciones de configuración en segundo plano. El siguiente diagrama muestra cómo funciona.



1. La extensión AWS AppConfig Agent Lambda se configura como una capa de la función Lambda.

2. Para acceder a sus datos de configuración, la función llama a la AWS AppConfig extensión en un punto final HTTP que se esté ejecutando. `localhost:2772`
3. La extensión mantiene una caché local de los datos de configuración. Si los datos no están en la caché, la extensión llama AWS AppConfig para obtener los datos de configuración.
4. Al recibir la configuración del servicio, la extensión la almacena en la memoria caché local y la pasa a la función de Lambda.
5. AWS AppConfig La extensión Agent Lambda comprueba periódicamente si hay actualizaciones en los datos de configuración en segundo plano. Cada vez que se invoca la función de Lambda, la extensión comprueba el tiempo transcurrido desde que recuperó una configuración. Si el tiempo transcurrido es superior al intervalo de sondeo configurado, la extensión llama AWS AppConfig para comprobar si hay datos recién implementados, actualiza la memoria caché local si se ha producido algún cambio y restablece el tiempo transcurrido.

> **Note**
>
> - Lambda crea instancias independientes correspondientes al nivel de simultaneidad que requiere la función. Cada instancia está aislada y mantiene su propia memoria caché local de los datos de configuración. Para obtener más información sobre las instancias de Lambda y la simultaneidad, consulte Administración de la simultaneidad para una función de Lambda.
> - El tiempo que tarda un cambio de configuración en aparecer en una función Lambda, después de implementar una configuración actualizada AWS AppConfig, depende de la estrategia de implementación que haya utilizado para la implementación y del intervalo de sondeo que haya configurado para la extensión.

Antes de empezar

Antes de activar la extensión AWS AppConfig Agent Lambda, haga lo siguiente:

- Organice las configuraciones de la función de Lambda para poder externalizarlas en AWS AppConfig.
- Cree AWS AppConfig artefactos y datos de configuración, incluidos indicadores de características o datos de configuración de formato libre. Para obtener más información, consulte Creación de indicadores de características y datos de configuración de formato libre en AWS AppConfig.

- Agregue `appconfig:StartConfigurationSession` y `appconfig:GetLatestConfiguration` a la política AWS Identity and Access Management (IAM) utilizada por el rol de ejecución de la función Lambda. Para obtener más información, consulte Rol de ejecución de AWS Lambda en la Guía para desarrolladores de AWS Lambda . Para obtener más información sobre permisos de AWS AppConfig , consulte Acciones, recursos y claves de condición para AWS AppConfig en la Referencia de autorizaciones de servicio.

Añadir la extensión AWS AppConfig Agent Lambda

Para utilizar la extensión AWS AppConfig Agent Lambda, debe añadir la extensión a su Lambda. Esto se puede hacer añadiendo la extensión AWS AppConfig Agent Lambda a la función Lambda como capa o activando la extensión en una función Lambda como imagen contenedora.

> **Note**
>
> La AWS AppConfig extensión es independiente del tiempo de ejecución y es compatible con todos los tiempos de ejecución.

Agregar la extensión de Lambda del agente de AWS AppConfig mediante una capa y un ARN

Para usar la extensión AWS AppConfig Agent Lambda, añada la extensión a la función Lambda como una capa. Para obtener información sobre cómo añadir una capa a la función, consulte Configuración de extensiones en la Guía del desarrollador de AWS Lambda . El nombre de la extensión en la AWS Lambda consola es AWS- AppConfig -Extension. Tenga en cuenta también que cuando agregue la extensión como una capa a su Lambda, debe especificar un nombre de recurso de Amazon (ARN). Elija un ARN de una de las siguientes listas que corresponda a la plataforma y al Región de AWS lugar donde creó la Lambda.

- plataformas x86-64
- Plataforma ARM64

Si desea probar la extensión antes de añadirla a la función, puede comprobar que funciona mediante el siguiente ejemplo de código.

```
import urllib.request
```

```
def lambda_handler(event, context):
    url = f'http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name'
    config = urllib.request.urlopen(url).read()
    return config
```

Para probarlo, cree una función de Lambda nueva para Python, agregue la extensión y ejecute la función de Lambda. Tras ejecutar la función Lambda, la función AWS AppConfig Lambda devuelve la configuración que especificó para la ruta http://localhost:2772. Para obtener información sobre la creación de una función de Lambda, consulte Creación de una función Lambda con la consola en la Guía para desarrolladores de AWS Lambda .

Para añadir la extensión AWS AppConfig Agent Lambda como imagen de contenedor, consulte. Uso de una imagen de contenedor para añadir la extensión AWS AppConfig Agent Lambda

Configuración de la extensión de Lambda del agente de AWS AppConfig

Puede configurar la extensión cambiando las siguientes variables de AWS Lambda entorno. Para obtener más información, consulte Uso de variables de AWS Lambda entorno en la Guía para AWS Lambda desarrolladores.

Recuperación previa de los datos de configuración

La variable de entorno `AWS_APPCONFIG_EXTENSION_PREFETCH_LIST` puede mejorar el tiempo de inicio de la función. Cuando se inicializa la extensión AWS AppConfig Agent Lambda, recupera la configuración especificada antes de que AWS AppConfig Lambda comience a inicializar la función e invocar el controlador. En algunos casos, los datos de configuración ya están disponibles en la memoria caché local antes de que la función los solicite.

Para utilizar la función de obtención previa, defina el valor de la variable de entorno en la ruta correspondiente a los datos de configuración. Por ejemplo, si la configuración corresponde a una aplicación, un entorno y un perfil de configuración denominados respectivamente "my_application", "my_environment" y "my_configuration_data", la ruta sería `/applications/my_application/environments/my_environment/configurations/my_configuration_data`. Puede especificar varios elementos de configuración enumerándolos en una lista separada por comas (si tiene un nombre de recurso que incluye una coma, utilice el valor de ID del recurso en lugar de su nombre).

Acceso a los datos de configuración desde otra cuenta

La extensión AWS AppConfig Agent Lambda puede recuperar datos de configuración de otra cuenta especificando un rol de IAM que concede permisos a los datos. Para establecer esta política, siga estos pasos:

1. En la cuenta en la que AWS AppConfig se utilizan para administrar los datos de configuración, cree un rol con una política de confianza que conceda a la cuenta que ejecuta la función Lambda acceso a las `appconfig:GetLatestConfiguration` acciones `appconfig:StartConfigurationSession` y, junto con los ARN parciales o completos correspondientes a los AWS AppConfig recursos de configuración.
2. En la cuenta que ejecuta la función de Lambda, agregue la variable de entorno `AWS_APPCONFIG_EXTENSION_ROLE_ARN` a la función de Lambda con el ARN del rol creado en el paso 1.
3. (Opcional) Si es necesario, se puede especificar un ID externo mediante la variable de entorno `AWS_APPCONFIG_EXTENSION_ROLE_EXTERNAL_ID`. Del mismo modo, se puede configurar un nombre de sesión mediante la variable de entorno `AWS_APPCONFIG_EXTENSION_ROLE_SESSION_NAME`.

> **Note**
>
> Observe la siguiente información.
>
> - La extensión AWS AppConfig Agent Lambda solo puede recuperar datos de una cuenta. Si especifica un rol de IAM, la extensión no podrá recuperar los datos de configuración de la cuenta en la que se ejecuta la función de Lambda.
> - AWS Lambda registra información sobre la extensión AWS AppConfig Agent Lambda y la función Lambda mediante Amazon Logs. CloudWatch

Variable de entorno	Detalles	Valor predeterminado
`AWS_APPCONFIG_EXTENSION_HTTP_PORT`	Esta variable de entorno especifica el puerto en el que se ejecuta el servidor HTTP local que aloja la extensión.	2772

Variable de entorno	Detalles	Valor predeterminado
`AWS_APPCONFIG_EXTENSION_LOG_LEVEL`	Esta variable de entorno especifica qué registros AWS AppConfig específicos de la extensión se envían a Amazon CloudWatch Logs para una función. Los valores válidos que no distinguen entre mayúsculas y minúsculas son `debug`, `info`, `warn`, `error` y `none`. La depuración incluye información detallada, incluida la información sobre el tiempo y la extensión.	`info`
`AWS_APPCONFIG_EXTENSION_MAX_CONNECTIONS`	Esta variable de entorno configura el número máximo de conexiones que la extensión utiliza para recuperar configuraciones de AWS AppConfig.	3
`AWS_APPCONFIG_EXTENSION_POLL_INTERVAL_SECONDS`	Esta variable de entorno controla la frecuencia con la que la extensión AWS AppConfig busca una configuración actualizada en cuestión de segundos.	45

Variable de entorno	Detalles	Valor predeterminado
`AWS_APPCONFIG_EXTENSION_POLL_TIMEOUT_MILLIS`	Esta variable de entorno controla el tiempo máximo, en milisegundos, durante el que la extensión espera una respuesta AWS AppConfig al actualizar los datos de la caché. Si AWS AppConfig no responde en el período de tiempo especificado, la extensión omite este intervalo de sondeo y devuelve los datos almacenados en caché previamente actualizados.	3 000
`AWS_APPCONFIG_EXTENSION_PREFETCH_LIST`	Esta variable de entorno especifica los datos de configuración que la extensión comienza a recuperar antes de que la función se inicialice y se ejecute el controlador. Puede reducir considerablemente el tiempo de inicio en frío de la función.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
AWS_APPCONFIG_EXTENSION_PROXY_HEADERS	Esta variable de entorno especifica los encabezados requeridos por el proxy al que se hace referencia en la variable de entorno AWS_APPCONFIG_EXTENSION_PROXY_URL . El valor es una lista de encabezados separados por comas. Todos los encabezados utilizan el siguiente formulario: `"header: value"`	Ninguna
AWS_APPCONFIG_EXTENSION_PROXY_URL	Esta variable de entorno especifica la URL del proxy que se utilizará para las conexiones desde la AWS AppConfig extensión a. Servicios de AWSHTTPSy se admiten las HTTP direcciones URL.	Ninguna
AWS_APPCONFIG_EXTENSION_ROLE_ARN	Esta variable de entorno especifica el ARN del rol de IAM correspondiente a un rol que la extensión debe asumir para recuperar AWS AppConfig la configuración.	Ninguna
AWS_APPCONFIG_EXTENSION_ROLE_EXTERNAL_ID	Esta variable de entorno especifica el identificador externo que se utilizará junto con el ARN del rol asumido.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
`AWS_APPCONFIG_EXTENSION_ROLE_SESSION_NAME`	Esta variable de entorno especifica el nombre de la sesión que se va a asociar a las credenciales del rol de IAM asumido.	Ninguna
`AWS_APPCONFIG_EXTENSION_SERVICE_REGION`	Esta variable de entorno especifica una región alternativa que la extensión debe usar para llamar al AWS AppConfig servicio. Si no se ha definido, la extensión usa el punto de conexión de la región actual.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
`AWS_APPCONFIG_EXTENSION_MANIFEST`	Esta variable de entorno configura el AWS AppConfig Agente para que aproveche las funciones adicionales por configuración, como las recuperaciones de varias cuentas y el almacenamiento de la configuración en el disco. Puede introducir uno de los siguientes valores: • `"app:env:manifest-config"` • `"file:/fully/qualified/path/to/manifest.json"` Para obtener más información sobre el uso de estas características, consulte Funciones de recuperación adicionales.	true
`AWS_APPCONFIG_EXTENSION_WAIT_ON_MANIFEST`	Esta variable de entorno configura el AWS AppConfig agente para que espere hasta que se procese el manifiesto antes de completar el inicio.	true

Recuperación de una o varias marcas de una configuración de marcas de características

Para las configuraciones de marcas de características (configuraciones de tipo `AWS.AppConfig.FeatureFlags`), la extensión de Lambda le permite recuperar una sola marca o

un subconjunto de marcas de una configuración. Recuperar una o dos marcas es útil si su Lambda solo necesita usar algunas marcas del perfil de configuración. El siguiente ejemplo usa Python.

> **Note**
>
> La posibilidad de llamar a un único indicador de función o a un subconjunto de indicadores en una configuración solo está disponible en la versión 2.0.45 y versiones posteriores de la extensión AWS AppConfig Agent Lambda.

Puede recuperar los datos de AWS AppConfig configuración de un punto final HTTP local. Para acceder a una marca específica o a una lista de marcas, utilice el parámetro de consulta ? `flag=flag_name` para un perfil de configuración de AWS AppConfig .

Para acceder a una única marca y sus atributos

```
import urllib.request

def lambda_handler(event, context):
    url = f'http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?flag=flag_name'
    config = urllib.request.urlopen(url).read()
    return config
```

Para acceder a varias marcas y sus atributos

```
import urllib.request

def lambda_handler(event, context):
    url = f'http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?
flag=flag_name_one&flag=flag_name_two'
    config = urllib.request.urlopen(url).read()
    return config
```

Versiones disponibles de la extensión AWS AppConfig Agent Lambda

En este tema se incluye información sobre las versiones de la extensión AWS AppConfig Agent Lambda. La extensión de Lambda del agente de AWS AppConfig admite las funciones de Lambda desarrolladas para las plataformas x86-64 y ARM64 (Graviton2). Para que funcione correctamente, la función Lambda debe estar configurada para usar el nombre de recurso de Amazon (ARN) específico

del Región de AWS lugar donde está alojada actualmente. Puede ver Región de AWS los detalles del ARN más adelante en esta sección.

> **Important**
>
> Tenga en cuenta los siguientes detalles importantes sobre la extensión AWS AppConfig Agent Lambda.
>
> - La acción de API `GetConfiguration` quedó obsoleta el 28 de enero de 2022. Las llamadas para recibir datos de configuración deben usar las API `StartConfigurationSession` y `GetLatestConfiguration` en su lugar. Si utiliza una versión de la extensión AWS AppConfig Agent Lambda creada antes del 28 de enero de 2022, es posible que tenga que configurar los permisos para las nuevas API. Para obtener más información, consulte Acerca del servicio AWS AppConfig de plano de datos.
> - AWS AppConfig es compatible con todas las versiones enumeradas enVersiones anteriores de la extensión. Se recomienda actualizar periódicamente a la última versión para aprovechar las mejoras de la extensión.

Temas

- Notas de la versión de la extensión de Lambda del agente de AWS AppConfig
- Búsqueda del número de versión de la extensión de Lambda
- plataformas x86-64
- Plataforma ARM64
- Versiones anteriores de la extensión

Notas de la versión de la extensión de Lambda del agente de AWS AppConfig

En la siguiente tabla se describen los cambios realizados en las versiones recientes de la AWS AppConfig extensión Lambda.

Versión	Fecha de lanzamiento	Notas
2.0.358	12/01/2023	Se agregó soporte para las siguientes funciones de recuperación:

Versión	Fecha de lanzamiento	Notas
		• Recuperación de varias cuentas: utilice el AWS AppConfig agente de una cuenta principal o una recuperación Cuenta de AWS para recuperar los datos de configuración de varias cuentas de proveedor es. • Escribir la copia de la configuración en el disco: utilice el AWS AppConfig agente para escribir los datos de configuración en el disco. Esta función permite a los clientes con aplicacio nes que leen los datos de configuración del disco integrarse en ellas AWS AppConfig.
2.0.181	14/08/2023	Se agregó soporte para el Región de AWS il-central-1 de Israel (Tel Aviv).

Versión	Fecha de lanzamiento	Notas
2.0.165	21/02/2023	Correcciones de errores menores. Ya no se restringe el uso de extensiones a versiones de ejecución específicas a través de la AWS Lambda consola. Añadida compatibilidad con los siguientes Regiones de AWS: • Medio Oriente (EAU): me-central-1 • Asia Pacífico (Hyderabad): ap-south-2 • Asia Pacífico (Melbourne): ap-southeast-4 • Europa (España): eu-south-2 • Europa (Zúrich): eu-central-2

Versión	Fecha de lanzamiento	Notas
2.0.122	23/08/2022	Se agregó soporte para un proxy de tunelización, que se puede configurar con las variables de entorno `AWS_APPCONFIG_EXTENSION_PROXY_URL` y `AWS_APPCONFIG_EXTENSION_PROXY_HEADERS`. Se agregó .NET 6 como tiempo de ejecución. Para obtener más información sobre las variables de entorno, consulte Configuración de la extensión de Lambda del agente de AWS AppConfig.
2.0.58	05/03/2022	Soporte mejorado para los procesadores Graviton2 (ARM64) en Lambda.

Versión	Fecha de lanzamiento	Notas
2.0.45	15/03/2022	Se agregó soporte para llamar a una sola marca de caracterí sticas. Anteriormente, los clientes llamaban a las marcas de características agrupados en un perfil de configura ción y tenían que analizar la respuesta desde el lado del cliente. Con esta versión, los clientes pueden usar un parámetro `flag=<flag-name>` al llamar al punto de conexión HTTP localhost para obtener el valor de una única marca. También se agregó soporte inicial para los procesadores Graviton2 (ARM64).

Búsqueda del número de versión de la extensión de Lambda

Utilice el siguiente procedimiento para localizar el número de versión de la extensión AWS AppConfig Agent Lambda actualmente configurada. Para que funcione correctamente, la función Lambda debe estar configurada para usar el nombre de recurso de Amazon (ARN) específico del Región de AWS lugar donde está alojada actualmente.

1. Inicie sesión en la AWS Lambda consola AWS Management Console y ábrala en https://console.aws.amazon.com/lambda/.
2. Elija la función de Lambda en la que desea agregar la capa de `AWS-AppConfig-Extension`.
3. En el área Capas, elija Agregar una capa.
4. En la sección Elegir una capa, selecciona AWS- AppConfig -Extensión en la lista de AWS capas.
5. Utilice la lista Versión para elegir un número de versión.
6. Elija Añadir.

7. Utilice la pestaña Probar para probar la función.
8. Una vez finalizada la prueba, consulte el resultado del registro. Localice la versión de la extensión AWS AppConfig Agent Lambda en la sección Detalles de la ejecución. Esta versión debe coincidir con las URL requeridas para esa versión.

plataformas x86-64

Al añadir la extensión como capa a la Lambda, debe especificar un ARN. Elija un ARN de la siguiente tabla que se corresponda con el Región de AWS lugar en el que creó la Lambda. Estos ARN son para funciones de Lambda desarrolladas para la plataforma x86-64.

Versión 2.0.358

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:128`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:93`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:141`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:161`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:93`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:106`

Región	ARN
Europa (Zúrich)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:47
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:125
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:93
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:98
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:159
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:83
Europa (España)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:44
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:76
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:76

Región	ARN
Asia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:83`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:98`
Asia-Pacífico (Seúl)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:108`
Asia-Pacífico (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:101`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:106`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:106`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:79`
Asia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:20`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:107`

Región	ARN
Asia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:47`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:128`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:83`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:22`
Medio Oriente (EAU)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:49`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:85`
AWS GovCloud (Este de EE. UU.)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:54`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:54`

Plataforma ARM64

Al añadir la extensión como capa a la Lambda, debe especificar un ARN. Elija un ARN de la siguiente tabla que se corresponda con el Región de AWS lugar en el que creó la Lambda. Estos ARN son para funciones de Lambda desarrolladas para la plataforma ARM64.

Versión 2.0.358

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:61`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:45`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:18`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:63`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:13`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:49`
Europa (Zúrich)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:5`

Región	ARN
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:63
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:45
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:17
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:18
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:11
Europa (España)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:5
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:11
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:51
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:16

Región	ARN
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:16
Asia-Pacífico (Singapur)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:58
Asia-Pacífico (Sídney)	arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:49
Asia-Pacífico (Yakarta)	arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:16
Asia-Pacífico (Melbourne)	arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:5
Asia-Pacífico (Bombay)	arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:49
Asia-Pacífico (Hyderabad)	arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:5
América del Sur (São Paulo)	arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:16
África (Ciudad del Cabo)	arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:11

Región	ARN
Medio Oriente (EAU)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:5`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:13`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:5`

Versiones anteriores de la extensión

En esta sección se enumeran los ARN y Regiones de AWS las versiones anteriores de la extensión AWS AppConfig Lambda. Esta lista no contiene información sobre todas las versiones anteriores de la extensión de Lambda del agente de AWS AppConfig , pero se actualizará cuando se publiquen nuevas versiones.

Versiones anteriores de la extensión (plataforma x86-64)

En las tablas siguientes se enumeran los ARN y las Regiones de AWS versiones anteriores de la extensión AWS AppConfig Agent Lambda desarrollada para la plataforma x86-64.

Fecha de sustitución por una extensión más reciente: 1 de diciembre de 2023

Versión 2.0.181

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:113`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:81`

Región	ARN
Oeste de EE. UU. (Norte de California)	arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:124
Oeste de EE. UU. (Oregón)	arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:146
Canadá (centro)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:81
Europa (Fráncfort)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:93
Europa (Zúrich)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:32
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:110
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:81
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:82
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:142

Región	ARN
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:73
Europa (España)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:29
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:68
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:68
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:73
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:84
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:93
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:86
Asia-Pacífico (Singapur)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:91

Región	ARN
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:93`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:64`
Asia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:5`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:94`
Asia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:32`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:113`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:73`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:7`
Medio Oriente (EAU)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:34`

Región	ARN
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:73`
AWS GovCloud (Este de EE. UU.)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:46`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:46`

Fecha de sustitución por una extensión más reciente: 08/14/2023

Versión 2.0.165

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:110`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:79`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:121`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:143`

Región	ARN
Canadá (centro)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:79
Europa (Fráncfort)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:91
Europa (Zúrich)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:29
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:108
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:79
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:80
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:139
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:71
Europa (España)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:26

Región	ARN
China (Pekín)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:66`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:66`
Asia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:71`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:82`
Asia-Pacífico (Seúl)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:91`
Asia-Pacífico (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:84`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:89`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:91`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:60`

Región	ARN
Asia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:2`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:92`
Asia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:29`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:110`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:71`
Medio Oriente (EAU)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:31`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:71`
AWS GovCloud (EE. UU.-Este)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:44`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:44`

Fecha de sustitución por una extensión más reciente: 02/21/2023

Versión 2.0.122

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:82`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:59`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:93`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:114`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:59`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:70`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:82`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:59`

Región	ARN
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:60
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:111
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:54
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:52
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:52
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:54
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:62
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:70
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:59

Región	ARN
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:64`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:70`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:37`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:71`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:82`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:54`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:54`
AWS GovCloud (EE. UU.-Este)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:29`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:29`

Fecha de sustitución por una extensión más reciente: 08/23/2022

Versión 2.0.58

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:69`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:50`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:78`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:101`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:50`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:59`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:69`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:50`

Región	ARN
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:51
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:98
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:47
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:46
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:46
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:47
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:49
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:59
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:46

Región	ARN
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:51`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:59`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:24`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:60`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:69`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:47`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:47`
AWS GovCloud (EE. UU.-Este)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:23`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:23`

Fecha de sustitución por una extensión más reciente: 04/21/2022

Versión 2.0.45

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:68`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:49`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:77`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:100`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:49`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:58`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:68`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:49`

Región	ARN
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:50
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:97
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:46
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:45
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:45
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:46
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:48
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:58
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:45

Región	ARN
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:50`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:58`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:23`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:59`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:68`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:46`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:46`
AWS GovCloud (EE. UU.-Este)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:22`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:22`

Fecha de sustitución por una extensión más reciente: 03/15/2022

Versión 2.0.30

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:61`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:47`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:61`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:89`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:47`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:54`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:59`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:47`

Región	ARN
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:48
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:86
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:44
China (Pekín)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:43
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:43
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:44
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:45
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:42
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:54

Región	ARN
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:45`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:54`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:13`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:55`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:61`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:44`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:44`
AWS GovCloud (EE. UU.-Este)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:20`
AWS GovCloud (Estados Unidos-Oeste)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:20`

Versiones anteriores de la extensión (plataforma ARM64)

En las siguientes tablas se enumeran los ARN y las Regiones de AWS versiones anteriores de la extensión AWS AppConfig Agent Lambda desarrollada para la plataforma ARM64.

Fecha de sustitución por una extensión más reciente: 1 de diciembre de 2023

Versión 2.0.181

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:46`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:33`
Oeste de EE. UU. (Norte de California)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:1`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:48`
Canadá (centro)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:36`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:48`

Región	ARN
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:33
Europa (París)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:1
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:1
Europa (Milán)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:1
Asia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:1
Asia-Pacífico (Tokio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:37
Asia-Pacífico (Seúl)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:1
Asia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:1
Asia-Pacífico (Singapur)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:43

Región	ARN
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:36`
Asia-Pacífico (Yakarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:1`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:36`
América del Sur (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:1`
África (Ciudad del Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:1`
Medio Oriente (Baréin)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:1`

Fecha de sustitución por una extensión más reciente: 03/30/2023

Versión 2.0.165

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:43`

Región	ARN
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:31`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:45`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:34`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:46`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:31`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:35`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:41`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:34`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:34`

Fecha de sustitución por una extensión más reciente: 02/21/2023

Versión 2.0.122

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:15`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:11`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:16`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:13`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:20`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:11`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:15`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:16`

Región	ARN
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:13`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:13`

Fecha de sustitución por una extensión más reciente: 08/23/2022

Versión 2.0.58

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:2`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:2`
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:3`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:2`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:7`

Región	ARN
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:2`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:2`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:3`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:2`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:2`

Fecha de sustitución por una extensión más reciente: 04/21/2022

Versión 2.0.45

Región	ARN
Este de EE. UU. (Norte de Virginia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:1`
Este de EE. UU. (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:1`

Región	ARN
Oeste de EE. UU. (Oregón)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:2`
Europa (Fráncfort)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:6`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:1`
Asia-Pacífico (Tokio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:1`
Asia-Pacífico (Singapur)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:2`
Asia-Pacífico (Sídney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:1`
Asia-Pacífico (Bombay)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:1`

Uso de una imagen de contenedor para añadir la extensión AWS AppConfig Agent Lambda

Puede empaquetar la extensión AWS AppConfig Agent Lambda como una imagen de contenedor para subirla a su registro de contenedores alojado en Amazon Elastic Container Registry (Amazon ECR).

Para añadir la extensión AWS AppConfig Agent Lambda como imagen de contenedor de Lambda

1. Introduzca el nombre del recurso de Amazon (ARN) Región de AWS y el nombre del recurso de Amazon en AWS Command Line Interface (AWS CLI), tal y como se muestra a continuación. Sustituya el valor de región y ARN por su región y el ARN correspondiente para recuperar una copia de la capa de Lambda.

   ```
   aws lambda get-layer-version-by-arn \
     --region us-east-1 \
     --arn arn:aws:lambda::layer:AWS-AppConfig-Extension:00
   ```

   A continuación, se muestra la respuesta.

   ```
   {
     "LayerVersionArn": "arn:aws:lambda::layer:AWS-AppConfig-Extension:00",
     "Description": "AWS AppConfig extension: Use dynamic configurations deployed via
    AWS AppConfig for your AWS Lambda functions",
     "CreatedDate": "2021-04-01T02:37:55.339+0000",
     "LayerArn": "arn:aws:lambda::layer:AWS-AppConfig-Extension",

     "Content": {
       "CodeSize": 5228073,
       "CodeSha256": "8otOgbLQbexpUm3rKlMhvcE6Q5TvwcLCKrc4Oe+vmMY=",
       "Location" : "S3-Bucket-Location-URL"
     },

     "Version": 30,
     "CompatibleRuntimes": [
       "python3.8",
       "python3.7",
       "nodejs12.x",
       "ruby2.7"
     ],
   }
   ```

2. En la respuesta anterior, el valor devuelto para `Location` es la URL del bucket de Amazon Simple Storage Service (Amazon S3) que contiene la extensión de Lambda. Pegue la URL en su navegador web para descargar un archivo.zip con la extensión de Lambda.

   > **Note**
   >
   > La URL del bucket de Amazon S3 solo estará disponible durante 10 minutos.

   (Opcional) También puede utilizar el siguiente comando `curl` para descargar la extensión de Lambda.

   ```
   curl -o extension.zip "S3-Bucket-Location-URL"
   ```

   (Opcional) Como alternativa, puede combinar el paso 1 y el paso 2 para recuperar el ARN y descargar el archivo de extensión .zip de una sola vez.

   ```
   aws lambda get-layer-version-by-arn \
     --arn arn:aws:lambda::layer:AWS-AppConfig-Extension:00 \
     | jq -r '.Content.Location' \
     | xargs curl -o extension.zip
   ```

3. Agregue las siguientes líneas en su `Dockerfile` para agregar la extensión a su imagen de contenedor.

   ```
   COPY extension.zip extension.zip
   RUN yum install -y unzip \
     && unzip extension.zip /opt \
     && rm -f extension.zip
   ```

4. Asegúrese de que la función de ejecución de la función Lambda tenga el conjunto de permisos appconfig: GetConfiguration.

Ejemplo

En esta sección se incluye un ejemplo para habilitar la extensión AWS AppConfig Agent Lambda en una función de Python Lambda basada en imágenes de contenedores.

1. Cree un `Dockerfile` similar al que se muestra a continuación:

```
FROM public.ecr.aws/lambda/python:3.8 AS builder
COPY extension.zip extension.zip
RUN yum install -y unzip \
  && unzip extension.zip -d /opt \
  && rm -f extension.zip

FROM public.ecr.aws/lambda/python:3.8
COPY --from=builder /opt /opt
COPY index.py ${LAMBDA_TASK_ROOT}
CMD [ "index.handler" ]
```

2. Descargue la capa de extensión en el mismo directorio que el `Dockerfile`.

```
aws lambda get-layer-version-by-arn \
  --arn arn:aws:lambda::layer:AWS-AppConfig-Extension:00 \
  | jq -r '.Content.Location' \
  | xargs curl -o extension.zip
```

3. Cree un archivo de Python con el nombre `index.py` en el mismo directorio que el `Dockerfile`.

```
import urllib.request

def handler(event, context):
    return {
       # replace parameters here with your application, environment, and
 configuration names
       'configuration': get_configuration('myApp', 'myEnv', 'myConfig'),
    }

def get_configuration(app, env, config):
    url = f'http://localhost:2772/applications/{app}/environments/{env}/
configurations/{config}'
    return urllib.request.urlopen(url).read()
```

4. Ejecute los siguientes pasos para crear la imagen de `docker` y cargarla en Amazon ECR.

```
// set environment variables
export ACCOUNT_ID = <YOUR_ACCOUNT_ID>
export REGION = <AWS_REGION>

// create an ECR repository
aws ecr create-repository --repository-name test-repository
```

```
// build the docker image
docker build -t test-image .

// sign in to AWS
aws ecr get-login-password \
  | docker login \
  --username AWS \
  --password-stdin "$ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com"

// tag the image
docker tag test-image:latest "$ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com/test-
repository:latest"

// push the image to ECR
docker push "$ACCOUNT_ID.dkr.ecr.$REGION.amazonaws.com/test-repository:latest"
```

5. Utilice la imagen de Amazon ECR que acaba de crear para crear la función de Lambda. Para obtener más información sobre una función de Lambda como contenedor, consulte Crear una función de Lambda definida como una imagen de contenedor.
6. Asegúrese de que la función de ejecución de la función Lambda tenga el conjunto de permisos appconfig: GetConfiguration.

Integración con OpenAPI

Puede usar la siguiente especificación de YAML para OpenAPI para crear un SDK con una herramienta como OpenAPI Generator. Puede actualizar esta especificación para incluir valores codificados para la aplicación, el entorno o la configuración. También puede agregar rutas adicionales (si tiene varios tipos de configuración) e incluir esquemas de configuración para generar modelos con tipo específicos de la configuración para sus clientes de SDK. Para obtener más información sobre OpenAPI (también conocida como Swagger), consulte la especificación de OpenAPI.

```
openapi: 3.0.0
info:
  version: 1.0.0
  title: AppConfig Agent Lambda extension API
  description: An API model for the AppConfig Agent Lambda extension.
servers:
  - url: https://localhost:{port}/
```

```
    variables:
      port:
        default:
          '2772'
paths:
  /applications/{Application}/environments/{Environment}/configurations/
{Configuration}:
    get:
      operationId: getConfiguration
      tags:
        - configuration
      parameters:
        - in: path
          name: Application
          description: The application for the configuration to get. Specify either the
 application name or the application ID.
          required: true
          schema:
            type: string
        - in: path
          name: Environment
          description: The environment for the configuration to get. Specify either the
 environment name or the environment ID.
          required: true
          schema:
            type: string
        - in: path
          name: Configuration
          description: The configuration to get. Specify either the configuration name
 or the configuration ID.
          required: true
          schema:
            type: string
      responses:
        200:
          headers:
            ConfigurationVersion:
              schema:
                type: string
          content:
            application/octet-stream:
              schema:
                type: string
                format: binary
```

```
          description: successful config retrieval
        400:
          description: BadRequestException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        404:
          description: ResourceNotFoundException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        500:
          description: InternalServerException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        502:
          description: BadGatewayException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        504:
          description: GatewayTimeoutException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'

components:
  schemas:
    Error:
      type: string
      description: The response error
```

Recuperación de datos de configuración de instancias de Amazon EC2

Puede realizar la integración AWS AppConfig con las aplicaciones que se ejecutan en sus instancias de Linux de Amazon Elastic Compute Cloud (Amazon EC2) mediante Agent. AWS AppConfig El agente mejora el procesamiento y la administración de las aplicaciones de las siguientes maneras:

- El agente llama AWS AppConfig en su nombre utilizando una función AWS Identity and Access Management (IAM) y gestionando una caché local de datos de configuración. Al extraer los datos de configuración de la memoria caché local, su aplicación necesita menos actualizaciones de código para gestionar los datos de configuración, recupera los datos de configuración en milisegundos y no se ve afectada por problemas de red que puedan interrumpir las llamadas a dichos datos.*
- El agente ofrece una experiencia nativa para recuperar y resolver los indicadores de AWS AppConfig funciones.
- En su estado original, el agente proporciona las prácticas recomendadas para las estrategias de almacenamiento en caché, los intervalos de sondeo y la disponibilidad de los datos de configuración local, al tiempo que rastrea los tokens de configuración necesarios para las siguientes llamadas de servicio.
- Mientras se ejecuta en segundo plano, el agente sondea periódicamente el plano de AWS AppConfig datos en busca de actualizaciones de los datos de configuración. La aplicación puede recuperar los datos conectándose a localhost en el puerto 2772 (un valor de puerto predeterminado personalizable) y llamando a HTTP GET para recuperar los datos.

* El AWS AppConfig agente almacena los datos en caché la primera vez que el servicio recupera los datos de configuración. Por este motivo, la primera llamada para recuperar datos es más lenta que las llamadas posteriores.

Temas

Paso 1 (obligatorio): Crear recursos y configurar los permisos

Para integrarse AWS AppConfig con las aplicaciones que se ejecutan en sus instancias de Amazon EC2, debe crear AWS AppConfig artefactos y datos de configuración, incluidos indicadores de características o datos de configuración de formato libre. Para obtener más información, consulte

Creación de indicadores de características y datos de configuración de formato libre en AWS AppConfig.

Para recuperar los datos de configuración alojados por AWS AppConfig, sus aplicaciones deben configurarse con acceso al plano de AWS AppConfig datos. Para dar acceso a sus aplicaciones, actualice la política de permisos de IAM que está asignada al rol de instancia de Amazon EC2. En concreto, debe añadir las acciones `appconfig:StartConfigurationSession` y `appconfig:GetLatestConfiguration` a la política. A continuación se muestra un ejemplo:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "appconfig:StartConfigurationSession",
                "appconfig:GetLatestConfiguration"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obtener información sobre cómo añadir permisos a la política, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

Paso 2: (obligatorio) Instalar e iniciar el AWS AppConfig agente en las instancias de Amazon EC2

AWS AppConfig El agente está alojado en un bucket de Amazon Simple Storage Service (Amazon S3) administrado por. AWS Siga este procedimiento para descargar la versión más reciente del agente en su instancia de Linux. Si la aplicación está distribuida en varias instancias, debe realizar este procedimiento en cada instancia que aloje la aplicación.

> **Note**
>
> Tenga en cuenta la siguiente información:
>
> - AWS AppConfig El agente está disponible para los sistemas operativos Linux que ejecutan la versión 4.15 o superior del kernel. Los sistemas basados en Debian, como por ejemplo Ubuntu, no son compatibles.

- El agente es compatible con las arquitecturas x86_64 y ARM64.
- Para aplicaciones distribuidas, recomendamos añadir los comandos “install” y “startup” a los datos de usuario de Amazon EC2 de su grupo de escalado automático. Si lo hace, cada instancia ejecuta los comandos automáticamente. Para obtener más información, consulte Ejecutar comandos en la instancia de Linux durante el lanzamiento en la Guía del usuario de Amazon EC2 para instancias de Linux. Además, puede consultar el Tutorial: Recuperar el estado de ciclo de vida de destino a través de los metadatos de instancia en la Guía del usuario de Amazon EC2 Auto Scaling.
- Los procedimientos de este tema describen la manera de realizar acciones, como por ejemplo instalar el agente iniciando sesión en la instancia para ejecutar el comando. Puede ejecutar los comandos desde un equipo cliente local y dirigirse a una o más instancias mediante Run Command, que es una capacidad de AWS Systems Manager. Para obtener más información, consulte Run Command de AWS Systems Manager en la Guía del usuario de AWS Systems Manager .
- AWS AppConfig El agente en las instancias Linux de Amazon EC2 es un `systemd` servicio.

Para instalar e iniciar el AWS AppConfig agente en una instancia

1. Conexión con su instancia de Linux.
2. Abra una terminal y ejecute el siguiente comando con permisos de administrador para las arquitecturas x86_64:

   ```
   sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/x86_64/latest/aws-appconfig-agent.rpm
   ```

   Para arquitecturas ARM64, ejecute el comando siguiente:

   ```
   sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/arm64/latest/aws-appconfig-agent.rpm
   ```

   Si quiere instalar una versión específica del AWS AppConfig agente, sustituya `latest` la URL por un número de versión específico. Este es un ejemplo para x86_64:

```
sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/x86_64/2.0.2/aws-appconfig-agent.rpm
```

3. Para iniciar el agente, ejecute el comando siguiente:

```
sudo systemctl start aws-appconfig-agent
```

4. Ejecute el siguiente comando para verificar que el agente está funcionando:

```
sudo systemctl status aws-appconfig-agent
```

En caso de éxito, este comando devuelve información similar a la siguiente:

```
aws-appconfig-agent.service - aws-appconfig-agent
  ...
  Active: active (running) since Mon 2023-07-26 00:00:00 UTC; 0s ago
  ...
```

> **Note**
>
> Para detener el agente, ejecute el comando siguiente:
>
> ```
> sudo systemctl stop aws-appconfig-agent
> ```

Paso 3: (opcional, pero recomendado) Enviar los archivos de registro a CloudWatch Logs

De forma predeterminada, el AWS AppConfig agente publica los registros en STDERR. Systemd redirige los códigos STDOUT y STDERR de todos los servicios que se ejecutan en la instancia de Linux al diario de systemd. Puede ver y administrar los datos de registro en el registro de systemd si ejecuta AWS AppConfig Agent solo en una o dos instancias. Una solución mejor, una solución que recomendamos encarecidamente para las aplicaciones distribuidas, es escribir los archivos de registro en el disco y, a continuación, utilizar el CloudWatch agente de Amazon para cargar los datos de registro AWS en la nube. Además, puedes configurar el CloudWatch agente para que elimine los

archivos de registro antiguos de la instancia, lo que evitará que la instancia se quede sin espacio en disco.

Para habilitar el registro en el disco, debe configurar la variable de entorno `LOG_PATH`, tal y como se describe en Paso 4: (opcional) Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon EC2.

Para empezar a utilizar el CloudWatch agente, consulte Recopilar métricas y registros de instancias de Amazon EC2 y servidores locales con el CloudWatch agente en la Guía del usuario de Amazon CloudWatch . Puede utilizar Quick Setup, una función de Systems Manager para instalar rápidamente el CloudWatch agente. Para obtener más información, consulte Administración de host con Configuración Rápida en la Guía del usuario de AWS Systems Manager .

> **Warning**
>
> Si decide escribir los archivos de registro en el disco sin utilizar el CloudWatch agente, debe eliminar los archivos de registro antiguos. AWS AppConfig El agente rota automáticamente los archivos de registro cada hora. Si no se eliminan los archivos de registro antiguos, es posible que la instancia se quede sin espacio en disco.

Tras instalar el CloudWatch agente en la instancia, cree un archivo de configuración del CloudWatch agente. El archivo de configuración indica al CloudWatch agente cómo trabajar con los archivos de registro AWS AppConfig del agente. Para obtener más información sobre la creación de un archivo de configuración del CloudWatch agente, consulte Crear el archivo de configuración del CloudWatch agente.

Añada la siguiente `logs` sección al archivo de configuración del CloudWatch agente de la instancia y guarde los cambios:

```
"logs": {
  "logs_collected": {
    "files": {
      "collect_list": [
        {
          "file_path": "/path_you_specified_for_logging",
          "log_group_name": "${YOUR_LOG_GROUP_NAME}/aws-appconfig-agent.log",
          "auto_removal": true
        },
```

```
        ...
      ]
    },
    ...
  },
  ...
}
```

Si el valor `auto_removal` es `true`, el CloudWatch agente elimina automáticamente los archivos de registro del AWS AppConfig agente rotados.

Paso 4: (opcional) Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon EC2

Puede configurar el AWS AppConfig agente para Amazon EC2 mediante variables de entorno. Para establecer las variables de entorno de un servicio de `systemd`, debe crear un archivo de unidad integrado. El siguiente ejemplo muestra cómo crear un archivo unitario integrado para establecer el nivel de registro del AWS AppConfig agente. `DEBUG`

Ejemplo de cómo crear un archivo de unidad integrada para variables de entorno

1. Conexión con su instancia de Linux.
2. Abra una terminal y ejecute el siguiente comando con permisos de administrador. El comando crea un directorio de configuración:

   ```
   sudo mkdir /etc/systemd/system/aws-appconfig-agent.service.d
   ```

3. Ejecute el siguiente comando para crear un archivo de unidad integrada. Sustituya *`file_name`* por el nombre del archivo. La extensión debe ser `.conf`:

   ```
   sudo touch /etc/systemd/system/aws-appconfig-agent.service.d/file_name.conf
   ```

4. Introduzca la información en el archivo de unidad integrada. En el siguiente ejemplo, se añade una sección `Service` que define una variable de entorno. El ejemplo establece AWS AppConfig el nivel de registro del agente de en `DEBUG`.

   ```
   [Service]
   Environment=LOG_LEVEL=DEBUG
   ```

5. Ejecute el siguiente comando para volver a cargar la configuración de systemd:

```
sudo systemctl daemon-reload
```

6. Ejecute el siguiente comando para reiniciar el AWS AppConfig agente:

```
sudo systemctl restart aws-appconfig-agent
```

Puede configurar AWS AppConfig Agent for Amazon EC2 especificando las siguientes variables de entorno en un archivo de unidad desplegable.

Variable de entorno	Detalles	Valor predeterminado

`ACCESS_TOKEN`	Esta variable de entorno define un token que se debe proporcionar al solicitar datos de configuración al servidor HTTP del agente. El valor del token debe estar establecido en el encabezado de autorización de la solicitud HTTP con un tipo de autorización de `Bearer`. A continuación se muestra un ejemplo.
	`GET /applications/my_app/...`
	`Host: localhost:2772`
	`Authorization: Bearer <token value>`
		Ninguna

`BACKUP_DIRECTORY`	Esta variable de entorno permite al AWS AppConfig agente guardar una copia de seguridad de cada configura	Ninguna

Variable de entorno	Detalles	Valor predeterminado
	ción que recupere en el directorio especificado. **Important** Las configuraciones respaldadas en el disco no están cifradas. Si su configuración contiene datos confidenciales, le AWS AppConfig recomendamos que practique el principio de privilegios mínimos con los permisos del sistema de archivos. Para obtener más información, consulte Seguridad en AWS AppConfig.	
`HTTP_PORT`	Esta variable de entorno especifica el puerto en el que se ejecuta el servidor HTTP del agente.	2772

Variable de entorno	Detalles	Valor predeterminado
`LOG_LEVEL`	Esta variable de entorno especifica el nivel de detalle que registra el agente. Cada nivel incluye el nivel actual y todos los niveles superiores. Los nombres de las variables distinguen entre mayúsculas y minúsculas. Del más detallado al menos detallado, los niveles de registro son: `debug`, `info`, `warn`, `error` y `none`. `Debug` incluye información detallada sobre el agente, incluida la información sobre el tiempo.	`info`
`LOG_PATH`	La ubicación del disco en la que se escriben los registros . Si no se especifica, los registros se escriben en stderr.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
`MANIFEST`	Esta variable de entorno configura el AWS AppConfig Agente para que aproveche las funciones adicionales por configuración, como las recuperaciones de varias cuentas y el almacenamiento de la configuración en disco. Puede introducir uno de los siguientes valores: • `"app:env:manifest-config"` • `"file:/fully/qualified/path/to/manifest.json"` Para obtener más información sobre el uso de estas características, consulte Funciones de recuperación adicionales.	true
`MAX_CONNECTIONS`	Esta variable de entorno configura el número máximo de conexiones que el agente utiliza para recuperar las configuraciones de AWS AppConfig.	3

Variable de entorno	Detalles	Valor predeterminado
`POLL_INTERVAL`	Esta variable de entorno controla la frecuencia con la que el agente AWS AppConfig consulta los datos de configuración actualizados. Puede especificar un número de segundos para el intervalo. También puede especificar un número con una unidad de tiempo: s para segundos, m para minutos y h para horas. Si no se especifica una unidad, el agente ejecuta de forma predeterminada los segundos. Por ejemplo, 60, 60s y 1m dan como resultado el mismo intervalo de sondeo.	45 segundos
`PREFETCH_LIST`	Esta variable de entorno especifica los datos de configuración que el agente solicita en AWS AppConfig cuanto se inicia.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
PRELOAD_BACKUPS	Si se establece en`true`, el AWS AppConfig agente carga las copias de seguridad de la configuración que se encuentran BACKUP_DI RECTORY en la memoria y comprueba inmediatamente si existe una versión más reciente del servicio. Si se establece en`false`, el AWS AppConfig agente solo carga el contenido de una copia de seguridad de la configura ción si no puede recuperar los datos de configuración del servicio, por ejemplo, si hay un problema con la red.	true
PROXY_HEADERS	Esta variable de entorno especifica los encabezad os que requiere el proxy al que se hace referencia en la variable de entorno PROXY_URL . El valor es una lista de encabezados separados por comas. Todos los encabezados utilizan el siguiente formulario. `"header: value"`	Ninguna

Variable de entorno	Detalles	Valor predeterminado
PROXY_URL	Esta variable de entorno especifica la URL del proxy que se utilizará para las conexiones entre AWS AppConfig el agente y Servicios de AWS, por ejemplo, HTTPSy se admiten HTTP las URL.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
`REQUEST_TIMEOUT`	Esta variable de entorno controla el tiempo que el agente espera una respuesta. AWS AppConfig Si el servicio no responde, se produce un error en la solicitud. Si la solicitud se emplea para la recuperación inicial de datos, el agente devuelve un error a su solicitud. Si el tiempo de espera se agota cuando se está comprobando en segundo plano si hay datos actualiza dos, el agente registra el error y lo vuelve a intentar tras un breve retraso. Puede especificar el número de milisegundos del tiempo de espera. También puede especificar un número con una unidad de tiempo: ms para milisegundos y s para segundos. Si no se especifica una unidad, el valor predeterm inado del agente es milisegun dos. Por ejemplo, 5000, 5000ms y 5s dan como resultado el mismo valor de tiempo de espera de la solicitud.	3000 milisegundos

Variable de entorno	Detalles	Valor predeterminado
ROLE_ARN	Esta variable de entorno especifica el nombre de recurso de Amazon (ARN) de un rol de IAM. AWS AppConfig El agente asume esta función para recuperar los datos de configuración.	Ninguna
ROLE_EXTERNAL_ID	Esta variable de entorno especifica el ID externo que se utilizará con el ARN del rol asumido.	Ninguna
ROLE_SESSION_NAME	Esta variable de entorno especifica el nombre de la sesión que se va a asociar a las credenciales del rol de IAM asumido.	Ninguna
SERVICE_REGION	Esta variable de entorno especifica una alternativa Región de AWS que el AWS AppConfig agente utiliza para llamar al AWS AppConfig servicio. Si no se define, el agente intenta determinar la región actual. Si no puede, el agente no podrá iniciarse.	Ninguna
WAIT_ON_MANIFEST	Esta variable de entorno configura el AWS AppConfig agente para que espere hasta que se procese el manifiesto antes de completar el inicio.	true

Paso 5 (obligatorio): Recuperar datos de configuración

Puede recuperar los datos de configuración del AWS AppConfig agente mediante una llamada HTTP localhost. Los siguientes ejemplos utilizan `curl` con un cliente HTTP. Puede llamar al agente mediante cualquier cliente HTTP disponible compatible con el idioma de su aplicación o con las bibliotecas disponibles, incluido un AWS SDK.

Para recuperar el contenido completo de cualquier configuración implementada

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name"
```

Para recuperar una única marca y sus atributos desde una configuración AWS AppConfig de tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?flag=flag_name"
```

Para acceder a varias marcas y sus atributos desde una configuración de AWS AppConfig de tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?
flag=flag_name_one&flag=flag_name_two"
```

Paso 6 (opcional, pero recomendado): Automatizar AWS AppConfig las actualizaciones del agente

AWS AppConfig El agente se actualiza periódicamente. Para asegurarse de que está ejecutando la versión más reciente del agente de AWS AppConfig en sus instancias, le recomendamos que añada los siguientes comandos a sus datos de usuario de Amazon EC2. Puede añadir los comandos a los datos de usuario de la instancia o del grupo de escalado automático de EC2. El script instala e inicia la última versión del agente cada vez que se inicia o se reinicia una instancia.

```
#!/bin/bash
# install the latest version of the agent
yum install -y https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/
linux/x86_64/latest/aws-appconfig-agent.rpm
# optional: configure the agent
```

```
mkdir /etc/systemd/system/aws-appconfig-agent.service.d
echo "${MY_AGENT_CONFIG}" > /etc/systemd/system/aws-appconfig-agent.service.d/
overrides.conf
systemctl daemon-reload
# start the agent
systemctl start aws-appconfig-agent
```

Recuperación de datos de configuración de Amazon ECS y Amazon EKS

Puede realizar la integración AWS AppConfig con Amazon Elastic Container Service (Amazon ECS) y Amazon Elastic Kubernetes Service (Amazon EKS) mediante Agent. AWS AppConfig El agente funciona como un contenedor asociado que se ejecuta junto con las aplicaciones de contenedores de Amazon ECS y Amazon EKS. El agente mejora el procesamiento y la administración de las aplicaciones en contenedores de las siguientes maneras:

- El agente llama AWS AppConfig en su nombre utilizando una función AWS Identity and Access Management (IAM) y gestionando una caché local de datos de configuración. Al extraer los datos de configuración de la memoria caché local, su aplicación necesita menos actualizaciones de código para gestionar los datos de configuración, recupera los datos de configuración en milisegundos y no se ve afectada por problemas de red que puedan interrumpir las llamadas a dichos datos.*
- El agente ofrece una experiencia nativa para recuperar y resolver los indicadores de AWS AppConfig funciones.
- En su estado original, el agente proporciona las prácticas recomendadas para las estrategias de almacenamiento en caché, los intervalos de sondeo y la disponibilidad de los datos de configuración local, al tiempo que rastrea los tokens de configuración necesarios para las siguientes llamadas de servicio.
- Mientras se ejecuta en segundo plano, el agente sondea periódicamente el plano de AWS AppConfig datos en busca de actualizaciones de los datos de configuración. La aplicación en contenedores puede recuperar los datos conectándose a localhost en el puerto 2772 (un valor de puerto predeterminado personalizable) y llamando a HTTP GET para recuperar los datos.
- AWS AppConfig El agente actualiza los datos de configuración de sus contenedores sin tener que reiniciarlos ni reciclarlos.

* El AWS AppConfig agente almacena los datos en caché la primera vez que el servicio recupera los datos de configuración. Por este motivo, la primera llamada para recuperar datos es más lenta que las llamadas posteriores.

Temas

Antes de empezar

Para integrarlo AWS AppConfig con sus aplicaciones contenedoras, debe crear AWS AppConfig artefactos y datos de configuración, incluidos indicadores de características o datos de configuración de formato libre. Para obtener más información, consulte Creación de indicadores de características y datos de configuración de formato libre en AWS AppConfig.

Para recuperar los datos de configuración alojados por AWS AppConfig, las aplicaciones contenedoras deben configurarse con acceso al plano de AWS AppConfig datos. Para dar acceso a sus aplicaciones, actualice la política de permisos de IAM que utiliza su rol de IAM de servicio de contenedores. En concreto, debe añadir las acciones `appconfig:StartConfigurationSession` y `appconfig:GetLatestConfiguration` a la política. Los roles de IAM del servicio de contenedores incluyen los siguientes:

- Rol de tarea de Amazon ECS
- Rol de nodo de Amazon EKS
- La función de ejecución del AWS Fargate (Fargate) pod (si sus contenedores Amazon EKS utilizan Fargate para el procesamiento informático)

Para obtener información sobre cómo añadir permisos a la política, consulte Adición y eliminación de permisos de identidad de IAM en la Guía del usuario de IAM.

Inicio del agente de AWS AppConfig para la integración de Amazon ECS

El contenedor AWS AppConfig Agent sidecar está disponible automáticamente en su entorno de Amazon ECS. Para utilizar el contenedor AWS AppConfig Agent sidecar, debe iniciarlo.

Para iniciar Amazon ECS (consola)

1. Abra la consola en https://console.aws.amazon.com/ecs/v2.
2. En el panel de navegación, elija Task Definitions (Definiciones de tareas).
3. Elija la definición de tarea para su aplicación y, a continuación, seleccione la revisión más reciente.
4. Elija Crear nueva revisión y Crear nueva revisión.
5. Elija Agregar más contenedores.
6. En Nombre, introduzca un nombre exclusivo para el contenedor del AWS AppConfig agente.
7. Para la URI de imagen, introduzca: **`public.ecr.aws/aws-appconfig/aws-appconfig-agent:2.x`**
8. En Contenedor esencial, elija Sí.
9. En la sección Asignaciones de puertos, elija Agregar asignación de puertos.
10. En Puerto del contenedor, ingrese **`2772`**.

> **Note**
>
> AWS AppConfig El agente se ejecuta en el puerto 2772, de forma predeterminada. O puede especificar un puerto diferente.

11. Seleccione Crear. Amazon ECS crea una nueva revisión del contenedor y muestra los detalles.
12. En el panel de navegación, elija Clústers y, a continuación, elija el nombre del clúster de la aplicación en la lista.
13. En la pestaña Servicios, seleccione el servicio para su aplicación.
14. Elija Actualizar.
15. En Configuración de implementación, en Revisión, elija la revisión más reciente.
16. Elija Actualizar. Amazon ECS implementa la definición de tareas más reciente.
17. Una vez finalizada la implementación, puede comprobar que el AWS AppConfig agente se está ejecutando en la pestaña Configuración y tareas. En la pestaña Tareas, elija la tarea que se está ejecutando.
18. En la sección Contenedores, compruebe que el contenedor del AWS AppConfig agente esté en la lista.

19. Para comprobar que el AWS AppConfig agente se inició, seleccione la pestaña Registros. Busque una declaración como la siguiente para el contenedor del AWS AppConfig agente: `[appconfig agent] 1970/01/01 00:00:00 INFO serving on localhost:2772`

> **Note**
>
> Puede ajustar el comportamiento predeterminado del AWS AppConfig agente introduciendo o cambiando variables de entorno. Para obtener más información sobre cómo configurar las variable de entorno, consulte Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon ECS y Amazon EKS. Para obtener información sobre cómo cambiar las variables de entorno en Amazon ECS, consulte Pasar variables de entorno a un contenedor en la Guía para desarrolladores de Amazon Elastic Container Service.

Inicio del agente de AWS AppConfig para la integración de Amazon EKS

El contenedor AWS AppConfig Agent sidecar está disponible automáticamente en su entorno Amazon EKS. Para utilizar el contenedor AWS AppConfig Agent sidecar, debe encenderlo. El siguiente procedimiento describe cómo utilizar la herramienta de línea de comandos `kubectl` de Amazon EKS para realizar cambios en el archivo `kubeconfig` de la aplicación de contenedor. Para obtener más información sobre cómo crear o editar un archivo `kubeconfig`, consulte Creación o actualización de un archivo kubeconfig para un clúster de Amazon EKS en la Guía del usuario de Amazon EKS.

Para iniciar AWS AppConfig Agent (herramienta de línea de comandos de kubectl)

1. Abra el archivo `kubeconfig` y compruebe que la aplicación Amazon EKS se está ejecutando como una implementación de un solo contenedor. El contenido del archivo es similar al siguiente.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
  namespace: my-namespace
  labels:
    app: my-application-label
spec:
  replicas: 1
```

```
  selector:
    matchLabels:
      app: my-application-label
  template:
    metadata:
      labels:
        app: my-application-label
    spec:
      containers:
      - name: my-app
        image: my-repo/my-image
        imagePullPolicy: IfNotPresent
```

2. Añada los detalles de la definición del contenedor del AWS AppConfig agente a su archivo de despliegue de YAML.

```
- name: appconfig-agent
        image: public.ecr.aws/aws-appconfig/aws-appconfig-agent:2.x
        ports:
        - name: http
          containerPort: 2772
          protocol: TCP
        env:
        - name: SERVICE_REGION
          value: region
        imagePullPolicy: IfNotPresent
```

> **Note**
>
> Observe la siguiente información.
>
> - AWS AppConfig El agente se ejecuta en el puerto 2772, de forma predeterminada. O puede especificar un puerto diferente.
> - Puede ajustar el comportamiento predeterminado del AWS AppConfig agente introduciendo variables de entorno. Para obtener más información, consulte Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon ECS y Amazon EKS.
> - Para *SERVICE_REGION*, especifique el Región de AWS código (por ejemplo`us-west-1`) en el que el AWS AppConfig agente recupera los datos de configuración.

3. Ejecute el siguiente comando en la herramienta `kubectl` para aplicar los cambios al clúster.

```
kubectl apply -f my-deployment.yml
```

4. Una vez finalizada la implementación, compruebe que el AWS AppConfig agente se esté ejecutando. Utilice el comando siguiente para ver el archivo de registro del pod de la aplicación.

```
kubectl logs -n my-namespace -c appconfig-agent my-pod
```

Busque una declaración como la siguiente para el contenedor del AWS AppConfig agente: `[appconfig agent] 1970/01/01 00:00:00 INFO serving on localhost:2772`

> **Note**
>
> Puede ajustar el comportamiento predeterminado del AWS AppConfig agente introduciendo o cambiando variables de entorno. Para obtener más información sobre cómo configurar las variable de entorno, consulte Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon ECS y Amazon EKS.

Uso de variables de entorno para configurar el AWS AppConfig agente para Amazon ECS y Amazon EKS

Puede configurar el AWS AppConfig agente cambiando las siguientes variables de entorno para su contenedor de agentes.

Variable de entorno	Detalles	Valor predeterminado
`ACCESS_TOKEN`	Esta variable de entorno define un token que se debe proporcionar al solicitar datos de configuración al servidor HTTP del agente. El valor del token debe estar establecido en el encabezado de autorización de la solicitud HTTP con un tipo de autorización de	Ninguna

Variable de entorno	Detalles	Valor predeterminado

	`Bearer`. A continuación se muestra un ejemplo.
	
	`GET /applications/my_app/...`
	`Host: localhost:2772`
	
	`Authorization: Bearer <token value>`	

Variable de entorno	Detalles	Valor predeterminado
`BACKUP_DIRECTORY`	Esta variable de entorno permite al AWS AppConfig Agente guardar una copia de seguridad de cada configura ción que recupera en el directorio especificado. **Important** Las configuraciones respaldadas en el disco no están cifradas. Si su configuración contiene datos confidenciales, le AWS AppConfig recomendamos que practique el principio de privilegios mínimos con los permisos del sistema de archivos. Para obtener más información, consulte Seguridad en AWS AppConfig.	Ninguna
`HTTP_PORT`	Esta variable de entorno especifica el puerto en el que se ejecuta el servidor HTTP del agente.	2772

Variable de entorno	Detalles	Valor predeterminado
`LOG_LEVEL`	Esta variable de entorno especifica el nivel de detalle que registra el agente. Cada nivel incluye el nivel actual y todos los niveles superiores. Los nombres de las variables distinguen entre mayúsculas y minúsculas. Del más detallado al menos detallado, los niveles de registro son: `debug`, `info`, `warn`, `error` y `none`. `Debug` incluye información detallada sobre el agente, incluida la información sobre el tiempo.	`info`

Variable de entorno	Detalles	Valor predeterminado
`MANIFEST`	Esta variable de entorno configura el AWS AppConfig Agente para que aproveche las funciones adicionales por configuración, como las recuperaciones de varias cuentas y el almacenamiento de la configuración en disco. Puede introducir uno de los siguientes valores: • `"app:env:manifest-config"` • `"file:/fully/qualified/path/to/manifest.json"` Para obtener más información sobre el uso de estas características, consulte Funciones de recuperación adicionales.	true
`MAX_CONNECTIONS`	Esta variable de entorno configura el número máximo de conexiones que el agente utiliza para recuperar las configuraciones de AWS AppConfig.	3

Variable de entorno	Detalles	Valor predeterminado
`POLL_INTERVAL`	Esta variable de entorno controla la frecuencia con la que el agente AWS AppConfig consulta los datos de configuración actualiza dos. Puede especificar un número de segundos para el intervalo. También puede especificar un número con una unidad de tiempo: s para segundos, m para minutos y h para horas. Si no se especifica una unidad, el agente ejecuta de forma predeterminada los segundos. Por ejemplo, 60, 60s y 1m dan como resultado el mismo intervalo de sondeo.	45 segundos
`PREFETCH_LIST`	Esta variable de entorno especifica los datos de configuración que el agente solicita en AWS AppConfig cuanto se inicia.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
PRELOAD_BACKUPS	Si se establece en`true`, el AWS AppConfig agente carga las copias de seguridad de la configuración que se encuentran BACKUP_DI RECTORY en la memoria y comprueba inmediatamente si existe una versión más reciente del servicio. Si se establece en`false`, el AWS AppConfig agente solo carga el contenido de una copia de seguridad de la configura ción si no puede recuperar los datos de configuración del servicio, por ejemplo, si hay un problema con la red.	true
PROXY_HEADERS	Esta variable de entorno especifica los encabezad os que requiere el proxy al que se hace referencia en la variable de entorno PROXY_URL . El valor es una lista de encabezados separados por comas. Todos los encabezados utilizan el siguiente formulario. `"header: value"`	Ninguna

Variable de entorno	Detalles	Valor predeterminado
PROXY_URL	Esta variable de entorno especifica la URL del proxy que se utilizará para las conexiones entre AWS AppConfig el agente y Servicios de AWS, por ejemplo, HTTPSy se admiten HTTP las URL.	Ninguna

Variable de entorno	Detalles	Valor predeterminado
`REQUEST_TIMEOUT`	Esta variable de entorno controla el tiempo que el agente espera una respuesta. AWS AppConfig Si el servicio no responde, se produce un error en la solicitud. Si la solicitud se emplea para la recuperación inicial de datos, el agente devuelve un error a su solicitud. Si el tiempo de espera se agota cuando se está comprobando en segundo plano si hay datos actualiza dos, el agente registra el error y lo vuelve a intentar tras un breve retraso. Puede especificar el número de milisegundos del tiempo de espera. También puede especificar un número con una unidad de tiempo: ms para milisegundos y s para segundos. Si no se especifica una unidad, el valor predeterm inado del agente es milisegun dos. Por ejemplo, 5000, 5000ms y 5s dan como resultado el mismo valor de tiempo de espera de la solicitud.	3000 milisegundos

Variable de entorno	Detalles	Valor predeterminado
ROLE_ARN	Esta variable de entorno especifica el nombre de recurso de Amazon (ARN) de un rol de IAM. AWS AppConfig El agente asume esta función para recuperar los datos de configuración.	Ninguna
ROLE_EXTERNAL_ID	Esta variable de entorno especifica el ID externo que se utilizará con el ARN del rol asumido.	Ninguna
ROLE_SESSION_NAME	Esta variable de entorno especifica el nombre de la sesión que se va a asociar a las credenciales del rol de IAM asumido.	Ninguna
SERVICE_REGION	Esta variable de entorno especifica una alternativa Región de AWS que el AWS AppConfig agente utiliza para llamar al AWS AppConfig servicio. Si no se define, el agente intenta determinar la región actual. Si no puede, el agente no podrá iniciarse.	Ninguna
WAIT_ON_MANIFEST	Esta variable de entorno configura el AWS AppConfig agente para que espere hasta que se procese el manifiesto antes de completar el inicio.	true

Recuperación de los datos de configuración

Puede recuperar los datos de configuración del AWS AppConfig agente mediante una llamada HTTP localhost. Los siguientes ejemplos utilizan `curl` con un cliente HTTP. Puede llamar al agente utilizando cualquier cliente HTTP disponible compatible con el lenguaje de su aplicación o con las bibliotecas disponibles.

> **Note**
>
> Para recuperar los datos de configuración si la aplicación utiliza una barra diagonal, por ejemplo, «test-backend/test-service», necesitará utilizar la codificación URL.

Para recuperar el contenido completo de cualquier configuración implementada

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name"
```

Para recuperar una única marca y sus atributos desde una configuración AWS AppConfig de tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?flag=flag_name"
```

Para acceder a varias marcas y sus atributos desde una configuración de AWS AppConfig de tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?
flag=flag_name_one&flag=flag_name_two"
```

Funciones de recuperación adicionales

AWS AppConfig El agente ofrece las siguientes funciones adicionales para ayudarle a recuperar las configuraciones de sus aplicaciones.

- Recuperación de varias cuentas: Utilice el AWS AppConfig agente de una cuenta principal o de recuperación Cuenta de AWS para recuperar los datos de configuración de varias cuentas de proveedores.
- Escriba la copia de la configuración en el disco: Utilice el AWS AppConfig agente para escribir los datos de configuración en el disco. Esta función permite a los clientes con aplicaciones que leen los datos de configuración del disco integrarse en ellas AWS AppConfig.

Acerca de los manifiestos de agentes

Para activar estas funciones del AWS AppConfig agente, debe crear un manifiesto. Un manifiesto es un conjunto de datos de configuración que se proporcionan para controlar las acciones que el agente puede realizar. Un manifiesto se escribe en JSON. Contiene un conjunto de claves de nivel superior que corresponden a las diferentes configuraciones que has utilizado AWS AppConfig para la implementación.

Un manifiesto puede incluir varias configuraciones. Además, cada configuración del manifiesto puede identificar una o más funciones del agente para utilizarlas en la configuración especificada. El contenido del manifiesto utiliza el siguiente formato:

```
{
    "application_name:environment_name:configuration_name": {
        "agent_feature_to_enable_1": {
            "feature-setting-key": "feature-setting-value"
        },
        "agent_feature_to_enable_2": {
            "feature-setting-key": "feature-setting-value"
        }
    }
}
```

A continuación, se muestra un ejemplo de JSON para un manifiesto con dos configuraciones. La primera configuración (*MyApp*) no utiliza ninguna función AWS AppConfig del agente. La segunda configuración (*My2ndApp*) utiliza las funciones de escritura, copia en disco y recuperación multicuenta:

```
{
        "MyApp:Test:MyAllowListConfiguration": {},

        "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
```

```
            "credentials": {
                "roleArn": "arn:us-west-1:iam::123456789012:role/MyTestRole",
                "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
                "roleSessionName": "AwsAppConfigAgent",
                "credentialsDuration": "2h"
            },
            "writeTo": {
                "path": "/tmp/aws-appconfig/my-2nd-app/beta/my-enable-payments-feature-
flag-configuration.json"
            }
        }
    }
```

¿Cómo proporcionar un manifiesto de agente?

Puede almacenar el manifiesto como un archivo en un lugar donde el AWS AppConfig agente pueda leerlo. O bien, puede almacenar el manifiesto como una AWS AppConfig configuración y dirigir al agente hacia él. Para proporcionar un manifiesto de agente, debe establecer una variable de `MANIFEST` entorno con uno de los siguientes valores:

Ubicación del manifiesto	Valor de la variable de entorno	Caso de uso
Archivos	archivo: /path/to/agent-manifest.json	Usa este método si tu manifiesto no va a cambiar con frecuencia.
AWS AppConfig configuración	*nombre de la aplicación: nombre del entorno: nombre de la configuración*	Utilice este método para las actualizaciones dinámicas. Puede actualizar e implementar un manifiesto almacenado AWS AppConfig como configuración del mismo modo que almacena otras AWS AppConfig configuraciones.
Variable de entorno	Contenido del manifiesto (JSON)	Usa este método si tu manifiesto no cambia con frecuencia. Este método resulta útil en entornos de contenedores donde es más

Ubicación del manifiesto	Valor de la variable de entorno	Caso de uso
		fácil establecer una variable de entorno que exponer un archivo.

Para obtener más información sobre la configuración de variables para AWS AppConfig Agent, consulte el tema correspondiente a su caso de uso:

- Configuración de la extensión de Lambda del agente de AWS AppConfig
- Uso de AWS AppConfig Agent con Amazon EC2
- Uso de AWS AppConfig Agent con Amazon ECS y Amazon EKS

Recuperación de varias cuentas

Puede configurar el AWS AppConfig agente para que recupere las configuraciones de varias de ellas Cuentas de AWS introduciendo las anulaciones de credenciales en el manifiesto del agente. AWS AppConfig Las anulaciones de credenciales incluyen el nombre de recurso de Amazon (ARN) de una función AWS Identity and Access Management (IAM), un ID de función, un nombre de sesión y la duración del tiempo que el agente puede asumir la función.

Introduce estos detalles en la sección de «credenciales» del manifiesto. La sección «credenciales» usa el siguiente formato:

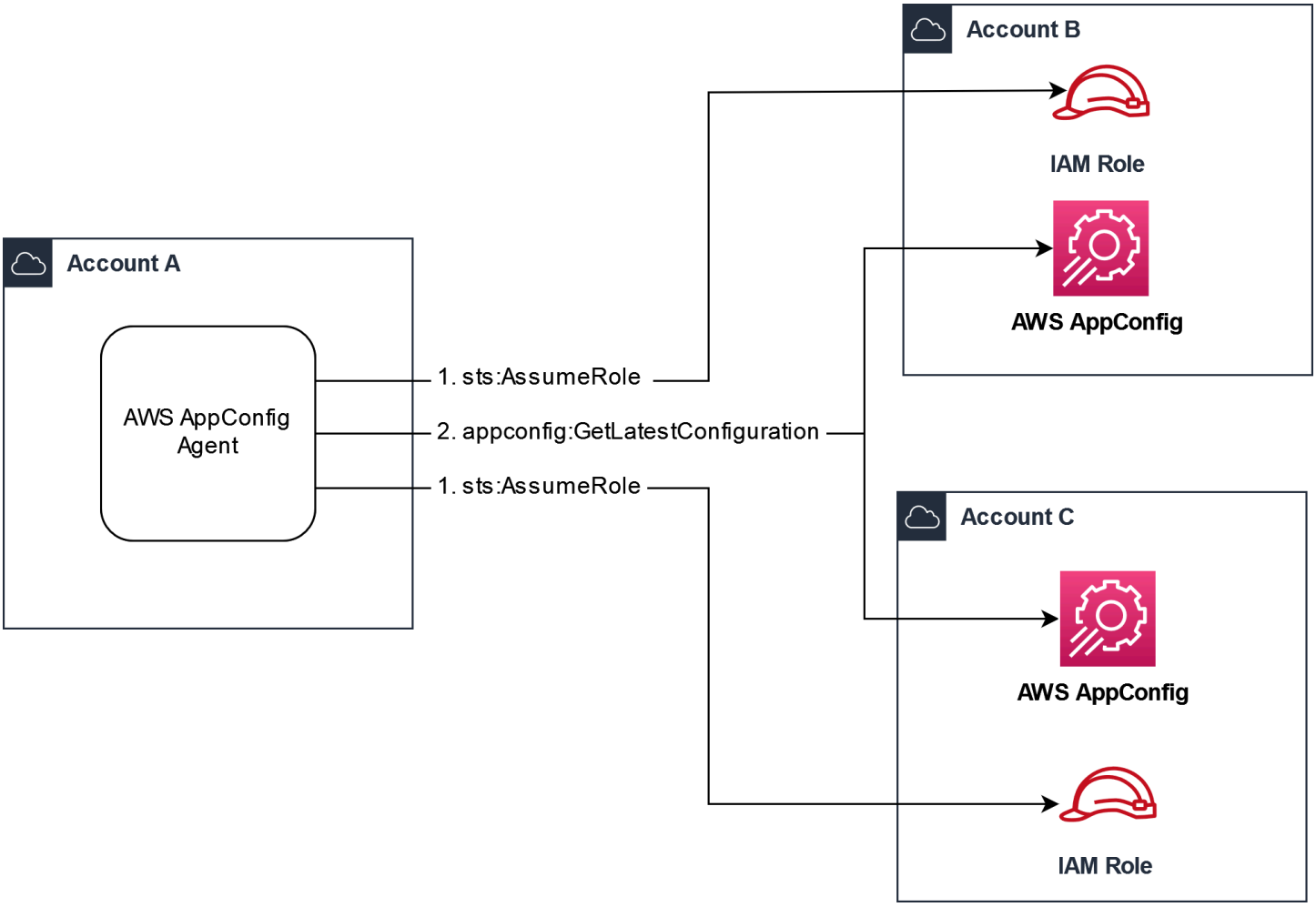
```
{
    "application_name:environment_name:configuration_name": {
        "credentials": {
            "roleArn": "arn:partition:iam::account_ID:role/roleName",
            "roleExternalId": "string",
            "roleSessionName": "string",
            "credentialsDuration": "time_in_hours"
        }
    }
}
```

A continuación se muestra un ejemplo:

```
{
    "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
```

```
        "credentials": {
            "roleArn": "arn:us-west-1:iam::123456789012:role/MyTestRole",
            "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
            "roleSessionName": "AWSAppConfigAgent",
            "credentialsDuration": "2h"
        }
    }
}
```

Antes de recuperar una configuración, el agente lee los detalles de las credenciales de la configuración en el manifiesto y, a continuación, asume la función de IAM especificada para esa configuración. Puedes especificar un conjunto diferente de anulaciones de credenciales para distintas configuraciones en un único manifiesto. En el siguiente diagrama, se muestra cómo el AWS AppConfig agente, mientras se ejecuta en la cuenta A (la cuenta de recuperación), asume funciones distintas especificadas para las cuentas B y C (las cuentas del proveedor) y, a continuación, llama a la operación de la GetLatestConfigurationAPI para recuperar los datos de configuración de la AWS AppConfig ejecución en esas cuentas:



Configure los permisos para recuperar los datos de configuración de las cuentas de los proveedores

AWS AppConfig El agente que se ejecuta en la cuenta de recuperación necesita permiso para recuperar los datos de configuración de las cuentas del proveedor. Para conceder el permiso al agente, debe crear un rol AWS Identity and Access Management (IAM) en cada una de las cuentas de los proveedores. AWS AppConfig El agente de la cuenta de recuperación asume esta función para obtener datos de las cuentas de los proveedores. Complete los procedimientos de esta sección para crear una política de permisos de IAM, una función de IAM y añadir sustituciones por parte de los agentes al manifiesto.

Antes de empezar

Recopile la siguiente información antes de crear una política de permisos y un rol en IAM.

- Los identificadores de cada uno Cuenta de AWS. La cuenta de recuperación es la cuenta que llamará a otras cuentas para obtener datos de configuración. Las cuentas de proveedor son las cuentas que venderán los datos de configuración a la cuenta de recuperación.
- El nombre de la función de IAM utilizada AWS AppConfig en la cuenta de recuperación. Esta es una lista de las funciones que utilizan AWS AppConfig, de forma predeterminada:
  - En el caso de Amazon Elastic Compute Cloud (Amazon EC2) AWS AppConfig , utiliza el rol de instancia.
  - Para AWS Lambda, AWS AppConfig utiliza la función de ejecución Lambda.
  - Para Amazon Elastic Container Service (Amazon ECS) y Amazon Elastic Kubernetes Service (Amazon AWS AppConfig EKS), utiliza la función de contenedor.

  Si ha configurado el AWS AppConfig agente para que utilice un rol de IAM diferente especificando la variable de `ROLE_ARN` entorno, anote ese nombre.

Cree la política de permisos

Utilice el siguiente procedimiento para crear una política de permisos mediante la consola de IAM. Complete el procedimiento de cada una Cuenta de AWS de ellas para vender los datos de configuración de la cuenta de recuperación.

Para crear una política de IAM

1. Inicie sesión AWS Management Console en una cuenta de proveedor.
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. Elige la opción JSON.
5. En el editor de políticas, sustituya el JSON predeterminado por la siguiente declaración de política. Actualice cada *marcador de posición de recurso de ejemplo* con los detalles de la cuenta del proveedor.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appconfig:StartConfigurationSession",
            "appconfig:GetLatestConfiguration"
        ],
        "Resource":
 "arn:partition:appconfig:region:vendor_account_ID:application/
vendor_application_ID/environment/vendor_environment_ID/
configuration/vendor_configuration_ID"
    }
   ]
}
```

A continuación se muestra un ejemplo:

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appconfig:StartConfigurationSession",
            "appconfig:GetLatestConfiguration"
        ],
        "Resource": "arn:aws:appconfig:us-east-2:111122223333:application/abc123/
environment/def456/configuration/hij789"
    }
   ]
}
```

6. Elija Siguiente.
7. En el campo Nombre de la política, introduzca un nombre.

8. (Opcional) En Agregar etiquetas, agregue uno o más pares de valores de etiqueta y clave para organizar, rastrear o controlar el acceso a esta política.
9. Elija Crear política. El sistema le devuelve a la página Policies (Políticas).
10. Repita este procedimiento en cada uno de los casos en Cuenta de AWS los que se vendan los datos de configuración de la cuenta de recuperación.

Creación del rol de IAM

Utilice el siguiente procedimiento para crear un rol de IAM mediante la consola de IAM. Complete el procedimiento de cada uno de ellos Cuenta de AWS que venderá los datos de configuración de la cuenta de recuperación.

Para crear un rol de IAM

1. Inicie sesión AWS Management Console en una cuenta de proveedor.
2. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
3. En el panel de navegación, selecciona Roles y, a continuación, selecciona Crear política.
4. En Tipo de entidad de confianza, elija Cuenta de AWS.
5. En la Cuenta de AWSsección, elija Otro Cuenta de AWS.
6. En el campo ID de cuenta, introduce el ID de la cuenta de recuperación.
7. (Opcional) Como práctica recomendada de seguridad para este supuesto rol, selecciona Requerir un identificador externo e introduce una cadena.
8. Elija Siguiente.
9. En la página Añadir permisos, utilice el campo de búsqueda para localizar la política que creó en el procedimiento anterior. Seleccione la casilla de verificación situada junto a su nombre.
10. Seleccione Next (Siguiente).
11. En Role name (Nombre de rol), escriba un nombre.
12. (Opcional) En Description (Descripción), introduzca una descripción.
13. Para el paso 1: Seleccione entidades de confianza, elija Editar. Sustituya la política de confianza de JSON predeterminada por la siguiente política. Actualiza cada *marcador de posición de recurso de ejemplo* con la información de tu cuenta de recuperación.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::retrieval_account_ID:role/appconfig_role_in_retrieval_account"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

14. (Opcional) En Tags (Etiquetas), agregue uno o varios pares de valor etiqueta-clave para organizar, realizar un seguimiento o controlar el acceso a este rol.
15. Elija Create role. El sistema le devuelve a la página Roles.
16. Busca el rol que acabas de crear. Elíjalo. En la sección ARN, copia el ARN. Especificará esta información en el siguiente procedimiento.

Agregue anulaciones de credenciales al manifiesto

Tras crear el rol de IAM en su cuenta de proveedor, actualice el manifiesto en la cuenta de recuperación. En concreto, añada el bloque de credenciales y el ARN del rol de IAM para recuperar los datos de configuración de la cuenta del proveedor. Este es el formato JSON:

```
{
    "vendor_application_name:vendor_environment_name:vendor_configuration_name": {
        "credentials": {
            "roleArn":
 "arn:partition:iam::vendor_account_ID:role/name_of_role_created_in_vendor_account",
            "roleExternalId": "string",
            "roleSessionName": "string",
            "credentialsDuration": "time_in_hours"
        }
    }
}
```

A continuación se muestra un ejemplo:

```
{
    "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
        "credentials": {
```

```
            "roleArn": "arn:us-west-1:iam::123456789012:role/MyTestRole",
            "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
            "roleSessionName": "AwsAppConfigAgent",
            "credentialsDuration": "2h"
        }
    }
}
```

Valide que la recuperación de varias cuentas esté funcionando

Puede comprobar que ese agente puede recuperar los datos de configuración de varias cuentas consultando los registros del AWS AppConfig agente. El registro de `INFO` nivel de los datos iniciales recuperados de 'YourApplicationName:YourEnvironmentName:YourConfigurationName' es el mejor indicador de que las recuperaciones se han realizado correctamente. Si las recuperaciones fallan, debería ver un registro de `ERROR` nivel que indique el motivo del error. A continuación, se muestra un ejemplo de una recuperación exitosa de una cuenta de proveedor:

```
[appconfig agent] 2023/11/13 11:33:27 INFO AppConfig Agent 2.0.x
[appconfig agent] 2023/11/13 11:33:28 INFO serving on localhost:2772
[appconfig agent] 2023/11/13 11:33:28 INFO retrieved initial data for
 'MyTestApplication:MyTestEnvironment:MyDenyListConfiguration' in XX.Xms
```

Escriba la copia de la configuración en el disco

Puede configurar el AWS AppConfig agente para que almacene automáticamente una copia de una configuración en el disco en texto plano. Esta función permite a los clientes con aplicaciones que leen los datos de configuración del disco integrarse con ellas AWS AppConfig.

Esta función no está diseñada para usarse como función de respaldo de la configuración. AWS AppConfig El agente no lee los archivos de configuración copiados en el disco. Si desea hacer una copia de seguridad de las configuraciones en disco, consulte las variables de `PRELOAD_BACKUP` entorno de Uso del AWS AppConfig agente con Amazon EC2 o Uso del AWS AppConfig agente con Amazon ECS `BACKUP_DIRECTORY` y Amazon EKS.

> **Warning**
>
> Tenga en cuenta la siguiente información importante acerca de esta función:
>
> - Las configuraciones guardadas en el disco se almacenan en texto plano y son legibles por humanos. No habilite esta función para configuraciones que incluyan datos confidenciales.

- Esta función graba en el disco local. Utilice el principio de privilegios mínimos para los permisos del sistema de archivos. Para obtener más información, consulte Implementación del acceso a los privilegios mínimos.

Para habilitar la configuración de escritura, copie en el disco

1. Edite el manifiesto.
2. Elija la configuración que desee AWS AppConfig escribir en el disco y añada un `writeTo` elemento. A continuación se muestra un ejemplo:

```
{
    "application_name:environment_name:configuration_name": {
        "writeTo": {
            "path": "path_to_configuration_file"
        }
    }
}
```

A continuación se muestra un ejemplo:

```
{
    "MyTestApp:MyTestEnvironment:MyNewConfiguration": {
        "writeTo": {
            "path": "/tmp/aws-appconfig/mobile-app/beta/enable-mobile-payments"
        }
    }
}
```

3. Guarde los cambios. El archivo configuration.json se actualizará cada vez que se implementen nuevos datos de configuración.

Valide que la copia de la configuración de escritura en el disco funcione

Para comprobar que se están grabando copias de una configuración en el disco, consulte los registros del AWS AppConfig agente. La entrada de `INFO` registro con la frase «INFO escribió la configuración '*application: environment*: *configuration*' en *file_path*» indica que el AWS AppConfig agente escribe copias de la configuración en el disco.

A continuación se muestra un ejemplo:

```
[appconfig agent] 2023/11/13 11:33:27 INFO AppConfig Agent 2.0.x
[appconfig agent] 2023/11/13 11:33:28 INFO serving on localhost:2772
[appconfig agent] 2023/11/13 11:33:28 INFO retrieved initial data for
 'MobileApp:Beta:EnableMobilePayments' in XX.Xms
[appconfig agent] 2023/11/13 17:05:49 INFO wrote configuration
 'MobileApp:Beta:EnableMobilePayments' to /tmp/configs/your-app/your-env/your-
config.json
```

AWS AppConfig Agente: desarrollo local

AWS AppConfig El agente apoya un modo de desarrollo local. Si habilita el modo de desarrollo local, el agente lee los datos de configuración de un directorio específico del disco. No recupera los datos de configuración de AWS AppConfig. Puede simular las implementaciones de configuración actualizando los archivos en el directorio especificado. Recomendamos el modo de desarrollo local para los siguientes casos de uso:

- Pruebe diferentes versiones de configuración antes de implementarlas utilizando AWS AppConfig.
- Pruebe diferentes opciones de configuración para una nueva función antes de realizar cambios en su repositorio de código.
- Pruebe diferentes escenarios de configuración para comprobar que funcionan según lo esperado.

> **Warning**
>
> No utilice el modo de desarrollo local en los entornos de producción. Este modo no admite funciones de AWS AppConfig seguridad importantes, como la validación de la implementación y las reversiones automatizadas.

Utilice el siguiente procedimiento para configurar el AWS AppConfig agente para el modo de desarrollo local.

Para configurar el AWS AppConfig agente para el modo de desarrollo local

1. Instale el agente mediante el método descrito para su entorno informático. AWS AppConfig El agente trabaja con lo siguiente Servicios de AWS:
   - AWS Lambda
   - Amazon EC2

- Amazon ECS y Amazon EKS

2. Si el agente está en ejecución, deténgalo.
3. `LOCAL_DEVELOPMENT_DIRECTORY`Añádalo a la lista de variables de entorno. Especifique un directorio en el sistema de archivos que proporcione al agente permisos de lectura. Por ejemplo, `/tmp/local_configs`.
4. Cree un archivo en el directorio. El nombre del archivo debe tener el siguiente formato:

```
application_name:environment_name:configuration_profile_name
```

A continuación se muestra un ejemplo:

```
Mobile:Development:EnableMobilePaymentsFeatureFlagConfiguration
```

> **Note**
>
> (Opcional) Puede controlar el tipo de contenido que devuelve el agente para sus datos de configuración en función de la extensión que le dé al archivo. Por ejemplo, si nombras el archivo con la extensión.json, el agente devolverá un tipo de contenido `application/json` cuando la aplicación lo solicite. Si omites la extensión, el agente la utilizará `application/octet-stream` para el tipo de contenido. Si necesita un control preciso, puede proporcionar una extensión en el formato `.type%subtype`. El agente devolverá un tipo de contenido de `.type/subtype`.

5. Ejecute el siguiente comando para reiniciar el agente y solicitar los datos de configuración.

```
curl http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name
```

El agente comprueba si hay cambios en el archivo local en el intervalo de sondeo especificado para el agente. Si no se especifica el intervalo de sondeo, el agente usa el intervalo predeterminado de 45 segundos. Esta comprobación en el intervalo de sondeo garantiza que el agente se comporte de la misma manera en un entorno de desarrollo local que cuando está configurado para interactuar con el AWS AppConfig servicio.

> **Note**
>
> Para implementar una nueva versión de un archivo de configuración de desarrollo local, actualice el archivo con datos nuevos.

Recuperación de configuraciones mediante una llamada directa a las API

La aplicación recupera los datos de configuración estableciendo primero una sesión de configuración mediante la operación de la StartConfigurationSessionAPI. A continuación, el cliente de la sesión realiza llamadas periódicas GetLatestConfigurationpara comprobar y recuperar los últimos datos disponibles.

Al llamar a `StartConfigurationSession`, el código envía la siguiente información:

- Identificadores (ID o nombre) de una AWS AppConfig aplicación, un entorno y un perfil de configuración que la sesión rastrea.
- (Opcional) Tiempo mínimo que el cliente de la sesión debe esperar entre llamadas a `GetLatestConfiguration`.

Como respuesta, AWS AppConfig proporciona un valor `InitialConfigurationToken` que se proporciona al cliente de la sesión y se utiliza la primera vez que llama a `GetLatestConfiguration` esa sesión.

> **Important**
>
> Este token solo debe usarse una vez en la primera llamada a `GetLatestConfiguration`. Debe usar el nuevo token en la respuesta de `GetLatestConfiguration` (`NextPollConfigurationToken`) en cada llamada posterior a `GetLatestConfiguration`. Para admitir los casos de uso de sondeos prolongados, los tokens son válidos durante un máximo de 24 horas. Si en una llamada a `GetLatestConfiguration` se utiliza un token caducado, el sistema devuelve `BadRequestException`.

Al llamar a `GetLatestConfiguration`, su código de cliente envía el valor más reciente de `ConfigurationToken` del que dispone y recibe como respuesta:

- `NextPollConfigurationToken`: el valor de `ConfigurationToken` que se utilizará en la siguiente llamada a `GetLatestConfiguration`.
- `NextPollIntervalInSeconds`: Tiempo que el cliente debe esperar antes de realizar su próxima llamada a `GetLatestConfiguration`.
- Configuración: los datos más recientes destinados a la sesión. Puede estar vacía si el cliente ya tiene la versión más reciente de la configuración.

Important

Tenga en cuenta la siguiente información importante.

- Solo se debe llamar a la StartConfigurationSessionAPI una vez por aplicación, entorno, perfil de configuración y cliente para establecer una sesión con el servicio. Por lo general, se hace al iniciar la aplicación o inmediatamente antes de recuperar una configuración por primera vez.
- Si la configuración se implementa mediante un `KmsKeyIdentifier`, la solicitud para recibir la configuración debe incluir el permiso para llamar a `kms:Decrypt`. Para obtener más información, consulte Descifrar en la Guía de referencia de la API de AWS Key Management Service .
- La operación de API que se utilizaba anteriormente para recuperar los datos de configuración, `GetConfiguration`, ha quedado obsoleta. La operación de API `GetConfiguration` no admite configuraciones cifradas.

Recuperación de un ejemplo de configuración

En el siguiente AWS CLI ejemplo, se muestra cómo recuperar los datos de configuración mediante las operaciones de AWS AppConfig datos `StartConfigurationSession` y `GetLatestConfiguration` API. El primer comando inicia una sesión de configuración. Esta llamada incluye los ID (o nombres) de la AWS AppConfig aplicación, el entorno y el perfil de configuración. La API devuelve un `InitialConfigurationToken` que se utiliza para recuperar los datos de configuración.

```
aws appconfigdata start-configuration-session \
    --application-identifier application_name_or_ID \
    --environment-identifier environment_name_or_ID \
    --configuration-profile-identifier configuration_profile_name_or_ID
```

El sistema devuelve información similar al siguiente formato.

```
{
   "InitialConfigurationToken": initial configuration token
}
```

Tras iniciar una sesión, utilice esta función InitialConfigurationTokenGetLatestConfigurationpara llamar y obtener los datos de configuración. Los datos de configuración se guardan en el archivo `mydata.json`.

```
aws appconfigdata get-latest-configuration \
    --configuration-token initial configuration token mydata.json
```

La primera llamada a `GetLatestConfiguration` utiliza el `ConfigurationToken` obtenido de `StartConfigurationSession`. Se devuelve la siguiente información.

```
{
    "NextPollConfigurationToken" : next configuration token,
    "ContentType" : content type of configuration,
    "NextPollIntervalInSeconds" : 60
}
```

Las llamadas posteriores a `GetLatestConfiguration` deben proporcionar el `NextPollConfigurationToken` de la respuesta anterior.

```
aws appconfigdata get-latest-configuration \
    --configuration-token next configuration token mydata.json
```

Important

Tenga en cuenta los siguientes detalles importantes acerca de la operación de API `GetLatestConfiguration`:

- La respuesta de `GetLatestConfiguration` incluye una sección `Configuration` que muestra los datos de configuración. La sección `Configuration` solo aparece si el sistema encuentra datos de configuración nuevos o actualizados. Si el sistema no encuentra datos de configuración nuevos o actualizados, los datos de `Configuration` están vacíos.
- Recibirá un nuevo `ConfigurationToken` en cada respuesta de `GetLatestConfiguration`.
- Recomendamos ajustar la frecuencia de sondeo de las llamadas a la API `GetLatestConfiguration` en función del presupuesto, la frecuencia esperada de las implementaciones de configuración y el número de destinos para una configuración.

Ampliación de los flujos de trabajo mediante extensiones

Una extensión aumenta la capacidad de introducir lógica o comportamiento en diferentes puntos del AWS AppConfig flujo de trabajo de creación o implementación de una configuración. Por ejemplo, puede utilizar las extensiones para realizar los siguientes tipos de tareas (por solo citar algunas):

- Enviar una notificación a un tema de Amazon Simple Notification Service (Amazon SNS) cuando se implemente un perfil de configuración.
- Eliminar los datos confidenciales del contenido de un perfil de configuración antes de que comience la implementación.
- Crear o actualizar un problema de Atlassian Jira cada vez que se realice un cambio en una marca de características.
- Combinar el contenido de un servicio o un origen de datos con sus datos de configuración al iniciar una implementación.
- Realizar una copia de seguridad de una configuración en un bucket de Amazon Simple Storage Service (Amazon S3) siempre que se implemente una configuración.

Puede asociar estos tipos de tareas a AWS AppConfig aplicaciones, entornos y perfiles de configuración.

Contenido

Acerca de AWS AppConfig las extensiones

En este tema se presentan los conceptos y la terminología de las AWS AppConfig extensiones. La información se analiza en el contexto de cada paso necesario para configurar y utilizar AWS AppConfig las extensiones.

Temas

- Paso 2: Determinar cuándo quiere que se ejecute la extensión
- Paso 3: Crear una asociación de extensión
- Paso 4: Implementar una configuración y comprobar que se llevan a cabo las acciones de la extensión

Paso 1: Determine lo que quiere hacer con las extensiones

¿Quieres recibir una notificación de un webhook que envía mensajes a Slack cada vez que se complete una AWS AppConfig implementación? ¿Desea hacer una copia de seguridad de un perfil de configuración en un bucket de Amazon Simple Storage Service (Amazon S3) antes de implementar la configuración? ¿Desea eliminar la información confidencial de los datos de configuración antes de implementar la configuración? Puede usar extensiones para realizar este tipo de tareas y más. Puedes crear extensiones personalizadas o usar las extensiones AWS creadas que vienen incluidas. AWS AppConfig

> **Note**
>
> En la mayoría de los casos de uso, para crear una extensión personalizada, debe crear una AWS Lambda función para realizar cualquier cálculo y procesamiento definidos en la extensión. Para obtener más información, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Las siguientes extensiones AWS creadas pueden ayudarle a integrar rápidamente las implementaciones de configuración con otros servicios. Puede usar estas extensiones en la AWS AppConfig consola o llamando a las acciones de la API de extensión directamente desde el AWS CLI AWS Tools for PowerShell, o el SDK.

Extensión	Descripción
Amazon CloudWatch Evidentemente, pruebas A/B	Esta extensión permite a su aplicación asignar variaciones a las sesiones de usuario de forma local en lugar de tener que llamar a la EvaluateFeatureoperación. Para obtener más información, consulte Trabajar con la extensión Amazon CloudWatch Evidently.

Extensión	Descripción
AWS AppConfig eventos de despliegue para EventBridge	Esta extensión envía los eventos al bus de eventos EventBridge predeterminado cuando se implementa una configuración.
AWS AppConfig eventos de despliegue en Amazon Simple Notification Service (Amazon SNS)	Esta extensión envía mensajes al tema de Amazon SNS que el usuario haya especificado cuando se implementa una configuración.
AWS AppConfig eventos de despliegue en Amazon Simple Queue Service (Amazon SQS)	Esta extensión coloca los mensajes en la cola de Amazon SQS cuando se implementa una configuración.
Extensión de integración: Atlassian Jira	Esta extensión permite AWS AppConfig crear y actualizar problemas cada vez que se realizan cambios en un indicador de función.

Paso 2: Determinar cuándo quiere que se ejecute la extensión

Una extensión define una o más acciones que realiza durante un AWS AppConfig flujo de trabajo. Por ejemplo, la `AWS AppConfig deployment events to Amazon SNS` extensión AWS creada incluye una acción para enviar una notificación a un tema de Amazon SNS. Cada acción se invoca cuando interactúas con un proceso AWS AppConfig o cuando AWS AppConfig lo realizas en tu nombre. Se denominan puntos de acción. AWS AppConfig las extensiones admiten los siguientes puntos de acción:

- `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`
- `PRE_START_DEPLOYMENT`
- `ON_DEPLOYMENT_START`
- `ON_DEPLOYMENT_STEP`
- `ON_DEPLOYMENT_BAKING`
- `ON_DEPLOYMENT_COMPLETE`
- `ON_DEPLOYMENT_ROLLED_BACK`

Las acciones de extensión configuradas en los puntos de `PRE_*` acción se aplican después de la validación de la solicitud, pero antes de AWS AppConfig realizar la actividad correspondiente al nombre del punto de acción. Estas invocaciones de acciones se procesan al mismo tiempo que una solicitud. Si se realiza más de una solicitud, las invocaciones a las acciones se ejecutan de forma secuencial. Tenga en cuenta también que los puntos de acción de `PRE_*` reciben y pueden cambiar el contenido de una configuración. Los puntos de acción `PRE_*` también pueden responder a un error e impedir que se lleve a cabo una acción.

Una extensión también se puede ejecutar en paralelo con un AWS AppConfig flujo de trabajo mediante un punto de `ON_*` acción. `ON_*`los puntos de acción se invocan de forma asíncrona. `ON_*`los puntos de acción no reciben el contenido de una configuración. Si una extensión experimenta un error durante un punto de acción de `ON_*`, el servicio ignora el error y continúa con el flujo de trabajo.

Paso 3: Crear una asociación de extensión

Para crear una extensión o configurar una extensión AWS creada, debe definir los puntos de acción que invocan una extensión cuando se utiliza un AWS AppConfig recurso específico. Por ejemplo, puede optar por ejecutar la extensión de `AWS AppConfig deployment events to Amazon SNS` y recibir notificaciones sobre un tema de Amazon SNS cada vez que se inicie una implementación de configuración para una aplicación específica. Definir qué puntos de acción invocan una extensión para un AWS AppConfig recurso específico se denomina asociación de extensiones. Una asociación de extensiones es una relación especificada entre una extensión y un AWS AppConfig recurso, como una aplicación o un perfil de configuración.

Una sola AWS AppConfig aplicación puede incluir varios entornos y perfiles de configuración. Si asocia una extensión a una aplicación o un entorno, AWS AppConfig invoca la extensión para cualquier flujo de trabajo relacionado con los recursos de la aplicación o el entorno, si corresponde.

Por ejemplo, supongamos que tiene una AWS AppConfig aplicación llamada MobileApps que incluye un perfil de configuración llamado AccessList. Supongamos que la MobileApps aplicación incluye entornos beta, de integración y de producción. Debe crear una asociación de extensión para la extensión AWS de notificación de Amazon SNS creada y asociar la extensión a MobileApps la aplicación. La extensión de notificación de Amazon SNS se invoca cada vez que se implementa la configuración de la aplicación en cualquiera de los tres entornos.

> **Note**
>
> No es necesario crear una extensión para utilizar las extensiones AWS creadas, pero sí una asociación de extensiones.

Paso 4: Implementar una configuración y comprobar que se llevan a cabo las acciones de la extensión

Tras crear una asociación, cuando se crea una configuración alojada o se implementa una configuración, AWS AppConfig invoca la extensión y realiza las acciones especificadas. Cuando se invoca una extensión, si el sistema experimenta un error durante un punto de `PRE-*` acción, AWS AppConfig devuelve información sobre ese error.

Trabajar con extensiones creadas de AWS

AWS AppConfig incluye las siguientes extensiones de AWS autor. Estas extensiones pueden ayudarle a integrar el AWS AppConfig flujo de trabajo con otros servicios. Puedes usar estas extensiones en el SDK AWS Management Console o mediante una llamada a las acciones de la API de extensiones directamente desde el AWS CLI AWS Tools for PowerShell, o el SDK.

Extensión	Descripción
Amazon CloudWatch Evidentemente, pruebas A/B	Esta extensión permite a su aplicación asignar variaciones a las sesiones de usuario de forma local en lugar de tener que llamar a la EvaluateFeatureoperación. Para obtener más información, consulte Trabajar con la extensión Amazon CloudWatch Evidently.
AWS AppConfig eventos de despliegue para EventBridge	Esta extensión envía los eventos al bus de eventos EventBridge predeterminado cuando se implementa una configuración.
AWS AppConfig eventos de despliegue en Amazon Simple Notification Service (Amazon SNS)	Esta extensión envía mensajes al tema de Amazon SNS que el usuario haya especificado cuando se implementa una configuración.

Extensión	Descripción
AWS AppConfig eventos de despliegue en Amazon Simple Queue Service (Amazon SQS)	Esta extensión coloca los mensajes en la cola de Amazon SQS cuando se implementa una configuración.
Extensión de integración: Atlassian Jira	Esta extensión permite AWS AppConfig crear y actualizar problemas cada vez que se realizan cambios en un indicador de función.

Trabajar con la extensión Amazon CloudWatch Evidently

Puede utilizar Amazon CloudWatch Evidently para validar nuevas funciones de forma segura ofreciéndolas a un porcentaje específico de sus usuarios mientras implementa la función. Puede monitorear el rendimiento de la nueva característica para decidir cuándo aumentar el tráfico hacia los usuarios. Esto ayuda a reducir los riesgos e identificar las consecuencias no deseadas antes de lanzar la característica por completo. También puede llevar a cabo experimentos A/B para tomar decisiones de diseño de características basadas en pruebas y datos.

La AWS AppConfig extensión de CloudWatch Evidently permite a la aplicación asignar variaciones a las sesiones de usuario de forma local, en lugar de tener que llamar a la EvaluateFeatureoperación. Esto mitiga los riesgos de latencia y disponibilidad que conlleva una llamada a la API. Para obtener información sobre cómo configurar y usar la extensión, consulte Realizar lanzamientos y experimentos A/B con CloudWatch Evidently en la Guía CloudWatch del usuario de Amazon.

Uso de la extensión **`AWS AppConfig deployment events to Amazon EventBridge`**

La `AWS AppConfig deployment events to Amazon EventBridge` extensión es una AWS extensión creada que le ayuda a supervisar el flujo de trabajo de implementación de la AWS AppConfig configuración y actuar en consecuencia. La extensión envía notificaciones de eventos al bus de eventos EventBridge predeterminado cada vez que se implementa una configuración. Una vez que haya asociado la extensión a una de sus AWS AppConfig aplicaciones, entornos o perfiles de configuración, AWS AppConfig envía notificaciones de eventos al bus de eventos cada vez que se inicia, finaliza y revierte la implementación de la configuración.

Si quieres tener más control sobre los puntos de acción que envían EventBridge las notificaciones, puedes crear una extensión personalizada e introducir el nombre de recurso de Amazon (ARN) del bus de eventos EventBridge predeterminado para el campo URI. Para obtener información sobre la creación de una extensión, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

> **Important**
>
> Esta extensión solo admite el bus de eventos EventBridge predeterminado.

Uso del paquete de extensión

Para usar la `AWS AppConfig deployment events to Amazon EventBridge` extensión, primero debe adjuntarla a uno de sus AWS AppConfig recursos mediante la creación de una asociación de extensiones. La asociación se crea mediante la AWS AppConfig consola o la acción de la CreateExtensionAssociationAPI. Al crear la asociación, se especifica el ARN de una AWS AppConfig aplicación, un entorno o un perfil de configuración. Si asocia la extensión a una aplicación o un entorno, se envía una notificación de evento para cualquier perfil de configuración contenido en la aplicación o el entorno especificados.

Tras crear la asociación, cuando se implementa una configuración para el AWS AppConfig recurso especificado, AWS AppConfig invoca la extensión y envía las notificaciones en función de los puntos de acción especificados en la extensión.

> **Note**
>
> Esta extensión se invoca mediante los siguientes puntos de acción:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> No se pueden personalizar los puntos de acción de esta extensión. Para invocar diferentes puntos de acción, puede crear su propia extensión. Para obtener más información, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Utilice los siguientes procedimientos para crear una asociación de AWS AppConfig extensiones mediante la AWS Systems Manager consola o el AWS CLI.

Para crear una extensión de asociación (consola)

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. En la pestaña Extensiones, seleccione Añadir al recurso.
4. En la sección de detalles del recurso de la extensión, en Tipo de recurso, elija un tipo de AWS AppConfig recurso. Según el recurso que elija, AWS AppConfig le solicitará que elija otros recursos.
5. Elija Crear una asociación al recurso.

Este es un ejemplo de evento que se envía EventBridge cuando se invoca la extensión.

```
{
   "version":"0",
   "id":"c53dbd72-c1a0-2302-9ed6-c076e9128277",
   "detail-type":"On Deployment Complete",
   "source":"aws.appconfig",
   "account":"111122223333",
   "time":"2022-07-09T01:44:15Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:appconfig:us-east-1:111122223333:extensionassociation/z763ff5"
   ],
   "detail":{
      "InvocationId":"5tfjcig",
       "Parameters":{

      },
      "Type":"OnDeploymentComplete",
      "Application":{
         "Id":"ba8toh7",
         "Name":"MyApp"
      },
      "Environment":{
         "Id":"pgil2o7",
         "Name":"MyEnv"
```

```
      },
      "ConfigurationProfile":{
         "Id":"ga3tqep",
         "Name":"MyConfigProfile"
      },
      "DeploymentNumber":1,
      "ConfigurationVersion":"1"
   }
}
```

Uso de la extensión **AWS AppConfig deployment events to Amazon SNS**

La `AWS AppConfig deployment events to Amazon SNS` extensión es una AWS extensión creada que le ayuda a supervisar el flujo de trabajo de implementación de la AWS AppConfig configuración y actuar en consecuencia. La extensión publica mensajes en un tema de Amazon SNS siempre que se implementa una configuración. Tras asociar la extensión a una de sus AWS AppConfig aplicaciones, entornos o perfiles de configuración, AWS AppConfig publica un mensaje sobre el tema cada vez que se inicie, finalice y revierta la implementación de la configuración.

Si desea tener más control sobre los puntos de acción que envían las notificaciones de Amazon SNS, puede crear una extensión personalizada e introducir el nombre de recurso de Amazon (ARN) del tema de Amazon SNS en el campo URI. Para obtener información sobre la creación de una extensión, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Uso del paquete de extensión

En esta sección, se explica cómo se utiliza la extensión `AWS AppConfig deployment events to Amazon SNS`.

Paso 1: Configurar AWS AppConfig la publicación de mensajes en un tema

Añada una política de control de acceso a su tema de Amazon SNS concediendo permisos de publicación de AWS AppConfig (`appconfig.amazonaws.com`) (`sns:Publish`). Para obtener más información, consulte Casos de ejemplo para el control de acceso de Amazon SNS.

Paso 2: Crear una asociación de extensión

Adjunta la extensión a uno de tus AWS AppConfig recursos creando una asociación de extensiones. La asociación se crea mediante la AWS AppConfig consola o la acción de la

CreateExtensionAssociationAPI. Al crear la asociación, se especifica el ARN de una AWS AppConfig aplicación, un entorno o un perfil de configuración. Si asocia la extensión a una aplicación o un entorno, se envía una notificación para cualquier perfil de configuración contenido en la aplicación o el entorno especificados. Al crear la asociación, debe introducir un valor para el parámetro `topicArn` que contiene el ARN del tema de Amazon SNS que desee utilizar.

Tras crear la asociación, cuando se implementa una configuración para el AWS AppConfig recurso especificado, AWS AppConfig invoca la extensión y envía las notificaciones en función de los puntos de acción especificados en la extensión.

> **Note**
>
> Esta extensión se invoca mediante los siguientes puntos de acción:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> No se pueden personalizar los puntos de acción de esta extensión. Para invocar diferentes puntos de acción, puede crear su propia extensión. Para obtener más información, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Utilice los siguientes procedimientos para crear una asociación de AWS AppConfig extensiones mediante la AWS Systems Manager consola o el AWS CLI.

Para crear una extensión de asociación (consola)

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. En la pestaña Extensiones, seleccione Añadir al recurso.
4. En la sección de detalles del recurso de la extensión, en Tipo de recurso, elija un tipo de AWS AppConfig recurso. Según el recurso que elija, AWS AppConfig le solicitará que elija otros recursos.
5. Elija Crear una asociación al recurso.

A continuación, se incluye un ejemplo del mensaje que se envía al tema de Amazon SNS cuando se invoca la extensión.

```
{
    "Type": "Notification",
    "MessageId": "ae9d702f-9a66-51b3-8586-2b17932a9f28",
    "TopicArn": "arn:aws:sns:us-east-1:111122223333:MySNSTopic",
    "Message": {
        "InvocationId": "7itcaxp",
        "Parameters": {
            "topicArn": "arn:aws:sns:us-east-1:111122223333:MySNSTopic"
        },
        "Application": {
            "Id": "1a2b3c4d",
            "Name": MyApp
        },
        "Environment": {
            "Id": "1a2b3c4d",
            "Name": MyEnv
        },
        "ConfigurationProfile": {
            "Id": "1a2b3c4d",
            "Name": "MyConfigProfile"
        },
        "Description": null,
        "DeploymentNumber": "3",
        "ConfigurationVersion": "1",
        "Type": "OnDeploymentComplete"
    },
    "Timestamp": "2022-06-30T20:26:52.067Z",
    "SignatureVersion": "1",
    "Signature": "<...>",
    "SigningCertURL": "<...>",
    "UnsubscribeURL": "<...>",
    "MessageAttributes": {
        "MessageType": {
            "Type": "String",
            "Value": "OnDeploymentStart"
        }
    }
}
```

Uso de la extensión **AWS AppConfig deployment events to Amazon SQS**

La `AWS AppConfig deployment events to Amazon SQS` extensión es una AWS extensión creada que le ayuda a supervisar el flujo de trabajo de implementación de la AWS AppConfig configuración y actuar en consecuencia. La extensión coloca los mensajes en la cola de Amazon Simple Queue Service (Amazon SQS) cada vez que se implementa una configuración. Tras asociar la extensión a una de sus AWS AppConfig aplicaciones, entornos o perfiles de configuración, coloca un mensaje en la AWS AppConfig cola cada vez que se inicie, finalice o revierta la implementación de la configuración.

Si desea tener más control sobre los puntos de acción que envían las notificaciones de Amazon SQS, puede crear una extensión personalizada e introducir el nombre de recurso de Amazon (ARN) de la cola de Amazon SQS en el campo URI. Para obtener información sobre la creación de una extensión, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Uso del paquete de extensión

En esta sección, se explica cómo se utiliza la extensión `AWS AppConfig deployment events to Amazon SQS`.

Paso 1: Configure para poner los mensajes en cola AWS AppConfig

Añada una política de Amazon SQS a su cola de Amazon SQS concediendo permisos de envío de mensajes AWS AppConfig (`appconfig.amazonaws.com`) (`sqs:SendMessage`). Para obtener más información, consulte Ejemplos básicos de políticas de Amazon SQS.

Paso 2: Crear una asociación de extensión

Adjunte la extensión a uno de sus AWS AppConfig recursos creando una asociación de extensiones. La asociación se crea mediante la AWS AppConfig consola o la acción de la CreateExtensionAssociationAPI. Al crear la asociación, se especifica el ARN de una AWS AppConfig aplicación, un entorno o un perfil de configuración. Si asocia la extensión a una aplicación o un entorno, se envía una notificación para cualquier perfil de configuración contenido en la aplicación o el entorno especificados. Al crear la asociación, debe introducir un parámetro `Here` que contenga el ARN de la cola de Amazon SQS que desee utilizar.

Tras crear la asociación, cuando se crea o implementa una configuración para el AWS AppConfig recurso especificado, AWS AppConfig invoca la extensión y envía las notificaciones en función de los puntos de acción especificados en la extensión.

> **Note**
>
> Esta extensión se invoca mediante los siguientes puntos de acción:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> No se pueden personalizar los puntos de acción de esta extensión. Para invocar diferentes puntos de acción, puede crear su propia extensión. Para obtener más información, consulte Tutorial: Creación de extensiones personalizadas AWS AppConfig.

Utilice los siguientes procedimientos para crear una asociación de AWS AppConfig extensiones mediante la AWS Systems Manager consola o el AWS CLI.

Para crear una extensión de asociación (consola)

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. En la pestaña Extensiones, seleccione Añadir al recurso.
4. En la sección de detalles del recurso de la extensión, en Tipo de recurso, elija un tipo de AWS AppConfig recurso. Según el recurso que elija, AWS AppConfig le solicitará que elija otros recursos.
5. Elija Crear una asociación al recurso.

A continuación, se incluye un ejemplo del mensaje que se envía a la cola de Amazon SQS cuando se invoca la extensión.

```
{
   "InvocationId":"7itcaxp",
   "Parameters":{
      "queueArn":"arn:aws:sqs:us-east-1:111122223333:MySQSQueue"
   },
   "Application":{
      "Id":"1a2b3c4d",
```

```
      "Name":MyApp
   },
   "Environment":{
      "Id":"1a2b3c4d",
      "Name":MyEnv
   },
   "ConfigurationProfile":{
      "Id":"1a2b3c4d",
      "Name":"MyConfigProfile"
   },
   "Description":null,
   "DeploymentNumber":"3",
   "ConfigurationVersion":"1",
   "Type":"OnDeploymentComplete"
}
```

Trabajando con la extensión Jira de Atlassian para AWS AppConfig

Gracias a la integración con Atlassian Jira, AWS AppConfig puedes crear y actualizar problemas en la consola de Atlassian siempre que realices cambios en una de tus marcas de características según lo especificado. Cuenta de AWS Región de AWS Cada problema de Jira incluye el nombre de la marca, el ID de la aplicación, el ID del perfil de configuración y los valores de la marca. Tras actualizar, guardar e implementar los cambios de las marcas, Jira actualiza los problemas existentes con los detalles del cambio.

> **Note**
>
> Jira actualiza los problemas cada vez que se crea o actualiza una marca de características. Jira también actualiza los problemas cuando se elimina un atributo de marca de nivel secundario de una marca de nivel principal. Jira no registra información cuando se elimina una marca de nivel principal.

Para configurar la integración, debe hacer lo siguiente:

- Configurar los permisos para la integración con Jira AWS AppConfig
- Configuración de la aplicación de integración de AWS AppConfig Jira

Configurar los permisos para la integración con Jira AWS AppConfig

Cuando configuras AWS AppConfig la integración con Jira, especificas las credenciales de un usuario. En concreto, se introduce el identificador de la clave de acceso y la clave secreta del usuario en la aplicación AWS AppConfig para Jira. Este usuario le da permiso a Jira para comunicarse con él. AWS AppConfig AWS AppConfig usa estas credenciales una vez para establecer una asociación entre AWS AppConfig y Jira. Las credenciales no se almacenan. Puedes eliminar la asociación desinstalando la aplicación AWS AppConfig para Jira.

La cuenta de usuario requiere una política de permisos que incluya las siguientes acciones:

- `appconfig:CreateExtensionAssociation`
- `appconfig:GetConfigurationProfile`
- `appconfig:ListApplications`
- `appconfig:ListConfigurationProfiles`
- `appconfig:ListExtensionAssociations`
- `sts:GetCallerIdentity`

Realice las siguientes tareas para crear una política de permisos de IAM y un usuario para la integración de AWS AppConfig y Jira:

Tareas

- Tarea 1: Crear una política de permisos de IAM para AWS AppConfig la integración con Jira
- Tarea 2: Crear un usuario para una integración con AWS AppConfig Jira

Tarea 1: Crear una política de permisos de IAM para AWS AppConfig la integración con Jira

Usa el siguiente procedimiento para crear una política de permisos de IAM que permita comunicarse con Atlassian Jira. AWS AppConfig Recomendamos crear una nueva política y adjuntarla a un nuevo rol de IAM. Añadir el permiso necesario a una política y un rol de IAM existentes va en contra del principio de privilegio mínimo y no se recomienda.

Para crear una política de IAM para la integración con Jira AWS AppConfig

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

2. En el panel de navegación, seleccione Policies (Políticas) y, a continuación, seleccione Create policy (Crear política).
3. En la página de Crear política, elija la pestaña JSON y, a continuación, sustituya el contenido predeterminado por la siguiente política JSON. En la siguiente política, sustituya *Region*, *account_ID*, *application_ID* y *configuration_profile_ID* por información de su entorno de marcas de características de AWS AppConfig .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                      "appconfig:CreateExtensionAssociation",
                      "appconfig:ListExtensionAssociations",
                      "appconfig:GetConfigurationProfile"
             ],
            "Resource": [

 "arn:aws:appconfig:Region:account_ID:application/application_ID",

 "arn:aws:appconfig:Region:account_ID:application/application_ID/
configurationprofile/configuration_profile_ID"
             ]
        },
       {
            "Effect": "Allow",
            "Action": [
                      "appconfig:ListApplications"

             ],
            "Resource": [
                      "arn:aws:appconfig:Region:account_ID:*"
             ]
        },
         {
            "Effect": "Allow",
            "Action": [
                      "appconfig:ListConfigurationProfiles"
             ],
            "Resource": [
```

```
 "arn:aws:appconfig:Region:account_ID:application/application_ID"
             ]
        },
        {
            "Effect": "Allow",
            "Action": "sts:GetCallerIdentity",
            "Resource": "*"
        }
    ]
}
```

4. Elija Siguiente: Etiquetas.
5. (Opcional) Agregue uno o varios pares de valor etiqueta-clave para organizar, realizar un seguimiento o controlar el acceso a esta política y, a continuación, elija Next: Review (Siguiente: Revisar).
6. En la página Review policy (Revisar política), ingrese un nombre en el cuadro Name (Nombre), como **`AppConfigJiraPolicy`**, y luego ingrese una descripción opcional.
7. Seleccione Crear política.

Tarea 2: Crear un usuario para una integración con AWS AppConfig Jira

Usa el siguiente procedimiento para crear un usuario para AWS AppConfig una integración de Atlassian Jira. Tras crear el usuario, puede copiar el ID de la clave de acceso y la clave secreta, que especificará cuando complete la integración.

Para crear un usuario para AWS AppConfig una integración con Jira

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, elija Users (Usuarios), y luego Add users (Agregar usuarios).
3. En el campo Nombre de usuario, introduzca un nombre, como **`AppConfigJiraUser`**.
4. En Seleccione el tipo de AWS credencial, elija Clave de acceso: Acceso programático.
5. Elija Siguiente: permisos.
6. En la página de Establecer permisos, elija Asociar directamente las políticas existentes. Busque la política que ha creado en Tarea 1: Crear una política de permisos de IAM para AWS AppConfig la integración con Jira, seleccione la casilla de verificación y, a continuación, elija Siguiente: Etiquetas.

7. En la página de Añadir etiquetas (opcional), agregue uno o varios pares clave-valor de etiqueta para organizar o controlar el acceso a este usuario o realizar su seguimiento. Elija Siguiente: Revisar.
8. En la página Revisar, verifique los detalles del usuario.
9. Seleccione la opción Crear un usuario. El sistema muestra el ID de clave de acceso y la clave secreta del usuario. Descargue el archivo.csv o copie estas credenciales en otra ubicación. Especificará estas credenciales al configurar la integración.

Configuración de la aplicación de integración de AWS AppConfig Jira

Utilice el siguiente procedimiento para configurar las opciones necesarias en la aplicación AWS AppConfig para Jira. Tras completar este procedimiento, Jira crea una nueva emisión para cada indicador de función especificado en el Cuenta de AWS suyo. Región de AWS Si realizas cambios en una marca de función AWS AppConfig, Jira registrará los detalles de las incidencias existentes.

> **Note**
>
> Una marca AWS AppConfig de entidad puede incluir varios atributos de marca de nivel secundario. Jira crea un problema para cada marca de características de nivel principal. Si cambia un atributo de marca de nivel secundario, puede ver los detalles de ese cambio en el problema de Jira de la marca de nivel principal.

Configurar la integración

1. Iniciar sesión en Atlassian Marketplace.
2. En el campo de búsqueda, escriba **`AWS AppConfig`** y pulse Intro.
3. Instale la aplicación en la nueva instancia de Jira.
4. En la consola de Atlassian, seleccione Administrar aplicaciones y, a continuación, elija AWS AppConfig para Jira.
5. Elija Configurar.
6. En Detalles de configuración, elija el proyecto de Jira y, a continuación, elija el proyecto que quiere asociar a su marca de características de AWS AppConfig .
7. Elija Región de AWS y, a continuación, elija la Región en la que se encuentra su marca de características de AWS AppConfig .

8. En el campo ID de la aplicación, introduzca el nombre de la aplicación de AWS AppConfig que contiene su marca de características.
9. En el campo ID del perfil de configuración, introduzca el nombre del perfil de configuración de AWS AppConfig de su marca de características.
10. En los campos ID de clave de acceso y Clave secreta, introduzca las credenciales que ha copiado en Tarea 2: Crear un usuario para una integración con AWS AppConfig Jira. Si lo desea, también puede especificar un token de sesión.
11. Elija Enviar.
12. En la consola de Atlassian, selecciona Proyectos y, a continuación, elige el proyecto que has seleccionado para la integración. AWS AppConfig La página de problemas muestra un problema para cada indicador de función en el y especificado Cuenta de AWS . Región de AWS

Eliminar la aplicación AWS AppConfig para Jira y los datos

Si ya no quieres usar la integración de Jira con los marcadores de AWS AppConfig funciones, puedes eliminar la aplicación AWS AppConfig para Jira en la consola de Atlassian. Al eliminar la aplicación de integración, ocurre lo siguiente:

- Elimina la asociación entre tu instancia de Jira y AWS AppConfig
- Elimina los detalles de tu instancia de Jira de AWS AppConfig

Para eliminar la aplicación AWS AppConfig para Jira

1. En la consola de Atlassian, seleccione Administrar aplicaciones.
2. Elija AWS AppConfig para Jira.
3. Elija Desinstalar.

Tutorial: Creación de extensiones personalizadas AWS AppConfig

Para crear una AWS AppConfig extensión personalizada, complete las siguientes tareas. Cada tarea se describe más detalladamente en temas posteriores.

> **Note**
>
> Puede ver ejemplos de AWS AppConfig extensiones personalizadas en GitHub:

- Ejemplo de extensión que impide las implementaciones con un calendario de `blocked day` moratorias mediante Systems Manager Change Calendar
- Ejemplo de extensión que evita que los secretos se filtren en los datos de configuración mediante git-secrets
- Ejemplo de extensión que evita que la información de identificación personal (PII) se filtre en los datos de configuración mediante Amazon Comprehend

1. Cree una función AWS Lambda

   En la mayoría de los casos de uso, para crear una extensión personalizada, debe crear una AWS Lambda función para realizar cualquier cálculo y procesamiento definidos en la extensión. Una excepción a esta regla es si se crean versiones personalizadas de las extensiones de notificación creadas de AWS para añadir o eliminar puntos de acción. Para obtener más información sobre esta excepción, consulte Crear una extensión personalizada AWS AppConfig.

2. Configurar los permisos para su extensión personalizada

   Para configurar los permisos para su extensión personalizada, puede hacer una de las siguientes acciones:

   - Cree un rol de servicio AWS Identity and Access Management (IAM) que incluya `InvokeFunction` permisos.
   - Cree una política de recursos mediante la acción de la AddPermissionAPI Lambda.

   Este tutorial describe cómo crear el rol de servicio de IAM.

3. Crear una extensión

   Puede crear una extensión mediante la AWS AppConfig consola o mediante una llamada a la acción de la CreateExtensionAPI desde el AWS CLI AWS Tools for PowerShell, o el SDK. En el tutorial se utiliza la consola.

4. Crear una asociación de extensión

   Puedes crear una asociación de extensiones mediante la AWS AppConfig consola o mediante una llamada a la acción de la CreateExtensionAssociationAPI desde el AWS CLI AWS Tools for PowerShell, o el SDK. En el tutorial se utiliza la consola.

5. Realizar una acción que invoque la extensión

   Tras crear la asociación, AWS AppConfig invoca la extensión cuando se producen los puntos de acción definidos por la extensión para ese recurso. Por ejemplo, si asocia una extensión que contiene una acción de `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`, se invocará la extensión cada vez que cree una nueva versión de la configuración alojada.

En los temas de esta sección, se describen las tareas necesarias para crear una extensión personalizada de AWS AppConfig . Cada tarea se describe en el contexto de un caso de uso en el que un cliente desea crear una extensión que haga automáticamente una copia de seguridad de una configuración en un bucket de Amazon Simple Storage Service (Amazon S3). La extensión se ejecuta cada vez que se crea (`PRE_CREATE_HOSTED_CONFIGURATION_VERSION`) o se implementa (`PRE_START_DEPLOYMENT`) una configuración alojada.

Temas

Creación de una función Lambda para una extensión personalizada AWS AppConfig

En la mayoría de los casos de uso, para crear una extensión personalizada, debe crear una AWS Lambda función para realizar cualquier cálculo y procesamiento definidos en la extensión. En esta sección se incluye un código de ejemplo de una función Lambda para una extensión personalizada AWS AppConfig . En esta sección también se incluyen detalles de referencia sobre la solicitud y la respuesta de la carga. Para obtener más información acerca de cómo crear funciones de Lambda, consulte Introducción a Lambda en la Guía del desarrollador de AWS Lambda .

Código de muestra

El siguiente código de ejemplo para una función Lambda, cuando se invoca, realiza automáticamente una copia de seguridad de la AWS AppConfig configuración en un bucket de Amazon S3. Se

hace una copia de seguridad de la configuración cada vez que se crea o implementa una nueva configuración. El ejemplo emplea parámetros en la extensión, por lo que el nombre del bucket no tiene que estar codificado en la función de Lambda. Al usar parámetros en la extensión, el usuario puede adjuntar la extensión a varias aplicaciones y hacer copias de seguridad de las configuraciones en diferentes buckets. El ejemplo de código incluye comentarios que explican mejor la función.

Ejemplo de función Lambda para una extensión AWS AppConfig

```
from datetime import datetime
import base64
import json

import boto3


def lambda_handler(event, context):
    print(event)

    # Extensions that use the PRE_CREATE_HOSTED_CONFIGURATION_VERSION and
 PRE_START_DEPLOYMENT
    # action points receive the contents of AWS AppConfig configurations in Lambda
 event parameters.
    # Configuration contents are received as a base64-encoded string, which the lambda
 needs to decode
    # in order to get the configuration data as bytes. For other action points, the
 content
    # of the configuration isn't present, so the code below will fail.
    config_data_bytes = base64.b64decode(event["Content"])

    # You can specify parameters for extensions. The CreateExtension API action lets
 you define
    # which parameters an extension supports. You supply the values for those
 parameters when you
    # create an extension association by calling the CreateExtensionAssociation API
 action.
    # The following code uses a parameter called S3_BUCKET to obtain the value
 specified in the
    # extension association. You can specify this parameter when you create the
 extension
    # later in this walkthrough.
    extension_association_params = event.get('Parameters', {})
    bucket_name = extension_association_params['S3_BUCKET']
    write_backup_to_s3(bucket_name, config_data_bytes)
```

```
    # The PRE_CREATE_HOSTED_CONFIGURATION_VERSION and PRE_START_DEPLOYMENT action
 points can
    # modify the contents of a configuration. The following code makes a minor change
    # for the purposes of a demonstration.
    old_config_data_string = config_data_bytes.decode('utf-8')
    new_config_data_string = old_config_data_string.replace('hello', 'hello!')
    new_config_data_bytes = new_config_data_string.encode('utf-8')

    # The lambda initially received the configuration data as a base64-encoded string
    # and must return it in the same format.
    new_config_data_base64string =
 base64.b64encode(new_config_data_bytes).decode('ascii')

    return {
        'statusCode': 200,
        # If you want to modify the contents of the configuration, you must include the
 new contents in the
        # Lambda response. If you don't want to modify the contents, you can omit the
 'Content' field shown here.
        'Content': new_config_data_base64string
    }


def write_backup_to_s3(bucket_name, config_data_bytes):
    s3 = boto3.resource('s3')
    new_object = s3.Object(bucket_name,
 f"config_backup_{datetime.now().isoformat()}.txt")
    new_object.put(Body=config_data_bytes)
```

Si desea utilizar este ejemplo durante este tutorial, guárdelo con el nombre **`MyS3ConfigurationBackUpExtension`** y copie el nombre de recurso de Amazon (ARN) de la función. El ARN se especifica al crear la función de asunción AWS Identity and Access Management (IAM) en la siguiente sección. El ARN y el nombre se especifican al crear la extensión.

Referencia de carga

En esta sección se incluyen los detalles de referencia de las solicitudes y respuestas de carga útil para trabajar con extensiones personalizadas. AWS AppConfig

Estructura de la solicitud

PreCreateHostedConfigurationVersion

```
{
    'InvocationId': 'vlns753', // id for specific invocation
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'ContentType': 'text/plain',
    'ContentVersion': '2',
    'Content': 'SGVsbG8gZWFydGgh', // Base64 encoded content
    'Application': {
        'Id': 'abcd123',
        'Name': 'ApplicationName'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'Description': '',
    'Type': 'PreCreateHostedConfigurationVersion',
    'PreviousContent': {
        'ContentType': 'text/plain',
        'ContentVersion': '1',
        'Content': 'SGVsbG8gd29ybGQh'
    }
}
```

PreStartDeployment

```
{
    'InvocationId': '765ahdm',
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'ContentType': 'text/plain',
    'ContentVersion': '2',
    'Content': 'SGVsbG8gZWFydGgh',
    'Application': {
        'Id': 'abcd123',
        'Name': 'ApplicationName'
    },
    'Environment': {
        'Id': 'ibpnqlq',
```

```
        'Name': 'EnvironmentName'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'DeploymentNumber': 2,
    'Description': 'Deployment description',
    'Type': 'PreStartDeployment'
}
```

Eventos asíncronos

OnStartDeployment, OnDeploymentStep, OnDeployment

```
{
    'InvocationId': 'o2xbtm7',
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'Type': 'OnDeploymentStart',
    'Application': {
        'Id': 'abcd123'
    },
    'Environment': {
        'Id': 'efgh456'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'DeploymentNumber': 2,
    'Description': 'Deployment description',
    'ConfigurationVersion': '2'
}
```

Estructura de una respuesta

Los siguientes ejemplos muestran lo que devuelve la función Lambda en respuesta a la solicitud de una extensión personalizada. AWS AppConfig

Eventos sincrónicos: respuesta exitosa

Si desea transformar el contenido, utilice lo siguiente:

```
"Content": "SomeBase64EncodedByteArray"
```

Si no quiere transformar el contenido, no devuelva nada.

Eventos asíncronos: respuesta exitosa

Devuelve: nada

Todos los eventos de error

```
{
        "Error": "BadRequestError",
        "Message": "There was malformed stuff in here",
        "Details": [{
            "Type": "Malformed",
            "Name": "S3 pointer",
            "Reason": "S3 bucket did not exist"
        }]
    }
```

Configurar los permisos para una extensión personalizada AWS AppConfig

Utilice el siguiente procedimiento para crear y configurar un rol de servicio AWS Identity and Access Management (o asumir un rol) de servicio (IAM). AWS AppConfig utiliza este rol para invocar la función Lambda.

Para crear un rol de servicio de IAM y permitir AWS AppConfig asumirlo

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y luego seleccione Crear rol.
3. En Seleccionar entidad de confianza, elija Tipo de entidad de confianza.
4. Pegue la siguiente política JSON en el campo Política de confianza personalizada.

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

```
        "Service": "appconfig.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Elija Siguiente.

5. En la página Asociar política de permisos, seleccione Crear política. La página Create policy (Crear política) se abre en una pestaña nueva.
6. Elija la pestaña JSON y pegue la siguiente política personalizada en el editor. La acción de `lambda:InvokeFunction` se usa para los puntos de acción de `PRE_*`. La acción de `lambda:InvokeAsync` se usa para los puntos de acción de `ON_*`. Sustituya el *ARN de Lambda* por el nombre de recurso de Amazon (ARN) de su Lambda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction",
        "lambda:InvokeAsync"
      ],
      "Resource": "Your Lambda ARN"
    }
  ]
}
```

7. Elija Siguiente: etiquetas.
8. En la página Agregar etiquetas (Opcional), añada uno o más pares clave-valor y, a continuación, elija Siguiente: Revisión.
9. En la página Revisar política, ingrese un nombre y una descripción, y luego elija Crear política.
10. En la pestaña del navegador de su política de confianza personalizada, seleccione el icono Actualizar y, a continuación, busque la política de permisos que acaba de crear.
11. Seleccione la casilla de verificación de la política y, a continuación, elija Siguiente.
12. En la página Nombrar, revisar y crear, ingrese un nombre en el cuadro Nombre del rol y, a continuación, escriba una descripción.

13. Elija Create role. El sistema le devuelve a la página Roles. Elija Ver rol en el banner.
14. Copie el ARN. Este ARN se especifica al crear la extensión.

Crear una extensión personalizada AWS AppConfig

Una extensión define una o más acciones que realiza durante un AWS AppConfig flujo de trabajo. Por ejemplo, la `AWS AppConfig deployment events to Amazon SNS` extensión AWS creada incluye una acción para enviar una notificación a un tema de Amazon SNS. Cada acción se invoca cuando interactúas con un proceso AWS AppConfig o cuando AWS AppConfig lo realizas en tu nombre. Se denominan puntos de acción. AWS AppConfig las extensiones admiten los siguientes puntos de acción:

- `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`
- `PRE_START_DEPLOYMENT`
- `ON_DEPLOYMENT_START`
- `ON_DEPLOYMENT_STEP`
- `ON_DEPLOYMENT_BAKING`
- `ON_DEPLOYMENT_COMPLETE`
- `ON_DEPLOYMENT_ROLLED_BACK`

Las acciones de extensión configuradas en los puntos de `PRE_*` acción se aplican después de la validación de la solicitud, pero antes de AWS AppConfig realizar la actividad correspondiente al nombre del punto de acción. Estas invocaciones de acciones se procesan al mismo tiempo que una solicitud. Si se realiza más de una solicitud, las invocaciones a las acciones se ejecutan de forma secuencial. Tenga en cuenta también que los puntos de acción de `PRE_*` reciben y pueden cambiar el contenido de una configuración. Los puntos de acción `PRE_*` también pueden responder a un error e impedir que se lleve a cabo una acción.

Una extensión también se puede ejecutar en paralelo con un AWS AppConfig flujo de trabajo mediante un punto de `ON_*` acción. `ON_*`los puntos de acción se invocan de forma asíncrona. `ON_*`los puntos de acción no reciben el contenido de una configuración. Si una extensión experimenta un error durante un punto de acción de `ON_*`, el servicio ignora el error y continúa con el flujo de trabajo.

El siguiente ejemplo de extensión define una acción que llama al punto de acción de `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`. En el campo `Uri`, la acción especifica el nombre de recurso de Amazon (ARN) de la función de Lambda de `MyS3ConfigurationBackUpExtension` creada anteriormente en este tutorial. La acción también especifica el ARN del rol de asunción AWS Identity and Access Management (IAM) creado anteriormente en este tutorial.

Ejemplo de extensión AWS AppConfig

```
{
    "Name": "MySampleExtension",
    "Description": "A sample extension that backs up configurations to an S3 bucket.",
    "Actions": {
        "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
            {
                "Name": "PreCreateHostedConfigVersionActionForS3Backup",
                "Uri": "arn:aws:lambda:aws-
region:111122223333:function:MyS3ConfigurationBackUpExtension",
                "RoleArn": "arn:aws:iam::111122223333:role/ExtensionsTestRole"
            }
        ]
    },
    "Parameters" : {
        "S3_BUCKET": {
            "Required": false
        }
    }
}
```

> **Note**
>
> Para ver la sintaxis de las solicitudes y las descripciones de los campos al crear una extensión, consulta el CreateExtensiontema en la Referencia de la AWS AppConfig API.

Crear una extensión (consola)

1. Abre la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.

3. En la pestaña Extensiones, elija Crear extensión.
4. En Nombre de extensión, escriba un nombre único. Para este tutorial, introduzca **`MyS3ConfigurationBackUpExtension`**. Si lo desea, introduzca una descripción.
5. En la sección Acciones, elija Añadir nueva acción.
6. En Nombre de acción, escriba un nombre único. Para este tutorial, introduzca **`PreCreateHostedConfigVersionActionForS3Backup`**. Este nombre describe el punto de acción utilizado por la acción y el propósito de la extensión.
7. En la lista Punto de acción, elija PRE_CREATE_HOSTED_CONFIGURATION_VERSION.
8. Para Uri, elija Función de Lambda y, a continuación, elija la función en la lista Función de Lambda. Si no ve la función, compruebe que se encuentra en el mismo Región de AWS lugar donde la creó.
9. Para el rol de IAM, elija el rol que creó anteriormente en este tutorial.
10. En la sección Parámetros de extensión (opcional), elija Agregar nuevo parámetro.
11. En Nombre de parámetro, introduzca un nombre. Para este tutorial, introduzca **`S3_BUCKET`**.
12. Repita los pasos del 5 al 11 para crear una segunda acción para el punto de acción de `PRE_START_DEPLOYMENT`.
13. Elija Crear extensión.

Personalización de las extensiones AWS de notificación creadas

No es necesario crear una Lambda o una extensión para utilizar las extensiones de notificación creadas de AWS. Solo hay que crear una asociación de extensión y, a continuación, realizar una operación que llame a uno de los puntos de acción admitidos. De forma predeterminada, las extensiones de notificación AWS creadas admiten los siguientes puntos de acción:

- `ON_DEPLOYMENT_START`
- `ON_DEPLOYMENT_COMPLETE`
- `ON_DEPLOYMENT_ROLLED_BACK`

Si crea versiones personalizadas de la extensión `AWS AppConfig deployment events to Amazon SNS` y extensiones de `AWS AppConfig deployment events to Amazon SQS`, puede especificar los puntos de acción de los que desea recibir notificaciones.

> **Note**
>
> La extensión `AWS AppConfig deployment events to EventBridge` no admite los puntos de acción `PRE_*`. Puede crear una versión personalizada si desea eliminar algunos de los puntos de acción predeterminados asignados a la versión AWS creada.

No necesita crear una función de Lambda si crea versiones personalizadas de las extensiones de notificación creadas de AWS . Solo necesita especificar un nombre de recurso de Amazon (ARN) en el campo `Uri` para la nueva versión de la extensión.

- Para una extensión de EventBridge notificación personalizada, introduzca el ARN de los eventos EventBridge predeterminados en el `Uri` campo.
- Para una extensión de notificación de Amazon SNS personalizada, introduzca el ARN de un tema de Amazon SNS en el campo `Uri`.
- Para una extensión de notificación de Amazon SQS personalizada, introduzca el ARN de una cola de mensajes de Amazon SQS en el campo `Uri`.

Crear una asociación de extensiones para una extensión personalizada AWS AppConfig

Para crear una extensión o configurar una extensión AWS creada, debe definir los puntos de acción que invocan una extensión cuando se utiliza un AWS AppConfig recurso específico. Por ejemplo, puede optar por ejecutar la extensión de `AWS AppConfig deployment events to Amazon SNS` y recibir notificaciones sobre un tema de Amazon SNS cada vez que se inicie una implementación de configuración para una aplicación específica. Definir qué puntos de acción invocan una extensión para un AWS AppConfig recurso específico se denomina asociación de extensiones. Una asociación de extensiones es una relación especificada entre una extensión y un AWS AppConfig recurso, como una aplicación o un perfil de configuración.

Una sola AWS AppConfig aplicación puede incluir varios entornos y perfiles de configuración. Si asocia una extensión a una aplicación o un entorno, AWS AppConfig invoca la extensión para cualquier flujo de trabajo relacionado con los recursos de la aplicación o el entorno, si corresponde.

Por ejemplo, supongamos que tiene una AWS AppConfig aplicación llamada MobileApps que incluye un perfil de configuración llamado AccessList. Supongamos que la MobileApps aplicación incluye entornos beta, de integración y de producción. Debe crear una asociación de extensión para la

extensión AWS de notificación de Amazon SNS creada y asociar la extensión a MobileApps la aplicación. La extensión de notificación de Amazon SNS se invoca cada vez que se implementa la configuración de la aplicación en cualquiera de los tres entornos.

Utilice los siguientes procedimientos para crear una asociación AWS AppConfig de extensiones mediante la AWS AppConfig consola.

Para crear una extensión de asociación (consola)

1. Abra la AWS Systems Manager consola en https://console.aws.amazon.com/systems-manager/appconfig/.
2. En el panel de navegación, elija AWS AppConfig.
3. En la pestaña Extensiones, elija un botón de opción para una extensión y, a continuación, seleccione Añadir al recurso. Para los fines de este tutorial, elija ConfigurationBackUpExtensionmyS3.
4. En la sección de detalles del recurso de la extensión, en Tipo de recurso, elija un tipo de AWS AppConfig recurso. Según el recurso que elija, AWS AppConfig le solicitará que elija otros recursos. Para este tutorial, elija Aplicación.
5. Elija su aplicación en la lista.
6. En la sección Parámetros, compruebe que S3_BUCKET aparezca en el campo Clave. En el campo Valor, pegue el ARN de las extensiones de Lambda. Por ejemplo: `arn:aws:lambda:`*`aws-region`*`:111122223333:function:MyS3ConfigurationBackUpExtension`.
7. Elija Crear una asociación al recurso.

Realizar una acción que invoque una extensión personalizada AWS AppConfig

Tras crear la asociación, puede invocar la extensión `MyS3ConfigurationBackUpExtension` creando un nuevo perfil de configuración que especifique `hosted` para su `SourceUri`. Como parte del flujo de trabajo para crear la nueva configuración, AWS AppConfig encuentra el punto de `PRE_CREATE_HOSTED_CONFIGURATION_VERSION` acción. Al encontrar este punto de acción, se invoca la extensión `MyS3ConfigurationBackUpExtension`, que automáticamente hace una copia de seguridad de la configuración que se acaba de crear en el bucket de S3 especificado en la sección `Parameter` de la asociación de extensión.

AWS AppConfig extensión: integración con Atlassian Jira

AWS AppConfig se integra con Atlassian Jira. La integración permite AWS AppConfig crear y actualizar problemas en la consola de Atlassian cada vez que realices cambios en una característica de tu marca para cumplir con las especificadas. Cuenta de AWS Región de AWS Cada problema de Jira incluye el nombre de la marca, el ID de la aplicación, el ID del perfil de configuración y los valores de la marca. Tras actualizar, guardar e implementar los cambios de las marcas, Jira actualiza los problemas existentes con los detalles del cambio. Para obtener más información, consulte Trabajando con la extensión Jira de Atlassian para AWS AppConfig.

Ejemplos de código de AWS AppConfig

Esta sección incluye ejemplos de código para realizar acciones comunes AWS AppConfig mediante programación. Le recomendamos que utilice estos ejemplos con Java, Python y JavaScriptlos SDK para realizar las acciones en un entorno de prueba. En esta sección se incluye un ejemplo de código para limpiar el entorno de prueba una vez que haya terminado.

Temas

Crear o actualizar una configuración de formato libre almacenada en el almacén de configuraciones hospedado

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones que realiza el código. Los ejemplos de esta sección llaman a las siguientes API:

- CreateApplication
- CreateConfigurationProfile
- CreateHostedConfigurationVersion

Java

```
public CreateHostedConfigurationVersionResponse createHostedConfigVersion() {
        AppConfigClient appconfig = AppConfigClient.create();

        // Create an application
```

```
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a hosted, freeform configuration profile
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("hosted")
            .type("AWS.Freeform"));

        // Create a hosted configuration version
        CreateHostedConfigurationVersionResponse hcv =
 appconfig.createHostedConfigurationVersion(req -> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .contentType("text/plain; charset=utf-8")
            .content(SdkBytes.fromUtf8String("my config data")));

        return hcv;
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')

# create a hosted, freeform configuration profile
config_profile = appconfig.create_configuration_profile(
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='hosted',
    Type='AWS.Freeform')

# create a hosted configuration version
hcv = appconfig.create_hosted_configuration_version(
    ApplicationId=application['Id'],
    ConfigurationProfileId=config_profile['Id'],
    Content=b'my config data',
```

```
    ContentType='text/plain')
```

JavaScript

```
import {
  AppConfigClient,
  CreateApplicationCommand,
  CreateConfigurationProfileCommand,
  CreateHostedConfigurationVersionCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
);

// create a hosted, freeform configuration profile
const profile = await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "hosted",
    Type: "AWS.Freeform",
  })
);

// create a hosted configuration version
await appconfig.send(
  new CreateHostedConfigurationVersionCommand({
    ApplicationId: application.Id,
    ConfigurationProfileId: profile.Id,
    ContentType: "text/plain",
    Content: "my config data",
  })
);
```

Crear un perfil de configuración para un secreto almacenado en Secrets Manager

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones realizadas por el código. Los ejemplos de esta sección llaman a las siguientes API:

- CreateApplication
- CreateConfigurationProfile

Java

```
private void createSecretsManagerConfigProfile() {
        AppConfigClient appconfig = AppConfigClient.create();

        // Create an application
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a configuration profile for Secrets Manager Secret
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("secretsmanager://MySecret")
            .retrievalRoleArn("arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret")
            .type("AWS.Freeform"));
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')

# create a configuration profile for Secrets Manager Secret
config_profile = appconfig.create_configuration_profile(
```

```
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='secretsmanager://MySecret',
    RetrievalRoleArn='arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret',
    Type='AWS.Freeform')
```

JavaScript

```
import {
  AppConfigClient,
  CreateConfigurationProfileCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
);

// create a configuration profile for Secrets Manager Secret
await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "secretsmanager://MySecret",
    RetrievalRoleArn: "arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret",
    Type: "AWS.Freeform",
  })
);
```

Implementación de un perfil de configuración

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones que realiza el código. Los ejemplos de esta sección llaman a las siguientes API:

- CreateApplication
- CreateConfigurationProfile
- CreateHostedConfigurationVersion

- CreateEnvironment
- StartDeployment
- GetDeployment

Java

```
private void createDeployment() throws InterruptedException {
        AppConfigClient appconfig = AppConfigClient.create();

        // Create an application
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a hosted, freeform configuration profile
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("hosted")
            .type("AWS.Freeform"));

        // Create a hosted configuration version
        CreateHostedConfigurationVersionResponse hcv =
 appconfig.createHostedConfigurationVersion(req -> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .contentType("text/plain; charset=utf-8")
            .content(SdkBytes.fromUtf8String("my config data")));


        // Create an environment
        CreateEnvironmentResponse env = appconfig.createEnvironment(req -> req
            .applicationId(app.id())
            .name("Beta")
            // If you have CloudWatch alarms that monitor the health of your
 service, you can add them here and they
            // will trigger a rollback if they fire during an appconfig deployment
            //.monitors(Monitor.builder().alarmArn("arn:aws:cloudwatch:us-
east-1:520900602629:alarm:MyAlarm")
            //
  .alarmRoleArn("arn:aws:iam::520900602629:role/MyAppConfigAlarmRole").build())
        );
```

```
        // Start a deployment
        StartDeploymentResponse deploymentResponse = appconfig.startDeployment(req -
> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .environmentId(env.id())
            .configurationVersion(hcv.versionNumber().toString())
            .deploymentStrategyId("AppConfig.Linear50PercentEvery30Seconds")
        );

        // Wait for deployment to complete
        List<DeploymentState> nonFinalDeploymentStates = Arrays.asList(
            DeploymentState.DEPLOYING,
            DeploymentState.BAKING,
            DeploymentState.ROLLING_BACK,
            DeploymentState.VALIDATING);
        GetDeploymentRequest getDeploymentRequest =
 GetDeploymentRequest.builder().applicationId(app.id())

 .environmentId(env.id())

 .deploymentNumber(deploymentResponse.deploymentNumber()).build();
        GetDeploymentResponse deployment =
 appconfig.getDeployment(getDeploymentRequest);
        while (nonFinalDeploymentStates.contains(deployment.state())) {
            System.out.println("Waiting for deployment to complete: " + deployment);
            Thread.sleep(1000L);
            deployment = appconfig.getDeployment(getDeploymentRequest);
        }

        System.out.println("Deployment complete: " + deployment);
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')
```

```
# create an environment
environment = appconfig.create_environment(
    ApplicationId=application['Id'],
    Name='MyEnvironment')

# create a configuration profile
config_profile = appconfig.create_configuration_profile(
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='hosted',
    Type='AWS.Freeform')

# create a hosted configuration version
hcv = appconfig.create_hosted_configuration_version(
    ApplicationId=application['Id'],
    ConfigurationProfileId=config_profile['Id'],
    Content=b'my config data',
    ContentType='text/plain')

# start a deployment
deployment = appconfig.start_deployment(
    ApplicationId=application['Id'],
    EnvironmentId=environment['Id'],
    ConfigurationProfileId=config_profile['Id'],
    ConfigurationVersion=str(hcv['VersionNumber']),
    DeploymentStrategyId='AppConfig.Linear20PercentEvery6Minutes')
```

JavaScript

```
import {
  AppConfigClient,
  CreateApplicationCommand,
  CreateEnvironmentCommand,
  CreateConfigurationProfileCommand,
  CreateHostedConfigurationVersionCommand,
  StartDeploymentCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
```

```
);

// create an environment
const environment = await appconfig.send(
  new CreateEnvironmentCommand({
    ApplicationId: application.Id,
    Name: "MyEnvironment",
  })
);

// create a configuration profile
const config_profile = await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "hosted",
    Type: "AWS.Freeform",
  })
);

// create a hosted configuration version
const hcv = await appconfig.send(
  new CreateHostedConfigurationVersionCommand({
    ApplicationId: application.Id,
    ConfigurationProfileId: config_profile.Id,
    Content: "my config data",
    ContentType: "text/plain",
  })
);

// start a deployment
await appconfig.send(
  new StartDeploymentCommand({
    ApplicationId: application.Id,
    EnvironmentId: environment.Id,
    ConfigurationProfileId: config_profile.Id,
    ConfigurationVersion: hcv.VersionNumber.toString(),
    DeploymentStrategyId: "AppConfig.Linear20PercentEvery6Minutes",
  })
);
```

Uso AWS AppConfig del agente para leer un perfil de configuración de formato libre

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones que realiza el código.

Java

```
public void retrieveConfigFromAgent() throws Exception {
        /*
        In this sample, we will retrieve configuration data from the AWS AppConfig
 Agent.
        The agent is a sidecar process that handles retrieving configuration data
 from AppConfig
        for you in a way that implements best practices like configuration caching.

        For more information about the agent, see Simplified retrieval methods
        */

        // The agent runs a local HTTP server that serves configuration data
        // Make a GET request to the agent's local server to retrieve the
 configuration data
        URL url = new URL("http://localhost:2772/applications/MyDemoApp/
environments/Beta/configurations/MyConfigProfile");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod("GET");
        StringBuilder content;
        try (BufferedReader in = new BufferedReader(new
 InputStreamReader(con.getInputStream()))) {
            content = new StringBuilder();
            int ch;
            while ((ch = in.read()) != -1) {
                content.append((char) ch);
            }
        }
        con.disconnect();
        System.out.println("Configuration from agent via HTTP: " + content);
    }
```

Python

```
# in this sample, we will retrieve configuration data from the AWS AppConfig Agent.
```

```
# the agent is a sidecar process that handles retrieving configuration data from AWS
 AppConfig
# for you in a way that implements best practices like configuration caching.
#
# for more information about the agent, see
# Simplified retrieval methods
#

import requests

application_name = 'MyDemoApp'
environment_name = 'MyEnvironment'
config_profile_name = 'MyConfigProfile'

# the agent runs a local HTTP server that serves configuration data
# make a GET request to the agent's local server to retrieve the configuration data
response = requests.get(f"http://localhost:2772/applications/{application_name}/
environments/{environment_name}/configurations/{config_profile_name}")
config = response.content
```

JavaScript

```
// in this sample, we will retrieve configuration data from the AWS AppConfig Agent.
// the agent is a sidecar process that handles retrieving configuration data from
 AppConfig
// for you in a way that implements best practices like configuration caching.

// for more information about the agent, see
// Simplified retrieval methods

const application_name = "MyDemoApp";
const environment_name = "MyEnvironment";
const config_profile_name = "MyConfigProfile";

// the agent runs a local HTTP server that serves configuration data
// make a GET request to the agent's local server to retrieve the configuration data
const url = `http://localhost:2772/applications/${application_name}/environments/
${environment_name}/configurations/${config_profile_name}`;
const response = await fetch(url);
const config = await response.text(); // (use `await response.json()` if your config
 is json)
```

Uso AWS AppConfig del agente para leer un indicador de función específico

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones realizadas por el código.

Java

```
public void retrieveSingleFlagFromAgent() throws Exception {
        /*
          You can retrieve a single flag's data from the agent by providing the
 "flag" query string parameter.
          Note: the configuration's type must be AWS.AppConfig.FeatureFlags
        */

        URL url = new URL("http://localhost:2772/applications/MyDemoApp/
environments/Beta/configurations/MyFlagsProfile?flag=myFlagKey");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod("GET");
        StringBuilder content;
        try (BufferedReader in = new BufferedReader(new
 InputStreamReader(con.getInputStream()))) {
            content = new StringBuilder();
            int ch;
            while ((ch = in.read()) != -1) {
                content.append((char) ch);
            }
        }
        con.disconnect();
        System.out.println("MyFlagName from agent: " + content);
    }
```

Python

```
import requests

application_name = 'MyDemoApp'
environment_name = 'MyEnvironment'
config_profile_name = 'MyConfigProfile'
flag_key = 'MyFlag'

# retrieve a single flag's data by providing the "flag" query string parameter
```

```
# note: the configuration's type must be AWS.AppConfig.FeatureFlags
response = requests.get(f"http://localhost:2772/applications/{application_name}/
environments/{environment_name}/configurations/{config_profile_name}?
flag={flag_key}")
config = response.content
```

JavaScript

```
const application_name = "MyDemoApp";
const environment_name = "MyEnvironment";
const config_profile_name = "MyConfigProfile";
const flag_name = "MyFlag";

// retrieve a single flag's data by providing the "flag" query string parameter
// note: the configuration's type must be AWS.AppConfig.FeatureFlags
const url = `http://localhost:2772/applications/${application_name}/environments/
${environment_name}/configurations/${config_profile_name}?flag=${flag_name}`;
const response = await fetch(url);
const flag = await response.json(); // { "enabled": true/false }
```

Uso de la acción GetLatestConfig de la API para leer un perfil de configuración de formato libre

Cada uno de los siguientes ejemplos incluye comentarios sobre las acciones que realiza el código. Los ejemplos de esta sección llaman a las siguientes API:

- GetLatestConfiguration
- StartConfigurationSession

Java

```
public void retrieveConfigFromApi() {
        /*
        The example below uses two AppConfigData APIs: StartConfigurationSession and
 GetLatestConfiguration.
        For more information on these APIs, see AWS AppConfig Data         */
        AppConfigDataClient appConfigData = AppConfigDataClient.create();

        /*
```

```
        Start a new configuration session using the StartConfigurationSession API.
This operation does not return configuration data.
        Rather, it returns an initial configuration token that should be passed to
GetLatestConfiguration.
        IMPORTANT: This operation should only be performed once (per configuration),
prior to the first GetLatestConfiguration
        call you preform. Each GetLatestConfiguration will return a new
configuration token that you should then use in the
        next GetLatestConfiguration call.
        */
        StartConfigurationSessionResponse session =
            appConfigData.startConfigurationSession(req -> req
                .applicationIdentifier("MyDemoApp")
                .configurationProfileIdentifier("MyConfigProfile")
                .environmentIdentifier("Beta"));

        /*
        Retrieve configuration data using the GetLatestConfiguration API. The first
time you call this API your configuration
        data will be returned. You should cache that data (and the configuration
token) and update that cache asynchronously
        by regularly polling the GetLatestConfiguration API in a background thread.
If you already have the latest configuration
        data, subsequent GetLatestConfiguration calls will return an empty response.
If you then deploy updated configuration
        data the next time you call GetLatestConfiguration it will return that
updated data.

        You can also avoid all the complexity around writing this code yourself by
leveraging our agent instead.
        For more information about the agent, see Simplified retrieval methods
        */

        // The first getLatestConfiguration call uses the token from
StartConfigurationSession
        String configurationToken = session.initialConfigurationToken();
        GetLatestConfigurationResponse configuration =

appConfigData.getLatestConfiguration(GetLatestConfigurationRequest.builder().configurationT

        System.out.println("Configuration retrieved via API: " +
configuration.configuration().asUtf8String());
```

```
        // You'll want to hold on to the token in the getLatestConfiguration
 response because you'll need to use it
        // the next time you call
        configurationToken = configuration.nextPollConfigurationToken();
        configuration =

 appConfigData.getLatestConfiguration(GetLatestConfigurationRequest.builder().configurationT

        // Try creating a new deployment at this point to see how the output below
 changes.
        if (configuration.configuration().asByteArray().length != 0) {
            System.out.println("Configuration contents have changed
 since the last GetLatestConfiguration call, new contents = " +
 configuration.configuration().asUtf8String());
        } else {
            System.out.println("GetLatestConfiguration returned an empty response
 because we already have the latest configuration");
        }
    }
```

Python

```
# the example below uses two AppConfigData APIs: StartConfigurationSession and
 GetLatestConfiguration.
#
# for more information on these APIs, see
# AWS AppConfig Data
#

import boto3

application_name = 'MyDemoApp'
environment_name = 'MyEnvironment'
config_profile_name = 'MyConfigProfile'

appconfigdata = boto3.client('appconfigdata')

# start a new configuration session.
# this operation does not return configuration data.
# rather, it returns an initial configuration token that should be passed to
 GetLatestConfiguration.
#
# note: this operation should only be performed once (per configuration).
```

```
#   all subsequent calls to AppConfigData should be via GetLatestConfiguration.
scs = appconfigdata.start_configuration_session(
    ApplicationIdentifier=application_name,
    EnvironmentIdentifier=environment_name,
    ConfigurationProfileIdentifier=config_profile_name)
initial_token = scs['InitialConfigurationToken']

# retrieve configuration data from the session.
# this operation returns your configuration data.
# each invocation of this operation returns a unique token that should be passed to
 the subsequent invocation.
#
# note: this operation does not always return configuration data after the first
 invocation.
#   data is only returned if the configuration has changed within AWS AppConfig
 (i.e. a deployment occurred).
#   therefore, you should cache the data returned by this call so that you can use
 it later.
glc = appconfigdata.get_latest_configuration(ConfigurationToken=initial_token)
config = glc['Configuration'].read()
```

JavaScript

```
// the example below uses two AppConfigData APIs: StartConfigurationSession and
 GetLatestConfiguration.

// for more information on these APIs, see
// AWS AppConfig Data

import {
  AppConfigDataClient,
  GetLatestConfigurationCommand,
  StartConfigurationSessionCommand,
} from "@aws-sdk/client-appconfigdata";

const appconfigdata = new AppConfigDataClient();

const application_name = "MyDemoApp";
const environment_name = "MyEnvironment";
const config_profile_name = "MyConfigProfile";

// start a new configuration session.
// this operation does not return configuration data.
```

```
// rather, it returns an initial configuration token that should be passed to
 GetLatestConfiguration.
//
// note: this operation should only be performed once (per configuration).
//   all subsequent calls to AppConfigData should be via GetLatestConfiguration.
const scs = await appconfigdata.send(
  new StartConfigurationSessionCommand({
    ApplicationIdentifier: application_name,
    EnvironmentIdentifier: environment_name,
    ConfigurationProfileIdentifier: config_profile_name,
  })
);
const { InitialConfigurationToken } = scs;

// retrieve configuration data from the session.
// this operation returns your configuration data.
// each invocation of this operation returns a unique token that should be passed to
 the subsequent invocation.
//
// note: this operation does not always return configuration data after the first
 invocation.
//   data is only returned if the configuration has changed within AWS AppConfig
 (i.e. a deployment occurred).
//   therefore, you should cache the data returned by this call so that you can use
 it later.
const glc = await appconfigdata.send(
  new GetLatestConfigurationCommand({
    ConfigurationToken: InitialConfigurationToken,
  })
);
const config = glc.Configuration.transformToString();
```

Limpiar su entorno

Si ejecutó uno o más de los ejemplos de código de esta sección, le recomendamos que utilice uno de los ejemplos siguientes para localizar y eliminar los AWS AppConfig recursos creados por esos ejemplos de código. Los ejemplos de esta sección llaman a las siguientes API:

- ListApplications
- DeleteApplication
- ListEnvironments

- DeleteEnvironments
- ListConfigurationProfiles
- DeleteConfigurationProfile
- ListHostedConfigurationVersions
- DeleteHostedConfigurationVersion

Java

```
/*
    This sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.

   WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
   an application that you may need in the future.
   */

   public void cleanUpDemoResources() {
       AppConfigClient appconfig = AppConfigClient.create();

       // The name of the application to delete
       // IMPORTANT: verify this name corresponds to the application you wish to
 delete
       String applicationToDelete = "MyDemoApp";


 appconfig.listApplicationsPaginator(ListApplicationsRequest.builder().build()).items().forE
 -> {
           if (app.name().equals(applicationToDelete)) {
               System.out.println("Deleting App: " + app);
               appconfig.listConfigurationProfilesPaginator(req ->
 req.applicationId(app.id())).items().forEach(cp -> {
                   System.out.println("Deleting Profile: " + cp);
                   appconfig
                       .listHostedConfigurationVersionsPaginator(req -> req
                           .applicationId(app.id())
                           .configurationProfileId(cp.id()))
                       .items()
                       .forEach(hcv -> {
```

```
                            System.out.println("Deleting HCV: " + hcv);
                            appconfig.deleteHostedConfigurationVersion(req -> req
                                .applicationId(app.id())
                                .configurationProfileId(cp.id())
                                .versionNumber(hcv.versionNumber()));
                        });
                    appconfig.deleteConfigurationProfile(req -> req
                        .applicationId(app.id())
                        .configurationProfileId(cp.id()));
                });

                appconfig.listEnvironmentsPaginator(req-
>req.applicationId(app.id())).items().forEach(env -> {
                    System.out.println("Deleting Environment: " + env);
                    appconfig.deleteEnvironment(req-
>req.applicationId(app.id()).environmentId(env.id()));
                });

                appconfig.deleteApplication(req -> req.applicationId(app.id()));
            }
        });
    }
```

Python

```
# this sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.
#
# WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
#   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
#   an application that you may need in the future.
#

import boto3

# the name of the application to delete
# IMPORTANT: verify this name corresponds to the application you wish to delete
application_name = 'MyDemoApp'

# create and iterate over a list paginator such that we end up with a list of pages,
 which are themselves lists of applications
```

```
# e.g. [ [{'Name':'MyApp1',...},{'Name':'MyApp2',...}], [{'Name':'MyApp3',...}] ]
list_of_app_lists = [page['Items'] for page in
 appconfig.get_paginator('list_applications').paginate()]
# retrieve the target application from the list of lists
application = [app for apps in list_of_app_lists for app in apps if app['Name'] ==
 application_name][0]
print(f"deleting application {application['Name']} (id={application['Id']})")

# delete all configuration profiles
list_of_config_lists = [page['Items'] for page in
 appconfig.get_paginator('list_configuration_profiles').paginate(ApplicationId=application['
for config_profile in [config for configs in list_of_config_lists for config in
 configs]:
    print(f"\tdeleting configuration profile {config_profile['Name']}
 (Id={config_profile['Id']})")

    # delete all hosted configuration versions
    list_of_hcv_lists = [page['Items'] for page in
 appconfig.get_paginator('list_hosted_configuration_versions').paginate(ApplicationId=applic
 ConfigurationProfileId=config_profile['Id'])]
    for hcv in [hcv for hcvs in list_of_hcv_lists for hcv in hcvs]:

 appconfig.delete_hosted_configuration_version(ApplicationId=application['Id'],
 ConfigurationProfileId=config_profile['Id'], VersionNumber=hcv['VersionNumber'])
        print(f"\t\tdeleted hosted configuration version {hcv['VersionNumber']}")

    # delete the config profile itself
    appconfig.delete_configuration_profile(ApplicationId=application['Id'],
 ConfigurationProfileId=config_profile['Id'])
    print(f"\tdeleted configuration profile {config_profile['Name']}
 (Id={config_profile['Id']})")

# delete all environments
list_of_env_lists = [page['Items'] for page in
 appconfig.get_paginator('list_environments').paginate(ApplicationId=application['Id'])]
for environment in [env for envs in list_of_env_lists for env in envs]:
    appconfig.delete_environment(ApplicationId=application['Id'],
 EnvironmentId=environment['Id'])
    print(f"\tdeleted environment {environment['Name']} (Id={environment['Id']})")

# delete the application itself
appconfig.delete_application(ApplicationId=application['Id'])
print(f"deleted application {application['Name']} (id={application['Id']})")
```

JavaScript

```
// this sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.

// WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
//   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
//   an application that you may need in the future.

import {
  AppConfigClient,
  paginateListApplications,
  DeleteApplicationCommand,
  paginateListConfigurationProfiles,
  DeleteConfigurationProfileCommand,
  paginateListHostedConfigurationVersions,
  DeleteHostedConfigurationVersionCommand,
  paginateListEnvironments,
  DeleteEnvironmentCommand,
} from "@aws-sdk/client-appconfig";

const client = new AppConfigClient();

// the name of the application to delete
// IMPORTANT: verify this name corresponds to the application you wish to delete
const application_name = "MyDemoApp";

// iterate over all applications, deleting ones that have the name defined above
for await (const app_page of paginateListApplications({ client }, {})) {
  for (const application of app_page.Items) {

    // skip applications that dont have the name thats set
    if (application.Name !== application_name) continue;

    console.log( `deleting application ${application.Name} (id=${application.Id})`);

    // delete all configuration profiles
    for await (const config_page of paginateListConfigurationProfiles({ client },
 { ApplicationId: application.Id })) {
      for (const config_profile of config_page.Items) {
        console.log(`\tdeleting configuration profile ${config_profile.Name} (Id=
${config_profile.Id})`);
```

```
        // delete all hosted configuration versions
        for await (const hosted_page of
 paginateListHostedConfigurationVersions({ client },
          { ApplicationId: application.Id, ConfigurationProfileId:
 config_profile.Id }
        )) {
          for (const hosted_config_version of hosted_page.Items) {
            await client.send(
              new DeleteHostedConfigurationVersionCommand({
                ApplicationId: application.Id,
                ConfigurationProfileId: config_profile.Id,
                VersionNumber: hosted_config_version.VersionNumber,
              })
            );
            console.log(`\t\tdeleted hosted configuration version
 ${hosted_config_version.VersionNumber}`);
          }
        }

        // delete the config profile itself
        await client.send(
          new DeleteConfigurationProfileCommand({
            ApplicationId: application.Id,
            ConfigurationProfileId: config_profile.Id,
          })
        );
        console.log(`\tdeleted configuration profile ${config_profile.Name} (Id=
${config_profile.Id})`)
      }

      // delete all environments
      for await (const env_page of paginateListEnvironments({ client },
 { ApplicationId: application.Id })) {
        for (const environment of env_page.Items) {
          await client.send(
            new DeleteEnvironmentCommand({
              ApplicationId: application.Id,
              EnvironmentId: environment.Id,
            })
          );
          console.log(`\tdeleted environment ${environment.Name} (Id=
${environment.Id})`)
        }
```

```
      }
    }

    // delete the application itself
    await client.send(
      new DeleteApplicationCommand({ ApplicationId: application.Id })
    );
    console.log(`deleted application ${application.Name} (id=${application.Id})`)
  }
}
```

Seguridad en AWS AppConfig

La seguridad en la nube de AWS es la mayor prioridad. Como cliente de AWS, se beneficiará de una arquitectura de red y un centro de datos que están 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 servicios de AWS en la Nube de AWS. AWS también le proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los Programas de conformidad de AWS . Para obtener información sobre los programas de conformidad que se aplican a AWS Systems Manager, 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 la normativa aplicables.

AWS AppConfig es una capacidad de AWS Systems Manager. Para comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza AWS AppConfig, consulte Seguridad en AWS Systems Manager. En esa sección se describe cómo configurar Systems Manager para cumplir los objetivos de seguridad y conformidad de AWS AppConfig.

Implementación del acceso a los privilegios mínimos

Como práctica recomendada de seguridad, conceda los permisos mínimos requeridos que las identidades requieren para realizar acciones específicas en recursos específicos en condiciones específicas. AWS AppConfig El agente ofrece dos funciones que permiten al agente acceder al sistema de archivos de una instancia o contenedor: realizar copias de seguridad y escribir en disco. Si habilita estas funciones, compruebe que solo el AWS AppConfig agente tiene permisos para escribir en los archivos de configuración designados del sistema de archivos. Compruebe también que solo los procesos necesarios para leer estos archivos de configuración tengan la capacidad de hacerlo. La implementación del acceso con privilegios mínimos es esencial a la hora de reducir los riesgos de seguridad y el impacto que podrían causar los errores o los intentos malintencionados.

Para obtener más información sobre la implementación del acceso con privilegios mínimos, consulte SEC03-BP02 Conceder acceso con privilegios mínimos en la Guía del usuario. AWS Well-Architected Tool Para obtener más información sobre las funciones del AWS AppConfig agente mencionadas en esta sección, consulte. Funciones de recuperación adicionales

Cifrado de datos en reposo en AWS AppConfig

AWS AppConfig proporciona cifrado de forma predeterminada para proteger los datos en reposo de los clientes utilizando Claves propiedad de AWS.

Claves propiedad de AWS: AWS AppConfig utiliza estas claves de forma predeterminada para cifrar automáticamente los datos desplegados por el servicio y alojados en el almacén de datos de AWS AppConfig. No puede ver, administrar o usar las Claves propiedad de AWS, ni auditar su uso. Sin embargo, no tiene que realizar ninguna acción ni cambiar ningún programa para proteger las claves que cifran sus datos. Para obtener más información, consulte Claves propiedad de AWS en la Guía para desarrolladores de AWS Key Management Service.

Si bien el usuario no puede deshabilitar esta capa de cifrado ni seleccionar otro tipo de cifrado, puede especificar una clave administrada por el cliente para usarla al guardar los datos de configuración alojados en el almacén de datos de AWS AppConfig y al implementar los datos de configuración.

Claves administradas por el cliente: AWS AppConfig admite el uso de una clave simétrica administrada por el cliente que usted crea, posee y administra para añadir una segunda capa de cifrado sobre la Clave propiedad de AWS existente. Como usted tiene el control total de esta capa de cifrado, puede realizar tareas como las siguientes:

- Establecer y mantener concesiones y políticas de claves
- Establecer y mantener concesiones y políticas de IAM
- Habilitar y deshabilitar políticas de claves
- Rotar el material criptográfico
- Agregar etiquetas.
- Crear alias de clave
- Programar la eliminación de claves

Para obtener más información, consulte las claves administradas por el cliente en la Guía para desarrolladores de AWS Key Management Service.

AWS AppConfigCompatibilidad con las claves administradas por el cliente

AWS AppConfig ofrece compatibilidad con el cifrado de claves administrado por el cliente para los datos de configuración. En el caso de las versiones de la configuración guardadas en el almacén de datos alojado de AWS AppConfig, los clientes pueden establecer una `KmsKeyIdentifier` en el perfil de configuración correspondiente. Cada vez que se crea una nueva versión de los datos de configuración mediante la operación de API `CreateHostedConfigurationVersion`, AWS AppConfig genera una clave de datos de AWS KMS a partir del `KmsKeyIdentifier` para cifrar los datos antes de almacenarlos. Cuando se accede a los datos más adelante, ya sea durante las operaciones de API `GetHostedConfigurationVersion` o `StartDeployment`, AWS AppConfig descifra los datos de configuración utilizando la información sobre la clave de datos generada.

AWS AppConfig también ofrece compatibilidad con el cifrado de claves administrado por el cliente para los datos de configuración implementados. Para cifrar los datos de configuración, los clientes pueden proporcionar un `KmsKeyIdentifier` a su implementación. AWS AppConfig genera la clave de datos de AWS KMS con este `KmsKeyIdentifier` para cifrar los datos sobre la operación de API `StartDeployment`.

Acceso de cifrado de AWS AppConfig

Al crear una clave administrada por el cliente, utilice la política de claves que se indica a continuación para asegurarse de que se pueda utilizar la clave.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
        "Sid": "Allow use of the key",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::account_ID:role/role_name"
        },
        "Action": [
            "kms:Decrypt",
            "kms:GenerateDataKey"
            ],
        "Resource": "*"
        }
 ]
```

Para cifrar los datos de configuración alojados con una clave administrada por el cliente, la identidad que llama a `CreateHostedConfigurationVersion` necesita la siguiente declaración de política, que se puede asignar a un usuario, grupo o rol:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey,
            "Resource": "arn:aws:kms:Region:account_ID:key_ID"
        }
    ]
}
```

Si utiliza un secreto de Secrets Manager o cualquier otro dato de configuración cifrado con una clave administrada por el cliente, su `retrievalRoleArn` necesitará `kms:Decrypt` para descifrar y recuperar los datos.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:Region:account_ID:configuration source/object"
    }
  ]
}
```

Al llamar a la operación de la AWS AppConfig StartDeploymentAPI, la llamada de identidad `StartDeployment` necesita la siguiente política de IAM, que se puede asignar a un usuario, grupo o rol:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey*"
```

```
      ],
      "Resource": "arn:aws:kms:Region:account_ID:key_ID"
    }
  ]
}
```

Al llamar a la operación de la AWS AppConfig GetLatestConfigurationAPI, la llamada de identidad `GetLatestConfiguration` necesita la siguiente política, que se puede asignar a un usuario, grupo o rol:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt,
            "Resource": "arn:aws:kms:Region:account_ID:key_ID"
        }
    ]
}
```

Contexto de cifrado

Un contexto de cifrado es un conjunto opcional de pares clave-valor que pueden contener información contextual adicional sobre los datos.

AWS KMS utiliza el contexto de cifrado como datos autenticados adicionales para admitir el cifrado autenticado. Cuando se incluye un contexto de cifrado en una solicitud para cifrar datos, AWS KMS vincula el contexto de cifrado a los datos cifrados. Para descifrar los datos, debe incluir el mismo contexto de cifrado en la solicitud.

Contexto de cifrado de AWS AppConfig: AWS AppConfig utiliza un contexto de cifrado en todas las operaciones criptográficas de AWS KMS para las implementaciones y los datos de configuración alojados cifrados. El contexto contiene una clave correspondiente al tipo de datos y un valor que identifica el elemento de datos específico.

Supervisar las claves de cifrado para AWS

Cuando utilizas claves gestionadas por el AWS KMS clienteAWS AppConfig, puedes utilizar AWS CloudTrail Amazon CloudWatch Logs para realizar un seguimiento de las solicitudes que se AWS AppConfig envían aAWS KMS.

El siguiente ejemplo es un CloudTrail evento `Decrypt` para monitorear AWS KMS las operaciones solicitadas para acceder AWS AppConfig a los datos cifrados por la clave administrada por el cliente:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "appconfig.amazonaws.com"
    },
    "eventTime": "2023-01-03T02:22:28z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "Region",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
            "aws:appconfig:deployment:arn":
 "arn:aws:appconfig:Region:account_ID:application/application_ID/
environment/environment_ID/deployment/deployment_ID"
        },
        "keyId": "arn:aws:kms:Region:account_ID:key/key_ID",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "account_ID",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:Region:account_ID:key_ID"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "account_ID",
    "sharedEventID": "dc129381-1d94-49bd-b522-f56a3482d088"
}
```

Acceso a AWS AppConfig a través de un punto de conexión de la interfaz (AWS PrivateLink)

Puede usar un AWS PrivateLink para crear una conexión privada entre la VPC y AWS AppConfig. Puede acceder a AWS AppConfig como si estuviera en su VPC, sin el uso de una puerta de enlace de Internet, un dispositivo NAT, una conexión VPN o una conexión AWS Direct Connect. Las instancias de la VPC no necesitan direcciones IP públicas para acceder a AWS AppConfig.

Esta conexión privada se establece mediante la creación de un punto de conexión de interfaz alimentado por AWS PrivateLink . Creamos una interfaz de red de punto de conexión en cada subred habilitada para el punto de conexión de interfaz. Se trata de interfaces de red administradas por el solicitante que sirven como punto de entrada para el tráfico destinado a AWS AppConfig.

Para obtener más información, consulte Acceso a Servicios de AWS a través de AWS PrivateLink en la Guía de AWS PrivateLink.

Consideraciones sobre AWS AppConfig

Antes de configurar un punto de conexión de interfaz para AWS AppConfig, consulte Consideraciones en la Guía de AWS PrivateLink.

AWS AppConfigpermite realizar llamadas a los `appconfigdata`servicios `appconfig`y a través del punto final de la interfaz.

Creación de un punto de conexión de interfaz para AWS AppConfig

Puede crear un punto de conexión de interfaz para AWS AppConfig mediante la consola de Amazon VPC o la AWS Command Line Interface (AWS CLI). Para obtener más información, consulte Creación de un punto de conexión de interfaz en la Guía de AWS PrivateLink.

Cree un punto de conexión para AWS AppConfig utilizando los siguientes nombres de servicio:

```
com.amazonaws.region.appconfig
```

```
com.amazonaws.region.appconfigdata
```

Si habilita DNS privado para el punto de conexión de interfaz, puede realizar solicitudes a la API para AWS AppConfig usando su nombre de DNS predeterminado para la región. Por ejemplo, `appconfig.us-east-1.amazonaws.com` y `appconfigdata.us-east-1.amazonaws.com`.

Creación de una política de puntos de conexión para el punto de conexión de interfaz

Una política de puntos de conexión es un recurso de IAM que puede adjuntar al punto de conexión de su interfaz. La política de puntos de conexión predeterminada permite acceso completo a AWS AppConfig a través del punto de conexión de interfaz. Para controlar el acceso permitido a AWS AppConfig desde la VPC, adjunte una política de puntos de conexión personalizada al punto de conexión de interfaz.

Una política de punto de conexión especifica la siguiente información:

- Las entidades principales que pueden llevar a cabo acciones (Cuentas de AWS, usuarios de IAM y roles de IAM).
- Las acciones que se pueden realizar.
- El recurso en el que se pueden realizar las acciones.

Para obtener más información, consulte Control del acceso a los servicios con políticas de punto de conexión en la Guía del usuario de AWS PrivateLink.

Ejemplo: política de punto de conexión de VPC para acciones de AWS AppConfig

A continuación, se muestra un ejemplo de una política de un punto de conexión personalizada. Cuando se asocia con un punto de conexión, esta política concede acceso a las acciones de AWS AppConfig mostradas para todas las entidades principales en todos los recursos.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "appconfig:CreateApplication",
            "appconfig:CreateEnvironment",
            "appconfig:CreateConfigurationProfile",
            "appconfig:StartDeployment",
            "appconfig:GetLatestConfiguration"
            "appconfig:StartConfigurationSession"
         ],
         "Resource":"*"
      }
```

```
    ]
}
```

Rotación de claves de Secrets Manager

En esta sección se describe información de seguridad importante sobre la integración de AWS AppConfig con Secrets Manager. Para obtener más información acerca de Secrets Manager, consulte ¿Qué es AWS Secrets Manager? en la Guía del usuario de AWS Secrets Manager.

Configuración de la rotación automática de los secretos de Secrets Manager implementación por AWS AppConfig

La rotación es el proceso de actualización periódica de un secreto almacenado en Secrets Manager. Cuando Secrets Manager rota un secreto, se actualizan las credenciales tanto en el secreto como en la base de datos o el servicio. Puede configurar la rotación automática de secretos en Secrets Manager utilizando una función de AWS Lambda para actualizar el secreto y la base de datos. Para obtener más información, consulte Rotación de sus secretos de AWS Secrets Manager en la guía del usuario de AWS Secrets Manager.

Para habilitar la rotación de claves de los secretos de Secrets Manager implementados por AWS AppConfig, actualice la función de Lambda de rotación e implemente el secreto rotado.

> **Note**
>
> Implemente su perfil de configuración de AWS AppConfig una vez que su secreto se haya rotado y actualizado completamente a la nueva versión. Puede determinar si el secreto ha cambiado porque el estado `VersionStage` cambia de `AWSPENDING` a `AWSCURRENT`. La finalización de la rotación de secretos se produce dentro de la función `finish_secret` de las plantillas de rotación de Secrets Manager.

A continuación, se muestra un ejemplo de función que inicia una implementación de AWS AppConfig después de rotar un secreto.

```
import time
import boto3
client = boto3.client('appconfig')
```

```
def finish_secret(service_client, arn, new_version):
    """Finish the rotation by marking the pending secret as current
    This method finishes the secret rotation by staging the secret staged AWSPENDING
 with the AWSCURRENT stage.
    Args:
        service_client (client): The secrets manager service client
        arn (string): The secret ARN or other identifier
        new_version (string): The new version to be associated with the secret
    """
    # First describe the secret to get the current version
    metadata = service_client.describe_secret(SecretId=arn)
    current_version = None
    for version in metadata["VersionIdsToStages"]:
        if "AWSCURRENT" in metadata["VersionIdsToStages"][version]:
            if version == new_version:
                # The correct version is already marked as current, return
                logger.info("finishSecret: Version %s already marked as AWSCURRENT for
 %s" % (version, arn))
                return
            current_version = version
            break

    # Finalize by staging the secret version current
    service_client.update_secret_version_stage(SecretId=arn, VersionStage="AWSCURRENT",
 MoveToVersionId=new_version, RemoveFromVersionId=current_version)

    # Deploy rotated secret
    response = client.start_deployment(
            ApplicationId='TestApp',
            EnvironmentId='TestEnvironment',
            DeploymentStrategyId='TestStrategy',
            ConfigurationProfileId='ConfigurationProfileId',
            ConfigurationVersion=new_version,
            KmsKeyIdentifier=key,
            Description='Deploy secret rotated at ' + str(time.time())
        )

    logger.info("finishSecret: Successfully set AWSCURRENT stage to version %s for
 secret %s." % (new_version, arn))
```

Supervisión de AWS AppConfig

La supervisión es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de AWS AppConfig y de sus otras soluciones de AWS. AWS ofrece las siguientes herramientas de supervisión para vigilar AWS AppConfig, informar cuando algo no va bien y tomar medidas automáticamente cuando proceda:

- AWS CloudTrail captura llamadas a la API y eventos relacionados efectuados por su cuenta de AWS o en su nombre, y entrega los archivos de registro al bucket de Amazon S3 que se haya especificado. También puede identificar qué usuarios y cuentas llamaron a AWS, la dirección IP de origen de las llamadas y el momento en que se hicieron. Para más información, consulte la Guía del usuario de AWS CloudTrail.
- Amazon CloudWatch Logs le permite supervisar, almacenar y acceder a sus archivos de registro desde instancias de Amazon EC2 y otras fuentes. CloudTrail CloudWatch Los registros pueden monitorear la información de los archivos de registro y notificarle cuando se alcancen ciertos umbrales. También se pueden archivar los datos del registro en un almacenamiento de larga duración. Para obtener más información, consulta la Guía del usuario CloudWatch de Amazon Logs.

Temas

Registrar llamadas a la API de AWS AppConfig mediante AWS CloudTrail

AWS AppConfigestá integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio enAWS AppConfig. CloudTrail captura todas las llamadas a la API AWS AppConfig como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de AWS AppConfig y las llamadas desde el código a las operaciones de la API de AWS AppConfig. Si crea una ruta, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon S3, incluidos los eventos paraAWS AppConfig. Si no configura una ruta, podrá ver los eventos más recientes en la CloudTrail consola, en el historial de eventos. Con la información recopilada por usted CloudTrail, puede determinar el destinatario de la solicitudAWS

AppConfig, la dirección IP desde la que se realizó la solicitud, quién la realizó, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la Guía AWS CloudTrail del usuario.

AWS AppConfiginformación en CloudTrail

CloudTrail está habilitada en tu cuenta Cuenta de AWS al crear la cuenta. Cuando se produce una actividad enAWS AppConfig, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio en el historial de eventos. Puede ver, buscar y descargar eventos recientes en su Cuenta de AWS. Para obtener más información, consulte Visualización de eventos con el historial de CloudTrail eventos.

Para mantener un registro continuo de eventos en la Cuenta de AWS, incluidos los eventos de AWS AppConfig, cree un registro de seguimiento. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma 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. Además, puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos. Para más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- CloudTrail servicios e integraciones compatibles
- Configuración de las notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de CloudTrail registro de varias regiones y recibir archivos de CloudTrail registro de varias cuentas

Todas AWS AppConfig las acciones se registran CloudTrail y se documentan en la referencia de la AWS AppConfig API. Por ejemplo, las llamadas a `GetApplication` y `ListApplications` las acciones generan entradas en los archivos de CloudTrail registro. `CreateApplication`

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario lo 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 .

AWS AppConfigeventos de datos en CloudTrail

Los eventos de datos proporcionan información sobre las operaciones de recursos realizadas en un recurso o dentro de él (por ejemplo, recuperar la última configuración implementada mediante una llamada GetLatestConfiguration). Se denominan también operaciones del plano de datos. Los eventos de datos suelen ser actividades de gran volumen. De forma predeterminada, CloudTrail no registra los eventos de datos. El historial de CloudTrail eventos no registra los eventos de datos.

Se aplican cargos adicionales a los eventos de datos. Para obtener más información sobre CloudTrail los precios, consulta AWS CloudTrailPrecios.

Puede registrar eventos de datos para los tipos de AWS AppConfig recursos mediante la CloudTrail consola o las operaciones de la CloudTrail API. AWS CLI En la tabla de esta sección se muestran los tipos de recursos disponibles paraAWS AppConfig.

- Para registrar eventos de datos mediante la CloudTrail consola, cree un almacén de datos de rutas o eventos para registrar eventos de datos, o actualice un banco de datos de seguimiento o evento existente para registrar eventos de datos.

  1. Elija Eventos de datos para registrar los eventos de datos.
  2. En la lista de tipos de eventos de datos, elija AWS AppConfig.
  3. Elija la plantilla de selección de registros que desee utilizar. Puede registrar todos los eventos de datos del tipo de recurso, registrar todos los `readOnly` eventos, registrar todos los `writeOnly` eventos o crear una plantilla de selección de registros personalizada para filtrar `resources.ARN` los campos y. `readOnly` `eventName`
  4. En Nombre del selector, introduzca AppConfigDataEvents. Para obtener información sobre cómo habilitar Amazon CloudWatch Logs para su registro de eventos de datos, consulteRegistrar las métricas de las llamadas al plano de AWS AppConfig datos.
- Para registrar los eventos de datos mediante elAWS CLI, configure el `--advanced-event-selectors` parámetro para que el `eventCategory` campo sea igual al valor del tipo de recurso `Data` y el `resources.type` campo sea igual al valor del tipo de recurso (consulte

la tabla). Puede agregar condiciones para filtrar los valores de los `resources.ARN` campos `readOnly`y `eventName`.

- Para configurar una ruta para registrar eventos de datos, ejecute el put-event-selectorscomando. Para obtener más información, consulte Registrar eventos de datos para senderos con la AWS CLI.
- Para configurar un banco de datos de eventos para registrar eventos de datos, ejecute el create-event-data-storecomando para crear un nuevo banco de datos de eventos para registrar eventos de datos, o ejecute el update-event-data-storecomando para actualizar un banco de datos de eventos existente. Para obtener más información, consulte Registrar eventos de datos para los almacenes de datos de eventos con AWS CLI.

En la tabla siguiente se enumeran los tipos de recursos de AWS AppConfig. La columna Tipo de evento de datos (consola) muestra el valor que se puede elegir en la lista de tipos de eventos de datos de la CloudTrail consola. La columna de valores resources.type muestra el `resources.type` valor que se debe especificar al configurar los selectores de eventos avanzados mediante las API o. AWS CLI CloudTrail La CloudTrail columna API de datos en la que se ha registrado muestra las llamadas a la API registradas CloudTrail para el tipo de recurso.

Tipo de evento de datos (consola)	resources.type value	Las API de datos registradas en CloudTrail *
AWS AppConfig	`AWS::AppConfig::Configuration`	• GetLatestConfiguration • StartConfigurationSession

*Puede configurar selectores de eventos avanzados para filtrar por `eventName`, `readOnly`, y `resources.ARN` campos para registrar solo aquellos eventos que sean importantes para usted. Para obtener más información acerca de estos campos, consulte AdvancedFieldSelector.

AWS AppConfigeventos de gestión en CloudTrail

Los eventos de administración proporcionan información sobre las operaciones de administración que se realizan en los recursos de su cuenta de AWS. Se denominan también operaciones del plano de control. De forma predeterminada, CloudTrail registra los eventos de administración.

AWS AppConfigregistra todas las operaciones del plano de AWS AppConfig control como eventos de administración. Para obtener una lista de las operaciones del plano de AWS AppConfig control en las que se AWS AppConfig registra CloudTrail, consulte la referencia de la AWS AppConfig API.

Descripción de las entradas de los archivos de registro de AWS AppConfig

Un rastro es una configuración que permite la entrega de eventos como archivos de registro a un bucket de Amazon S3 que usted especifique. CloudTrail Los archivos de registro contienen una o más entradas de registro. Un evento representa una solicitud única de cualquier fuente e incluye información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra una entrada de CloudTrail registro que demuestra la StartConfigurationSessionacción.


```
{
      "eventVersion": "1.09",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Administrator",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
          "sessionIssuer": {},
          "attributes": {
            "creationDate": "2024-01-11T14:37:02Z",
            "mfaAuthenticated": "false"
          }
        }
      },
      "eventTime": "2024-01-11T14:45:15Z",
      "eventSource": "appconfig.amazonaws.com",
      "eventName": "StartConfigurationSession",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "203.0.113.0",
      "userAgent": "Boto3/1.34.11 md/Botocore#1.34.11 ua/2.0 os/macos#22.6.0
 md/arch#x86_64 lang/python#3.11.4 md/pyimpl#CPython cfg/retry-mode#legacy
 Botocore/1.34.11",
      "requestParameters": {
        "applicationIdentifier": "rrfexample",
```

```
        "environmentIdentifier": "mexampleqe0",
        "configurationProfileIdentifier": "3eexampleu1"
      },
      "responseElements": null,
      "requestID": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
      "eventID": "a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE",
      "readOnly": false,
      "resources": [
        {
          "accountId": "123456789012",
          "type": "AWS::AppConfig::Configuration",
          "ARN": "arn:aws:appconfig:us-east-1:123456789012:application/rrfexample/
environment/mexampleqe0/configuration/3eexampleu1"
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "123456789012",
      "eventCategory": "Data",
      "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "appconfigdata.us-east-1.amazonaws.com"
      }
    }
```


Registrar las métricas de las llamadas al plano de AWS AppConfig datos

Si ha configurado AWS CloudTrail el registro de eventos de AWS AppConfig datos, puede permitir que Amazon CloudWatch Logs registre las métricas de las llamadas al plano de AWS AppConfig datos. A continuación, puede buscar y filtrar los datos de registro en CloudWatch los registros creando uno o más filtros de métricas. Los filtros métricos definen los términos y patrones que se deben buscar en los datos de registro a medida que se envían a CloudWatch los registros. CloudWatch Logs utiliza filtros métricos para convertir los datos de registro en CloudWatch métricas numéricas. Puede graficar las métricas o configurarlas con una alarma.

Antes de empezar

Habilite el registro de eventos de AWS AppConfig datosAWS CloudTrail. El siguiente procedimiento describe cómo habilitar el registro de métricas para una entrada de AWS AppConfigseguimiento

existente CloudTrail. Para obtener información sobre cómo habilitar el CloudTrail registro de las llamadas del plan de AWS AppConfig datos, consulteAWS AppConfigeventos de datos en CloudTrail.

Utilice el siguiente procedimiento para permitir que CloudWatch los registros registren las métricas de las llamadas al plano de AWS AppConfig datos.

Para permitir que CloudWatch Logs registre las métricas de las llamadas al plano AWS AppConfig de datos

1. Abra la CloudTrail consola en https://console.aws.amazon.com/cloudtrail/.
2. En el salpicadero, elige tu AWS AppConfig ruta.
3. En la sección CloudWatch Registros, seleccione Editar.
4. Elija Enabled (Habilitado).
5. Para el nombre del grupo de registros, deje el nombre predeterminado o introduzca uno. Anote el nombre. Más adelante, elegirá el grupo de CloudWatch registros en la consola de registros.
6. En Role name (Nombre de rol), escriba un nombre.
7. Elija Guardar cambios.

Utilice el siguiente procedimiento para crear una métrica y un filtro de métricas para AWS AppConfig los CloudWatch registros. El procedimiento describe cómo crear un filtro métrico para las llamadas realizadas por `operation` y (opcionalmente) las llamadas realizadas por `operation` y`Amazon Resource Name (ARN)`.

Para crear una métrica y un filtro de métricas para «AWS AppConfigIn CloudWatch Logs»

1. Abra la CloudWatch consola enhttps://console.aws.amazon.com/cloudwatch/.
2. En el panel de navegación, elija Logs (Registros) y, luego, Log groups (Grupos de registros).
3. Seleccione la casilla de verificación situada junto al grupo de AWS AppConfig registros.
4. Elija Actions (Acciones) y, a continuación, seleccione Create metric filter (Crear filtro de métrica).
5. En Nombre del filtro, introduzca un nombre.
6. En Patrón de filtro, introduzca lo siguiente:

```
{ $.eventSource = "appconfig.amazonaws.com" }
```

7. (Opcional) En la sección Patrón de prueba, elija su grupo de registros en la lista Seleccione los datos de registro que desee probar. Si CloudTrail no has registrado ninguna llamada, puedes saltarte este paso.
8. Elija Siguiente.
9. Para el espacio de nombres métrico, ingresa. **`AWS AppConfig`**
10. En Metric name (Nombre de métrica), ingrese **`Calls`** (Tiempo de carga de página).
11. En Metric Value (Valor de métrica), ingrese **`1`**.
12. Omita el valor y la unidad predeterminados.
13. Para el nombre de la dimensión, introduzca**`operation`**.
14. En Valor de dimensión, introduzca**`$.eventName`**.

    (Opcional) Puede introducir una segunda dimensión que incluya el nombre del recurso de Amazon (ARN) que realiza la llamada. Para añadir una segunda dimensión, introduzca **`resource`** el nombre de la dimensión. En Valor de dimensión, introduzca**`$.resources[0].ARN`**.

    Elija Siguiente.
15. Revise los detalles del filtro y cree un filtro métrico.

(Opcional) Puede repetir este procedimiento para crear un nuevo filtro de métricas para un código de error específico, como AccessDenied. Si lo hace, introduzca los siguientes detalles:

1. En Nombre del filtro, introduzca un nombre.
2. En Patrón de filtro, introduzca lo siguiente:

   ```
   { $.errorCode = "codename" }
   ```

   Por ejemplo

   ```
   { $.errorCode = "AccessDenied" }
   ```
3. Para el espacio de nombres métrico, introduzca. **`AWS AppConfig`**
4. En Metric name (Nombre de métrica), ingrese **`Errors`** (Tiempo de carga de página).
5. En Metric Value (Valor de métrica), ingrese **`1`**.
6. En Valor predeterminado, introduzca un cero (0).

7. Omita la unidad, las dimensiones y las alarmas.

Después de CloudTrail registrar las llamadas a la API, puedes ver las métricas en ellas CloudWatch. Para obtener más información, consulta Cómo ver tus métricas y registros en la consola en la Guía del CloudWatch usuario de Amazon. Para obtener información sobre cómo localizar una métrica que has creado, consulta Buscar las métricas disponibles.

> **Note**
>
> Si configuras la métrica de error sin dimensión, como se describe aquí, puedes ver esas métricas en la página Métricas sin dimensión.

Crear una alarma para una CloudWatch métrica

Después de crear las métricas, puede crear alarmas métricas en CloudWatch. Por ejemplo, puede crear una alarma para la métrica de AWS AppConfigllamadas que creó en el procedimiento anterior. En concreto, puede crear una alarma para las llamadas a la acción de la AWS AppConfig `StartConfigurationSession` API que superen un umbral. Para obtener información sobre cómo crear una alarma para una métrica, consulta Crear una CloudWatch alarma basada en un umbral estático en la Guía del CloudWatch usuario de Amazon. Para obtener información sobre los límites predeterminados para las llamadas al plano de AWS AppConfig datos, consulte los límites predeterminados del plano de datos en Referencia general de Amazon Web Services.

AWS AppConfig Historial de documentos de la Guía del usuario

En la siguiente tabla se describen los cambios importantes en la documentación desde la última versión de AWS AppConfig.

Versión actual de la API: 2019-10-09

Cambio	Descripción	Fecha
AWS AppConfig ejemplos de extensiones personalizadas	El tema Tutorial: creación de AWS AppConfig extensiones personalizadas ahora incluye enlaces a los siguientes ejemplos de extensiones en GitHub: • Ejemplo de extensión que impide las implementaciones con un calendario de `blocked day` moratorias mediante Systems Manager Change Calendar • Ejemplo de extensión que evita que los secretos se filtren en los datos de configuración mediante git-secrets • Ejemplo de extensión que evita que la información de identificación personal (PII) se filtre en los datos de configuración mediante Amazon Comprehend	28 de febrero de 2024

Tema nuevo: Registro de llamadas a la AWS AppConfig API mediante AWS CloudTrail	AWS AppConfig está integrado con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio en AWS AppConfig. CloudTrail captura todas las llamadas a la API AWS AppConfig como eventos. Este nuevo tema proporciona contenido AWS AppConfig específico en lugar de incluir enlaces al contenido correspondiente en la Guía del AWS Systems Manager usuario. Para obtener más información, consulte Registrar llamadas a AWS AppConfig la API mediante AWS CloudTrail.	18 de enero de 2024

AWS AppConfig ahora admite AWS PrivateLink	Puede usarlo AWS PrivateLink para crear una conexión privada entre su VPC y. AWS AppConfig Puede acceder AWS AppConfig como si estuviera en su VPC, sin el uso de una puerta de enlace a Internet, un dispositivo NAT, una conexión VPN o AWS Direct Connect una conexión. Las instancias de la VPC no necesitan direcciones IP públicas para acceder a AWS AppConfig. Para obtener más información, consulte Acceso AWS AppConfig mediante un punto final de interfaz (AWS PrivateLink).	6 de diciembre de 2023

Funciones adicionales de recuperación de AWS AppConfig agentes y un nuevo modo de desarrollo local	AWS AppConfig Agent ofrece las siguientes funciones adicionales para ayudarle a recuperar las configuraciones de sus aplicaciones. Funciones de recuperación adicionales • Recuperación de varias cuentas: utilice el AWS AppConfig agente de una cuenta principal o la recuperación Cuenta de AWS para recuperar los datos de configuración de varias cuentas de proveedor es. • Escribir la copia de la configuración en el disco: utilice el AWS AppConfig agente para escribir los datos de configuración en el disco. Esta función permite a los clientes con aplicacio nes que leen los datos de configuración del disco integrarse en ellas AWS AppConfig. Note La escritura de la configuración en el disco no está diseñada como una función de	1 de diciembre de 2023

copia de seguridad de la configuración. AWS AppConfig El agente no lee los archivos de configuración copiados en el disco. Si desea hacer una copia de seguridad de las configuraciones en disco, consulte las variables de `PRELOAD_BACKUP` entorno de Uso del AWS AppConfig agente con Amazon EC2 o Uso del AWS AppConfig agente con Amazon ECS `BACKUP_DIRECTORY` y Amazon EKS.

Modo de desarrollo local

AWS AppConfig El agente apoya un modo de desarroll o local. Si habilita el modo de desarrollo local, el agente lee los datos de configuración de un directorio específico del disco. No recupera los datos de configuración de AWS AppConfig. Puede simular las implementaciones de configuración actualizando los archivos en el directorio especificado. Recomendamos

	el modo de desarrollo local para los siguientes casos de uso: • Pruebe diferentes versiones de configuración antes de implementarlas utilizando AWS AppConfig. • Pruebe diferentes opciones de configuración para una nueva función antes de realizar cambios en su repositorio de código. • Pruebe diferentes escenario s de configuración para comprobar que funcionan según lo esperado.	
Nuevo tema de ejemplos de código	Se ha añadido un nuevo tema sobre ejemplos de código a esta guía. El tema incluye ejemplos en Java, Python y JavaScript para realizar mediante programación seis acciones comunes AWS AppConfig .	17 de noviembre de 2023

Se ha revisado el índice para reflejar mejor el flujo de trabajo de AWS AppConfig	El contenido de esta guía del usuario ahora se agrupa bajo los encabezados Creación, Implementación, Recuperación y Ampliación de los flujos de trabajo. Esta organización refleja mejor el flujo de trabajo de uso de AWS AppConfig y tiene como objetivo ayudar a que el contenido sea más fácil de localizar.	7 de noviembre de 2023
Se ha añadido una referencia a la carga	El tema Creación de una función de Lambda para una extensión personalizada de AWS AppConfig ahora incluye una referencia a la carga de solicitud y respuesta.	7 de noviembre de 2023
Nueva estrategia de AWS despliegue predefinida	AWS AppConfig ahora ofrece y recomienda la estrategia de despliegue `AppConfig.Linear20PercentEvery6Minutes` predefinida. Para obtener más información, consulte Estrategias de implementación predefinidas.	11 de agosto de 2023

AWS AppConfig integración con Amazon EC2	Puede realizar la integración AWS AppConfig con las aplicaciones que se ejecutan en sus instancias de Linux de Amazon Elastic Compute Cloud (Amazon EC2) mediante Agent. AWS AppConfig El agente es compatible con las arquitecturas x86_64 y ARM64 para Amazon EC2. Para obtener más información, consulte Integración de AWS AppConfig con Amazon EC2.	20 de julio de 2023
AWS CloudFormation soporte para nuevos AWS AppConfig recursos y un ejemplo de indicador de funcionalidad	AWS CloudFormation ahora es compatible con los AWS::AppConfig::ExtensionAssociationrecursos AWS::AppConfig::Extensiony que le ayudarán a empezar con AWS AppConfig las extensiones. Los recursos AWS::AppConfig::ConfigurationProfiley AWS::AppConfig::HostedConfigurationVersion ahora incluyen un ejemplo para crear un perfil de configuración de indicadores de características en el almacén de configuración AWS AppConfig hospedado.	12 de abril de 2023

AWS AppConfig integración con AWS Secrets Manager	AWS AppConfig se integra con AWS Secrets Manager. Secrets Manager ayuda a cifrar, almacenar y recuperar de forma segura las credencia les de sus bases de datos y otros servicios. En lugar de codificar las credenciales de sus aplicaciones, puede hacer llamadas a Secrets Manager para recuperar sus credenciales siempre que las necesite. Secrets Manager le ayuda a proteger el acceso a sus recursos y datos de TI al permitirle rotar y administrar el acceso a sus secretos. Al crear un perfil de configura ción de formato libre, puede elegir Secrets Manager como origen de los datos de configuración. Debe incorpora rse a Secrets Manager y crear un secreto antes de crear el perfil de configura ción. Para obtener más información sobre Secrets Manager, consulte ¿Qué es AWS Secrets Manager? en la Guía AWS Secrets Manager del usuario. Para obtener información sobre la creación de un perfil de configuración, consulte Creación de un perfil	2 de febrero de 2023

de configuración de formato libre.

AWS AppConfig integración con Amazon ECS y Amazon EKS	Puede realizar la integración AWS AppConfig con Amazon Elastic Container Service (Amazon ECS) y Amazon Elastic Kubernetes Service (Amazon EKS) mediante el agente. AWS AppConfig El agente funciona como un contenedor asociado que se ejecuta junto con las aplicaciones de contenedores de Amazon ECS y Amazon EKS. El agente mejora el procesamiento y la administración de las aplicaciones en contenedores de las siguientes maneras: • El agente llama AWS AppConfig en su nombre utilizando una función AWS Identity and Access Management (IAM) y gestionando una caché local de datos de configuración. Al extraer los datos de configuración de la memoria caché local, su aplicación necesita menos actualizaciones de código para gestionar los datos de configuración, recupera los datos de configuración en milisegundos y no se ve afectada por problemas de red que puedan interrumpir las llamadas a dichos datos.	2 de diciembre de 2022

- El agente ofrece una experiencia nativa para recuperar y resolver los indicadores de AWS AppConfig funciones.
- En su estado original, el agente proporciona las prácticas recomendadas para las estrategias de almacenamiento en caché, los intervalos de sondeo y la disponibilidad de los datos de configuración local, al tiempo que rastrea los tokens de configura ción necesarios para las siguientes llamadas de servicio.
- Mientras se ejecuta en segundo plano, el agente sondea periódica mente el plano de AWS AppConfig datos en busca de actualizaciones de los datos de configuración. La aplicación en contenedores puede recuperar los datos conectándose a localhost en el puerto 2772 (un valor de puerto predeterminado personalizable) y llamando a HTTP GET para recuperar los datos.
- El AWS AppConfig agente actualiza los datos de configuración de sus

contenedores sin tener que reiniciarlos ni reciclarlos.

Para obtener más información, consulte AWS AppConfig Integración de con Amazon ECS y Amazon EKS.

Nueva extensión: AWS AppConfig extensión para CloudWatch Evidently	Puede utilizar Amazon CloudWatch Evidently para validar nuevas funciones de forma segura ofreciéndolas a un porcentaje específico o de sus usuarios mientras implementa la función. Puede monitorear el rendimiento de la nueva característica para decidir cuándo aumentar el tráfico hacia los usuarios. Esto ayuda a reducir los riesgos e identificar las consecuencias no deseadas antes de lanzar la característica por completo. También puede llevar a cabo experimentos A/B para tomar decisiones de diseño de características basadas en pruebas y datos. La AWS AppConfig extensión de CloudWatch Evidently permite a la aplicación asignar variaciones a las sesiones de usuario de forma local, en lugar de tener que llamar a la EvaluateFeatureoperación . Esto mitiga los riesgos de latencia y disponibilidad que conlleva una llamada a la API. Para obtener información sobre cómo configurar y usar la extensión, consulte Realizar lanzamientos y experimentos A/B con CloudWatch Evidently	13 de septiembre de 2022

	en la Guía CloudWatch del usuario de Amazon.	
La acción de la API `GetConfiguration` ha quedado obsoleta	El 18 de noviembre de 2021, AWS AppConfig lanzó un nuevo servicio de plano de datos. Este servicio reemplaza el proceso anterior de recuperación de datos de configuración mediante la acción de API `GetConfiguration` . El servicio de plano de datos utiliza dos nuevas acciones de API, StartConfigurationSessiony GetLatestConfiguration. El servicio de plano de datos también utiliza nuevos puntos de conexión. Para obtener más información, consulte Acerca del servicio AWS AppConfig de plano de datos.	13 de septiembre de 2022
Nueva versión de la extensión AWS AppConfig Agent Lambda	Ya está disponible la versión 2.0.122 de la extensión AWS AppConfig Agent Lambda. La nueva extensión usa nombres de recursos de Amazon (ARN) diferentes. Para más informaci ón, consulte AWS AppConfig Notas de la versión de la extensión de Lambda del agente de.	23 de agosto de 2022

Lanzamiento de extensiones AWS AppConfig	Una extensión aumenta la capacidad de introducir lógica o comportamiento en diferentes puntos del AWS AppConfig flujo de trabajo de creación o implementación de una configuración. Puede usar extensiones de su AWS autoría o crear las suyas propias. Para obtener más información, consulte Trabajar con AWS AppConfig extensiones.	12 de julio de 2022
Nueva versión de la extensión AWS AppConfig Agent Lambda	Ya está disponible la versión 2.0.58 de la extensión AWS AppConfig Agent Lambda. La nueva extensión usa nombres de recursos de Amazon (ARN) diferentes. Para obtener más información, consulte Versiones disponibles de la AWS AppConfig extensión Lambda.	3 de mayo de 2022

AWS AppConfig integración con Atlassian Jira	La integración con Atlassian Jira permite crear y actualizar problemas en AWS AppConfig la consola de Atlassian cada vez que realices cambios en una característica marcada en tu lista para cumplir con las especificadas. Cuenta de AWS Región de AWS Cada problema de Jira incluye el nombre de la marca, el ID de la aplicación, el ID del perfil de configuración y los valores de la marca. Tras actualiza r, guardar e implementar los cambios de las marcas, Jira actualiza los problemas existentes con los detalles del cambio. Para obtener más información, consulte Integraci ón de AWS AppConfig con Atlassian Jira.	7 de abril de 2022

Disponibilidad general de marcas de características y compatibilidad con extensiones de Lambda para procesadores ARM64 (Graviton2)	Con los marcadores de AWS AppConfig funciones , puedes desarrollar una nueva función e implementarla en producción, sin dejar de ocultarla a los usuarios. Para empezar, añada la marca a AWS AppConfig como datos de configuración. Una vez que la característica esté lista para su lanzamiento, puede actualizar los datos de configuración de la marca sin implementar ningún código. Esta característica mejora la seguridad de su entorno de operaciones de desarrollo, ya que no es necesario implementar código nuevo para lanzar la característica. Para obtener más información, consulte Crear un perfil de configuración de la marca de características. La disponibilidad general de las marcas de características de AWS AppConfig incluye las siguientes mejoras: • La consola incluye una opción para designar una marca como marca de corta duración. Puede filtrar y ordenar la lista de marcas por marcas de corta duración.	15 de marzo de 2022

- Para los clientes que utilizan marcas de características en AWS Lambda, la nueva extensión de Lambda permite llamar a marcas de características individua les mediante un punto de conexión HTTP. Para obtener más información, consulte Recuperación de una o varias marcas de una configuración de marcas de características.

Esta actualización también proporciona compatibilidad con las extensiones de AWS Lambda desarrolladas para los procesadores ARM64 (Graviton2). Para obtener más información, consulte Versiones disponibles de la extensión AWS AppConfig Lambda.

La acción de la GetConfiguration API está obsoleta	La acción de API `GetConfiguration` ha quedado obsoleta. Las llamadas para recibir datos de configuración deben usar las API `StartConfigurationSession` y `GetLatestConfiguration` en su lugar. Para obtener más información acerca de estas API y cómo usarlas, consulte Recuperación de la configuración.	28 de enero de 2022
Nuevo ARN de región para la extensión Lambda AWS AppConfig	AWS AppConfig La extensión Lambda está disponible en la nueva región de Asia Pacífico (Osaka). Se necesita el nombre de recurso de Amazon (ARN) para crear una Lambda en la región. Para obtener más información sobre el ARN de la región Asia Pacífico (Osaka), consulte Añadir la extensión Lambda AWS AppConfig.	4 de marzo de 2021

AWS AppConfig Extensión Lambda	Si lo utiliza AWS AppConfig para gestionar las configuraciones de una función Lambda, le recomendamos que añada la extensión Lambda AWS AppConfig . Esta extensión incluye las mejores prácticas que simplifican el uso y AWS AppConfig reducen los costos. La reducción de los costos se debe a una menor cantidad de llamadas a la API al AWS AppConfig servicio y, por separado, a la reducción de los costos debido a la reducción de los tiempos de procesamiento de las funciones de Lambda. Para más información, consulte Integración de AWS AppConfig con extensiones de Lambda.	8 de octubre de 2020
Sección nueva	Se añadió una nueva sección que proporciona las instrucciones para configurar AWS AppConfig. Para más información, consulte Configuración AWS AppConfig .	30 de septiembre de 2020

Se han añadido procedimientos de línea de comandos	Los procedimientos de esta guía del usuario ahora incluyen los pasos de línea de comandos para AWS Command Line Interface (AWS CLI) y las herramientas para Windows. PowerShell Para obtener más información, consulte Trabajar con AWS AppConfig.	30 de septiembre de 2020
Lanzamiento de la guía AWS AppConfig del usuario	Utilice AWS AppConfig una capacidad para crear AWS Systems Manager, administrar e implementar rápidamente configuraciones de aplicaciones. AWS AppConfig admite despliegues controlados en aplicaciones de cualquier tamaño e incluye controles de validación y supervisión integrados. Puede usarlo AWS AppConfig con aplicaciones alojadas en instancias EC2 AWS Lambda, contenedores, aplicaciones móviles o dispositivos de IoT.	31 de julio de 2020

Glosario de AWS

Para ver la terminología más reciente de AWS, consulte el Glosario de AWS en la Referencia de Glosario de AWS.

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.