Documento técnico de AWS

Arquitecturas de varios niveles sin servidor de AWS con Amazon API Gateway y AWS Lambda

Arquitecturas de varios niveles sin servidor de AWS con Amazon API Gateway y AWS Lambda : Documento técnico de AWS

Table of Contents

Arquitecturas de varios niveles sin servidor de AWS con Amazon API Gateway y AWS Lambda

Fecha de publicación: 20 de octubre de 2021 (Revisiones del documento)

Resumen

Este documento técnico Ilustra cómo las innovaciones de Amazon Web Services (AWS) se pueden utilizar para cambiar la forma de diseñar arquitecturas de multnivel e implementar patrones populares como microservicios, backend móviles y aplicaciones de una sola página. Los arquitectos y desarrolladores pueden usar Amazon API Gateway, AWS Lambda y otros servicios para reducir los ciclos de desarrollo y operaciones necesarios para crear y administrar aplicaciones de varios niveles.

Introducción

La aplicación de varios niveles (tres niveles,n niveles, etc.) ha sido un patrón de arquitectura fundamental durante décadas y sigue siendo un patrón popular para las aplicaciones orientadas al usuario. Si bien el lenguaje utilizado para describir una arquitectura de varios niveles varía, una aplicación de varios niveles generalmente consta de los siguientes componentes:

- Nivel de presentación: componente con el que el usuario interactúa directamente (por ejemplo, páginas web y interfaces de usuario de aplicaciones móviles).
- Nivel lógico: código necesario para traducir las acciones del usuario a la funcionalidad de la aplicación (por ejemplo, operaciones de base de datos CRUD y procesamiento de datos).
- Nivel de datos: medios de almacenamiento (por ejemplo, bases de datos, almacenes de objetos, cachés y sistemas de archivos) que contienen los datos relevantes para la aplicación.

El patrón de arquitectura de varios niveles proporciona un marco general para garantizar que los componentes de aplicaciones desacoplados y escalables de manera independiente se puedan desarrollar, administrar y mantener por separado (a menudo por equipos distintos).

Como consecuencia de este patrón en el que la red (un nivel debe realizar una llamada de red para interactuar con otro nivel) actúa como el límite entre los niveles, el desarrollo de una aplicación de varios niveles a menudo requiere la creación de muchos componentes de aplicación indiferenciados. Algunos de estos componentes incluyen:

- Código que define una cola de mensajes para la comunicación entre niveles
- Código que define una interfaz de programa de aplicación (API) y un modelo de datos
- Código relacionado con la seguridad que garantiza un acceso adecuado a la aplicación

Todos estos ejemplos se pueden considerar componentes «repetitivos» que, si bien son necesarios en aplicaciones de varios niveles, no varían mucho en su implementación de una aplicación a otra.

AWS ofrece una serie de servicios que permiten la creación de aplicaciones de varios niveles sin servidor, lo que simplifica en gran medida el proceso de implementación de dichas aplicaciones en producción y elimina la sobrecarga asociada con la administración tradicional de servidores. Amazon API Gateway, un servicio de creación y administración de API, y AWS Lambda, un servicio para ejecutar funciones de código arbitrarias, se pueden utilizar juntos para simplificar la creación de aplicaciones robustas de varios niveles.

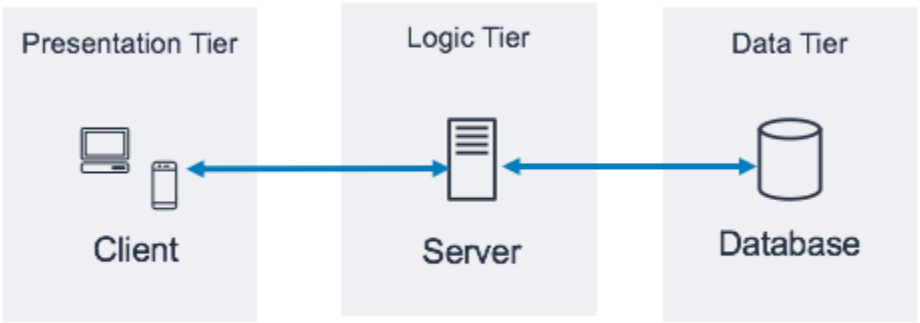
La integración de Amazon API Gateway con AWS Lambda permite que las funciones de código definidas por el usuario se inicien directamente a través de solicitudes HTTPS. Independientemente del volumen de solicitudes, tanto API Gateway como Lambda se escalan automáticamente para admitir con exactitud las necesidades de su aplicación (consulte Cuotas de Amazon API Gateway y notas importantes para obtener información sobre la escalabilidad). Al combinar estos dos servicios, puede crear un nivel que le permita escribir solo el código relevante para su aplicación y no centrarse en otros aspectos no diferenciados de la implementación de una arquitectura de varios niveles, como la arquitectura de alta disponibilidad, la escritura de SDK de cliente, la administración del servidor y el sistema operativo (SO), el escalado y la implementación de un mecanismo de autorización de clientes.

API Gateway y Lambda permiten la creación de un nivel lógico sin servidor. Según los requisitos de su aplicación, AWS también ofrece opciones para crear un nivel de presentación sin servidor (por ejemplo, con Amazon CloudFront y Amazon Simple Storage Service) y un nivel de datos (por ejemplo, Amazon Aurora, Amazon DynamoDB).

Este documento técnico se centra en el ejemplo más popular de una arquitectura de varios niveles, la aplicación web de tres niveles. Sin embargo, puede aplicar este patrón de varios niveles mucho más allá de una aplicación web típica de tres niveles.

Información general de la arquitectura de tres niveles

La arquitectura de tres niveles es la implementación más popular de una arquitectura de varios niveles y consiste en un solo nivel de presentación, nivel lógico y nivel de datos. En la siguiente ilustración se muestra un ejemplo de una aplicación simple y genérica de tres niveles.



Patrón arquitectónico de una aplicación de tres niveles

Hay muchos recursos en línea excelentes donde puede obtener más información sobre el patrón de arquitectura general de tres niveles. Este documento técnico se centra en un patrón de implementación específico para esta arquitectura mediante Amazon API Gateway y AWS Lambda.

Nivel lógico sin servidor

El nivel lógico de la arquitectura de tres niveles representa el cerebro de la aplicación. Aquí es donde el uso de Amazon API Gateway y AWS Lambda puede tener el mayor impacto en comparación con una implementación tradicional basada en servidor. Las características de estos dos servicios le permiten crear una aplicación sin servidor de alta disponibilidad, escalable y segura. En un modelo tradicional, su aplicación podría requerir miles de servidores; sin embargo, al usar Amazon API Gateway y AWS Lambda, no es responsable de la administración de servidores en ninguna capacidad. Además, al usar estos servicios administrados en conjunto, obtiene los siguientes beneficios:

- AWS Lambda:
  - Sin sistema operativo que elegir, proteger, aplicar parches o administrar
  - Sin servidores que precisen adaptación de tamaño, monitoreo o escalabilidad correctos
  - Reducción del riesgo de costes derivados de aprovisionamiento excesivo
  - Reducción del riesgo para su rendimiento por insuficiente aprovisionamiento

- Amazon API Gateway:
  - Mecanismos simplificados para implementar, monitorear y proteger las API
  - Rendimiento mejorado de la API mediante almacenamiento en caché y entrega de contenido

AWS Lambda

AWS Lambda es un servicio de computación que le permite ejecutar funciones de código arbitrarias en cualquiera de los lenguajes admitidos (Node.js, Python, Ruby, Java, Go, .NET. Para obtener más información, consulte las preguntas frecuentes sobre Lambda) sin aprovisionar, administrar ni escalar servidores. Las funciones de Lambda se ejecutan en un contenedor aislado y administrado y se lanzan en respuesta a un evento que puede ser uno de los varios desencadenadores programáticos que AWS pone a su disposición, lo que se conoce como origen de eventos. Consulte las preguntas frecuentes sobre Lambda para ver todos los orígenes de eventos

Muchos casos de uso populares de Lambda giran en torno a los flujos de trabajo de procesamiento de datos basados en eventos, como el procesamiento de archivos almacenados en Amazon S3 o la transmisión de registros de datos de Amazon Kinesis. Cuando se usa junto con Amazon API Gateway, una función de Lambda realiza la funcionalidad de un servicio web típico: inicia el código

en respuesta a una solicitud HTTPS del cliente; API Gateway actúa como puerta de entrada para su nivel lógico y AWS Lambda invoca el código de la aplicación.

Su lógica empresarial va aquí, sin necesidad de servidores

Lambda requiere que escriba funciones de código, llamadas controladores, que se ejecutarán cuando un evento las inicie. Para usar Lambda con API Gateway, puede configurar API Gateway para que inicie funciones de controlador cuando se produzca una solicitud HTTPS a su API. En una arquitectura de varios niveles sin servidor, cada una de las API que cree en API Gateway se integrará con una función de Lambda (y el controlador que contiene) que invoca la lógica empresarial requerida.

El uso de funciones de AWS Lambda para componer el nivel lógico le permite definir el nivel de granularidad deseado para exponer la funcionalidad de la aplicación (una función de Lambda por API o una función de Lambda por método de API). Dentro de la función de Lambda, el controlador puede comunicarse con cualquier otra dependencia (por ejemplo, otros métodos que haya cargado con su código, bibliotecas, binarios nativos y servicios web externos), o incluso otras funciones de Lambda.

La creación o actualización de una función de Lambda requiere cargar el código como un paquete de implementación de Lambda en un archivo zip en un bucket de Amazon S3 o empaquetar el código como una imagen de contenedor junto con todas las dependencias. Las funciones pueden usar diferentes métodos de implementación, como la consola de administración de AWS, ejecución AWS Command Line Interface (AWS CLI) o infraestructura en ejecución como plantillas de código o marcos como AWS CloudFormation, AWS Serverless Application Model (AWS SAM) o AWS Cloud Development Kit (AWS CDK). Cuando crea su función con cualquiera de estos métodos, especifica qué método dentro del paquete de implementación actuará como controlador de solicitudes. Puede reutilizar el mismo paquete de implementación para varias definiciones de funciones de Lambda, donde cada función de Lambda puede tener un controlador único dentro del mismo paquete de implementación.

Seguridad de Lambda

Para ejecutar una función de Lambda, debe invocarla un evento o servicio que esté permitido por una política de AWS Identity and Access Management (IAM). Con las políticas de IAM, puede crear una función de Lambda que no se puede iniciar en absoluto a menos que sea invocada por un recurso de API Gateway que usted haya definido. Dicha política se puede definir mediante políticas basadas en recursos en varios servicios de AWS.

Cada función de Lambda asume un rol de IAM que se asigna cuando se implementa la función de Lambda. Este rol de IAM define los demás servicios y recursos de AWS con los que puede interactuar la función de Lambda (por ejemplo, Amazon DynamoDB, Amazon S3). En el contexto de la función de Lambda, esto se denomina rol de ejecución.

No almacene información confidencial dentro de una función de Lambda. IAM gestiona el acceso a los servicios de AWS a través del rol de ejecución Lambda; si tiene que acceder a otras credenciales (por ejemplo, credenciales de base de datos y claves de API) desde la función de Lambda, puede usar AWS Key Management Service (AWS KMS) con variables de entorno o usar un servicio como AWS Secrets Manager para mantener esta información segura cuando no esté en uso.

Rendimiento a escala

El código extraído como imagen de contenedor de Amazon Elastic Container Registry (Amazon ECR) o de un archivo zip cargado en Amazon S3, se ejecuta en un entorno aislado administrado por AWS. No tiene que escalar sus funciones de Lambda: cada vez que la función reciba una notificación de evento, AWS Lambda localiza la capacidad disponible dentro de su flota de computación y ejecuta el código con las configuraciones de tiempo de ejecución, memoria, disco y tiempo de espera que usted haya definido. Con este patrón, AWS puede iniciar tantas copias de su función como sea necesario.

Un nivel lógico basado en Lambda siempre tiene el tamaño adecuado para las necesidades de sus clientes. La capacidad de absorber rápidamente los aumentos repentinos en el tráfico a través del escalado administrado y la iniciación de código concurrente, junto con los precios de pago por uso de Lambda, le permite atender siempre las solicitudes de los clientes y, al mismo tiempo, no pagar por la capacidad de computación inactiva.

Implementación y administración sin servidor

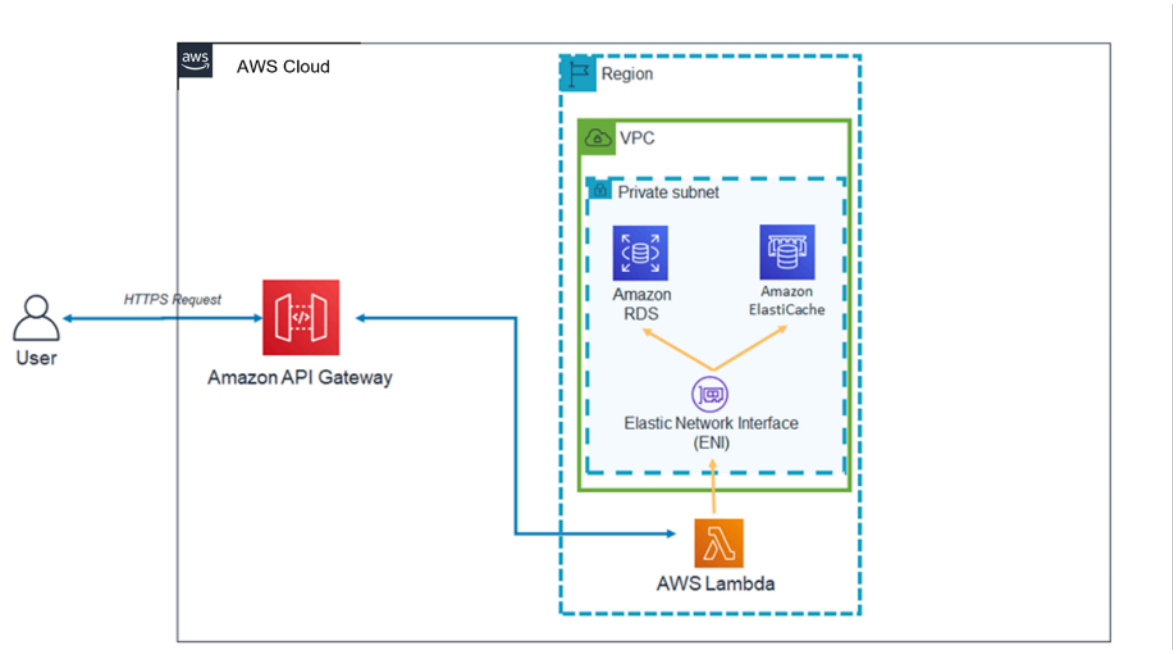
Para ayudarle a implementar y administrar sus funciones de Lambda, use AWS SAM AWS Serverless Application Model (AWS SAM), un marco de código abierto que incluye:

- Especificación de plantilla de AWS SAM: sintaxis utilizada para definir las funciones y describir los entornos, permisos, configuraciones y eventos para una carga e implementación simplificadas.
- CLI de AWS SAM: comandos que le permiten verificar la sintaxis de la plantilla SAM, invocar funciones localmente, depurar funciones de Lambda e implementar paquetes de funciones.

También puede utilizar AWS CDK, que es un marco de desarrollo de software para definir la infraestructura en la nube mediante lenguajes de programación y aprovisionarla a través de CloudFormation. CDK proporciona una forma imperativa de definir los recursos de AWS, mientras que AWS SAM proporciona una forma declarativa.

Por lo general, cuando implementa una función de Lambda, se invoca con los permisos definidos por su rol de IAM asignado y puede llegar a los puntos de conexión de Internet. Como el núcleo del nivel lógico, AWS Lambda es el componente que se integra directamente con el nivel de datos. Si su nivel de datos contiene información empresarial o de usuario confidencial, es importante asegurarse de que este nivel de datos esté aislado adecuadamente (en una subred privada).

Puede configurar una función de Lambda para que se conecte a subredes privadas en una nube privada virtual (VPC) en su cuenta de AWS si desea que la función de Lambda acceda a recursos que no puede exponer públicamente, como una instancia de base de datos privada. Cuando conecta una función a una VPC, Λ crea una interfaz de red elástica para cada subred en la configuración de VPC de su función y se utiliza la interfaz de red elástica para acceder a sus recursos internos de forma privada.



Patrón de arquitectura Lambda dentro de una VPC

El uso de Lambda con VPC significa que las bases de datos y otros medios de almacenamiento de los que depende su lógica empresarial pueden quedar inaccesibles a través de Internet. La VPC también garantiza que la única forma de interactuar con sus datos desde Internet sea a través de las API que ha definido y las funciones de código de Lambda que ha escrito.

Amazon API Gateway

Amazon API Gateway es un servicio completamente administrado que facilita que los desarrolladores creen, publiquen, mantengan, monitoricen y protejan las API a cualquier escala.

Los clientes (es decir, niveles de presentación) se integran con las API expuestas a través de API Gateway mediante solicitudes HTTPS estándar. La aplicabilidad de las API expuestas a través de API Gateway para una arquitectura de varios niveles orientada a servicios es la capacidad de separar partes individuales de la funcionalidad de la aplicación y exponer esta funcionalidad a través de puntos de conexión REST. Amazon API Gateway tiene características y cualidades específicas que pueden añadir potentes capacidades a su nivel lógico.

Integración con AWS Lambda

Amazon API Gateway admite los tipos de API REST y HTTP. Una API de API Gateway está formada por recursos y métodos. Un recurso es una entidad lógica a la que una aplicación puede acceder a través de una ruta de recursos (por ejemplo, `/tickets`). Un método corresponde a una solicitud de API que se envía a un recurso de API (por ejemplo,`GET /tickets`). API Gateway le permite respaldar cada método con una función de Lambda, es decir, cuando llama a la API a través del punto de conexión HTTPS expuesto en la API Gateway, API Gateway invoca la función de Lambda.

Puede conectar las funciones de API Gateway y Lambda mediante integraciones proxy e integraciones no proxy.

Integraciones proxy

En una integración de proxy, toda la solicitud HTTPS del cliente se envía tal cual a la función de Lambda. API Gateway pasa toda la solicitud del cliente como el parámetro de evento de la función de controlador de Lambda y la salida de la función de Lambda se devuelve directamente al cliente (incluido el código de estado, los encabezados, etc.).

Integraciones sin proxy

En una integración sin proxy, se configura la forma en que los parámetros, encabezados y cuerpo de la solicitud del cliente se transfieren al parámetro del evento de la función de controlador de Lambda. Además, puede configurar la forma en que la salida de Lambda se vuelve a traducir al usuario.

> **Note**
>
> API Gateway también puede usar proxy para recursos sin servidor adicionales fuera de AWS Lambda, como integraciones simuladas (útiles para el desarrollo inicial de aplicaciones) y proxy directo a objetos de S3.

Rendimiento de API estable en todas las regiones

Cada implementación de Amazon API Gateway incluye una distribución de Amazon CloudFront integrada. CloudFront es un servicio de entrega de contenido que utiliza la red global de ubicaciones de borde de Amazon como puntos de conexión para los clientes que utilizan su API. Esto ayuda a disminuir la latencia de respuesta de la API. Mediante el uso de varias ubicaciones de borde en todo el mundo, Amazon CloudFront también proporciona capacidades para combatir los escenarios de ataques de denegación de servicio distribuido (DDoS). Para obtener más información, consulte el documento técnico AWS Best Practices for DDoS Resiliency.

Puede mejorar el rendimiento de solicitudes de API específicas mediante el uso de API Gateway para almacenar las respuestas en una caché en memoria opcional. Este enfoque no solo proporciona beneficios de rendimiento para las solicitudes de API repetidas, sino que también reduce la cantidad de veces que se invocan las funciones de Lambda, lo que puede reducir el coste general.

Fomente la innovación y reduzca los gastos generales con funciones integradas

El coste de desarrollo para crear cualquier aplicación nueva es una inversión. El uso de API Gateway puede reducir la cantidad de tiempo necesario para ciertas tareas de desarrollo y reducir el coste total de desarrollo, lo que permite a las organizaciones experimentar e innovar con más libertad.

Durante las fases iniciales de desarrollo de aplicaciones, la implementación del registro y la recopilación de métricas a menudo se descuidan para entregar una nueva aplicación más rápidamente. Esto puede generar una deuda técnica y un riesgo operativo al implementar estas características en una aplicación que se ejecuta en producción. Amazon API Gateway se integra a la perfección con Amazon CloudWatch, que recopila y procesa datos sin procesar de API Gateway en métricas legibles y casi en tiempo real para supervisar la ejecución de la API. API Gateway también admite el registro de acceso con informes configurables y el seguimiento de AWS X-Ray para la depuración. Ninguna de estas características requiere que se escriba código y se pueden ajustar en aplicaciones que se ejecutan en producción sin riesgo para la lógica empresarial principal.

Es posible que se desconozca la vida útil general de una aplicación o que se sepa que es de corta duración. La creación de un caso empresarial para crear dichas aplicaciones puede ser más fácil si su punto de partida ya incluye las funciones administradas que proporciona API Gateway y si solo incurre en costes de infraestructura después de que las API comiencen a recibir solicitudes. Para obtener más información, consulte los Precios de API Gateway.

Iteraciones rápidas y agilidad

El uso de Amazon API Gateway y AWS Lambda para crear el nivel lógico de la API le permite adaptarse rápidamente a las cambiantes demandas de su base de usuarios al simplificar la implementación de la API y la administración de versiones.

Implementación por etapas

Cuando implementa una API en API Gateway, debe asociar la implementación con una etapa de API Gateway; cada etapa es una instantánea de la API y está disponible para que las aplicaciones cliente las llamen. Con esta convención, puede implementar aplicaciones fácilmente en etapas dev, test, stage o prod y mover implementaciones entre etapas. Cada vez que implementa la API en una etapa, crea una versión diferente de la API que se puede revertir si es necesario. Estas características permiten que la funcionalidad existente y las dependencias del cliente continúen sin interrupciones mientras se lanza la nueva funcionalidad como una versión de API independiente.

Integración desacoplada con Lambda

La integración entre la API en API Gateway y la función de Lambda se puede desacoplar mediante variables de etapa de API Gateway y un alias de función de Lambda. Esto simplifica y acelera la implementación de la API. En lugar de configurar el nombre o el alias de la función de Lambda directamente en la API, puede configurar la variable de etapa en la API que puede apuntar a un alias en particular en la función de Lambda. Durante la implementación, cambie el valor de la variable de etapa para que apunte a un alias de función de Lambda y la API ejecutará la versión de función de Lambda detrás del alias de Lambda para una etapa en particular.

Implementación del lanzamiento de valores controlados

El lanzamiento de valores controlados es una estrategia de implementación de software en la que una nueva versión de una API se implementa como lanzamiento de valores controlados para realizar pruebas, mientras que la versión base se implementa en la misma etapa como versión de producción para realizar las operaciones normales. Cuando se implementa un lanzamiento de valores controlados, el tráfico total de la API se separa aleatoriamente entre la versión de producción

y la versión de lanzamiento de valores controlados en un porcentaje preestablecido. Las API de API Gateway se pueden configurar para la implementación del lanzamiento de valores controlados para probar nuevas funciones con un conjunto limitado de usuarios.

Nombres de dominio personalizados

Puede proporcionar un nombre de URL intuitivo y fácil de usar para la API en lugar de la URL proporcionada por API Gateway. API Gateway proporciona funciones para configurar un dominio personalizado para las API. Con los nombres de dominio personalizados, puede configurar el nombre de host de la API y elegir una ruta base de varios niveles (por ejemplo, `myservice`, `myservice/cat/v1` o `myservice/dog/v2`) para asignarle una URL alternativa.

Priorizar la seguridad de API

Todas las aplicaciones deben garantizar que solo los clientes autorizados tengan acceso a sus recursos de API. Al diseñar una aplicación de varios niveles, puede aprovechar varias formas diferentes en las que Amazon API Gateway contribuye a proteger su nivel lógico:

Seguridad del tránsito

Todas las solicitudes a sus API se pueden realizar a través de HTTPS para habilitar el cifrado en tránsito.

API Gateway proporciona certificados SSL/TLS integrados; si utiliza la opción de nombre de dominio personalizado para las API públicas, puede proporcionar su propio certificado SSL/TLS con AWS Certificate Manager. API Gateway también admite la autenticación TLS mutua (mTLS). La TLS mutua mejora la seguridad de la API y ayuda a proteger los datos de ataques como la suplantación de identidad de clientes o los ataques de intermediarios.

Autorización de API

A cada combinación de recursos y métodos que cree como parte de su API se le concede un nombre de recurso de Amazon (ARN) único al que se puede hacer referencia en las políticas de AWS Identity and Access Management (IAM).

Hay tres métodos generales para agregar autorización a una API en API Gateway:

- Políticas y roles de IAM: los clientes utilizan la autorización de AWS Signature Version 4 (SIGv4) y las políticas de IAM para el acceso a la API. Las mismas credenciales pueden restringir o permitir el acceso a otros servicios y recursos de AWS según sea necesario (por ejemplo, buckets de Amazon S3 o tablas de Amazon DynamoDB).

- Grupos de usuarios de Amazon Cognito: los clientes inician sesión a través de un grupo de usuarios de Amazon Cognito y obtienen tokens, que se incluyen en el encabezado de autorización de una solicitud.
- Autorizador de Lambda: defina una función de Lambda que implemente un esquema de autorización personalizado que utilice una estrategia de token de portador (por ejemplo, OAuth y SAML) o use parámetros de solicitud para identificar a los usuarios.

Limitaciones de acceso

API Gateway admite la generación de claves de API y la asociación de estas claves con un plan de uso configurable. Puede supervisar el uso de claves de API con CloudWatch.

API Gateway admite limitación, límites de velocidad y límites de velocidad de ráfaga para cada método de la API.

API privadas

Mediante API Gateway, puede crear API REST privadas a las que solo se puede acceder desde la nube virtual privada en Amazon VPC mediante un punto de conexión de la VPC de interfaz. Se trata de una interfaz de red de punto de conexión que se crea en la VPC.

Mediante las políticas de recursos, puede habilitar o denegar el acceso a las API desde las VPC y los puntos de conexión de la VPC seleccionados, incluso entre diferentes cuentas de AWS. Cada punto de enlace se puede utilizar para tener acceso a varias API privadas. También puede utilizar AWS Direct Connect para establecer una conexión entre una red en las instalaciones y Amazon VPC, y obtener acceso a la API privada a través de esa conexión.

En cualquier caso, el tráfico dirigido a la API privada utilizará conexiones seguras y no saldrá de la red de Amazon, ya que está aislado de la red pública de Internet.

Protección de firewall con AWS WAF

Las API orientadas a Internet son vulnerables a ataques maliciosos. AWS WAF es un firewall de aplicaciones web que ayuda a proteger las API de dichos ataques. Protege las API de ataques web habituales, como ataques de inyección SQL y scripting entre sitios. Puede usar AWS WAF con API Gateway para ayudar a proteger las API.

Nivel de datos

El uso de AWS Lambda como nivel lógico no limita las opciones de almacenamiento de datos disponibles en el nivel de datos. Las funciones de Lambda se conectan a cualquier opción de almacenamiento de datos mediante la inclusión del controlador de base de datos adecuado en el paquete de implementación de Lambda y el uso de acceso basado en roles de IAM o credenciales cifradas (a través de AWS KMS o AWS Secrets Manager).

La elección de un almacén de datos para su aplicación depende en gran medida de los requisitos de la aplicación. AWS ofrece varios almacenes de datos sin servidor y almacenes de datos que no son sin servidor que puede utilizar para componer el nivel de datos de su aplicación.

Opciones de almacenamiento de datos sin servidor

Amazon S3 es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes en el sector.

Amazon Aurora es una base de datos relacional compatible con MySQL y PostgreSQL creada para la nube. Combina el rendimiento y la disponibilidad de las bases de datos empresariales tradicionales con la simplicidad y la rentabilidad de las bases de datos de código abierto. Aurora ofrece modelos de uso tradicionales y sin servidor.

Amazon DynamoDB es una base de datos de clave-valor y documentos que ofrece un rendimiento de milisegundos de un solo dígito a cualquier escala. Se trata de una base de datos completamente administrada, sin servidor, de varias regiones, multimaestra y duradera, con seguridad integrada, copia de seguridad y restauración, así como almacenamiento en caché en memoria para aplicaciones a escala de Internet.

Amazon Timestream es un servicio de base de datos de serie temporal ágil, de escala ajustable y completamente administrado para aplicaciones operativas y compatibles con IoT que simplifica el almacenamiento y análisis de billones de eventos diarios a una décima parte del coste de las bases de datos relacionales. Impulsados por el aumento de dispositivos de IoT, sistemas de TI y máquinas industriales inteligentes, los datos de serie temporal (datos que miden cómo cambian las cosas con el tiempo) son uno de los tipos de datos de más rápido crecimiento.

Amazon Quantum Ledger Database (Amazon QLDB) es una base de datos de libro mayor completamente administrada que ofrece un registro de transacciones transparente e inmutable, que se puede verificar de manera criptográfica y que es propiedad de una autoridad central de confianza.

Amazon QLDB registra cada uno de los cambios que se producen en los datos de las aplicaciones y mantiene un historial completo y que se pueda verificar.

Amazon Keyspaces (for Apache Cassandra) es un servicio de base de datos administrado, de alta disponibilidad y escalable compatible con Apache Cassandra. Con Amazon Keyspaces, puede ejecutar las cargas de trabajo de Cassandra en AWS con las mismas herramientas para desarrolladores y el mismo código de aplicación de Cassandra que utiliza en la actualidad. No necesita implementar parches en los servidores, aprovisionarlos ni administrarlos. Tampoco es necesario instalar, mantener ni utilizar sistemas de software. Amazon Keyspaces funciona sin servidores, por lo que solo paga por los recursos que utiliza. Además, el servicio escala verticalmente las tablas de forma automática en función del tráfico de la aplicación.

Amazon Elastic File System (Amazon EFS) proporciona un sistema de archivos elástico, sencillo, sin servidor y práctico que le permite compartir datos de archivos sin necesidad de aprovisionar o administrar el almacenamiento. Se puede utilizar con los servicios en la nube de AWS y con los recursos locales y está diseñado para escalar bajo demanda a petabytes sin interrumpir las aplicaciones. Con Amazon EFS puede incrementar o reducir los sistemas de archivos automáticamente a medida que agrega y quita archivos, lo que elimina la necesidad de aprovisionar y administrar la capacidad para dar lugar al crecimiento. Amazon EFS se puede montar con la función de Lambda, lo que lo convierte en una opción de almacenamiento de archivos viable para las API.

Opciones de almacenamiento de datos que no son sin servidor

Amazon Relational Database Service (Amazon RDS) es un servicio web administrado que facilita la configuración, el funcionamiento y el escalado de una base de datos relacional mediante cualquiera de los motores disponibles (Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle y Microsoft SQL Server) y que se ejecuta en varios diferentes tipos de instancias de base de datos optimizados para memoria, rendimiento o E/S.

Amazon Redshift es un servicio de almacenamiento de datos de escala de petabytes completamente administrado en la nube.

Amazon ElastiCache es una implementación completamente administrada de Redis o Memcached. Implemente, ejecute y ajuste la escala de conocidos almacenes de datos en memoria compatibles y de código abierto sin inconvenientes.

Amazon Neptune es un servicio de base de datos de grafos rápido, fiable y completamente administrado que le permite crear y ejecutar fácilmente aplicaciones que funcionen con conjuntos

de datos altamente conectados. Neptune admite modelos de gráficos populares (gráficos de propiedades y Marco de Descripción de Recursos [RDF] del W3C) y sus respectivos lenguajes de consulta, lo que le permite crear consultas con facilidad que naveguen de manera eficiente en conjuntos de datos altamente conectados.

Amazon DocumentDB (compatible con MongoDB) es un servicio de base de datos de documentos ágil, escalable, de alta disponibilidad y completamente administrado, compatible con cargas de trabajo de MongoDB.

Por último, también puede utilizar almacenes de datos que se ejecutan de forma independiente en Amazon EC2 como el nivel de datos de una aplicación de varios niveles.

Nivel de presentación

El nivel de presentación es responsable de interactuar con el nivel lógico a través de los puntos de conexión REST de API Gateway expuestos en Internet. Cualquier cliente o dispositivo con capacidad HTTPS puede comunicarse con estos puntos de conexión, lo que le da a su nivel de presentación la flexibilidad de adoptar muchas formas (aplicaciones de escritorio, aplicaciones móviles, páginas web, dispositivos de IoT, etc.). Según sus requisitos, el nivel de presentación puede utilizar las siguientes ofertas sin servidor de AWS: cualquier cliente o dispositivo con capacidad HTTPS puede comunicarse con estos puntos de conexión, lo que le da a su nivel de presentación la flexibilidad de adoptar muchas formas (aplicaciones de escritorio, aplicaciones móviles, páginas web, dispositivos de IoT, etc.). Según sus requisitos, el nivel de presentación puede utilizar las siguientes ofertas sin servidor de AWS:

- Amazon Cognito: un servicio sin servidor de sincronización de datos e identidades de usuario que le permite agregar el registro, el inicio de sesión y el control de acceso de los usuarios a sus aplicaciones web y móviles de manera rápida y eficiente. El escalado de Amazon Cognito le permite admitir a millones de usuarios e iniciar sesión mediante proveedores de identidad social, como Facebook, Google y Amazon, y proveedores de identidad empresarial mediante SAML 2.0.
- Amazon S3 con CloudFront: le permite ofrecer sitios web estáticos, como aplicaciones de una sola página, directamente desde un bucket de S3 sin necesidad de aprovisionar un servidor web. Puede usar CloudFront como una red de entrega de contenido (CDN) administrada para mejorar el rendimiento y habilitar SSL/TL mediante certificados administrados o personalizados.

AWS Amplify es un conjunto de herramientas y servicios que se pueden utilizar juntos o de forma individual para ayudar a los desarrolladores de frontend web y móvil a crear aplicaciones de pila completa escalables, con la tecnología de AWS. AWS Amplify es una oferta de servicio completamente administrado para implementar y alojar aplicaciones web estáticas a nivel mundial, que se ofrecen mediante la CDN de confianza de Amazon con cientos de puntos de presencia mundial y con flujos de trabajo CI/CD integrados que aceleran el ciclo de lanzamiento de la aplicación. Amplify admite marcos web conocidos, como JavaScript, React, Angular, Vue y Next.js, así como plataformas móviles, incluidas Android, iOS, React Native, Ionic y Flutter. Según las configuraciones de red y los requisitos de la aplicación, es posible que deba habilitar las API de API Gateway para que cumplan con el uso compartido de recursos de origen cruzado (CORS). El cumplimiento de CORS permite a los navegadores web invocar directamente sus API desde páginas web estáticas.

Cuando implementa un sitio web con CloudFront, se le proporciona un nombre de dominio de CloudFront para llegar a su aplicación (por ejemplo, `d2d47p2vcczkh2.cloudfront.net`). Puede usar Amazon Route 53 para registrar nombres de dominio y dirigirlos a su distribución de CloudFront, o dirigir los nombres de dominio que ya son de su propiedad a su distribución de CloudFront. Esto permite a los usuarios acceder a su sitio con un nombre de dominio familiar. Tenga en cuenta que también puede asignar un nombre de dominio personalizado mediante Route 53 a su distribución de API Gateway, lo que permite a los usuarios invocar las API con nombres de dominio conocidos.

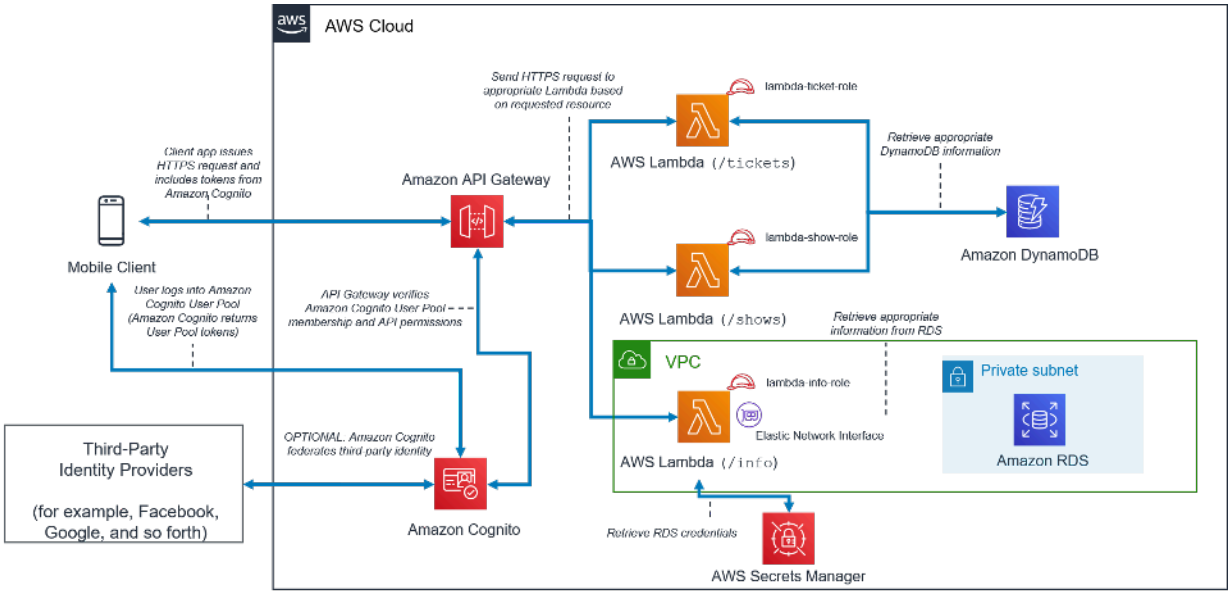
Ejemplos de patrones de arquitectura

Puede implementar patrones de arquitectura populares con API Gateway y AWS Lambda como nivel lógico. Este documento técnico incluye los patrones de arquitectura más populares que aprovechan los niveles lógicos basados en AWS Lambda:

- Backend móvil: una aplicación móvil se comunica con API Gateway y Lambda para acceder a los datos de la aplicación. Este patrón se puede ampliar a clientes HTTPS genéricos que no utilizan recursos de AWS sin servidor para alojar recursos de nivel de presentación (como clientes de escritorio, servidores web que se ejecutan en EC2, etc.).
- Aplicación de una sola página: una aplicación de una sola página alojada en Amazon S3 y CloudFront se comunica con API Gateway y AWS Lambda para acceder a los datos de la aplicación.
- Aplicación web: la aplicación web es un backend de aplicación web de propósito general, basado en eventos, que utiliza AWS Lambda con API Gateway para su lógica empresarial. También usa Amazon DynamoDB como su base de datos y Amazon Cognito para la administración de usuarios. Todo el contenido estático se aloja mediante Amplify.

Además de estos dos patrones, este documento técnico analiza la aplicabilidad de Lambda y API Gateway a una arquitectura general de microservicios. Una arquitectura de microservicios es un patrón popular que, aunque no es una arquitectura estándar de tres niveles, implica desacoplar los componentes de la aplicación e implementarlos como unidades de funcionalidad individuales sin estado que se comunican entre sí.

Backend móvil



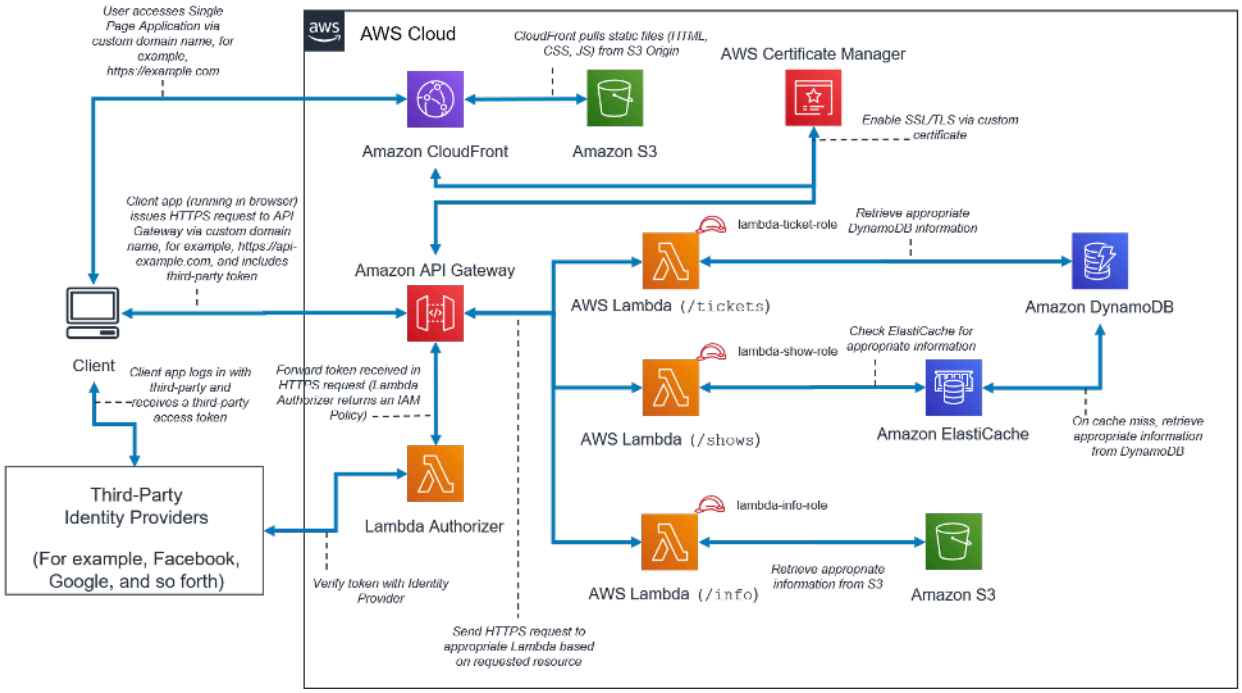
Patrón arquitectónico de backend móvil sin servidor

Tabla 1: Componentes del nivel de backend móvil

Nivel	Componentes
Presentación	Aplicación móvil que se ejecuta en un dispositivo de usuario.
Lógico	Amazon API Gateway con AWS Lambda. Esta arquitectura muestra tres servicios expuestos (`/tickets`, `/shows` y `/info`). Los puntos de conexión de API Gateway están protegidos por grupos de usuarios de Amazon Cognito. En este método, los usuarios inician sesión en los grupos de usuarios de Amazon Cognito (con un tercero federado si es necesario) y reciben tokens de acceso e ID que se utilizan para autorizar llamadas de API Gateway.

Nivel	Componentes
	A cada función de Lambda se le asigna su propio rol de Identity and Access Managemen t (IAM) para proporcionar acceso al origen de datos adecuado.
Datos	Los servicios `/tickets` y `/shows` utilizan DynamoDB. El servicio `/info` utiliza Amazon RDS. Esta función de Lambda recupera las credenciales de Amazon RDS de AWS Secrets Manager y utiliza una interfaz de red elástica para acceder a la subred privada.

Aplicación de una sola página

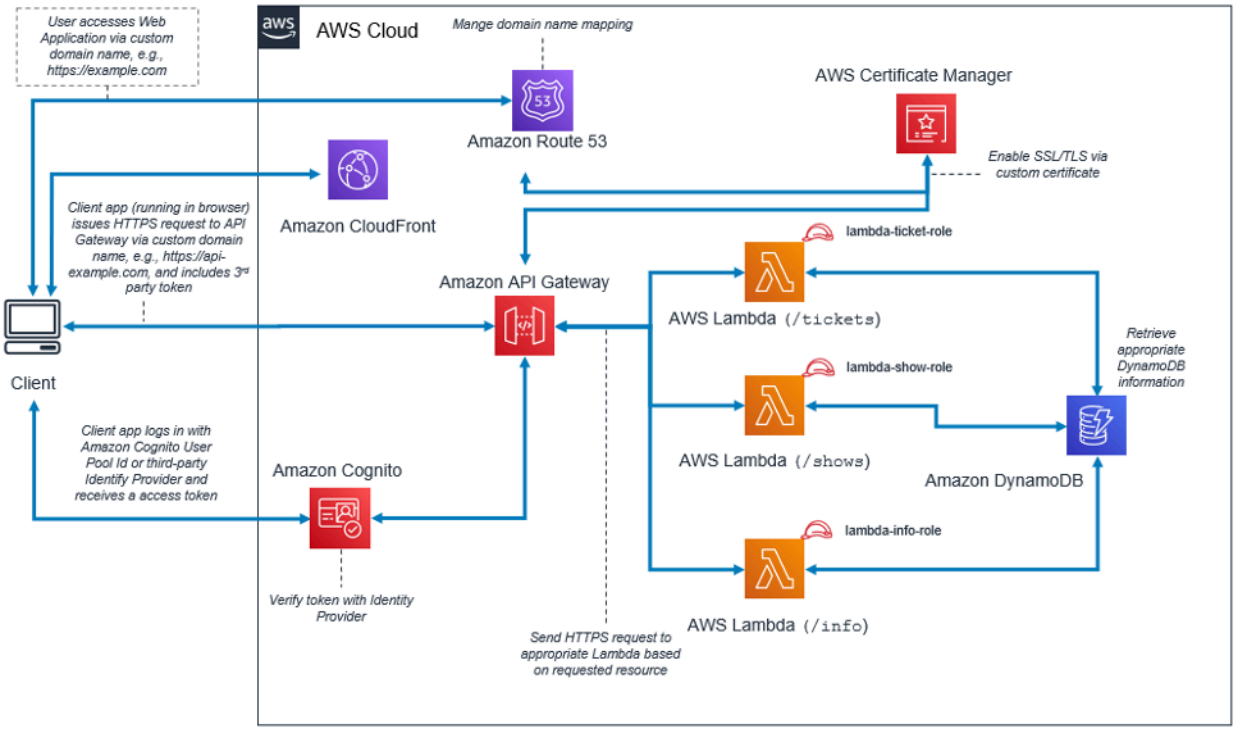


Patrón arquitectónico de aplicaciones de una sola página sin servidor

Tabla 2: Componentes de la aplicación de una sola página

Nivel	Componentes
Presentación	Contenido de sitio web estático alojado en Amazon S3, distribuido por CloudFront. AWS Certificate Manager permite utilizar un certificado SSL/TLS personalizado.
Lógico	API Gateway con AWS Lambda. Esta arquitectura muestra tres servicios expuestos (`/tickets`, `/shows` y `/info`). Los puntos de conexión de API Gateway están protegidos por un autorizador de Lambda. En este método, los usuarios inician sesión a través de un proveedor de identidad de terceros y obtienen tokens de acceso e ID. Estos tokens se incluyen en las llamadas de API Gateway y el autorizador de Lambda valida estos tokens y genera una política de IAM que contiene los permisos de inicio de la API. A cada función de Lambda se le asigna su propio rol de IAM para proporcionar acceso al origen de datos adecuado.
Datos	Los servicios `/tickets` y `/shows` utilizan Amazon DynamoDB. El servicio `/shows` utiliza Amazon ElastiCac he para mejorar el rendimiento de la base de datos. Los errores de caché se envían a DynamoDB. Amazon S3 se utiliza para alojar contenido estático utilizado por `/info service`.

Aplicación web



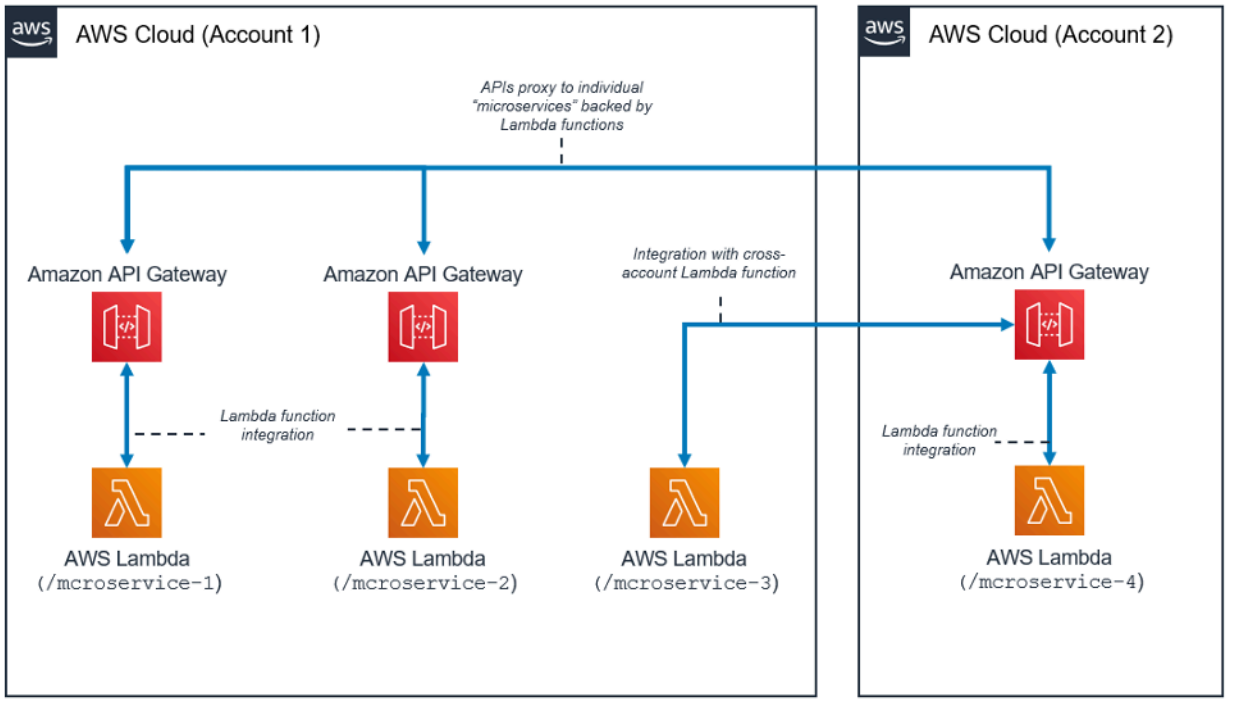
Patrón arquitectónico de aplicación web

Tabla 3: Componentes de aplicaciones web

Nivel	Componentes
Presentación	La aplicación frontend es todo contenido estático (HTML, CSS, JavaScript e imágenes) generado por las utilidades de React como create-react-app. Amazon CloudFront aloja todos estos objetos. La aplicación web, cuando se usa, descarga todos los recursos en el navegador y comienza a ejecutarse desde allí. La aplicación web se conecta al backend y llama a las API.
Lógico	El nivel lógica se construye con funciones de Lambda encabezadas por las API REST de API Gateway.

Nivel	Componentes
	Esta arquitectura muestra varios servicios expuestos. Hay varias funciones de Lambda diferentes, cada una de las cuales maneja un aspecto diferente de la aplicación. Las funciones de Lambda están detrás de API Gateway y se puede acceder a ellas mediante rutas de URL. La autenticación de usuarios se maneja mediante grupos de usuarios de Amazon Cognito o proveedores de usuarios federados . API Gateway utiliza la integración inmediata con Amazon Cognito. Solo después de autenticar a un usuario, el cliente recibirá un token JSON Web Token (JWT) que debe usar al realizar las llamadas a la API. A cada función de Lambda se le asigna su propio rol de IAM para proporcionar acceso al origen de datos adecuado.
Datos	En este ejemplo concreto, DynamoDB se usa para el almacenamiento de datos, pero se pueden usar otros servicios de almacenam iento o bases de datos de Amazon especialm ente diseñados según el caso de uso y el escenario de uso.

Microservicios con Lambda



Patrón arquitectónico de microservicios con Lambda

El patrón de arquitectura de microservicios no está vinculado a la arquitectura típica de tres niveles; sin embargo, este popular patrón puede obtener beneficios significativos del uso de recursos sin servidor.

En esta arquitectura, cada uno de los componentes de la aplicación se desacopla y se implementa y opera de forma independiente. Todo lo que necesita para crear un microservicio es una API creada con Amazon API Gateway y las funciones que AWS Lambda lanza posteriormente. Su equipo puede usar estos servicios para desacoplar y fragmentar su entorno hasta el nivel de granularidad deseado.

En general, un entorno de microservicios puede presentar las siguientes dificultades: sobrecarga reiterada para crear cada microservicio nuevo, problemas con la optimización de la densidad y la utilización del servidor, complejidad de ejecutar varias versiones de varios microservicios simultáneamente y proliferación de requisitos de código del lado del cliente para la integración con muchos servicios distintos.

Cuando crea microservicios con recursos sin servidor, estos problemas se vuelven menos difíciles de resolver y, en algunos casos, simplemente desaparecen. El patrón de microservicios sin servidor reduce la barrera para la creación de cada microservicio posterior (API Gateway incluso permite la clonación de las API existentes y el uso de funciones de Lambda en otras cuentas). La optimización

de la utilización del servidor ya no es relevante con este patrón. Por último, Amazon API Gateway proporciona SDK de cliente generados mediante programación en varios lenguajes populares para reducir la sobrecarga de la integración.

Conclusión

El patrón de arquitectura de varios niveles fomenta la práctica recomendada de crear componentes de aplicaciones que sean fáciles de mantener, desacoplar y escalar. Cuando crea un nivel lógico en el que la integración se produce mediante API Gateway y el cómputo se produce dentro de AWS Lambda, logra estos objetivos al tiempo que reduce la cantidad de esfuerzo para alcanzarlos. Juntos, estos servicios proporcionan una interfaz de API HTTPS para sus clientes y un entorno seguro para aplicar su lógica empresarial y, al mismo tiempo, eliminar la sobrecarga que implica la administración de la infraestructura típica basada en servidores.

Colaboradores

Entre los colaboradores de este documento, están las siguientes personas:

- Andrew Baird, arquitecto de soluciones de AWS
- Bryant Bost, consultor de AWS ProServe
- Stefano Buliani, administrador de productos sénior, tecnología, AWS Mobile
- Vyom Nagrani, administrador de productos sénior, AWS Mobile
- Ajay Nair, administrador de productos sénior, AWS Mobile
- Rahul Popat, arquitecto de soluciones globales
- Brajendra Singh, arquitecto de soluciones sénior

Documentación adicional

Para obtener información adicional, consulte:

- Documentos técnicos y guías de AWS

Revisiones del documento

Para recibir notificaciones sobre las actualizaciones de este documento técnico, suscríbase a la fuente RSS.

update-history-change	update-history-description	update-history-date
Documento técnico actualizado	Actualizado con las nuevas características y patrones de servicio.	20 de octubre de 2021
Documento técnico actualizado	Actualizado con las nuevas características y patrones de servicio.	1 de junio de 2021
Documento técnico actualizado	Actualizado con nuevas funciones de servicio.	25 de septiembre de 2019
Publicación inicial	Publicación del documento técnico.	1 de noviembre de 2015

Avisos

Los clientes son responsables de realizar sus propias evaluaciones de la información contenida en este documento. Este documento: (a) solo tiene fines informativos, (b) representa las prácticas y las ofertas de productos vigentes de AWS, que están sujetas a cambios sin previo aviso, y (c) no crea ningún compromiso ni garantía de AWS y sus empresas afiliadas, proveedores o concesionarios de licencias. Los productos o servicios de AWS se proporcionan “tal cual”, sin garantías, representaciones ni condiciones de ningún tipo, ya sean explícitas o implícitas. Las responsabilidades y obligaciones de AWS en relación con sus clientes se rigen por los acuerdos de AWS, y este documento no modifica ni forma parte de ningún acuerdo entre AWS y sus clientes.