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

Este documento técnico es únicamente de referencia histórica. Es posible que parte del contenido esté desactualizado y que algunos enlaces no estén disponibles.

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.

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

¿Usa Well-Architected?

El marco de AWS Well-Architected le ayuda a entender las ventajas y desventajas de las decisiones que toma al crear sistemas en la nube. Los seis pilares del marco le permitirán aprender las prácticas recomendadas de arquitectura para diseñar y utilizar sistemas fiables, seguros, eficientes, rentables y sostenibles. Mediante AWS Well-Architected Tool, disponible sin costo alguno en la Consola de administración de AWS, puede comparar las cargas de trabajo con estas prácticas recomendadas respondiendo a una serie de preguntas para cada pilar.

Desde el punto de vista de las aplicaciones sin servidor, nos centramos en las mejores prácticas para diseñar la arquitectura de sus aplicaciones sin servidor. AWS

Para obtener asesoramiento más experto y conocer las prácticas recomendadas para la arquitectura en la nube (implementaciones de arquitectura de referencia, diagramas y documentos técnicos), consulte el Centro de arquitectura de AWS.

Introducción

La aplicación de varios niveles (tres niveles, niveles n, etc.) ha sido un patrón arquitectónico fundamental durante décadas y sigue siendo un patrón popular en 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 aplicaciones móviles). UIs
- Nivel lógico: código necesario para traducir las acciones del usuario a la funcionalidad de la aplicación (por ejemplo, las operaciones de la base de datos CRUD y el procesamiento de datos).
- Nivel de datos: medio 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 la aplicación disociados y escalables de forma 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 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 programación de aplicaciones (API) y un modelo de datos
- Código relacionado con la seguridad que garantiza el acceso adecuado a la aplicación

Todos estos ejemplos pueden considerarse componentes «repetitivos» que, si bien son necesarios en aplicaciones de varios niveles, su implementación no varía mucho 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 considerablemente el proceso de implementación de dichas aplicaciones en producción y elimina la sobrecarga asociada a la administración de servidores tradicional. Amazon API Gateway, un servicio para crear y administrar APIs y AWS Lambdaun servicio para ejecutar funciones de código arbitrario, se pueden usar juntos para simplificar la creación de aplicaciones sólidas de varios niveles.

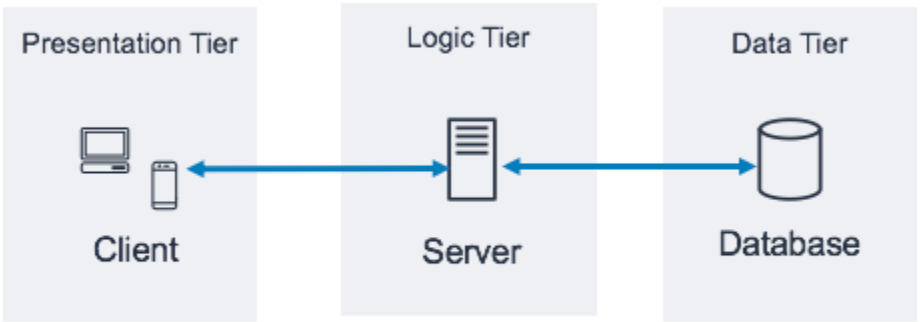
La integración de Amazon API Gateway con AWS Lambda permite que las funciones de código definidas por el usuario se inicien directamente a través de solicitudes HTTPS. Independientemente del volumen de solicitudes, tanto API Gateway como Lambda se escalan automáticamente para adaptarse exactamente a las necesidades de su aplicación (consulte API Gateway, las cuotas de Amazon API Gateway y las notas importantes para obtener información sobre escalabilidad). Al combinar estos dos servicios, puede crear un nivel que le permita escribir solo el código que sea importante para su aplicación y no centrarse en otros aspectos indiferenciantes de la implementación de una arquitectura de varios niveles, como la arquitectura para alta disponibilidad, la redacción de la administración de clientes SDKs, servidores y sistemas operativos (OS), 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 la 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 o 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.

Descripción general de la arquitectura de tres niveles

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



Patrón arquitectónico para una aplicación de tres niveles

Hay muchos recursos en línea excelentes en los que puede obtener más información sobre el patrón general de arquitectura 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 de lógica 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 AWS Lambda puede tener el mayor impacto en comparación con una implementación tradicional basada en servidores. 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 necesitar miles de servidores; sin embargo, si utiliza Amazon API Gateway, AWS Lambda usted no es responsable de la administración de los servidores en ningún aspecto. Además, al utilizar estos servicios gestionados en conjunto, obtendrá las siguientes ventajas:

- AWS Lambda:
  - No hay ningún sistema operativo que elegir, proteger, parchear o administrar
  - No hay servidores con el tamaño, el monitoreo o la escalabilidad correctos
  - Reducción del riesgo de costes derivado del sobreaprovisionamiento
  - Reducción del riesgo que supone para su rendimiento el aprovisionamiento insuficiente

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

AWS Lambda

AWS Lambda es un servicio informático que permite ejecutar funciones de código arbitrario sin aprovisionar, administrar ni escalar servidores. Los lenguajes compatibles incluyen Python, Ruby, Java, Go y.NET. Las funciones Lambda se ejecutan en un contenedor gestionado y aislado y se lanzan en respuesta a un evento que puede ser uno de los varios activadores programáticos disponibles, denominado fuente de eventos. AWS Para obtener más información sobre los idiomas y las fuentes de eventos compatibles, consulte Lambda FAQs.

Muchos casos de uso populares de Lambda giran en torno a 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 desde Amazon Kinesis. Cuando se utiliza junto con Amazon API Gateway, una función de Lambda ofrece la funcionalidad de un servicio web típico: inicia el código en

respuesta a una solicitud HTTPS de un cliente; API Gateway actúa como puerta de entrada al nivel lógico e AWS Lambda invoca el código de la aplicación.

Aquí reside la lógica empresarial, sin necesidad de servidores

Lambda requiere que escriba funciones de código, denominadas controladores, que se ejecutarán cuando se inicien mediante un evento. Para usar Lambda con API Gateway, puede configurar API Gateway para lanzar funciones de controlador cuando se produzca una solicitud HTTPS a su API. En una arquitectura de varios niveles sin servidor, cada uno de los APIs que cree en API Gateway se integrará con una función de Lambda (y el controlador interno) que invoca la lógica empresarial necesaria.

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

La creación o actualización de una función de Lambda requiere cargar el código como un paquete de despliegue 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, ejecutar AWS Command Line Interface (AWS CLI) o ejecutar la infraestructura como plantillas de código o marcos como CloudFormation, AWS Serverless Application Model(AWS SAM) o AWS Cloud Development Kit (AWS CDK). Al crear la función con alguno de estos métodos, debe especificar qué método del paquete de implementación servirá como gestor de solicitudes. Puede reutilizar el mismo paquete de despliegue para varias definiciones de funciones de Lambda, donde cada función de Lambda puede tener un controlador único dentro del mismo paquete de despliegue.

Seguridad Lambda

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

Cada función de Lambda asume una función de IAM que se asigna cuando se implementa la función de Lambda. Esta función de IAM define los demás AWS servicios y recursos con los que puede

interactuar su función Lambda (por ejemplo, Amazon DynamoDB Amazon S3). En el contexto de la función Lambda, esto se denomina rol de ejecución.

No almacene información confidencial dentro de una función Lambda. IAM gestiona el acceso a AWS los servicios mediante la función de ejecución de Lambda; si necesita acceder a otras credenciales (por ejemplo, credenciales de bases de datos y claves de API) desde la función de Lambda, puede AWS Key Management Serviceutilizar AWS KMS() con variables de entorno o utilizar un servicio como Secrets AWSManager para proteger esta información 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 las funciones de Lambda: cada vez que su función recibe una notificación de evento, AWS Lambda localiza la capacidad disponible en su flota de procesamiento y ejecuta el código con las configuraciones de tiempo de ejecución, memoria, disco y tiempo de espera que usted defina. Con este patrón, AWS puede iniciar tantas copias de la función como necesite.

Un nivel lógico basado en Lambda siempre tiene el tamaño adecuado para las necesidades de sus clientes. La capacidad de absorber rápidamente los picos de tráfico mediante el escalado gestionado y el inicio simultáneo de código, combinada con los pay-per-use precios de Lambda, le permite satisfacer siempre las solicitudes de los clientes y, al mismo tiempo, no pagar por la capacidad informática inactiva.

Implementación y administración sin servidor

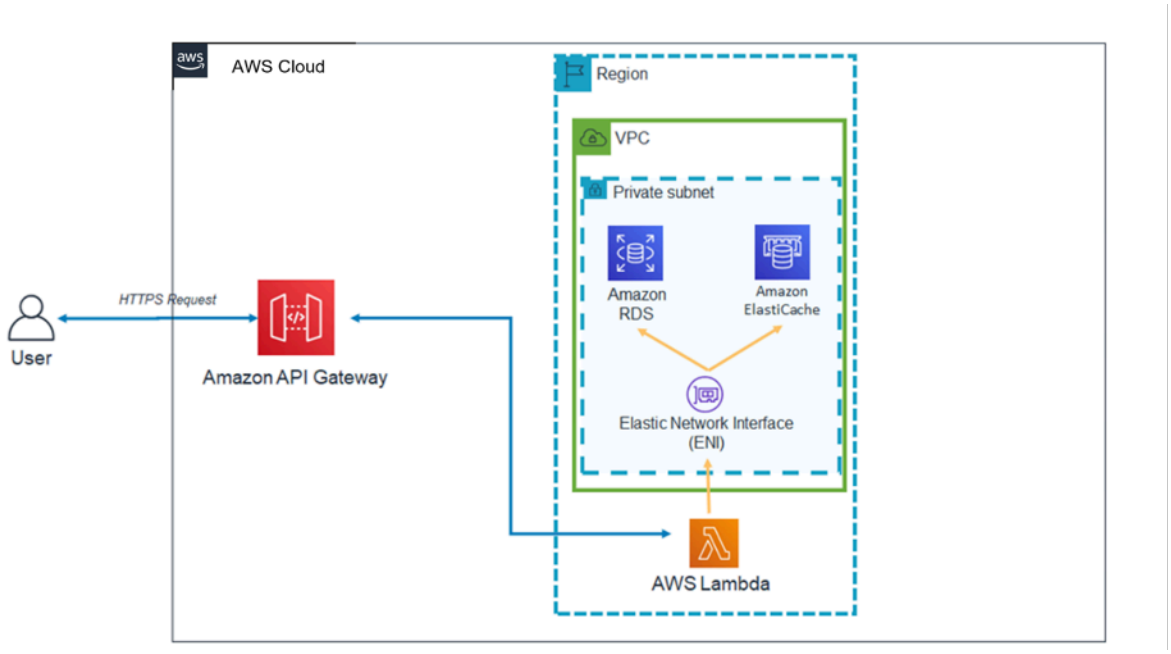
Para ayudarlo a implementar y administrar sus funciones de Lambda, utilice el modelo de aplicaciones sin servidor de AWS (AWS SAM), un marco de código abierto que incluye:

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

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

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

Puede configurar una función de Lambda para que se conecte a subredes privadas de una nube privada virtual (VPC) de su AWS cuenta si desea que la función de Lambda acceda a recursos que no puede exponer públicamente, como una instancia de base de datos privada. Al conectar una función a una VPC, Lambda crea una interfaz de red elástica para cada subred de la configuración de VPC de la función, y la interfaz de red elástica se utiliza para acceder a los 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 la lógica empresarial pueden quedar inaccesibles a través de Internet. La VPC también garantiza que la única forma de interactuar con los datos de Internet sea a través de las funciones APIs que haya definido y de código Lambda que haya escrito.

Amazon API Gateway

Amazon API Gateway es un servicio totalmente gestionado que permite a los desarrolladores crear, publicar, mantener, supervisar y proteger APIs a cualquier escala.

Los clientes (es decir, los niveles de presentación) se integran con lo APIs expuesto mediante API Gateway mediante solicitudes HTTPS estándar. La aplicabilidad de la APIs exposición a través de API Gateway a 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 finales 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 REST y HTTP de APIs. Una API de API Gateway se compone de recursos y métodos. Un recurso es una entidad lógica a la que puede acceder una aplicación 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 Lambda, es decir, cuando llama a la API a través del punto final HTTPS expuesto en API Gateway, API Gateway invoca la función Lambda.

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

Integraciones de proxy

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

Integraciones que no son de proxy

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

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

Rendimiento estable de las API en todas las regiones

Cada implementación de Amazon API Gateway incluye una CloudFront distribución de Amazon interna. CloudFrontes un servicio de entrega de contenido que utiliza la red global de ubicaciones periféricas de Amazon como puntos de conexión para los clientes que utilizan su API. Esto ayuda a reducir la latencia de respuesta de tu API. Al utilizar múltiples ubicaciones de borde en todo el mundo, Amazon CloudFront también proporciona capacidades para combatir los escenarios de ataques de denegación de servicio (DDoS) distribuidos. Para obtener más información, consulte el documento técnico de AWS Best Practices for DDo S Resiliency.

Puedes mejorar el rendimiento de solicitudes de API específicas mediante 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 costo total.

Fomente la innovación y reduzca los gastos generales con las funciones integradas

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

Durante las fases iniciales de desarrollo de la aplicación, a menudo se descuida la implementación del registro y la recopilación de métricas para ofrecer una nueva aplicación más rápidamente. Esto puede generar problemas técnicos y riesgos operativos al implementar estas funciones en una aplicación que se ejecuta en producción. Amazon API Gateway se integra perfectamente con Amazon CloudWatch, que recopila y procesa datos sin procesar de API Gateway para convertirlos en métricas legibles y prácticamente en tiempo real para supervisar la ejecución de las API. API Gateway también admite el registro de acceso con informes configurables y el AWS X-Rayseguimiento para la depuración. Cada una de estas funciones no requiere escribir código y

se pueden ajustar en las aplicaciones que se ejecutan en producción sin poner en peligro la lógica empresarial principal.

Es posible que se desconozca la vida útil total de una aplicación o que se sepa que es de corta duración. La creación de un modelo de negocio para crear este tipo de aplicaciones puede resultar más fácil si su punto de partida ya incluye las funciones gestionadas que ofrece API Gateway y si solo incurre en costes de infraestructura una vez que APIs comience a recibir solicitudes. Para obtener más información, consulte los precios de Amazon API Gateway.

Realice iteraciones con rapidez y manténgase ágil

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

Implementación por etapas

Al implementar una API en API Gateway, debes asociar la implementación a una etapa de API Gateway: cada etapa es una instantánea de la API y está disponible para que las aplicaciones cliente la llamen. Con esta convención, puede implementar fácilmente aplicaciones en etapas de desarrollo, prueba, puesta en escena o producción, y mover las implementaciones de una etapa a otra. Cada vez que despliegas tu API en una fase, creas 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 que las nuevas funcionalidades se lanzan como una versión de API independiente.

Integración desacoplada con Lambda

La integración entre la API de API Gateway y la función 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 Lambda directamente en la API, puede configurar la variable de etapa en la API que puede apuntar a un alias concreto de la función 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 la función de Lambda detrás del alias de Lambda para una etapa determinada.

Implementación de la versión Canary

La versión Canary es una estrategia de desarrollo de software en la que se implementa una nueva versión de una API con fines de prueba, y la versión base permanece implementada como una

versión de producción para operaciones normales en la misma fase. En una implementación de versión estándar, el tráfico total de la API se divide aleatoriamente en una versión de producción y una versión estándar con una proporción preconfigurada. APIs en API Gateway se puede configurar para la implementación de la versión canaria a fin de probar nuevas funciones con un conjunto limitado de usuarios.

Nombres de dominio personalizados

Puedes proporcionar a la API un nombre de URL intuitivo y adecuado para las empresas en lugar de la URL proporcionada por API Gateway. API Gateway proporciona funciones para configurar un dominio personalizado para APIs. Con los nombres de dominio personalizados, puedes configurar el nombre de host de tu API y elegir una ruta base de varios niveles (por ejemplo,, `myservicemyservice/cat/v1`, o`myservice/dog/v2`) para asignar la URL alternativa a tu API.

Dé prioridad a la seguridad de

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 las diferentes formas en las que Amazon API Gateway contribuye a proteger su nivel lógico:

Seguridad de tránsito

Todas las solicitudes que se le APIs envíen se pueden realizar a través de HTTPS para permitir el cifrado en tránsito.

API Gateway proporciona SSL/TLS certificados integrados: si utiliza la opción de nombre de dominio personalizado para uso público APIs, puede proporcionar su propio SSL/TLS certificado mediante AWS Certificate Manager. API Gateway también admite la autenticación TLS mutua (mTLS). El TLS mutuo mejora la seguridad de su API y ayuda a proteger sus datos de ataques, como la suplantación de identidad de clientes o los ataques intermedios. man-in-the

Autorización de API

A cada resource/method combinación 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 AWS Identity and Access Management (IAM).

Existen tres métodos generales para añadir autorización a una API en API Gateway:

- Funciones y políticas de IAM: los clientes utilizan la autorización de AWS Signature versión 4 (SiGv4) y las políticas de IAM para acceder a la API. Las mismas credenciales pueden restringir o

permitir el acceso a otros AWS servicios y recursos 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 los 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 portador (por ejemplo, OAuth SAML) o utilice parámetros de solicitud para identificar a los usuarios.

Restricciones 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 las claves de API con CloudWatch.

API Gateway admite la regulación, los límites de velocidad y los límites de velocidad de ráfaga para cada método de tu API.

Privada APIs

Con API Gateway, puede crear un REST privado al APIs que solo se puede acceder desde su nube privada virtual en Amazon VPC mediante un punto de enlace de interfaz de VPC. Se trata de una interfaz de red de punto de enlace que se crea en la VPC.

Con las políticas de recursos, puede habilitar o denegar el acceso a su API desde puntos de enlace seleccionados VPCs y de VPC, incluidas las cuentas de AWS. Cada punto de conexión se puede utilizar para acceder a varios puntos privados. APIs 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 mediante AWS WAF

Las personas con acceso a Internet APIs son vulnerables a los ataques malintencionados. AWS WAF es un firewall de aplicaciones web que ayuda a protegerse APIs de este tipo de ataques. APIs Protege contra los ataques web más comunes, como las inyecciones de SQL y los ataques de secuencias de comandos entre sitios. Puede usarlo AWS WAFcon API Gateway para ayudar a proteger APIs.

Nivel de datos

Su uso AWS Lambda como nivel lógico no limita las opciones de almacenamiento de datos disponibles en su nivel de datos. Las funciones de Lambda se conectan a cualquier opción de almacenamiento de datos al incluir el controlador de base de datos correspondiente en el paquete de despliegue de Lambda y utilizan credenciales cifradas o de acceso basado en roles de IAM (a través de Secrets Manager). AWS KMS AWS

La elección de un almacén de datos para la aplicación depende en gran medida de los requisitos de la aplicación. AWS ofrece varios almacenes de datos sin servidor y 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 del sector.

Amazon Aurora es una base de datos relacional compatible con MySQL y PostgreSQL creada para la nube, que 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 tradicional y sin servidor.

Amazon DynamoDB es una base de datos de documentos y valores clave que ofrece un rendimiento de milisegundos de un solo dígito a cualquier escala. Se trata de una base de datos duradera, multirregional, multiactiva y totalmente gestionada, sin servidores, con funciones integradas de seguridad, copia de seguridad y restauración y almacenamiento en caché en memoria para aplicaciones a escala de Internet.

Amazon Timestream es un servicio de base de datos de series temporales rápido, escalable y totalmente gestionado para aplicaciones operativas y de IoT que facilita el almacenamiento y el análisis de billones de eventos por día a una décima parte del coste de las bases de datos relacionales. Impulsados por el auge de los dispositivos de IoT, los sistemas de TI y las máquinas industriales inteligentes, los datos de series temporales (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 contable totalmente gestionada que proporciona un registro de transacciones transparente, inmutable y verificable criptográficamente, propiedad de una autoridad central de confianza. Amazon QLDB realiza un

seguimiento de todos y cada uno de los cambios en los datos de las aplicaciones y mantiene un historial completo y verificable de los cambios a lo largo del tiempo.

Amazon Keyspaces (para Apache Cassandra) es un servicio de bases de datos escalable, de alta disponibilidad y gestionado compatible con Apache Cassandra. Con Amazon Keyspaces, puede ejecutar sus cargas de trabajo de Cassandra con el mismo código de aplicación de Cassandra y las mismas herramientas de desarrollador que AWS utiliza actualmente. No tiene que aprovisionar, parchear ni administrar los servidores, ni tampoco tiene que instalar, mantener ni utilizar software. Amazon Keyspaces no tiene servidor, por lo que solo paga por los recursos que utilice y el servicio puede escalar automáticamente las tablas hacia arriba y hacia abajo en respuesta al tráfico de las aplicaciones.

Amazon Elastic File System (Amazon EFS) proporciona un sistema de archivos elástico set-and-forget, simple y sin servidor que le permite compartir datos de archivos sin aprovisionar ni administrar el almacenamiento. Se puede usar con los servicios en la nube y los recursos locales de AWS, y está diseñado para escalarse bajo demanda a petabytes sin interrumpir las aplicaciones. Con Amazon EFS, puede ampliar y reducir sus sistemas de archivos automáticamente a medida que agrega y elimina archivos, lo que elimina la necesidad de aprovisionar y administrar la capacidad para adaptarse al crecimiento. Amazon EFS se puede montar con la función Lambda, lo que lo convierte en una opción viable de almacenamiento de archivos. APIs

Opciones de almacenamiento de datos sin servidor

Amazon Relational Database Service (Amazon RDS) es un servicio web gestionado 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 se ejecuta en varios tipos de instancias de bases de datos diferentes que están optimizadas para la memoria, el rendimiento o la E/S.

Amazon Redshift es un servicio de almacenamiento de datos en la nube totalmente gestionado y a escala de petabytes.

Amazon ElastiCache es una implementación totalmente gestionada de Redis o Memcached. Implemente, ejecute y escale sin problemas los populares almacenes de datos en memoria compatibles con código abierto.

Amazon Neptune es un servicio de base de datos de gráficos rápido, fiable y totalmente gestionado que facilita la creación y ejecución de aplicaciones que funcionan con conjuntos de datos altamente

conectados. Neptune admite los modelos de gráficos más populares (gráficos de propiedades y el marco de descripción de recursos (RDF) del W3C, y sus respectivos lenguajes de consulta, lo que le permite crear consultas fácilmente que naveguen de manera eficiente por conjuntos de datos altamente conectados.

Amazon DocumentDB (compatible con MongoDB) es un servicio de base de datos de documentos rápido, escalable, de alta disponibilidad y totalmente gestionado que admite cargas de trabajo de MongoDB.

Por último, también puede utilizar los 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 finales REST de API Gateway expuestos en Internet. Cualquier cliente o dispositivo compatible con HTTPS puede comunicarse con estos puntos finales, 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.). En función de sus requisitos, el nivel de presentación puede utilizar las siguientes ofertas AWS sin servidor:

- Amazon Cognito: un servicio de sincronización de datos e identidad de usuarios sin servidor que le permite añadir el registro, el inicio de sesión y el control de acceso de los usuarios a sus aplicaciones web y móviles de forma rápida y eficaz. Amazon Cognito se amplía a millones de usuarios y admite el inicio de sesión con proveedores de identidad social, como Facebook, Google y Amazon, y con 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 disponer de un servidor web. Puede utilizarla CloudFront como una red de entrega de contenido gestionada (CDN) para mejorar el rendimiento y permitir el SSL/TL uso de certificados gestionados o personalizados.

AWS Amplifyes un conjunto de herramientas y servicios que se pueden usar juntos o por separado para ayudar a los desarrolladores web y móviles de front-end a crear aplicaciones completas y escalables, con la tecnología de. AWS Amplify ofrece un servicio totalmente gestionado para implementar y alojar aplicaciones web estáticas en todo el mundo, gestionado por la fiable CDN de Amazon, con cientos de puntos de presencia en todo el mundo y con CI/CD flujos de trabajo integrados que aceleran el ciclo de lanzamiento de las aplicaciones. Amplify es compatible con marcos web populares JavaScript, como React, Angular, Vue, Next.js, y plataformas móviles, como Android, iOS, React Native, Ionic y Flutter. Según las configuraciones de red y los requisitos de la aplicación, es posible que deba habilitar su API Gateway APIs para que cumpla con el uso compartido de recursos entre orígenes (CORS). La conformidad con el CORS permite a los navegadores web invocar directamente tus APIs páginas web estáticas.

Cuando despliega un sitio web con CloudFront, se le proporciona un nombre de CloudFront dominio para acceder a su aplicación (por ejemplo,). `d2d47p2vcczkh2.cloudfront.net` Puede usar Amazon Route 53 para registrar nombres de dominio y dirigirlos a su CloudFront distribución, o bien dirigir los nombres de dominio que ya posee a su CloudFront distribución. Esto permite a los usuarios acceder a su sitio con un nombre de dominio conocido. 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 APIs con nombres de dominio conocidos.

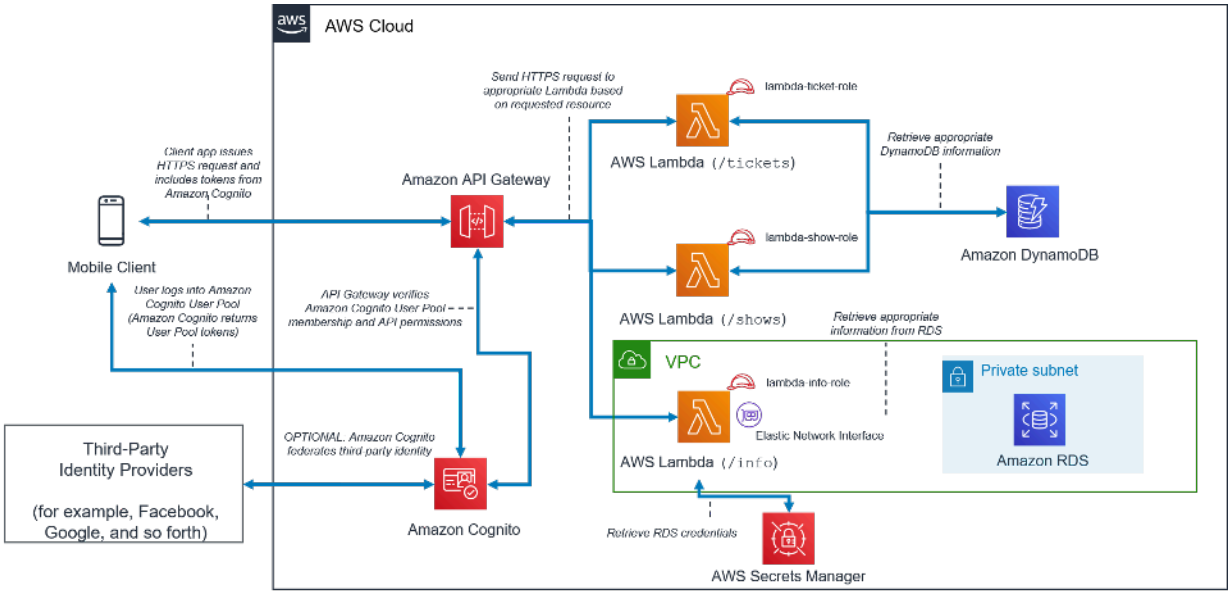
Ejemplos de patrones de arquitectura

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

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

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

Backend móvil



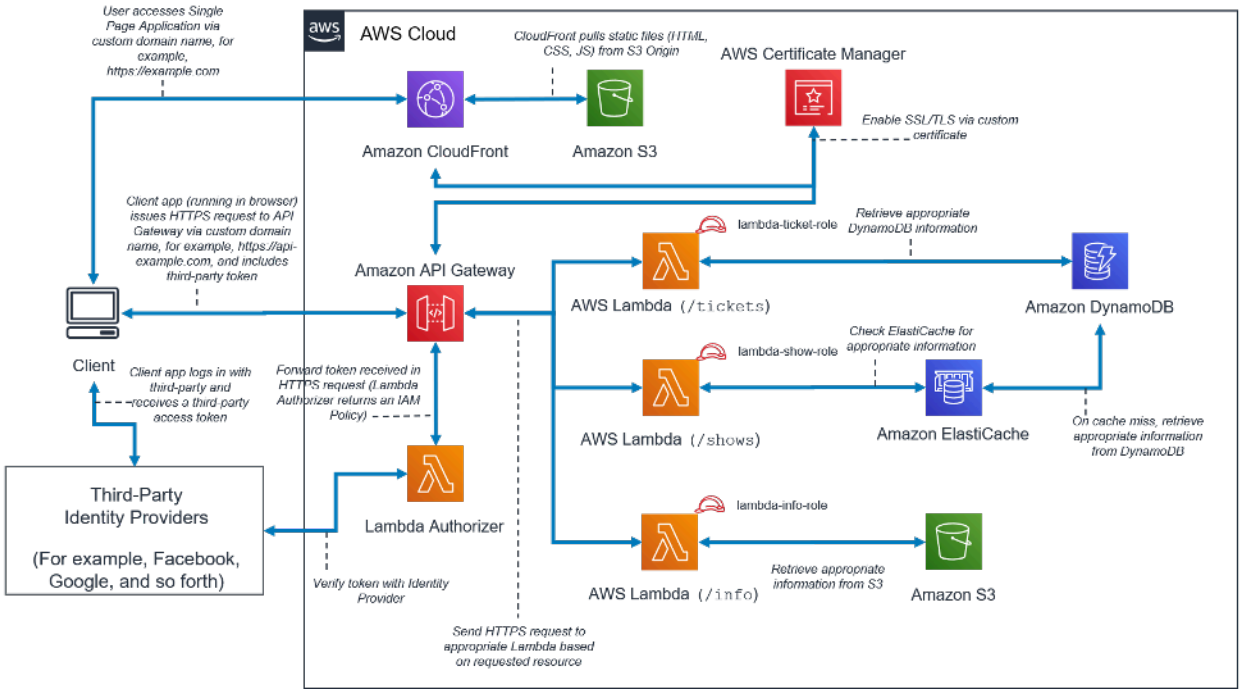
Patrón arquitectónico para el backend móvil sin servidor

Tabla 1: Componentes del nivel de backend móvil

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

Nivel	Componentes
	A cada función de Lambda se le asigna su propia función de Identity and Access Management (IAM) (Identity and Access Management, IAM) para proporcionar acceso a la fuente de datos adecuada.
Datos	DynamoDB se utiliza para `/tickets` los servicios y. `/shows` Para el `/info` servicio se utiliza Amazon RDS. Esta función Lambda recupera las credenciales de Amazon RDS de Secrets AWS Manager y utiliza una interfaz de red elástica para acceder a la subred privada.

Aplicación de una sola página

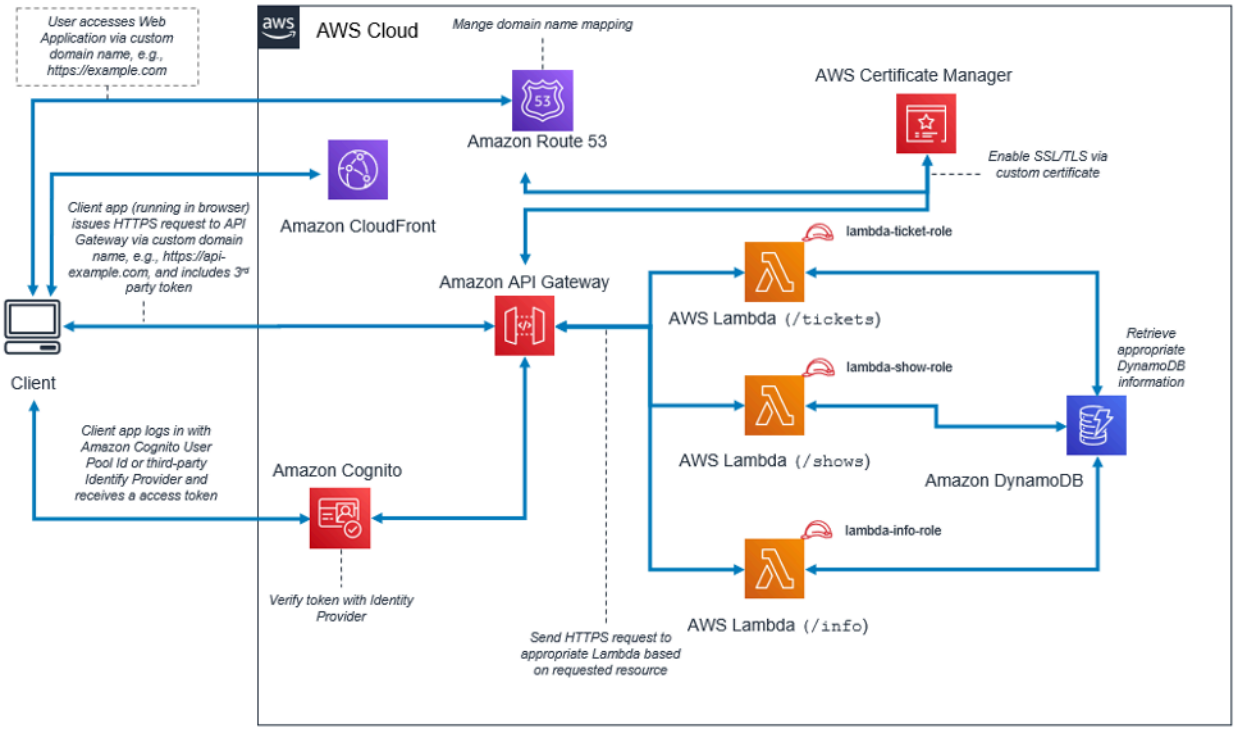


Patrón arquitectónico para aplicaciones de una sola página sin servidor

Tabla 2: Componentes de la aplicación de una sola página

Nivel	Componentes
Presentación	Contenido de sitios web estáticos alojado en Amazon S3, distribuido por CloudFront. AWS Certificate Manager permite utilizar un certificado SSL/TLS personalizado.
Logic (Lógica)	API Gateway con AWS Lambda. Esta arquitectura muestra tres servicios expuestos (`/tickets`/`shows`, y`/info`). Los puntos finales de API Gateway están protegidos por un autorizador Lambda. Con este método, los usuarios inician sesión a través de un proveedor de identidad externo y obtienen tokens de acceso e identificación. Estos tokens se incluyen en las llamadas a API Gateway, y el autorizador de Lambda los valida y genera una política de IAM que contiene los permisos de inicio de la API. A cada función de Lambda se le asigna su propia función de IAM para proporcionar acceso a la fuente de datos adecuada.
Datos	Amazon DynamoDB se utiliza para `/tickets` los servicios y. `/shows` El `/shows` servicio ElastiCache utiliza Amazon 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



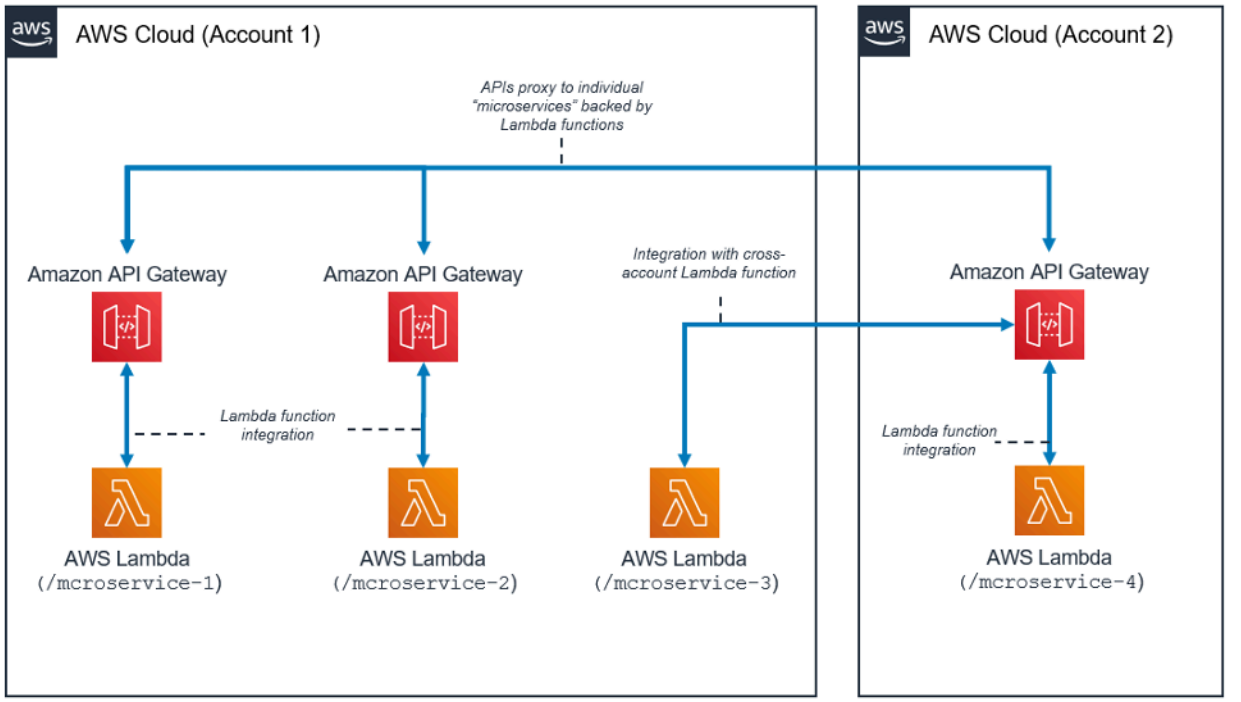
Patrón arquitectónico para la aplicación web

Tabla 3: Componentes de la aplicación web

Nivel	Componentes
Presentación	La aplicación front-end es todo contenido estático (HTML, CSS JavaScript e imágenes) generado por utilidades como create-react-app React. Amazon CloudFront aloja todos estos objetos. La aplicación web, cuando se usa, descarga todos los recursos al navegador y comienza a ejecutarse desde allí. La aplicación web se conecta al backend llamando al APIs.
Logic (Lógica)	La capa lógica se crea con funciones de Lambda dirigidas por API Gateway REST. APIs

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

Microservicios con Lambda



Patrón arquitectónico para microservicios con Lambda

El patrón de arquitectura de microservicios no está limitado a la arquitectura típica de tres niveles; sin embargo, este patrón popular puede reportar beneficios significativos al usar recursos sin servidor.

En esta arquitectura, cada uno de los componentes de la aplicación está desacoplado 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 funciones lanzadas AWS Lambda posteriormente por él. Su equipo puede usar estos servicios para desacoplar y fragmentar su entorno con el nivel de granularidad deseado.

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

Cuando se crean 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 funciones Lambda existentes APIs y el uso de las mismas en otras cuentas).

La optimización de la utilización de los servidores ya no es relevante con este patrón. Por último, Amazon API Gateway proporciona un cliente generado mediante programación SDKs en varios lenguajes populares para reducir la sobrecarga de integración.

Conclusión

El patrón de arquitectura de varios niveles fomenta la mejor práctica de crear componentes de aplicaciones que sean fáciles de mantener, desacoplar y escalar. Cuando se crea un nivel lógico en el que la integración se realiza mediante API Gateway y el cómputo se lleva a cabo en él AWS Lambda, se logran estos objetivos y se reduce el esfuerzo necesario para lograrlos. En conjunto, estos servicios proporcionan una interfaz de API HTTPS para sus clientes y un entorno seguro en el que aplicar su lógica empresarial, al tiempo que eliminan la sobrecarga que implica la administración de una infraestructura típica basada en servidores.

Colaboradores

Los colaboradores de este documento son:

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

Documentación adicional

Para obtener información adicional, consulte los siguientes recursos:

- Guías y documentos técnicos de AWS

Revisiones del documento

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

Cambio	Descripción	Fecha
Actualizaciones menores	Correcciones de errores y numerosos cambios menores en todo momento.	1 de abril de 2022
Documento técnico actualizado	Actualizado para incorporar nuevas características y patrones de servicio.	20 de octubre de 2021
Documento técnico actualizado	Actualizado para incorporar nuevas características y patrones de servicio.	1 de junio de 2021
Documento técnico actualizado	Actualizado para incorporar nuevas funciones del servicio.	25 de septiembre de 2019
Publicación inicial	Documento técnico publicado.	1 de noviembre de 2015

Avisos

Es responsabilidad de los clientes realizar su propia evaluación independiente de la información que contiene este documento. El presente documento: (a) tiene solo fines informativos, (b) representa las ofertas y prácticas actuales de los productos de AWS, que están sujetas a cambios sin previo aviso, y (c) no supone ningún compromiso ni garantía por parte de AWS y sus filiales, proveedores o licenciantes. Los productos o servicios de AWS se proporcionan “tal cual” sin garantías, declaraciones ni condiciones de ningún tipo, ya sean expresas o implícitas. Las responsabilidades y obligaciones de AWS con respecto a sus clientes se controlan mediante los acuerdos de AWS y este documento no forma parte ni modifica ningún acuerdo entre AWS y sus clientes.