AWS-Whitepaper

Serverlose mehrstufige AWS-Architekturen mit Amazon API Gateway und AWS Lambda

Serverlose mehrstufige AWS-Architekturen mit Amazon API Gateway und AWS Lambda: AWS-Whitepaper

Table of Contents

Dieses Whitepaper dient nur als historische Referenz. Einige Inhalte sind möglicherweise veraltet und einige Links sind möglicherweise nicht verfügbar.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.

Serverlose mehrstufige AWS-Architekturen mit Amazon API Gateway und AWS Lambda

Datum der Veröffentlichung: 20. Oktober 2021 () Dokumentversionen

Überblick

Dieses Whitepaper zeigt, wie Innovationen von Amazon Web Services (AWS) genutzt werden können, um die Art und Weise zu ändern, wie Sie mehrstufige Architekturen entwerfen und beliebte Muster wie Microservices, mobile Backends und Single-Page-Anwendungen implementieren. Architekten und Entwickler können Amazon API Gateway und andere Services verwenden AWS Lambda, um die Entwicklungs- und Betriebszyklen zu reduzieren, die für die Erstellung und Verwaltung mehrschichtiger Anwendungen erforderlich sind.

Sind Sie Well-Architected?

Das AWS Well-Architected Framework hilft Ihnen dabei, die Vor- und Nachteile der Entscheidungen zu verstehen, die Sie beim Aufbau von Systemen in der Cloud treffen. Die sechs Säulen des Frameworks ermöglichen es Ihnen, bewährte Architekturpraktiken für den Entwurf und Betrieb zuverlässiger, sicherer, effizienter, kostengünstiger und nachhaltiger Systeme kennenzulernen. Mithilfe des AWS Well-Architected Tool, das kostenlos im verfügbar ist AWS-Managementkonsole, können Sie Ihre Workloads anhand dieser bewährten Methoden überprüfen, indem Sie für jede Säule eine Reihe von Fragen beantworten.

Im Bereich Serverless Application Lens konzentrieren wir uns auf bewährte Methoden für die Architektur Ihrer serverlosen Anwendungen. AWS

Weitere Expertentipps und Best Practices für Ihre Cloud-Architektur — Referenzarchitekturbereitstellungen, Diagramme und Whitepapers — finden Sie im Architecture Center.AWS

Einführung

Die mehrstufige Anwendung (dreistufig, n-Tier usw.) ist seit Jahrzehnten ein Eckpfeiler der Architektur und ist nach wie vor ein beliebtes Muster für benutzerorientierte Anwendungen. Obwohl die Sprache, die zur Beschreibung einer mehrschichtigen Architektur verwendet wird, unterschiedlich ist, besteht eine mehrschichtige Anwendung im Allgemeinen aus den folgenden Komponenten:

- Präsentationsebene: Komponente, mit der der Benutzer direkt interagiert (z. B. Webseiten und mobile Apps). UIs
- Logikebene: Code, der erforderlich ist, um Benutzeraktionen in Anwendungsfunktionen zu übersetzen (z. B. CRUD-Datenbankoperationen und Datenverarbeitung).
- Datenschicht: Speichermedien (z. B. Datenbanken, Objektspeicher, Caches und Dateisysteme), auf denen die für die Anwendung relevanten Daten gespeichert sind.

Das mehrstufige Architekturmuster bietet einen allgemeinen Rahmen, der sicherstellt, dass entkoppelte und unabhängig skalierbare Anwendungskomponenten separat entwickelt, verwaltet und gewartet werden können (oft von unterschiedlichen Teams).

Aufgrund dieses Musters, bei dem das Netzwerk (eine Ebene muss einen Netzwerkaufruf tätigen, um mit einer anderen Ebene zu interagieren) als Grenze zwischen den Ebenen fungiert, erfordert die Entwicklung einer mehrschichtigen Anwendung häufig die Erstellung vieler undifferenzierter Anwendungskomponenten. Zu diesen Komponenten gehören:

- Code, der eine Nachrichtenwarteschlange für die Kommunikation zwischen Ebenen definiert
- Code, der eine Anwendungsprogrammierschnittstelle (API) und ein Datenmodell definiert
- Sicherheitsrelevanter Code, der einen angemessenen Zugriff auf die Anwendung gewährleistet

All diese Beispiele können als Standardkomponenten betrachtet werden, die zwar für mehrstufige Anwendungen notwendig sind, sich aber in ihrer Implementierung von einer Anwendung zur nächsten kaum unterscheiden.

AWS bietet eine Reihe von Services, die die Erstellung serverloser Multi-Tier-Anwendungen ermöglichen. Dadurch wird der Prozess der Bereitstellung solcher Anwendungen in der Produktion erheblich vereinfacht und der mit der herkömmlichen Serververwaltung verbundene Aufwand entfällt. Amazon API Gateway, ein Service zum Erstellen und Verwalten APIs sowie ein Service

zum Ausführen beliebiger Codefunktionen, können zusammen verwendet werden, um die Erstellung robuster mehrstufiger Anwendungen zu vereinfachen. AWS Lambda

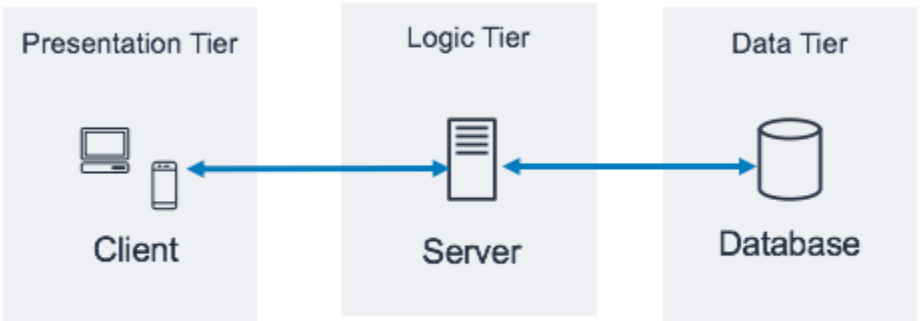
AWS Lambda Durch die Integration von Amazon API Gateway können benutzerdefinierte Codefunktionen direkt über HTTPS-Anfragen initiiert werden. Unabhängig vom Anforderungsvolumen skalieren sowohl API Gateway als auch Lambda automatisch, um genau die Anforderungen Ihrer Anwendung zu unterstützen (Informationen zur Skalierbarkeit finden Sie unter API Gateway Amazon API Gateway Gateway-Kontingente und wichtige Hinweise). Durch die Kombination dieser beiden Dienste können Sie eine Ebene erstellen, die es Ihnen ermöglicht, nur den Code zu schreiben, der für Ihre Anwendung wichtig ist, und sich nicht auf verschiedene andere undifferenzierte Aspekte der Implementierung einer mehrstufigen Architektur zu konzentrieren, z. B. Architektur für hohe Verfügbarkeit, Schreiben von Client- SDKs, Server- und Betriebssystemmanagement (OS), Skalierung und Implementierung eines Client-Autorisierungsmechanismus.

API Gateway und Lambda ermöglichen die Erstellung einer serverlosen Logikschicht. Abhängig von Ihren Anwendungsanforderungen bietet AWS auch Optionen zum Erstellen einer serverlosen Präsentationsebene (z. B. mit Amazon CloudFront und Amazon Simple Storage Service) und einer Datenebene (z. B. Amazon Aurora, Amazon DynamoDB).

Dieses Whitepaper konzentriert sich auf das beliebteste Beispiel für eine mehrstufige Architektur, die dreistufige Webanwendung. Sie können dieses mehrstufige Muster jedoch weit über eine typische dreistufige Webanwendung hinaus anwenden.

Überblick über die dreistufige Architektur

Die dreistufige Architektur ist die beliebteste Implementierung einer mehrstufigen Architektur und besteht aus einer einzigen Präsentationsebene, einer Logikebene und einer Datenebene. Die folgende Abbildung zeigt ein Beispiel für eine einfache, generische dreistufige Anwendung.



Architekturmuster für eine dreistufige Anwendung

Es gibt viele großartige Online-Ressourcen, in denen Sie mehr über das allgemeine dreistufige Architekturmuster erfahren können. Dieses Whitepaper konzentriert sich auf ein bestimmtes Implementierungsmuster für diese Architektur unter Verwendung von Amazon API Gateway und AWS Lambda.

Serverlose Logikebene

Die Logikebene der dreistufigen Architektur stellt das Gehirn der Anwendung dar. Hier AWS Lambda kann die Verwendung von Amazon API Gateway die größte Wirkung im Vergleich zu einer herkömmlichen, serverbasierten Implementierung haben. Die Funktionen dieser beiden Dienste ermöglichen es Ihnen, eine serverlose Anwendung zu erstellen, die hochverfügbar, skalierbar und sicher ist. In einem herkömmlichen Modell könnte Ihre Anwendung Tausende von Servern benötigen. Wenn Sie jedoch Amazon API Gateway verwenden, sind AWS Lambda Sie in keiner Weise für die Serververwaltung verantwortlich. Wenn Sie diese verwalteten Services zusammen nutzen, erhalten Sie außerdem die folgenden Vorteile:

- AWS Lambda:
  - Es gibt kein Betriebssystem, das ausgewählt, gesichert, gepatcht oder verwaltet werden muss
  - Es gibt keine Server, die die richtige Größe, Überwachung oder Skalierung haben
  - Geringeres Risiko für Ihre Kosten aufgrund von Überprovisionierung
  - Geringeres Risiko für Ihre Leistung durch unzureichende Bereitstellung

- Amazon API Gateway:
  - Vereinfachte Mechanismen zur Bereitstellung, Überwachung und Sicherung APIs
  - Verbesserte API-Leistung durch Caching und Inhaltsbereitstellung

AWS Lambda

AWS Lambda ist ein Rechendienst, mit dem Sie beliebige Codefunktionen ausführen können, ohne Server bereitstellen, verwalten oder skalieren zu müssen. Zu den unterstützten Sprachen gehören Python, Ruby, Java, Go und .NET. Lambda-Funktionen werden in einem verwalteten, isolierten Container ausgeführt und als Reaktion auf ein Ereignis gestartet, bei dem es sich um einen von mehreren programmatischen Triggern handeln kann, AWS die als Ereignisquelle bezeichnet werden. Weitere Informationen zu unterstützten Sprachen und Ereignisquellen finden Sie unter Lambda FAQs.

Viele beliebte Anwendungsfälle für Lambda drehen sich um ereignisgesteuerte Datenverarbeitungsworkflows, wie z. B. die Verarbeitung von in Amazon S3 gespeicherten Dateien oder das Streamen von Datensätzen aus Amazon Kinesis. In Verbindung mit Amazon API Gateway erfüllt eine Lambda-Funktion die Funktionalität eines typischen Webdienstes: Sie initiiert Code

als Antwort auf eine HTTPS-Anfrage eines Clients; API Gateway fungiert als Eingangstür für Ihre Logikebene und AWS Lambda ruft den Anwendungscode auf.

Hier kommt Ihre Geschäftslogik zum Einsatz, es sind keine Server erforderlich

Lambda erfordert, dass Sie Codefunktionen, sogenannte Handler, schreiben, die ausgeführt werden, wenn sie durch ein Ereignis ausgelöst werden. Um Lambda mit API Gateway zu verwenden, können Sie API Gateway so konfigurieren, dass Handlerfunktionen gestartet werden, wenn eine HTTPS-Anfrage an Ihre API erfolgt. In einer serverlosen mehrstufigen Architektur wird jede der in API Gateway APIs erstellten Architekturen mit einer Lambda-Funktion (und dem darin enthaltenen Handler) integriert, die die erforderliche Geschäftslogik aufruft.

Mithilfe von AWS Lambda Funktionen zur Erstellung der Logikebene können Sie eine gewünschte Granularität für die Bereitstellung der Anwendungsfunktionalität definieren (eine Lambda-Funktion pro API oder eine Lambda-Funktion pro API-Methode). Innerhalb der Lambda-Funktion kann der Handler auf alle anderen Abhängigkeiten (z. B. andere Methoden, die Sie mit Ihrem Code hochgeladen haben, Bibliotheken, native Binärdateien und externe Webdienste) oder sogar auf andere Lambda-Funktionen zugreifen.

Das Erstellen oder Aktualisieren einer Lambda-Funktion erfordert entweder das Hochladen von Code als Lambda-Bereitstellungspaket in einer ZIP-Datei in einen Amazon S3 S3-Bucket oder das Verpacken des Codes als Container-Image zusammen mit allen Abhängigkeiten. Die Funktionen können verschiedene Bereitstellungsmethoden wie AWS Management Console, running AWS Command Line Interface (AWS CLI) oder running infrastructure as Codevorlagen oder Frameworks wie CloudFormation, AWS Serverless Application Model(AWS SAM) oder verwenden AWS Cloud Development Kit (AWS CDK). Wenn Sie Ihre Funktion mit einer dieser Methoden erstellen, geben Sie an, welche Methode in Ihrem Bereitstellungspaket als Anforderungshandler fungiert. Sie können dasselbe Bereitstellungspaket für mehrere Lambda-Funktionsdefinitionen wiederverwenden, wobei jede Lambda-Funktion einen eindeutigen Handler innerhalb desselben Bereitstellungspakets haben kann.

Lambda-Sicherheit

Um eine Lambda-Funktion auszuführen, muss sie durch ein Ereignis oder einen Dienst aufgerufen werden, der durch eine AWS Identity and Access Management (IAM -) Richtlinie zugelassen ist. Mithilfe von IAM-Richtlinien können Sie eine Lambda-Funktion erstellen, die erst initiiert werden kann, wenn sie von einer von Ihnen definierten API-Gateway-Ressource aufgerufen wird. Eine solche

Richtlinie kann mithilfe ressourcenbasierter Richtlinien für verschiedene Dienste definiert werden. AWS

Jede Lambda-Funktion nimmt eine IAM-Rolle an, die zugewiesen wird, wenn die Lambda-Funktion bereitgestellt wird. Diese IAM-Rolle definiert die anderen AWS Dienste und Ressourcen, mit denen Ihre Lambda-Funktion interagieren kann (z. B. Amazon DynamoDB Amazon S3). Im Zusammenhang mit der Lambda-Funktion wird dies als Ausführungsrolle bezeichnet.

Speichern Sie keine vertraulichen Informationen in einer Lambda-Funktion. IAM wickelt den Zugriff auf AWS Dienste über die Lambda-Ausführungsrolle ab. Wenn Sie von Ihrer Lambda-Funktion aus auf andere Anmeldeinformationen (z. B. Datenbankanmeldeinformationen und API-Schlüssel) zugreifen müssen, können Sie AWS Key Management Service(AWS KMS) mit Umgebungsvariablen verwenden oder einen Dienst wie AWSSecrets Manager verwenden, um diese Informationen zu schützen, wenn sie nicht verwendet werden.

Leistung im großen Maßstab

Code, der als Container-Image aus Amazon Elastic Container Registry (Amazon ECR) oder aus einer auf Amazon S3 hochgeladenen Zip-Datei abgerufen wird, wird in einer isolierten Umgebung ausgeführt, die von AWS verwaltet wird. Sie müssen Ihre Lambda-Funktionen nicht skalieren — jedes Mal, wenn Ihre Funktion eine Ereignisbenachrichtigung empfängt, AWS Lambda sucht sie nach verfügbarer Kapazität innerhalb ihrer Rechenflotte und führt Ihren Code mit von Ihnen definierten Laufzeit-, Speicher-, Festplatten- und Timeout-Konfigurationen aus. Mit diesem Muster kann AWS so viele Kopien Ihrer Funktion wie nötig starten.

Eine Lambda-basierte Logikebene hat immer die richtige Größe für die Bedürfnisse Ihrer Kunden. Die Fähigkeit, Datenverkehrsspitzen durch verwaltete Skalierung und gleichzeitige Codeinitiierung schnell abzufangen, ermöglicht es Ihnen in Kombination mit der pay-per-use Lambda-Preisgestaltung, Kundenanfragen stets zu erfüllen und gleichzeitig nicht für ungenutzte Rechenkapazität zu zahlen.

Serverlose Bereitstellung und Verwaltung

Verwenden Sie das AWS Serverless Application Model (AWS SAM), ein Open-Source-Framework, das Folgendes umfasst, um Sie bei der Bereitstellung und Verwaltung Ihrer Lambda-Funktionen zu unterstützen:

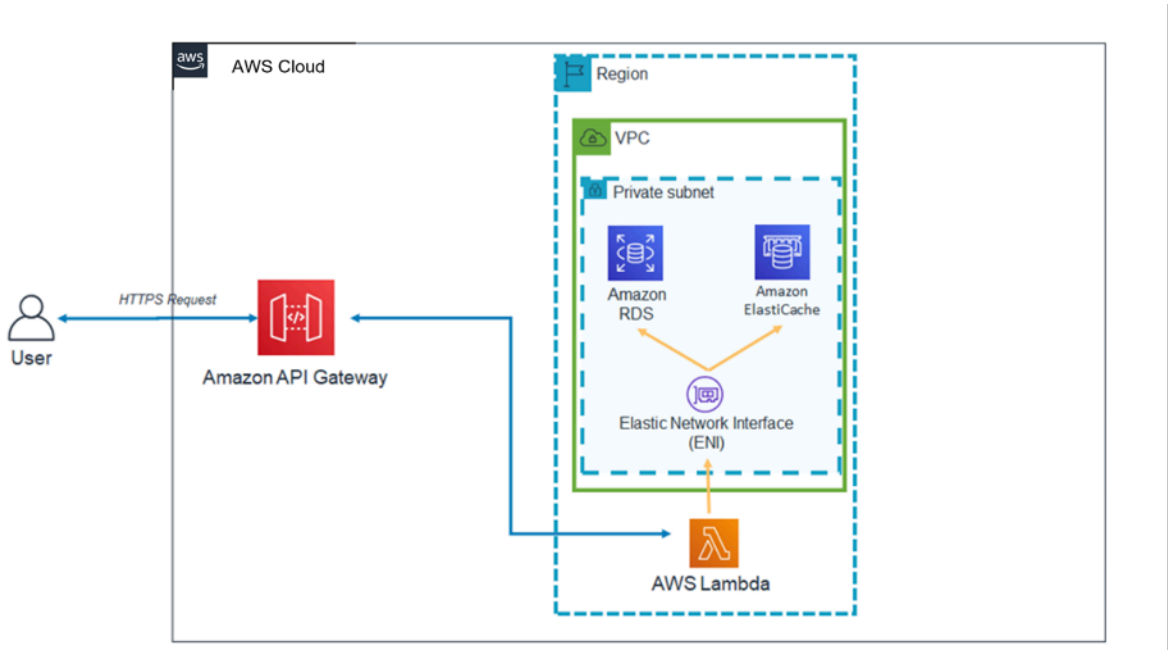
- AWS-SAM-Vorlagenspezifikation — Syntax, die zur Definition Ihrer Funktionen und zur Beschreibung ihrer Umgebungen, Berechtigungen, Konfigurationen und Ereignisse verwendet wird, um Upload und Bereitstellung zu vereinfachen.

- AWS SAM CLI — Befehle, mit denen Sie die SAM-Vorlagensyntax überprüfen, Funktionen lokal aufrufen, Lambda-Funktionen debuggen und Paketfunktionen bereitstellen können.

Sie können auch ein Framework für die Softwareentwicklung verwenden AWS CDK, mit dem Sie die Cloud-Infrastruktur mithilfe von Programmiersprachen definieren und bereitstellen können. CloudFormation CDK bietet eine zwingende Methode zur Definition von AWS Ressourcen, wohingegen es eine deklarative Methode AWS SAM bietet.

Wenn Sie eine Lambda-Funktion bereitstellen, wird sie in der Regel mit Berechtigungen aufgerufen, die durch die zugewiesene IAM-Rolle definiert sind, und sie kann mit dem Internet verbundene Endpunkte erreichen. Als Herzstück Ihrer Logikebene AWS Lambda ist die Komponente direkt in die Datenebene integriert. Wenn Ihre Datenschicht vertrauliche Geschäfts- oder Benutzerinformationen enthält, müssen Sie sicherstellen, dass diese Datenschicht angemessen isoliert ist (in einem privaten Subnetz).

Sie können eine Lambda-Funktion so konfigurieren, dass sie eine Verbindung zu privaten Subnetzen in einer Virtual Private Cloud (VPC) in Ihrem AWS Konto herstellt, wenn Sie möchten, dass die Lambda-Funktion auf Ressourcen zugreift, die Sie nicht öffentlich verfügbar machen können, z. B. eine private Datenbankinstanz. Wenn Sie eine Funktion mit einer VPC verbinden, erstellt Lambda eine elastic network interface für jedes Subnetz in der VPC-Konfiguration Ihrer Funktion, und die elastic network interface wird verwendet, um privat auf Ihre internen Ressourcen zuzugreifen.



Lambda-Architekturmuster innerhalb einer VPC

Die Verwendung von Lambda mit VPC bedeutet, dass Datenbanken und andere Speichermedien, von denen Ihre Geschäftslogik abhängt, über das Internet unzugänglich gemacht werden können. Die VPC stellt außerdem sicher, dass Sie mit Ihren Daten aus dem Internet nur über die von Ihnen definierten Funktionen und APIs die von Ihnen geschriebenen Lambda-Code-Funktionen interagieren können.

Amazon API Gateway

Amazon API Gateway ist ein vollständig verwalteter Service, der es Entwicklern ermöglicht, in jeder Größenordnung zu erstellen, zu veröffentlichen, zu warten, APIs zu überwachen und zu sichern.

Clients (d. h. Präsentationsebenen) lassen sich mithilfe von Standard-HTTPS-Anfragen in das APIs über API Gateway bereitgestellte Daten integrieren. Die Anwendbarkeit von APIs Exposed by API Gateway auf eine serviceorientierte mehrstufige Architektur besteht in der Möglichkeit, einzelne Anwendungsfunktionen zu trennen und diese Funktionalität über REST-Endpunkte verfügbar zu machen. Amazon API Gateway verfügt über spezifische Funktionen und Eigenschaften, mit denen Sie Ihre Logikebene um leistungsstarke Funktionen erweitern können.

Integration mit AWS Lambda

Amazon API Gateway unterstützt sowohl REST- als auch HTTP-Typen von APIs. Eine API-Gateway-API besteht aus Ressourcen und Methoden. Eine Ressource ist eine logische Einheit, auf die eine App über einen Ressourcenpfad zugreifen kann (z. B.`/tickets`). Eine Methode entspricht einer API-Anfrage, die an eine API-Ressource gesendet wird (z. B.`GET /tickets`). Mit API Gateway können Sie jede Methode mit einer Lambda-Funktion unterstützen, d. h. wenn Sie die API über den in API Gateway bereitgestellten HTTPS-Endpunkt aufrufen, ruft API Gateway die Lambda-Funktion auf.

Sie können API Gateway- und Lambda-Funktionen mithilfe von Proxy-Integrationen und Nicht-Proxy-Integrationen verbinden.

Proxy-Integrationen

Bei einer Proxyintegration wird die gesamte Client-HTTPS-Anfrage unverändert an die Lambda-Funktion gesendet. API Gateway übergibt die gesamte Client-Anfrage als Event-Parameter der Lambda-Handler-Funktion, und die Ausgabe der Lambda-Funktion wird direkt an den Client zurückgegeben (einschließlich Statuscode, Headern usw.).

Nicht-Proxy-Integrationen

In einer Nicht-Proxy-Integration konfigurieren Sie, wie die Parameter, Header und der Hauptteil der Client-Anfrage an den Event-Parameter der Lambda-Handler-Funktion übergeben werden. Darüber hinaus konfigurieren Sie, wie die Lambda-Ausgabe an den Benutzer zurückübersetzt wird.

> **Note**
>
> API Gateway kann auch als Proxy für zusätzliche serverlose Ressourcen außerhalb verwendet werden AWS Lambda, wie z. B. Scheinintegrationen (nützlich für die anfängliche Anwendungsentwicklung), und direkte Proxys für S3-Objekte.

Stabile API-Leistung in allen Regionen

Jede Bereitstellung von Amazon API Gateway beinhaltet eine CloudFrontAmazon-Distribution unter der Haube. CloudFrontist ein Service zur Bereitstellung von Inhalten, der das globale Netzwerk der Edge-Standorte von Amazon als Verbindungspunkte für Kunden nutzt, die Ihre API verwenden. Dies trägt dazu bei, die Antwortlatenz Ihrer API zu verringern. Durch die Nutzung mehrerer Edge-Standorte auf der ganzen Welt bietet Amazon CloudFront auch Funktionen zur Bekämpfung von Distributed-Denial-of-Service (DDoS) -Angriffsszenarien. Weitere Informationen finden Sie im Whitepaper AWS Best Practices for DDo S Resiliency.

Sie können die Leistung bestimmter API-Anfragen verbessern, indem Sie API Gateway verwenden, um Antworten in einem optionalen In-Memory-Cache zu speichern. Dieser Ansatz bietet nicht nur Leistungsvorteile bei wiederholten API-Anfragen, sondern reduziert auch die Häufigkeit, mit der Ihre Lambda-Funktionen aufgerufen werden, was Ihre Gesamtkosten senken kann.

Fördern Sie Innovationen und reduzieren Sie den Overhead mit integrierten Funktionen

Die Entwicklungskosten für die Erstellung einer neuen Anwendung sind eine Investition. Die Verwendung von API Gateway kann den Zeitaufwand für bestimmte Entwicklungsaufgaben reduzieren und die Gesamtentwicklungskosten senken, sodass Unternehmen freier experimentieren und innovativ sein können.

In den ersten Phasen der Anwendungsentwicklung werden die Implementierung der Protokollierung und die Erfassung von Metriken oft vernachlässigt, um eine neue Anwendung schneller

bereitzustellen. Dies kann zu technischen Schulden und Betriebsrisiken führen, wenn diese Funktionen in einer Anwendung bereitgestellt werden, die in der Produktion läuft. Amazon API Gateway lässt sich nahtlos in Amazon CloudWatch integrieren. Amazon sammelt und verarbeitet Rohdaten von API Gateway zu lesbaren, nahezu in Echtzeit verfügbaren Metriken zur Überwachung der API-Ausführung. API Gateway unterstützt auch die Zugriffsprotokollierung mit konfigurierbaren Berichten und die AWS X-RayAblaufverfolgung zum Debuggen. Für jede dieser Funktionen muss kein Code geschrieben werden und sie können in Anwendungen, die in der Produktion ausgeführt werden, angepasst werden, ohne dass die Kerngeschäftslogik gefährdet wird.

Die Gesamtlebensdauer einer Anwendung ist möglicherweise unbekannt, oder es ist bekannt, dass sie kurzlebig ist. Die Erstellung eines Geschäftsszenarios für die Entwicklung solcher Anwendungen kann vereinfacht werden, wenn Ihr Ausgangspunkt bereits die verwalteten Funktionen umfasst, die API Gateway bietet, und wenn Ihnen erst Infrastrukturkosten entstehen, nachdem Sie APIs Anfragen erhalten haben. Weitere Informationen finden Sie unter Amazon API Gateway Gateway-Preise.

Iterieren Sie schnell, bleiben Sie agil

Durch die Verwendung von Amazon API Gateway und AWS Lambda die Erstellung der Logikebene Ihrer API können Sie sich schnell an die sich ändernden Anforderungen Ihrer Benutzerbasis anpassen, indem Sie die API-Bereitstellung und das Versionsmanagement vereinfachen.

Bereitstellung in der Phase

Wenn Sie eine API in API Gateway bereitstellen, müssen Sie die Bereitstellung einer API-Gateway-Phase zuordnen — jede Phase ist ein Snapshot der API und wird für Client-Apps zum Aufrufen zur Verfügung gestellt. Mithilfe dieser Konvention können Sie Apps problemlos in der Entwicklungs -, Test -, Phase - oder Produktionsphase bereitstellen und Bereitstellungen zwischen den Phasen verschieben. Jedes Mal, wenn Sie Ihre API in einer Phase bereitstellen, erstellen Sie eine andere Version der API, die bei Bedarf rückgängig gemacht werden kann. Durch diese Funktionen können bestehende Funktionen und Kundenabhängigkeiten ungestört weitergeführt werden, während neue Funktionen als separate API-Version veröffentlicht werden.

Entkoppelte Integration mit Lambda

Die Integration zwischen API in API Gateway und Lambda-Funktion kann mithilfe von API Gateway Gateway-Stufenvariablen und einem Lambda-Funktionsalias entkoppelt werden. Dies vereinfacht und beschleunigt die API-Bereitstellung. Anstatt den Namen oder Alias der Lambda-Funktion direkt in der API zu konfigurieren, können Sie die Stage-Variable in der API konfigurieren, die auf einen

bestimmten Alias in der Lambda-Funktion verweisen kann. Ändern Sie während der Bereitstellung den Wert der Stage-Variablen so, dass er auf einen Lambda-Funktionsalias verweist, und die API führt die Lambda-Funktionsversion hinter dem Lambda-Alias für eine bestimmte Phase aus.

Canary-Release-Bereitstellung

Bei der Canary-Version handelt es sich um eine Softwareentwicklungsstrategie, bei der eine neue Version einer API zu Testzwecken bereitgestellt wird und die Basisversion weiterhin als Produktionsversion für den normalen Betrieb in derselben Phase bereitgestellt wird. Bei einer Canary-Release-Bereitstellung wird der gesamte API-Verkehr nach dem Zufallsprinzip in eine Produktionsversion und eine Canary-Version mit einem vorkonfigurierten Verhältnis aufgeteilt. APIs in API Gateway kann für die Bereitstellung der Canary-Version konfiguriert werden, um neue Funktionen mit einer begrenzten Anzahl von Benutzern zu testen.

Benutzerdefinierte Domainnamen

Sie können der API einen intuitiven, unternehmensfreundlichen URL-Namen anstelle der von API Gateway bereitgestellten URL angeben. API Gateway bietet Funktionen zur Konfiguration einer benutzerdefinierten Domain für APIs. Mit benutzerdefinierten Domainnamen können Sie den Hostnamen Ihrer API einrichten und einen mehrstufigen Basispfad (z. B.,, oder`myservice/dog/v2`) wählen `myservicemyservice/cat/v1`, um die alternative URL Ihrer API zuzuordnen.

Priorisieren Sie die API-Sicherheit

Alle Anwendungen müssen sicherstellen, dass nur autorisierte Kunden Zugriff auf ihre API-Ressourcen haben. Beim Entwerfen einer mehrstufigen Anwendung können Sie verschiedene Möglichkeiten nutzen, wie Amazon API Gateway zur Sicherung Ihrer Logikebene beiträgt:

Sicherheit im Transit

Alle Anfragen an Sie APIs können über HTTPS gestellt werden, um die Verschlüsselung bei der Übertragung zu ermöglichen.

API Gateway bietet integrierte SSL/TLS Zertifikate — wenn Sie die Option für benutzerdefinierte Domainnamen für die Öffentlichkeit verwenden APIs, können Sie mit AWS SSL/TLS Certificate Manager Ihr eigenes Zertifikat bereitstellen. API Gateway unterstützt auch die gegenseitige TLS-Authentifizierung (mTLS). Mutual TLS verbessert die Sicherheit Ihrer API und trägt dazu bei, Ihre Daten vor Angriffen wie Client-Spoofing oder man-in-the Middle Attacks zu schützen.

API-Autorisierung

Jeder resource/method Kombination, die Sie als Teil Ihrer API erstellen, wird ein eindeutiger Amazon-Ressourcenname (ARN) zugewiesen, auf den in AWS Identity and Access Management (IAM) -Richtlinien verwiesen werden kann.

Es gibt drei allgemeine Methoden, um einer API in API Gateway eine Autorisierung hinzuzufügen:

- IAM-Rollen und -Richtlinien: Kunden verwenden die Autorisierung und die IAM-Richtlinien von AWS Signature Version 4 (Sigv4) für den API-Zugriff. Dieselben Anmeldeinformationen können den Zugriff auf andere AWS Dienste und Ressourcen nach Bedarf einschränken oder zulassen (z. B. Amazon S3 S3-Buckets oder Amazon DynamoDB-Tabellen).
- Amazon Cognito Cognito-Benutzerpools: Kunden melden sich über einen Amazon Cognito Cognito-Benutzerpool an und erhalten Token, die im Autorisierungsheader einer Anfrage enthalten sind.
- Lambda-Autorisierer: Definieren Sie eine Lambda-Funktion, die ein benutzerdefiniertes Autorisierungsschema implementiert, das eine Bearer-Token-Strategie (z. B. OAuth und SAML) verwendet oder Anforderungsparameter verwendet, um Benutzer zu identifizieren.

Zugriffsbeschränkungen

API Gateway unterstützt die Generierung von API-Schlüsseln und die Zuordnung dieser Schlüssel zu einem konfigurierbaren Nutzungsplan. Sie können die Verwendung von API-Schlüsseln mit überwachen CloudWatch.

API Gateway unterstützt Drosselung, Ratenbegrenzungen und Burst-Ratenbegrenzungen für jede Methode in Ihrer API.

Privat APIs

Mit API Gateway können Sie private REST erstellen, auf APIs die nur von Ihrer virtuellen privaten Cloud in Amazon VPC aus zugegriffen werden kann, indem Sie einen VPC-Schnittstellen-Endpunkt verwenden. Dies ist eine Endpunkt-Netzwerkschnittstelle, die Sie in Ihrer VPC erstellen.

Mithilfe von Ressourcenrichtlinien können Sie den Zugriff auf Ihre API von ausgewählten Endpunkten VPCs und VPC-Endpunkten aus aktivieren oder verweigern, auch für AWS-Konten. Jeder Endpunkt kann für den Zugriff auf mehrere private Endgeräte verwendet werden. APIs Sie können außerdem AWS Direct Connect verwenden, um eine Verbindung von einem On-Premise-Netzwerk zu Amazon VPC herzustellen und über diese Verbindung auf Ihre private API zuzugreifen.

In allen Fällen verwendet der Datenverkehr zu Ihrer privaten API eine sichere Verbindung und verlässt nicht das Amazon-Netzwerk. Er ist vom öffentlichen Internet isoliert.

Firewall-Schutz mit AWS WAF

Mit dem Internet verbundene Geräte APIs sind anfällig für böswillige Angriffe. AWS WAF ist eine Firewall für Webanwendungen, die zum Schutz vor solchen APIs Angriffen beiträgt. Sie APIs schützt vor gängigen Web-Exploits wie SQL-Injection und Cross-Site-Scripting-Angriffen. Sie können es AWS WAFzusammen mit API Gateway verwenden, um zum Schutz beizutragen APIs.

Datenebene

Die Verwendung AWS Lambda als Logikebene schränkt die in Ihrer Datenschicht verfügbaren Datenspeicheroptionen nicht ein. Lambda-Funktionen stellen eine Verbindung zu jeder Datenspeicheroption her, indem sie den entsprechenden Datenbanktreiber in das Lambda-Bereitstellungspaket aufnehmen, und verwenden rollenbasierten IAM-Zugriff oder verschlüsselte Anmeldeinformationen (über AWS KMS oder Secrets Manager). AWS

Die Auswahl eines Datenspeichers für Ihre Anwendung hängt stark von Ihren Anwendungsanforderungen ab. AWS bietet eine Reihe von serverlosen und nicht serverlosen Datenspeichern, mit denen Sie die Datenschicht Ihrer Anwendung zusammenstellen können.

Optionen zur serverlosen Datenspeicherung

Amazon S3 ist ein Objektspeicherservice, der branchenführende Skalierbarkeit, Datenverfügbarkeit, Sicherheit und Leistung bietet.

Amazon Aurora ist eine MySQL- und PostgreSQL-kompatible relationale Datenbank, die für die Cloud entwickelt wurde und die Leistung und Verfügbarkeit herkömmlicher Unternehmensdatenbanken mit der Einfachheit und Wirtschaftlichkeit von Open-Source-Datenbanken kombiniert. Aurora bietet sowohl serverlose als auch traditionelle Nutzungsmodelle.

Amazon DynamoDB ist eine Schlüsselwert- und Dokumentendatenbank, die in jeder Größenordnung eine Leistung im einstelligen Millisekundenbereich bietet. Es handelt sich um eine vollständig verwaltete, serverlose, multiregionale, multiaktive und langlebige Datenbank mit integrierter Sicherheit, Sicherung und Wiederherstellung sowie In-Memory-Caching für Anwendungen im Internet.

Amazon Timestream ist ein schneller, skalierbarer, vollständig verwalteter Zeitreihen-Datenbankservice für IoT- und Betriebsanwendungen, der das Speichern und Analysieren von Billionen von Ereignissen pro Tag zu einem Zehntel der Kosten relationaler Datenbanken vereinfacht. Aufgrund der Zunahme von IoT-Geräten, IT-Systemen und intelligenten Industriemaschinen gehören Zeitreihendaten — Daten, die messen, wie sich Dinge im Laufe der Zeit verändern — zu den am schnellsten wachsenden Datentypen.

Amazon Quantum Ledger Database (Amazon QLDB) ist eine vollständig verwaltete Ledger-Datenbank, die ein transparentes, unveränderliches und kryptografisch verifizierbares Transaktionsprotokoll bereitstellt, das einer zentralen vertrauenswürdigen Behörde gehört. Amazon

QLDB verfolgt jede Änderung der Anwendungsdaten und führt einen vollständigen und überprüfbaren Verlauf der Änderungen im Laufe der Zeit.

Amazon Keyspaces (für Apache Cassandra) ist ein skalierbarer, hochverfügbarer und verwalteter Apache Cassandra-kompatibler Datenbankservice. Mit Amazon Keyspaces können Sie Ihre Cassandra-Workloads mit demselben Cassandra-Anwendungscode und denselben Entwicklertools ausführen, die Sie heute AWS verwenden. Sie müssen keine Server bereitstellen, patchen oder verwalten und Sie müssen keine Software installieren, warten oder betreiben. Amazon Keyspaces ist serverlos, sodass Sie nur für die Ressourcen zahlen, die Sie nutzen, und der Service kann Tabellen als Reaktion auf den Anwendungsdatenverkehr automatisch nach oben und unten skalieren.

Amazon Elastic File System (Amazon EFS) bietet ein einfaches, serverloses, elastisches Dateisystem set-and-forget, mit dem Sie Dateidaten gemeinsam nutzen können, ohne Speicher bereitzustellen oder zu verwalten. Es kann mit AWS-Cloud-Services und lokalen Ressourcen verwendet werden und ist so konzipiert, dass es bei Bedarf auf Petabyte skaliert werden kann, ohne Anwendungen zu unterbrechen. Mit Amazon EFS können Sie Ihre Dateisysteme beim Hinzufügen und Entfernen von Dateien automatisch vergrößern und verkleinern, sodass Sie keine Kapazität bereitstellen und verwalten müssen, um dem Wachstum gerecht zu werden. Amazon EFS kann mit der Lambda-Funktion bereitgestellt werden, was es zu einer praktikablen Dateispeicheroption für APIs macht.

Datenspeicheroptionen ohne Server

Amazon Relational Database Service (Amazon RDS) ist ein verwalteter Webservice, der die Einrichtung, den Betrieb und die Skalierung einer relationalen Datenbank mithilfe einer der verfügbaren Engines (Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle und Microsoft SQL Server) vereinfacht und auf verschiedenen Datenbank-Instance-Typen läuft, die für Speicher, Leistung oder I/O optimiert sind.

Amazon Redshift ist ein vollständig verwalteter Data-Warehouse-Service im Petabyte-Bereich in der Cloud.

Amazon ElastiCache ist eine vollständig verwaltete Bereitstellung von Redis oder Memcached. Nahtlose Bereitstellung, Ausführung und Skalierung beliebter Open-Source-kompatibler In-Memory-Datenspeicher.

Amazon Neptune ist ein schneller, zuverlässiger und vollständig verwalteter Graphdatenbank-Service, der es einfach macht, Anwendungen zu erstellen und auszuführen, die mit

stark verbundenen Datensätzen arbeiten. Neptune unterstützt beliebte Graphmodelle — Eigenschaftsdiagramme und W3C Resource Description Framework (RDF) — und ihre jeweiligen Abfragesprachen, sodass Sie auf einfache Weise Abfragen erstellen können, mit denen Sie effizient in stark vernetzten Datensätzen navigieren können.

Amazon DocumentDB (mit MongoDB-Kompatibilität) ist ein schneller, skalierbarer, hochverfügbarer und vollständig verwalteter Dokumentendatenbankservice, der MongoDB-Workloads unterstützt.

Schließlich können Sie auch Datenspeicher, die unabhängig auf Amazon laufen, EC2 als Datenebene einer mehrstufigen Anwendung verwenden.

Stufe „Präsentation“

Die Präsentationsebene ist für die Interaktion mit der Logikebene über die API Gateway Gateway-REST-Endpunkte verantwortlich, die über das Internet verfügbar sind. Jeder HTTPS-fähige Client oder jedes Gerät kann mit diesen Endpunkten kommunizieren, sodass Ihre Präsentationsebene die Flexibilität hat, viele Formen anzunehmen (Desktop-Anwendungen, mobile Apps, Webseiten, IoT-Geräte usw.). Je nach Ihren Anforderungen kann Ihre Präsentationsebene die folgenden AWS serverlosen Angebote nutzen:

- Amazon Cognito — Ein serverloser Service zur Benutzeridentität und Datensynchronisierung, mit dem Sie Ihren Web- und mobilen Apps schnell und effizient Benutzerregistrierung, Anmeldung und Zugriffskontrolle hinzufügen können. Amazon Cognito ist auf Millionen von Benutzern skalierbar und unterstützt die Anmeldung bei Anbietern sozialer Identitäten wie Facebook, Google und Amazon sowie bei Anbietern von Unternehmensidentitäten über SAML 2.0.
- Amazon S3 mit CloudFront — Ermöglicht es Ihnen, statische Websites, wie z. B. einseitige Anwendungen, direkt von einem S3-Bucket aus bereitzustellen, ohne dass ein Webserver bereitgestellt werden muss. Sie können es CloudFront als verwaltetes Content Delivery Network (CDN) verwenden, um die Leistung zu verbessern und die SSL/TL Verwendung verwalteter oder benutzerdefinierter Zertifikate zu ermöglichen.

AWS Amplifyist eine Reihe von Tools und Diensten, die zusammen oder einzeln verwendet werden können, um Frontend-Web- und Mobilgeräteentwicklern dabei zu helfen, skalierbare Full-Stack-Anwendungen zu entwickeln, die von. AWS Amplify bietet einen vollständig verwalteten Service für die weltweite Bereitstellung und das Hosting statischer Webanwendungen, der über das zuverlässige CDN von Amazon mit Hunderten von Präsenzpunkten weltweit und mit integrierten CI/CD Workflows bereitgestellt wird, die den Veröffentlichungszyklus Ihrer Anwendungen beschleunigen. Amplify unterstützt beliebte Web-Frameworks wie React JavaScript, Angular, Vue, Next.js und mobile Plattformen wie Android, iOS, React Native, Ionic und Flutter. Abhängig von Ihren Netzwerkkonfigurationen und Anwendungsanforderungen müssen Sie Ihr API Gateway APIs möglicherweise so einrichten, dass es Cross-Origin Resource Sharing (CORS) -kompatibel ist. Die CORS-Konformität ermöglicht es Webbrowsern, Ihre Daten direkt von statischen Webseiten APIs aus aufzurufen.

Wenn Sie eine Website mit bereitstellen CloudFront, erhalten Sie einen CloudFront Domainnamen, über den Sie auf Ihre Anwendung zugreifen können (z. B.). `d2d47p2vcczkh2.cloudfront.net` Sie können Amazon Route 53 verwenden, um Domainnamen zu registrieren und sie an Ihren

CloudFront Vertrieb weiterzuleiten, oder Sie können bereits eigene Domainnamen an Ihren CloudFront Vertrieb weiterleiten. Auf diese Weise können Benutzer mit einem vertrauten Domainnamen auf Ihre Website zugreifen. Beachten Sie, dass Sie Ihrer API-Gateway-Verteilung auch einen benutzerdefinierten Domainnamen mithilfe von Route 53 zuweisen können, sodass Benutzer APIs mit vertrauten Domainnamen aufrufen können.

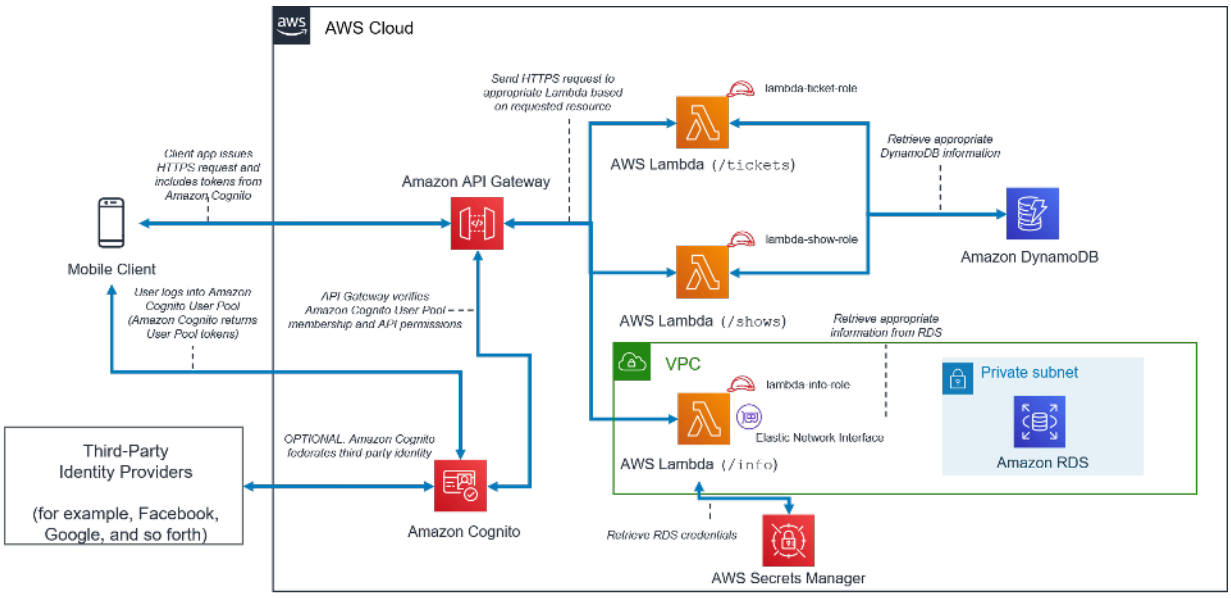
Beispiele für Architekturmuster

Sie können beliebte Architekturmuster mit API Gateway und AWS Lambda als Logikebene implementieren. Dieses Whitepaper enthält die beliebtesten Architekturmuster, die AWS Lambda basierte Logikstufen nutzen:

- Mobiles Backend — Eine mobile Anwendung kommuniziert mit API Gateway und Lambda, um auf Anwendungsdaten zuzugreifen. Dieses Muster kann auf generische HTTPS-Clients ausgedehnt werden, die keine serverlosen AWS-Ressourcen zum Hosten von Ressourcen auf Präsentationsebene verwenden (z. B. Desktop-Clients, Webserver EC2, auf denen ausgeführt wird usw.).
- Einseitige Anwendung — Eine einseitige Anwendung, die in Amazon S3 gehostet wird und mit API Gateway CloudFront kommuniziert und auf Anwendungsdaten AWS Lambda zugreift.
- Webanwendung — Die Webanwendung ist ein ereignisgesteuertes Allzweck-Back-End für Webanwendungen, das AWS Lambda mit API Gateway für seine Geschäftslogik verwendet wird. Es verwendet auch DynamoDB als Datenbank und Amazon Cognito für die Benutzerverwaltung. Alle statischen Inhalte werden mit Amplify gehostet.

Zusätzlich zu diesen beiden Mustern wird in diesem Whitepaper die Anwendbarkeit von Lambda und API Gateway auf eine allgemeine Microservice-Architektur erörtert. Eine Microservice-Architektur ist ein beliebtes Muster, obwohl es sich nicht um eine dreistufige Standardarchitektur handelt, bei der Anwendungskomponenten entkoppelt und als zustandslose, individuelle Funktionseinheiten bereitgestellt werden, die miteinander kommunizieren.

Mobiles Backend



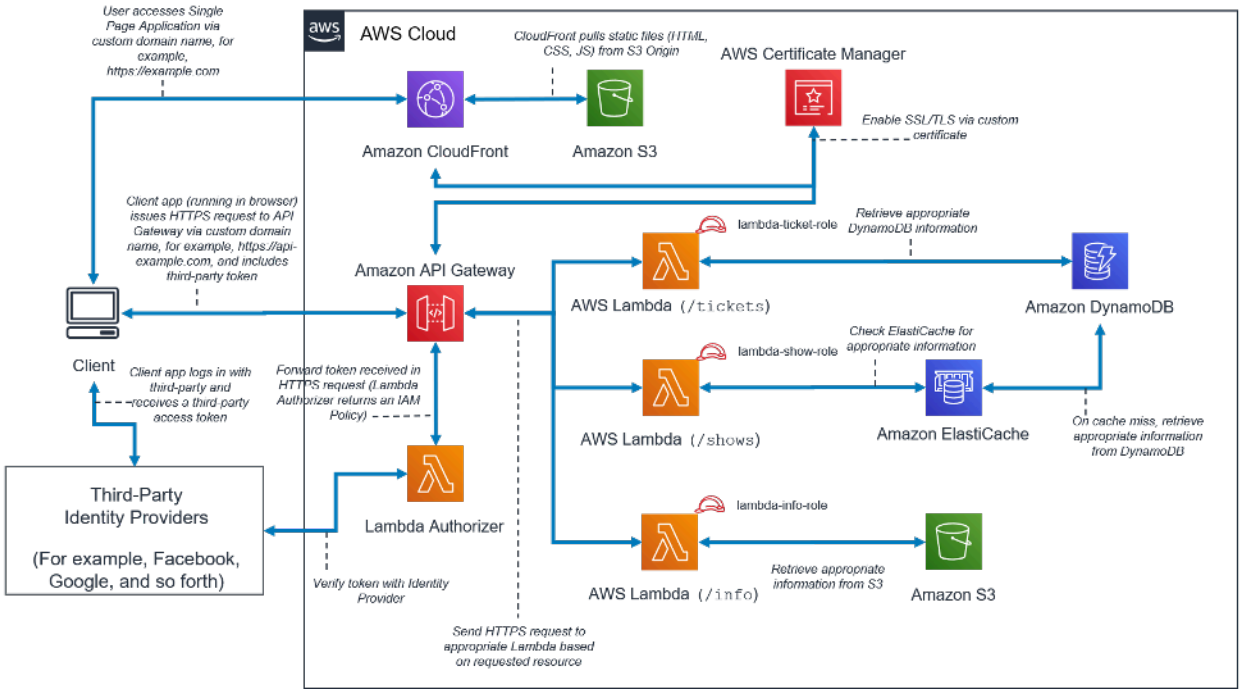
Architekturmuster für serverloses mobiles Backend

Tabelle 1: Komponenten der Stufe „Mobiles Backend“

Stufe	Komponenten
Präsentation	Mobile Anwendung, die auf einem Benutzergerät ausgeführt wird.
Logic (Logik)	Amazon API Gateway mit AWS Lambda. Diese Architektur zeigt drei exponierte Dienste (`/tickets`/`shows`, und/`info`). API Gateway Gateway-Endpunkte werden durch Amazon Cognito Cognito-Benutzerpools gesichert. Bei dieser Methode melden sich Benutzer Amazon Cognito Cognito-Benutzerpools an (falls erforderlich mit einem verbundenen Drittanbieter) und erhalten Zugriffs- und ID-Tokens, die zur Autorisierung von API Gateway Gateway-Aufrufen verwendet werden.

Stufe	Komponenten
	Jeder Lambda-Funktion wird ihre eigene Identity and Access Management (IAM) -Rolle zugewiesen, um Zugriff auf die entsprechende Datenquelle zu gewähren.
Daten	DynamoDB wird für die `/tickets` UND-Servi ces verwendet. `/shows` Amazon RDS wird für den `/info` Service verwendet. Diese Lambda-Funktion ruft Amazon RDS-Anmeldeinformationen von AWS Secrets Manager ab und verwendet eine elastic network interface, um auf das private Subnetz zuzugreifen.

Einseitige Anwendung

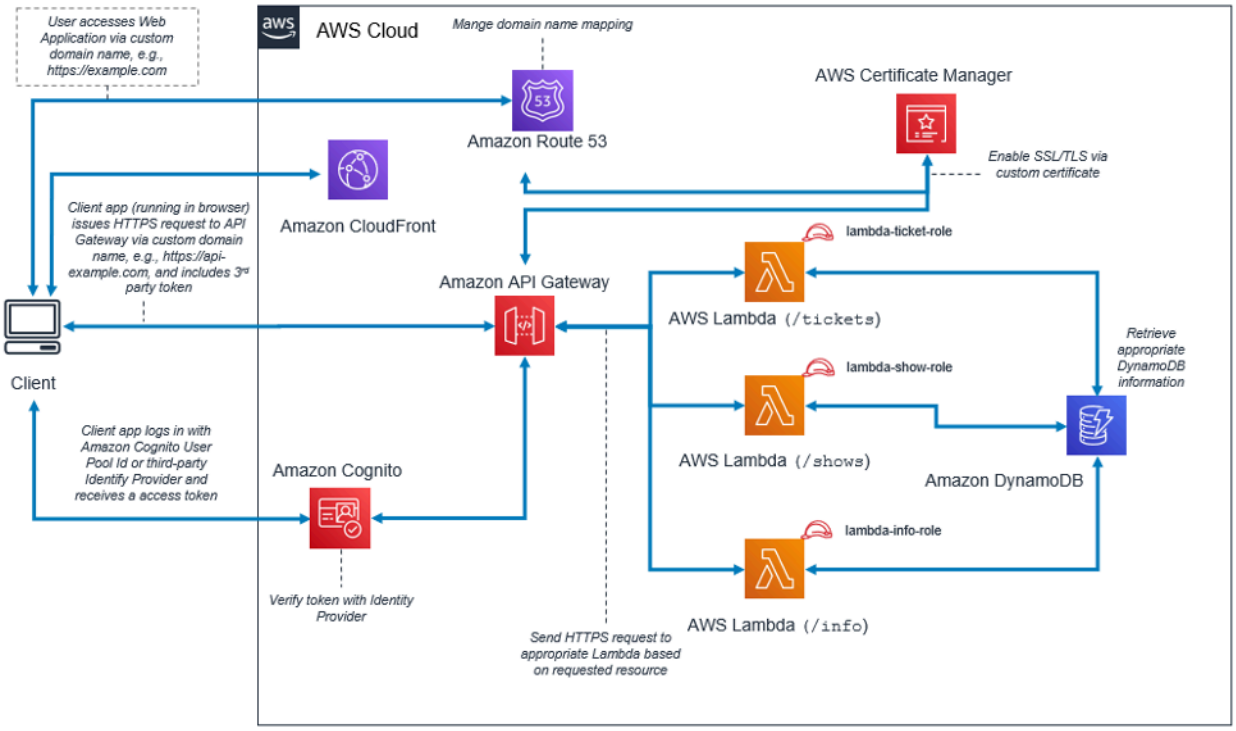


Architekturmuster für serverlose einseitige Anwendungen

Tabelle 2: Einseitige Anwendungskomponenten

Stufe	Komponenten
Präsentation	Statischer Webseiteninhalt, gehostet in Amazon S3, vertrieben von CloudFront. AWS Certificate Manager ermöglicht die Verwendung eines benutzerdefinierten SSL/TLS-Zertifikats.
Logic (Logik)	API Gateway mit AWS Lambda. Diese Architektur zeigt drei exponierte Dienste (`/tickets`/`shows`, und`/info`). API Gateway Gateway-Endpunkte werden durch einen Lambda-Authorizer gesichert. Bei dieser Methode melden sich Benutzer über einen externen Identitätsanbieter an und erhalten Zugriffs- und ID-Token. Diese Token sind in API-Gateway-Aufrufen enthalten, und der Lambda-Autorisierer validiert diese Token und generiert eine IAM-Richtlinie mit API-Initi ierungsberechtigungen. Jeder Lambda-Funktion wird eine eigene IAM-Rolle zugewiesen, um Zugriff auf die entsprech ende Datenquelle zu gewähren.
Daten	Amazon DynamoDB wird für die Dienste `/tickets` und `/shows` verwendet. Amazon ElastiCache wird vom `/shows` Service verwendet, um die Datenbankleistung zu verbessern. Cachefehler werden an DynamoDB gesendet. Amazon S3 wird verwendet, um statische Inhalte zu hosten, die von der verwendet werden`/info service`.

Webanwendung



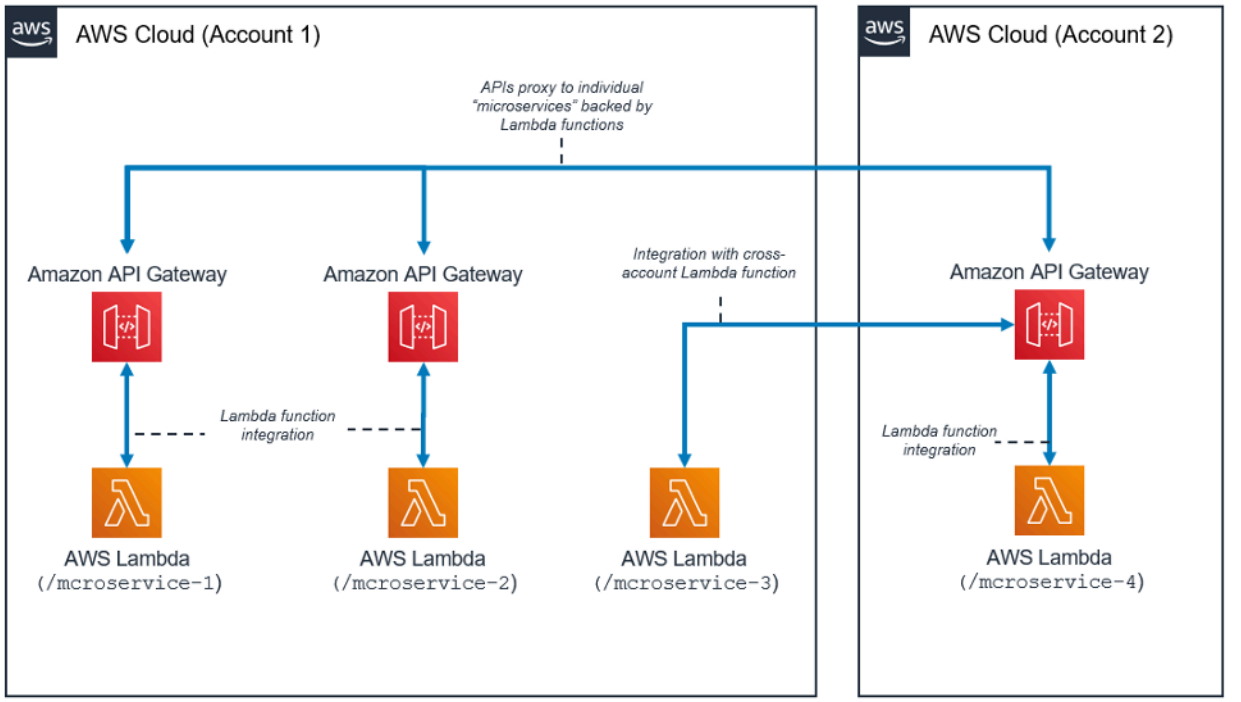
Architekturmuster für Webanwendungen

Tabelle 3: Komponenten der Webanwendung

Stufe	Komponenten
Präsentation	Die Frontend-Anwendung besteht aus allen statischen Inhalten (HTML, CSS JavaScript und Bilder), die von React-Dienstprogrammen wie create-react-app generiert werden. Amazon CloudFront hostet all diese Objekte. Wenn die Webanwendung verwendet wird, lädt sie alle Ressourcen in den Browser herunter und wird von dort aus ausgeführt. Die Webanwendung stellt eine Verbindung zum Backend her und ruft die auf APIs.

Stufe	Komponenten
Logic (Logik)	Die Logikschicht wird mithilfe von Lambda-Funktionen erstellt, die von API Gateway REST APIs unterstützt werden. Diese Architektur zeigt mehrere exponierte Dienste. Es gibt mehrere verschiedene Lambda-Funktionen, von denen jede einen anderen Aspekt der Anwendung behandelt. Die Lambda-Funktionen befinden sich hinter API Gateway und sind über API-URL-Pfade zugänglich. Die Benutzerauthentifizierung wird mithilfe von Amazon Cognito Cognito-Benutzerpools oder Verbundbenutzeranbietern abgewickelt. API Gateway verwendet die sofort einsatzbereite Integration mit Amazon Cognito. Erst nachdem ein Benutzer authentifiziert wurde, erhält der Client ein JSON-Web-Token (JWT), das er dann bei den API-Aufrufen verwenden sollte. Jeder Lambda-Funktion wird eine eigene IAM-Rolle zugewiesen, um Zugriff auf die entsprechende Datenquelle zu gewähren.
Daten	In diesem speziellen Beispiel wird DynamoDB für die Datenspeicherung verwendet, aber je nach Anwendungsfall und Nutzungsszenario können auch andere speziell entwickelte Amazon-Datenbank- oder Speicherdienste verwendet werden.

Microservices mit Lambda



Architekturmuster für Microservices mit Lambda

Das Mikroservice-Architekturmuster ist nicht an die typische dreistufige Architektur gebunden. Dieses beliebte Muster kann jedoch erhebliche Vorteile aus der Verwendung serverloser Ressourcen ziehen.

In dieser Architektur sind die einzelnen Anwendungskomponenten entkoppelt und werden unabhängig voneinander bereitgestellt und betrieben. Eine API, die mit Amazon API Gateway erstellt wurde, und Funktionen, die anschließend von gestartet werden AWS Lambda, sind alles, was Sie benötigen, um einen Microservice zu erstellen. Ihr Team kann diese Services nutzen, um Ihre Umgebung zu entkoppeln und zu fragmentieren, bis die gewünschte Granularität erreicht ist.

Im Allgemeinen kann eine Microservices-Umgebung zu den folgenden Schwierigkeiten führen: wiederholter Mehraufwand für die Erstellung jedes neuen Microservices, Probleme bei der Optimierung der Serverdichte und -auslastung, Komplexität der gleichzeitigen Ausführung mehrerer Versionen mehrerer Microservices und Zunahme von clientseitigen Codeanforderungen für die Integration in viele separate Dienste.

Wenn Sie Microservices mithilfe serverloser Ressourcen erstellen, ist es weniger schwierig, diese Probleme zu lösen, und in einigen Fällen verschwinden sie einfach. Das serverlose Microservices-Muster senkt die Barriere für die Erstellung jedes nachfolgenden Microservices (API Gateway ermöglicht sogar das Klonen vorhandener APIs und die Verwendung von Lambda-Funktionen in

anderen Konten). Die Optimierung der Serverauslastung ist bei diesem Muster nicht mehr relevant. Schließlich bietet Amazon API Gateway programmgesteuert generierte Clients SDKs in einer Reihe gängiger Sprachen, um den Integrationsaufwand zu reduzieren.

Schlussfolgerung

Das mehrstufige Architekturmuster fördert die bewährte Methode zur Erstellung von Anwendungskomponenten, die einfach zu warten, zu entkoppeln und zu skalieren sind. Wenn Sie eine Logikebene erstellen, in der die Integration durch API Gateway und die Berechnung innerhalb erfolgt AWS Lambda, erreichen Sie diese Ziele und reduzieren gleichzeitig den Aufwand, sie zu erreichen. Zusammen bieten diese Dienste ein HTTPS-API-Frontend für Ihre Kunden und eine sichere Umgebung, in der Sie Ihre Geschäftslogik anwenden und gleichzeitig den mit der Verwaltung einer typischen serverbasierten Infrastruktur verbundenen Aufwand vermeiden können.

Mitwirkende

Zu den Mitwirkenden an diesem Dokument gehören:

- Andrew Baird, AWS-Lösungsarchitekt
- Bryant Bost, AWS-Berater ProServe
- Stefano Buliani, leitender Produktmanager, Technik, AWS Mobile
- Vyom Nagrani, leitender Produktmanager, AWS Mobile
- Ajay Nair, leitender Produktmanager, AWS Mobile
- Rahul Popat, Architekt für globale Lösungen
- Brajendra Singh, leitender Lösungsarchitekt

Weitere Informationen

Zusätzliche Informationen finden Sie unter:

- AWS-Whitepapers und Leitfäden

Dokumentversionen

Abonnieren Sie den RSS-Feed, um über Aktualisierungen des Whitepapers benachrichtigt zu werden.

Änderung	Beschreibung	Datum
Kleinere Updates	Durchweg Bugfixes und zahlreiche kleinere Änderungen.	1. April 2022
Whitepaper aktualisiert	Für neue Servicefunktionen und -muster aktualisiert.	20. Oktober 2021
Whitepaper aktualisiert	Für neue Servicefunktionen und -muster aktualisiert.	1. Juni 2021
Whitepaper aktualisiert	Für neue Servicefunktionen aktualisiert.	25. September 2019
Erste Veröffentlichung	Whitepaper veröffentlicht.	1. November 2015

Hinweise

Kunden sind dafür verantwortlich, Ihre eigene unabhängige Bewertung der Informationen in diesem Dokument vorzunehmen. Dieses Dokument: (a) dient nur zu Informationszwecken, (b) stellt aktuelle AWS-Produktangebote und -praktiken dar, die ohne vorherige Ankündigung geändert werden können, und (c) stellt keine Verpflichtungen oder Zusicherungen von AWS und seinen verbundenen Unternehmen, Lieferanten oder Lizenzgebern dar. AWS-Produkte oder -Services werden „wie sie sind“ ohne ausdrückliche oder stillschweigende Garantien, Zusicherungen oder Bedingungen jeglicher Art bereitgestellt. Die Verantwortung und Haftung von AWS gegenüber seinen Kunden werden durch AWS-Vereinbarungen geregelt. Dieses Dokument gehört, weder ganz noch teilweise, nicht zu den Vereinbarung von AWS mit seinen Kunden und ändert diese Vereinbarungen auch nicht.