SERVERLESS COMPUTING

Ein Überblick zu den Services von Amazon und Microsoft

SERVERLESS

Im Serverless-Modell werden Anwendungskomponenten wie Datenbanken oder Komponenten zur Datenverarbeitung automatisch und bedarfsabhängig vom Cloud-Provider zur Verfügung gestellt und betrieben. Die Verantwortung des Cloud-Nutzers liegt darin, diese Ressourcen zu kombinieren und zu konfigurieren – etwa durch eigenen Code oder anwendungsspezifische Parameter.
Kosten fallen nach verbrauchten Kapazitäten an und die Skalierung erfolgt automatisch auf Basis der Last. Bereitstellung, Skalierung, Wartung, Hochverfügbarkeit und Verwaltung von Ressourcen liegen in der Verantwortung des Cloud-Providers. Serverless eignet sich besonders für schwer vorhersehbare oder kurzlebige Arbeitslasten, für Automatisierungsaufgaben oder Prototypen. Serverless ist weniger ideal für ressourcenintensive, langlebige und planbare Aufgaben, da in diesem Fall die Kosten signifikant höher sein können als bei selbstverwalteten Ausführungsumgebungen.

COMPUTE

SERVERLESS FUNCTION

- Vom Cloud-Provider verwaltete Ausführungsumgebung für Code
- Automatische Provisionierung und Skalierung
- Ereignisgesteuerte Ausführung
- Kosten nur für Ausführungszeit

AWS Lambda | **Azure Functions**

SERVERLESS CONTAINER

- Vom Cloud-Provider verwaltete Ausführungsumgebung für Container
- Container-Cluster ohne Management und Betrieb von virtuellen Maschinen
- Automatische Provisionierung und Skalierung
- Kosten nur für Laufzeit des Containers

AWS Fargate, Amazon ECS/EKS | **Azure Container Instances / AKS**

NETWORK

LOAD BALANCER

- Routing von HTTP-Anfragen auf verschiedene Backends (z. B. Lambda oder Functions)
- Vom Cloud-Provider verwaltete automatische Skalierung und Redundanz
- Kosten für Bereitstellung und verarbeitete Anfragen

Application Load Balancer | **Azure Application Gateway**

CONTENT DELIVERY NETWORK

- Caching von statischen Ressourcen an global verteilten Edge-Standorten
- Eintrittspunkt ins AWS-/Azure-Backbone-Netz zur beschleunigten Übertragung von dynamischen Inhalten
- Kosten für Anfragen und übertragene Daten

Amazon CloudFront | **Azure CDN**

DOMAIN NAME SERVICE

- DNS-Dienst mit fortgeschrittenen Funktionen (z. B. Routing-Strategien)
- Globale Verteilung für geringe Abfragelatenzen
- Kosten für verwaltete DNS-Zonen und Anfragen

Amazon Route 53 | **Azure DNS**

DATA

NOSQL-DATENBANK

- Dokumenten-/Key-Value-Datenbank
- Manuelle oder automatische Skalierung
- Automatische Ausfallsicherheit
- Kosten für Speicher und bereitgestellte Lese-/Schreibkapazität (manuell) oder Vorgänge (automatisch skaliert)

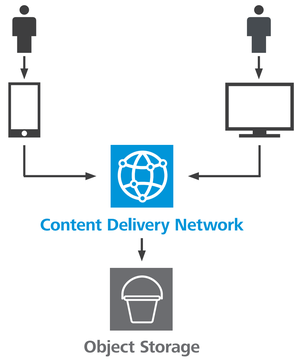
Amazon DynamoDB | **Azure Table Storage**

RELATIONALE DATENBANK

- Vollständig vom Provider verwaltete Datenbank
- Automatische Skalierung und Redundanz
- Kosten für genutzten Speicherplatz und bereitgehaltene automatisch skalierte Kapazität

Amazon Aurora Serverless | **Azure SQL Database Serverless**

STATIC WEBSITE HOSTING



ANALYTICS

DATA STREAM

- Erfassung, Pufferung, Analyse und Verarbeitung von Streaming-Daten in Echtzeit
- Integration mit anderen Services
- Ausgelegt auf hohen Durchsatz und viele Datenquellen
- Kosten pro Stream und bereitgestelltem Durchsatz

Amazon Kinesis | **Azure Stream Analytics**

STORAGE QUERY SERVICE

- Interaktiver Abfrageservice zur Analyse von Objektspeicherdaten
- Abfragen über SQL-Dialekt
- Kosten für analysierte Datenmenge

Amazon Athena | **Azure Data Lake Analytics**

ETL-SERVICE

- Serverless-Dienst zur Datenaufbereitung (Extract, Transform, Load)
- Automatische Generierung des ETL-Codes passend zu Quell- und Ziel-Daten
- Kosten für Verarbeitungszeit und bereitgestellte Kapazität

AWS Glue | **Azure Data Factory**

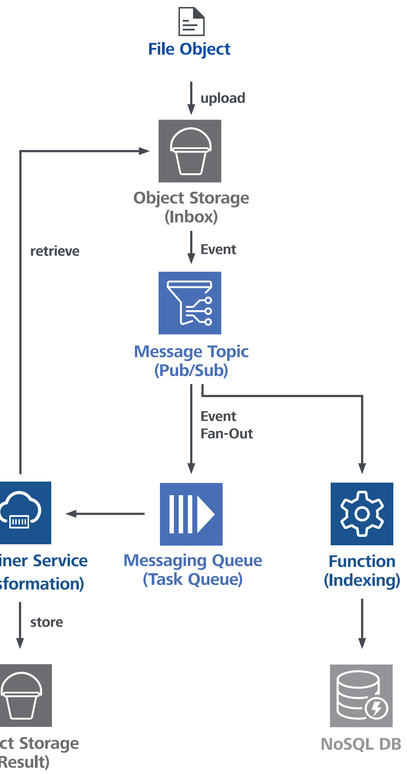
STORAGE

OBJECT STORAGE

- Mitwachsender skalierbarer Objektspeicher
- Upload und Download über HTTP-API
- Eignet sich für die Bereitstellung statischer Webseiten
- Kosten für genutzten Speicherplatz und abgehende Datenmengen

Amazon S3 | **Azure Storage Account**

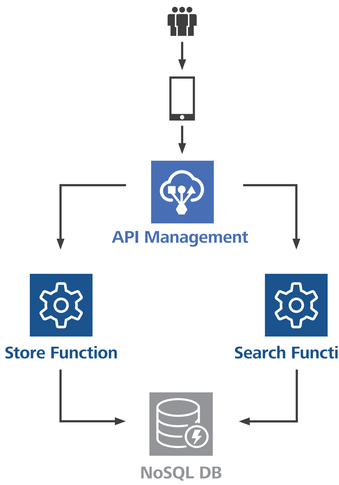
FILE PROCESSING



BEST PRACTICES FÜR SERVERLESS-FUNKTIONEN

Jede Funktion sollte nur eine Sache tun (Single Responsibility Principle). Dies verbessert Wartbarkeit und Wiederverwendbarkeit. Speicherkapazität, Zugriffsrechte und Timeout-Einstellung können gezielter konfiguriert werden.
Mit der Erhöhung des zugewiesenen Speichers einer Lambda-Funktion werden auch CPU und Netzwerkkapazität erhöht. Ein optimales Verhältnis aus Ausführungszeit und Kosten sollte per Benchmarking gefunden werden.
Eine Funktion sollte keine weitere Funktion synchron aufrufen. Das Warten führt zu unnötigen Kosten und erhöhter Kopplung. Stattdessen ist asynchrone Verarbeitung, z. B. über Message Queues, einzusetzen.
Das Deployment-Paket einer Funktion sollte so klein wie möglich sein. Auf große externe Bibliotheken ist zu verzichten. Das verbessert die Kaltstartzeit. Wiederkehrende Initialisierungen von Abhängigkeiten sollten außerhalb der Handler-Funktion stattfinden, damit sie nur einmalig beim Kaltstart ausgeführt werden müssen. Es ist ratsam, betriebliche Parameter über Umgebungsvariablen einer Funktion abzubilden. Das verbessert die Wiederverwendbarkeit. Die Zugriffsberechtigungen auf andere Cloud-Ressourcen sind für jede Funktion individuell und so restriktiv wie möglich zu definieren. Zustandsbehaftete Datenbankverbindungen sind zu vermeiden. Stattdessen sollten Service-APIs verwendet werden.

SIMPLE WEBSERVICE



SECURITY

IDENTITY PROVIDER

- OAuth-kompatibler Identitätsanbieter für die Benutzerverwaltung
- Registrierung, Passwort-Rücksetzung, Kontoauflösung, Multifaktor-Anmeldung
- Integration mit Social-Logins und OAuth-kompatiblen Providern
- Kosten für monatlich aktive Nutzer

Amazon Cognito | **Azure Active Directory B2C**

KEY MANAGEMENT

- Zentraler Dienst für die sichere Ablage und Verwaltung von kryptografischen Schlüsseln
- Ausführung von Ver- und Entschlüsselungsvorgängen mit Integration in andere Dienste
- Kosten nach Anzahl und Nutzung der Schlüssel

AWS KMS | **Azure Key Vault**

WEB APPLICATION FIREWALL

- Individuelle, regelbasierte Firewall
- Anwendungsbasierte Regeln möglich
- Integration in andere Dienste, wie z. B. CDN und Load Balancer
- Kosten für definierte Regeln und verarbeitete Anfragen

AWS WAF | **Azure Web Application Firewall**

INTEGRATION

WORKFLOW ENGINE

- Workflow Engine zur Kombination von Serverless-Diensten
- Konfiguration komplexer Workflows (inkl. Fehlerhandling, Bedingungen etc.)
- Kosten für durchgeführte Arbeitsschritte

AWS Step Functions | **Azure Logic App**

API MANAGEMENT

- Vollständig verwaltetes API-Management für REST APIs
- Verteilung von API-Anfragen auf verschiedene Dienste (z. B. AWS Lambda bzw. Azure Functions)
- Integration von Authentifizierung und API-Mapping möglich
- Kosten für verarbeitete Anfragen

Amazon API Gateway | **Azure API Management**

MESSAGE QUEUES

- Nachrichtenzustellung über Warteschlangen
- Exakt einmalige Zustellung und FIFO-Reihenfolge möglich
- Kosten für Anfragen und übertragene Datenmenge

Amazon SQS | **Azure Service Bus**

GRAPHQL API

- Vollständig verwaltetes GraphQL-Gateway
- Weiterleitung der Anfragen an ein Daten-Backend
- Kosten für durchgeführte Anfragen

AWS AppSync | **Azure Functions mit Apollo Server**

PUB/SUB MESSAGING

- Mehrpunkt-Kommunikation zwischen verschiedenen Systemen
- Eignet sich zur Entkopplung in einer ereignisgesteuerten Architektur
- Kosten für Nachrichten und übertragene Datenmenge

Amazon SNS | **Azure Event Grid**

IHR ANSPRECHPARTNER

Carl Zeiss Digital Innovation GmbH
Fritz-Foerster-Platz 2
01069 Dresden

Sven Jänicke
Sales & Business Development Manager

Telefon: +49 351 49701 – 450
E-Mail: sven.jaenicke@zeiss.com
www.zeiss.de/digital-innovation

Inhaltliche Entwicklung:
Michael Muck, Leo Lindhorst, Christian Heinemann, Christian Wißmann