Microservices und Microservices-Architektur

Microservices werden dazu verwendet, Cloud-basierte Anwendungen zu erstellen, die in einem verteilten Modell eingesetzt werden. Sie sind extrem skalierbar und erfordern in der Regel weniger Zeit und Aufwand für die Konstruktion und Wartung im Vergleich zu einer monolithischen Architektur.

Microservices können schnell in der Cloud bereitgestellt werden

- Die Microservices-Architektur ist eine Designmethodik für die schnelle Bereitstellung und Aktualisierung von Cloud-basierten Anwendungen.
- Microservices arbeiten normalerweise unabhängig, oft in Containern, die lose über eine Standardschnittstelle verbunden sind.
- Die Vorteile von Microservices sind die Geschwindigkeit der Bereitstellung, die einfache Aufrüstung und der Support für Skalierung und Komplexität.

Microservices sind locker gekoppelte Module mit geringen Anforderungen, die als Bausteine für Cloud-basierte Anwendungen dienen können. Eine Microservices-Architektur ist agil und skalierbar und unterstützt eine schnelle Bereitstellung im Vergleich zu einer monolithischen Architektur.

Microservices sind locker gekoppelte Module mit geringen Anforderungen, die als Bausteine für Cloud-basierte Anwendungen dienen können. Eine Microservices-Architektur ist agil und skalierbar und unterstützt eine schnelle Bereitstellung im Vergleich zu einer monolithischen Architektur.

Was sind Microservices?

Microservices sind locker gekoppelte Module mit geringen Anforderungen, die als Bausteine für komplexe Cloud-basierte Anwendungen dienen können. Während einzelne Microservices unabhängig arbeiten können, werden sie unter einem vereinheitlichten Interface locker gekoppelt.

Die Microservices-Architektur wird als ein moderner, flexibler Ersatz für das herkömmlichere Entwicklungsmodell der monolithischen Architektur angesehen.

In einer monolithischen Architektur gibt es in der Regel einen physischen oder virtualisierten Server, der jeder Anwendung zugeordnet ist, und diese Server laufen ständig. Die Skalierung und Verfügbarkeit von Anwendungen sind vollständig von der zugrunde liegenden Hardware abhängig, die eine feste Einheit darstellt.

Microservices hingegen können mehrere Instanzen auf einem einzelnen Server oder über mehrere Server hinweg betreiben, da sich die Ressourcen dynamisch skalieren, um die Anforderungen der Workload zu unterstützen. Einzelne Microservices werden oft containerisiert, um die Portabilität und Skalierbarkeit zu verbessern.

Die verschiedenen Merkmale von Microservices

Als Entwicklungsprozess hat das Microservices-Framework bestimmte Merkmale, die häufig, aber nicht universell sind.

Diese Merkmale umfassen:

- Komponenten. Microservices werden normalerweise aus separaten Software-Komponenten zusammengesetzt, die einzeln ausgetauscht und aktualisiert werden können. Diese Architektur hat Auswirkungen auf die Cloud-Management-Technologie, weil jeder der Microservices separat bereitgestellt, überwacht und aktualisiert werden muss.
- Dienstleistungen: Die Komponenten umfassen Dienstleistungen, die verfügbar sind, um auf Anforderung zu kommunizieren, können aber nicht ständig zwischen Anfragen oder Anrufen aktiv sein.
- Unabhängige Bereitstellung Meistens arbeiten die einzelnen Servicekomponenten unabhängig voneinander im Microservices-Framework. Wenn eine Komponente verändert oder aktualisiert wird, hat das geringe Auswirkungen auf andere Services und Komponenten im Vergleich zu einer herkömmlicheren monolithischen Architektur.
- Sicherheit. Die Kommunikation zwischen Microservices ist oft verschlüsselt, mit gegenseitiger Transport Layer Security (mTLS), um Daten während der Übertragung vor Malware und Eindringungen zu schützen.
- Containerisierung Microservices werden oft in Containern bereitgestellt, was zusätzliche Portabilität und Skalierbarkeit bietet.

  Microservices-Architekturen erleichtern die Skalierung von Anwendungen und das Tempo der Entwicklung, wodurch die Innovation ermöglicht und die Markteinführung für neue Funktionen beschleunigt wird.

  —Amazon Web Services (AWS)

Vorteile von Microservices und Cloud-Anwendungen

Microservices werden aufgrund der immanenten Flexibilität der Architektur in der Cloud immer beliebter. Eine kürzlich von Intel durchgeführte Studie zeigte, dass 83 Prozent aller neuen Cloud-nativen Anwendungen und SaaS-Lösungen Microservices verwenden.[1]

Einer der größten Vorteile von Microservices ist die Geschwindigkeit und Flexibilität der Bereitstellung. In dem Maße, wie Cloud-Anwendungen und Workloads in Größe und Umfang wachsen, wird es zunehmend schwieriger und zeitintensiver, monolithische Architekturen an die neuen Anforderungen anzupassen. Microservices werden disaggregiert, sodass ihre Entwicklung, Bereitstellung und Wartung in einem verteilten Modell verwaltet werden kann.

So können beispielsweise verschiedene unabhängige Entwicklungsteams die Verantwortung für einen oder mehrere der Microservices erhalten. Die Verteilung der Verantwortung erleichtert die Neuordnung der DevOps-Funktion gemäß der geschäftlichen Fähigkeiten jeder Einheit. Das kann zur Modernisierung des gesamten Entwicklungsprozesses und der Beseitigung von Silos führen.

Microservices stellen auch einige Herausforderungen bei der Bereitstellung dar. Einzelne Module können auf verschiedenen Systemen ausgeführt werden, um einen Auftrag zu erfüllen, was dazu führen kann, dass die Leistung von einem Modul zum anderen unterschiedlich ist.

Zudem kann die Latenz bei einigen Microservices ein Problem werden, wenn es einen plötzlichen Anstieg der Nachfrage gibt. Diese Probleme können oft durch eine Überversorgung mit Ressourcen gelöst werden, um Nachfragespitzen abzudecken.

Da die Microservices disaggregiert sind, kann ein Serviceausfall in einem Modul die anderen kaum oder überhaupt nicht beeinträchtigen. Das ist ein Vorteil, aber ein Serviceausfall kann auch zu Problemen führen.

So kann es beispielsweise schwierig sein, einen ausgefallenen Service zu debuggen, nachdem er den Betrieb eingestellt hat. Eine Lösung besteht darin, die Infrastruktur zu koordinieren und alle Prozesse und Datenflüsse im Voraus zu überwachen. Diese Überwachungsprotokolle können die hardwaregestützten Telemetry Collectors und PMUs (Performance Monitoring Units) nutzen, die in Cloud-Plattformen auf der Basis von Intel® Technik integriert sind.

So funktioniert die Microservices-Architektur

In einer Microservices-Architektur wird eine komplexe Anwendung in separate Fähigkeiten disaggregiert, die unabhängig entwickelt und benutzt werden können. Die einzelnen Microservices kommunizieren miteinander, oft über APIs, und sie sind locker verbunden. Allerdings kann jeder Microservice separat entwickelt, bereitgestellt und aktualisiert werden.

Die Bereitstellung von Microservices folgt oft einem von drei Musten:

1. Cloud-native. Einige etablierte, hochvolumige Anwendungen und Services beginnen als Microservices und bleiben in der Cloud. Laut der International Data Corporation (IDC) sind etwa 56 Prozent der Microservices Cloud-native, während die verbleibenden 44 Prozent als Legacy-Anwendungen begannen.[2]
2. Neustrukturierung und Verlagerung. Diese Bereitstellungen beginnen vor Ort oder in einem Rechenzentrum am Netzwerkrand und werden umstrukturiert, um sich an die Cloud-basierte Microservices-Architektur anzupassen. Die Neustrukturierung kann eine Neuzuweisung von Datenbanken und anderen Ressourcen in Verbindung mit der monolithischen Architektur umfassen, sodass sie mit den entsprechenden Microservices verbunden sind.
3. Anhebung und Verlagerung. Einige Organisationen migrieren ihre Anwendungen zu einer Microservices-Architektur ohne Neustrukturierung, in einem einfachen Verfahren der Anhebung und Verlagerung.

Microservices und DevOps

Die disaggregierte Natur der Microservices-Architektur führt oft dazu, dass DevOps-Teams einen funktionsübergreifenden Ansatz für die Anwendungsentwicklung einsetzen.

Anstatt Software in einem Stack zu entwickeln, wobei ein Team an der Firmware, ein anderes an der Middleware, und ein drittes an der Benutzerschnittstelle arbeitet, konzentriert sich die Entwicklung von Microservices mehr auf Geschäftsfunktionen.

Auf gewisse Weise können die Entwicklungsprozesse und die Organisation der DevOps-Teams den Strukturen der Microservices selbst ähneln, mit quasi-unabhängigen geschlossenen Einheiten, die lose zusammenarbeiten und Barrieren und Silos hinter sich lassen.

> Es ist sehr wichtig, dass alle Stakeholder verstehen, dass die Modernisierung einer monolithischen Anwendung zu einer Microservices-Architektur eine epische Reise ist und viele Iterationen benötigen kann. Es ist für die Architekten und Entwickler erforderlich, verschiedene Aspekte des Monolithen genau zu bewerten und einen Migrationsansatz für jeden davon zu entwickeln.
>
> —IBM, „Challenges and Patterns for Modernizing a Monolithic Application into Microservices“

Sicherheit von Microservices

Da Microservices oft so konzipiert sind, dass sie miteinander interagieren können, ist es wichtig, Daten zu schützen, während sie an jedem Ende der Interaktion oder im Übergang zwischen diesen beiden Punkten benutzt werden.

Verschlüsselung mit der Mutual Transport Layer Security (mTLS) ist eine übliche Lösung, die das Risiko des Eindringens und der Malware für Daten an jedem Endpunkt und auf den Transit reduziert. Das Ziel von mTLS ist die Verschlüsselung jeder Anfrage von jedem Microservice. So ein ganzheitlicher Ansatz für die Sicherheit bildet die Grundlage einer Zero-Trust-Umgebung, in der jeder Microservice, jeder Benutzer und jede Verbindung unabhängig verifiziert und überprüft werden müssen.

Es ist nicht immer immer erforderlich, die Authentifizierung und Verschlüsselung in jeden einzelnen Microservice z verwenden. Um Redundanzen zu vermeiden verwenden viele Entwickler stattdessen einen Service-Mesh. Der Mesh dient als Infrastruktur-Layer oder Proxy-Instanz in der Microservices-Architektur, um die Kommunikation zu sichern, kontrollieren und überwachen,

Sicherheitsprotokolle können eine erhebliche Rechenleistung erfordern, die Ressourcen aufbrauchen und die Bereitstellung von Microservices verlangsamen können. Um Verschlüsselungsalgorithmen zu beschleunigen und die Latenz zu reduzieren, können Entwickler von Microservices Intel® Data Protection Technology (Intel® DPT) bereitstellen.

Intel® DPT umfasst die Intel® Advanced Encryption Standard – New Instructions (Intel® AES-NI) und Intel® Secure Key Instructions, sowie den Intel® Digital Random Number Generator (Intel® DRNG) zur schnellen Erstellung von Verschlüsselungsschlüsseln.

Diese fortschrittlichen algorithmischen Schutzfunktionen werden für die integrierten Sicherheitsfunktionen der skalierbaren Intel® Xeon® Prozessoren optimiert. Beispielsweise bieten Intel® Advanced Vector Extensions 512 Intel® AVX-512) und algorithmische Innovationen wie symmetrische Verschlüsselung, sicheres Hashing und Function Stitching deutliche Leistungsverbesserung für die Kryptografie.

Diese und andere Hardware-fähige Sicherheitsfunktionen sind bei den großen Cloud-Service-Providern in Instanzen mit skalierbaren Intel® Xeon® Prozessoren verfügbar.

Beispiele für die Microservices-Architektur und ihr sich entwickelndes Framework

Da Cloud-Anwendungen weiterhin in Größe und Umfang wachsen, verlassen sich Entwickler zunehmend auf Microservices, um komplexe, multifunktionale Anwendungen zu erstellen und sie schnell zu skalieren, um sich an unterschiedliche Nutzungsmodelle anzupassen.

Für jede auf Microservices basierende Bereitstellung gibt es normalerweise mehrere locker verbundene Komponenten-Services, die separat ausgeführt werden. Diese modularen Komponenten arbeiten zusammen, um eine integrierte Benutzererfahrung oder Anwendung zu erschaffen.

In einigen auf Microservices basierenden Anwendungen, in denen die Benutzererfahrungen sich nach den Gruppeneigenschaften unterscheiden kann, ist ein Vorteil von Microservices, dass Code geteilt und wiederverwendet werden kann. Dieser Ansatz macht es überflüssig, mehrere Instanzen mit dem gleichen Service zu erstellen und zu warten. Wenn eine der differenzierten Erfahrungen eine Anpassung benötigt, können zusätzliche Microservices eingeschlossen werden.

Beispielsweise basieren beliebte Mitfahr-Apps auf Microservices, aber Fahrer und Passagiere haben unterschiedliche Benutzererfahrungen. Fahrerverwaltung, Standortverfolgung, Passagierprofile und Zahlungsverarbeitung gehören zu den unterschiedlichen Microservices, die zusammen die Benutzer- und Fahrerinterfaces auf ihren entsprechenden Mobilgeräten unterstützen. Alle Interfaces teilen die gleiche Marke, aber einige Funktionalitäten können für jede Gruppe unterschiedlich sein.

Eine Microservices-Architektur wird auch oft in E-Commerce-Shops verwendet. Die Produktempfehlung und der Kassiervorgang werden möglicherweise als einzelne Microservices entwickelt und bereitgestellt, aber der Käufer würde beide Prozesse innerhalb der Markenumgebung und des Interface des Online-Shops erleben.

Wann sollte man Microservices verwenden?

Ein Microservices-Ansatz kann dabei helfen, Anwendungen zu migrieren und skalieren, indem die komplexesten Lösungen in ihre Komponenten zerlegt werden. Jeder Microservice wird unabhängig entwickelt und bereitgestellt, und die verschiedenen Microservices arbeiten als locker integrierte Einheit zusammen.

Unternehmen verwenden wahrscheinlich Microservices, wenn sie eine neue Cloud-native Lösung entwickeln. Sie können auch Microservices bereitstellen, wenn eine bestehende monolithische Architektur zu schwerfällig wird, um ihre veränderten Anforderungen zu erfüllen.

Der Wechsel zu einer Microservices-Architektur kann zu Änderungen in der Organisation und der Struktur der DevOps-Teams selbst führen. Im Laufe der Zeit richten sich Teams vielleicht an ihren neuen, funktionsübergreifenden Unternehmensfunktionen aus, statt an den Silos, die den funktionalen Schichten im Technologie-Stack entsprechen.

FAQs

Häufig gestellte Fragen

Microservices sind Module mit geringen Anforderungen, die als Bausteine von komplexen Cloud-basierten Anwendungen dienen können. Während einzelne Microservices unabhängig arbeiten können, werden sie unter einem vereinheitlichten Interface locker gekoppelt.

Microservices können im Vergleich zu monolithischen Architekturen schnell und leicht bereitgestellt werden. Das Microservices-Framework ist auch leicht skalierbar, um immer komplexere Workloads und erweiterte Nutzungsmodelle zu erfüllen.

Cloud-Architekturen und Modelle

Microservices werden im Kontext von verschiedenen Cloud-Architekturen und Cloud-Servicemodellen bereitgestellt.

Design der Cloud-Architektur

Um die Skalierung, Widerstandsfähigkeit und Agilität zu maximieren, muss ein Cloud-Design die einzigartigen Anforderungen von Workloads und Benutzern sowie die Betriebskosten in Betracht ziehen.

Cloud-Architektur entdecken

Container und die Cloud

Microservices werden oft in Containern bereitgestellt, um die Flexibilität der Workloads und die Portabilität zu verbessern.

Erfahren Sie mehr über Containerisierung

Cloud-Orchestrierung und Ressourcennutzung

Cloud-Orchestrierung ist die Verwaltung und Koordination von Cloud-Ressourcen – Computing, Datenspeicher und Netzwerke. Spezifische Workloads werden auf der Grundlage der Anforderungen der Anwendung den entsprechenden Knoten zugewiesen.

Lesen Sie mehr über die Cloud-Orchestrierung

Automatisierung und Optimierung des Cloud-Management

Tools für vereinheitlichtes Cloud-Management können die Einrichtung und die Wartung dieser komplexen Multicloud-Umgebungen vereinfachen.

Erfahren Sie mehr über Cloud-Management-Tools

5 Hindernisse für Microservices

Erfahren Sie mehr über fünf entscheidende Herausforderungen bei Microservices und wie Sie mit den Tools und Ressourcen von Intel das Beste aus Ihrer Umgebung herausholen können.

Weitere Infos

Haftungsausschluss[3]