AWSWeißbuch

AWS-Grundlagen für mehrere Regionen

AWS-Grundlagen für mehrere Regionen: AWSWeißbuch

Table of Contents

AWS-Grundlagen für mehrere Regionen

Datum der Veröffentlichung: 20. Dezember 2022 () Dokumentversionen

Überblick

Dieses anspruchsvolle, 300-stufige paper richtet sich an Cloud-Architekten und Führungskräfte, die Workloads aufbauen und daran interessiert sindAWS, eine multiregionale Architektur zu verwenden, um die Resilienz ihrer Workloads zu verbessern. In diesem paper wird von Grundkenntnissen über AWS Infrastruktur und Dienste ausgegangen. Es beschreibt allgemeine Anwendungsfälle für mehrere Regionen, erläutert grundlegende Konzepte und Implikationen für mehrere Regionen in Bezug auf Design, Entwicklung und Bereitstellung und bietet präskriptive Anleitungen, anhand derer Sie besser bestimmen können, ob eine Architektur mit mehreren Regionen für Ihre Workloads geeignet ist.

Sind Sie Well-Architected?

Das AWSWell-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 Management Console, können Sie Ihre Workloads anhand dieser bewährten Methoden überprüfen, indem Sie für jede Säule eine Reihe von Fragen beantworten.

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

Einführung

Jede AWS-Regionbesteht aus mehreren unabhängigen und physisch getrennten Availability Zones innerhalb eines geografischen Gebiets. Eine strikte logische Trennung zwischen den Softwarediensten in jeder Region wird beibehalten. Durch dieses gezielte Design wird sichergestellt, dass ein Ausfall der Infrastruktur oder der Dienste in einer Region nicht zu einem korrelierten Ausfall in einer anderen Region führt.

Die meisten AWS Kunden können ihre Stabilitätsziele für einen Workload in einer einzelnen Region mithilfe mehrerer Availability Zones (AZs) oder regionaler Dienste erreichen. AWS Ein Teil der Kunden setzt jedoch aus drei Gründen auf Architekturen mit mehreren Regionen.

- Sie haben hohe Anforderungen an Verfügbarkeit und Betriebskontinuität für ihre Workloads der höchsten Stufe, die ihrer Meinung nach nicht in einer einzigen Region erfüllt werden können.
- Sie müssen die Anforderungen an die Datenhoheit erfüllen (z. B. die Einhaltung lokaler Gesetze und Vorschriften und die Einhaltung der Vorschriften), die erfordern, dass Workloads innerhalb einer bestimmten Jurisdiktion betrieben werden.
- Sie müssen die Leistung und das Kundenerlebnis für die Workloads verbessern, indem sie die Workloads an Standorten ausführen, die den Endbenutzern am nächsten sind.

Dieses paper konzentriert sich auf die Anforderungen an hohe Verfügbarkeit und Betriebskontinuität und hilft Ihnen, sich mit den Überlegungen zur Einführung einer multiregionalen Architektur für einen Workload vertraut zu machen. Wir beschreiben grundlegende Konzepte, die für den Entwurf, die Entwicklung und die Bereitstellung eines Workloads mit mehreren Regionen gelten, sowie einen verbindlichen Rahmen, anhand dessen Sie feststellen können, ob eine Architektur mit mehreren Regionen die richtige Wahl für einen bestimmten Workload ist. Sie müssen sicherstellen, dass eine Architektur mit mehreren Regionen die richtige Wahl für Ihren Workload ist, da diese Architekturen eine Herausforderung darstellen und es möglich ist, dass die Gesamtverfügbarkeit des Workloads sinkt, wenn sie nicht korrekt ausgeführt wird.

Entwicklung und Betrieb für Resilienz in einer einzigen Region

Bevor Sie sich mit Konzepten für mehrere Regionen befassen, sollten Sie zunächst sicherstellen, dass Ihr Workload in einer einzigen Region bereits so belastbar wie möglich ist. Um dies zu erreichen, bewerten Sie Ihre Arbeitslast anhand der Säulen Zuverlässigkeit und Operational Excellence des AWS Well-Architected Framework und nehmen Sie alle erforderlichen Änderungen vor, um die empfohlenen Best Practices zu übernehmen. Die folgenden Konzepte werden im AWS Well-Architected Framework behandelt:

- Workload-Segmentierung auf der Grundlage von Domänengrenzen
- Klar definierte Serviceverträge
- Verwaltung und Kopplung von Abhängigkeiten
- Umgang mit Ausfällen, Wiederholungsversuchen und Back-off-Strategien
- Idempotente Operationen und statusbehaftete und zustandslose Transaktionen
- Betriebsbereitschaft und Change-Management
- Den Zustand der Arbeitslast verstehen
- Auf Ereignisse reagieren

Um die Resilienz einzelner Regionen weiter zu verbessern, sollten Sie sich die Konzepte ansehen und anwenden, die in Advanced Multi-AZ Resilience Patterns for Handling Gray Failures erörtert wurden. In diesem paper werden bewährte Methoden zur Verwendung von Replikaten in jeder Availability Zone zur Eindämmung von Ausfällen beschrieben. Außerdem werden die in AWS Well Architected eingeführten Multi-AZ-Konzepte näher erläutert. Sobald Sie die empfohlenen Konzepte und bewährten Methoden zur Erzielung der höchsten Ausfallsicherheit in einer einzelnen Region vollständig angewendet haben, kann ein bestimmter Workload anhand der Grundlagen für Architekturen mit mehreren Regionen bewertet werden, um festzustellen, ob die Resilienz des Workloads mithilfe eines multiregionalen Ansatzes erhöht werden kann.

Grundlegendes Prinzip für mehrere Regionen 1: Die Anforderungen verstehen

Wie bereits erwähnt, sind hohe Verfügbarkeit und Betriebskontinuität die häufigsten Gründe für Architekturen mit mehreren Regionen. Verfügbarkeitsmetriken messen den Prozentsatz der Zeit, in der ein Workload über einen bestimmten Zeitraum genutzt werden kann, während Kennzahlen zur Betriebskontinuität die Wiederherstellung bei groß angelegten und in der Regel länger andauernden Ereignissen messen.

Die Messung der Verfügbarkeit ist ein nahezu kontinuierlicher Prozess. Spezifische Messungen oder Metriken können variieren, ergeben sich jedoch in der Regel aus einer Zielverfügbarkeit, die am häufigsten als Neun bezeichnet wird (z. B. Verfügbarkeit von 99,99%). Bei Verfügbarkeitszielen gibt es keine Einheitslösung. Verfügbarkeitsziele müssen auf Workload-Ebene festgelegt werden, anstatt ein einziges Ziel für alle Workloads festzulegen, bei dem die unkritischen Komponenten von den kritischen getrennt werden.

Um die Kontinuität des Betriebs zu gewährleisten, werden in der Regel die folgenden point-in-time Messwerte verwendet:

- Recovery Time Objective (RTO) — RTO ist die maximal zulässige Verzögerung zwischen der Betriebsunterbrechung und der Wiederherstellung des Dienstes. Dieser Wert bestimmt eine akzeptable Dauer, für die der Service beeinträchtigt ist.
- Recovery Point Objective (RPO) — RPO ist die maximal zulässige Zeitspanne seit dem letzten Datenwiederherstellungspunkt. Dies bestimmt, was als akzeptabler Datenverlust zwischen dem letzten Wiederherstellungspunkt und einer Betriebsunterbrechung angesehen wird.

Ähnlich wie bei der Festlegung von Verfügbarkeitszielen sollten RTO und RPO auch auf Workload-Ebene definiert werden. Um strengere Anforderungen an die Betriebskontinuität oder hohe Verfügbarkeit zu erreichen, sind höhere Investitionen erforderlich. Allerdings kann oder erfordert nicht jede Anwendung das gleiche Maß an Belastbarkeit. Die Schaffung eines Tiering-Mechanismus kann dazu beitragen, den Rahmen zu schaffen, der Geschäfts- und IT-Verantwortliche bei der Identifizierung der anspruchsvollsten Anwendungen auf der Grundlage ihrer geschäftlichen Auswirkungen und deren entsprechender Einstufung aufeinander abstimmt. Beispiele für Tiering finden Sie in den folgenden Tabellen.

Tabelle 1 — Beispiel für ein Resilienz-Tiering für SLA

Verfügbarkeit Service Level Agreement (SLA)	Stufe der Resilienz	Zulässige Ausfallzeit pro Jahr
99,99 %	Platin	52.60 Minuten
99,90%	Gold	8,77 Stunden
99,5 %	Silber	1,83 Tage

Tabelle 2 — Beispiel für ein Resilienz-Tiering für RTO und RPO

Stufe	Maximaler RTO	Maximales RPO	Kriterien	Kosten
Platin	15 Minuten	fünf Minuten	Unternehmenskritische Workloads	$$$
Gold	15 Minuten — sechs Stunden	zwei Stunden	Wichtige, aber nicht geschäftskritische Workloads	$$
Silber	sechs Stunden — ein paar Tage	24 Stunden	Unkritische Workloads	$

Bei der Gestaltung von Workloads im Hinblick auf Ausfallsicherheit ist ein Verständnis des Zusammenhangs zwischen Hochverfügbarkeit und Betriebskontinuität erforderlich. Wenn ein Workload beispielsweise eine Verfügbarkeit von 99,99% erfordert, sind nicht mehr als 53 Minuten Ausfallzeit pro Jahr tolerierbar. Es kann mindestens fünf Minuten dauern, bis ein Fehler erkannt wird, und weitere zehn Minuten, bis ein Bediener eingreift, Entscheidungen über Wiederherstellungsschritte trifft und diese Schritte ausführt. Es ist nicht ungewöhnlich, dass es 30 bis 45 Minuten dauert, bis ein einzelnes Problem behoben ist. In diesem Fall kann eine regionsübergreifende Strategie zur Bereitstellung einer isolierten Instanz, die korrelierte Auswirkungen beseitigt, die Fortsetzung des Betriebs ermöglichen, indem innerhalb einer bestimmten Zeit ein Failover durchgeführt wird, während die anfängliche Beeinträchtigung unabhängig geprüft wird. In diesem Fall ist die Definition des geeigneten RTO und RPO erforderlich.

Für unternehmenskritische Workloads mit extremen Verfügbarkeitsanforderungen (z. B. Verfügbarkeit von 99,99% oder höher) oder strengen Anforderungen an die Betriebskontinuität, die nur durch ein Failover in eine andere Region erfüllt werden können, kann ein regionsübergreifender Ansatz angemessen sein. Diese Anforderungen gelten jedoch in der Regel nur für einen kleinen Teil des Workload-Portfolios eines Unternehmens, für das eine begrenzte Wiederherstellungszeit gilt, die in Minuten oder Stunden gemessen wird. Sofern für eine Anwendung keine Wiederherstellungszeit von Minuten oder einigen Stunden erforderlich ist, ist es möglicherweise besser, darauf zu warten, dass eine regionale Unterbrechung der Anwendung in der betroffenen Region behoben wird. Dies ist in der Regel auf Workloads niedrigerer Stufen ausgerichtet.

Vor der Implementierung einer multiregionalen Architektur sollten sich Entscheidungsträger und technische Teams im Hinblick auf die Auswirkungen auf die Kosten, einschließlich betrieblicher und infrastruktureller Kostentreiber, einig sein. Eine typische Architektur mit mehreren Regionen kann im Vergleich zu einem Ansatz mit nur einer Region zu einem zweifachen Kostenanstieg führen. Zwar gibt es mehrere regionsübergreifende Muster für die Geschäftskontinuität, wie z. B. den Betrieb mit Hot-Standby-, Warm-Standby- und Pilotbetrieb, aber das Muster mit dem geringsten Risiko, die Wiederherstellungsziele zu erreichen, beinhaltet den Betrieb im Hot-Standby-Modus, wodurch sich die Kosten für Ihre Arbeitslast verdoppeln.

Wichtige Leitlinien

- Ziele für Verfügbarkeit und Kontinuität des Betriebs wie RTO und RPO sollten pro Workload festgelegt und mit den Geschäfts- und IT-Stakeholdern abgestimmt werden.
- Die meisten Ziele in Bezug auf Verfügbarkeit und Kontinuität des Betriebs können innerhalb einer einzigen Region erreicht werden. Bei Zielen, die mit einer einzelnen Region nicht erreicht werden können, sollten mehrere Regionen in Betracht gezogen werden, wobei ein klares Bild von den Kompromissen zwischen Kosten, Komplexität und Nutzen gezogen werden muss.

Multi-Region-Grundlagen 2: Die Daten verstehen

Die Verwaltung von Daten ist bei Architekturen mit mehreren Regionen kein triviales Problem. Die geografische Entfernung zwischen Regionen führt zu einer unvermeidlichen Latenz, die sich in der Zeit äußert, die für die Replikation von Daten zwischen Regionen benötigt wird. Kompromisse zwischen Verfügbarkeit, Datenkonsistenz und der Einführung höherer Latenzzeiten bei einem Workload, der eine Architektur mit mehreren Regionen verwendet, werden notwendig sein. Unabhängig davon, ob Sie asynchrone oder synchrone Replikation verwenden, müssen Sie Ihre Anwendung an die Verhaltensänderungen anpassen, die die Replikationstechnologie mit sich bringt. Aufgrund von Problemen im Zusammenhang mit Datenkonsistenz und Latenz ist es sehr schwierig, eine bestehende Anwendung, die für eine einzelne Region konzipiert wurde, in mehrere Regionen umzuwandeln. Es ist wichtig, die Anforderungen an die Datenkonsistenz und die Datenzugriffsmuster für bestimmte Workloads zu verstehen, um die Kompromisse abzuwägen.

2a: Die Anforderungen an die Datenkonsistenz verstehen

Das CAP-Theorem bietet eine Referenz für Überlegungen zu den Kompromissen zwischen Datenkonsistenz, Verfügbarkeit und Netzwerkpartitionen, von denen für eine Arbeitslast nur zwei gleichzeitig erfüllt werden können. Multiregion umfasst definitionsgemäß Netzwerkpartitionen zwischen Regionen, sodass Sie zwischen Verfügbarkeit und Konsistenz wählen müssen.

Wenn Sie sich für die regionsübergreifende Verfügbarkeit der Daten entscheiden, treten bei transaktionalen Schreibvorgängen keine nennenswerten Latenzen auf, da auf die asynchrone Replikation festgeschriebener Daten zwischen Regionen angewiesen ist, was zu einer verringerten Konsistenz zwischen den Regionen führt, bis die Replikation abgeschlossen ist. Wenn bei der asynchronen Replikation ein Fehler in der primären Region auftritt, besteht eine hohe Wahrscheinlichkeit, dass Schreibvorgänge aus der primären Region noch nicht repliziert werden. Dies führt zu einem Szenario, in dem die neuesten Daten erst verfügbar sind, wenn die Replikation wieder aufgenommen wird, und ein Abgleichprozess erforderlich ist, um laufende Transaktionen zu verarbeiten, die nicht aus der Region repliziert wurden, in der der Ausfall aufgetreten ist.

Für Workloads, bei denen asynchrone Replikation bevorzugt wird, können Sie Services wie Amazon Aurora und Amazon DynamoDB verwenden, die eine asynchrone regionsübergreifende Replikation ermöglichen. Sowohl die globalen Tabellen von Amazon Aurora Global Database als auch Amazon DynamoDB verfügen über standardmäßige CloudWatchAmazon-Metriken, um die Überwachung von Replikationsverzögerungen zu unterstützen.

Der Workload so zu gestalten, dass er die Vorteile ereignisgesteuerter Architekturen nutzt, ist für eine Strategie mit mehreren Regionen von Vorteil, da der Workload dadurch die asynchrone Replikation von Daten umfassen kann und der Status durch die Wiedergabe von Ereignissen wiederhergestellt werden kann. Da Streaming- und Messaging-Dienste Nachrichtennutzdaten in einer einzigen Region zwischenspeichern, muss ein regionaler Failover-/Failback-Prozess einen Mechanismus zur Umleitung von Client-Eingabedatenströmen sowie zum Abgleich von laufenden und/oder nicht zugestellten Payloads, die in der Region gespeichert sind, in der der Ausfall aufgetreten ist, beinhalten.

Wenn Konsistenz ausgewählt ist, kommt es zu einer erheblichen Latenz, da Daten während transaktionaler Schreibvorgänge synchron repliziert werden. Wenn Sie synchron in mehrere Regionen schreiben und der Schreibvorgang nicht in allen Regionen erfolgreich ist, wird die Verfügbarkeit möglicherweise beeinträchtigt, da die Transaktion nicht festgeschrieben wird und erneut versucht werden muss. Wiederholte Versuche, die Daten synchron in alle Regionen zu schreiben, gehen bei jedem Versuch auf Kosten der Latenz. Irgendwann, wenn die Wiederholungsversuche ausgeschöpft sind, muss entschieden werden, ob die Transaktion entweder komplett fehlschlagen soll, wodurch die Verfügbarkeit reduziert wird, oder ob die Transaktion nur auf verfügbare Regionen übertragen werden soll, was zu Inkonsistenzen führt. Es gibt Technologien zur Quorumbildung wie Paxos, mit denen Daten synchron repliziert und übertragen werden können, für die jedoch erhebliche Investitionen von Entwicklern erforderlich sind.

Wenn Schreibvorgänge synchrone Replikationen über mehrere Regionen hinweg beinhalten, um hohe Konsistenzanforderungen zu erfüllen, erhöht sich die Schreiblatenz um eine Größenordnung. Eine höhere Schreiblatenz kann normalerweise nicht ohne wesentliche Änderungen in eine Anwendung nachgerüstet werden. Im Idealfall muss dies bei der ersten Entwicklung der Anwendung berücksichtigt werden. Bei Workloads mit mehreren Regionen, bei denen synchrone Replikation Priorität hat, können AWSPartnerlösungen helfen.

2b: Verständnis der Datenzugriffsmuster

Datenzugriffsmuster für Workloads lassen sich in einen der folgenden Typen einteilen: leseintensiv oder schreibintensiv. Wenn Sie dieses Merkmal für einen bestimmten Workload verstehen, können Sie sich bei der Auswahl einer geeigneten Architektur für mehrere Regionen entscheiden.

Für leseintensive Workloads wie statische Inhalte, die vollständig schreibgeschützt sind, kann eine aktive/aktive Multiregions-Architektur ohne nennenswerte Komplexität erreicht werden. Die Bereitstellung statischer Inhalte am Netzwerkrand mithilfe eines Content Distribution Network (CDN) gewährleistet die Verfügbarkeit, indem Inhalte zwischengespeichert werden, die dem Endbenutzer

am nächsten sind. Die Verwendung von Funktionen wie Origin-Failover innerhalb von Amazon CloudFront kann dazu beitragen, dies zu erreichen. Eine weitere Option ist die Bereitstellung von statusfreiem Computing in mehreren Regionen und die Verwendung von DNS, um Benutzer zur nächstgelegenen Region weiterzuleiten, wo sie die Inhalte lesen können. Um dies zu erreichen, kann Route 53 mit Geolocation-Routing-Richtlinie verwendet werden.

Für leseintensive Workloads mit einem höheren Prozentsatz an Lese- als Schreibvorgängen kann eine lokale Lesestrategie und eine globale Schreibstrategie verwendet werden. Das bedeutet, dass alle Schreibvorgänge in eine Datenbank in einer bestimmten Region gehen und die Daten asynchron in alle anderen Regionen repliziert werden. Lesevorgänge können zu diesem Zweck in jeder Region durchgeführt werden. Bei diesem Ansatz ist ein Workload erforderlich, der letztendlich für Konsistenz sorgen muss, da lokale Lesevorgänge aufgrund der erhöhten Latenz bei der regionsübergreifenden Replikation von Schreibvorgängen veraltet sein können.

Aurora Global Database kann bei der Bereitstellung von Read Replicas in einer Standby-Region helfen, die ausschließlich den gesamten Lesetraffic lokal verarbeiten kann, und einem einzigen primären Datenspeicher in einer bestimmten Region für Schreibvorgänge. Daten werden asynchron von der Primär- in die Standby-Datenbank (Read Replicas) repliziert, und die Standby-Datenbanken können zur Primärdatenbank heraufgestuft werden, wenn Sie Failover-Operationen auf die Standby-Region umstellen müssen. Wenn ein Workload besser für nicht-relationale Datenmodelle geeignet ist, kann DynamoDB auch in diesem Ansatz verwendet werden. Auch hier muss der Workload letztendlich konsistent sein, weshalb er möglicherweise neu geschrieben werden muss, wenn er nicht von Anfang an darauf ausgelegt war.

Bei schreibintensiven Workloads sollte eine primäre Region ausgewählt werden, und die Fähigkeit zum Failover auf eine Standby-Region sollte in den Workload integriert werden. Im Vergleich zu einem aktiven/aktiven Ansatz ist ein primärer/Standby-Ansatz weniger kompliziert. Dies liegt daran, dass bei einer aktiven/aktiven Architektur der Workload neu geschrieben werden muss, um intelligentes Routing zu Regionen zu ermöglichen, Sitzungsaffinität herzustellen, idempotente Transaktionen sicherzustellen und potenzielle Konflikte zu bewältigen.

Für die meisten Workloads, die aus Gründen der Resilienz auf mehrere Regionen abzielen, ist kein aktiver/aktiver Ansatz erforderlich. Eine Sharding-Strategie kann eingesetzt werden, um die Widerstandsfähigkeit zu erhöhen, indem der Explosionsradius einer Beeinträchtigung innerhalb des Kundenstamms begrenzt wird. Wenn Sie einen Kundenstamm effektiv teilen können, können Sie für jeden Shard verschiedene Hauptregionen auswählen. Wenn Sie beispielsweise Kunden so teilen können, dass die Hälfte der Kunden auf Region Eins und die andere Hälfte auf Region Zwei

ausgerichtet ist und Regionen als Zellen behandelt werden, kann ein zellenübergreifender Ansatz geschaffen werden, der den Wirkungsradius Ihrer Arbeitslast reduziert.

Der Sharding-Ansatz kann mit einem Primär-/Standby-Ansatz kombiniert werden, um Failover-Funktionen für die Shards bereitzustellen. Ein getesteter Failover-Prozess muss in die Arbeitslast integriert werden, und es muss auch ein Prozess für den Datenabgleich entwickelt werden, um die Transaktionskonsistenz der Datenspeicher nach dem Failover sicherzustellen. Diese werden später in diesem paper ausführlicher behandelt.

Wichtige Leitlinien

- Es besteht eine hohe Wahrscheinlichkeit, dass Schreibvorgänge, die zur Replikation ausstehen, nicht in die Standby-Region übernommen werden, wenn ein Fehler auftritt. Daten sind erst verfügbar, wenn die Replikation wieder aufgenommen wird (unter der Annahme einer asynchronen Replikation).
- Im Rahmen des Failovers ist ein Datenabgleich erforderlich, um sicherzustellen, dass ein transaktionskonsistenter Status für Datenspeicher, die asynchrone Replikation verwenden, aufrechterhalten wird.
- Wenn eine hohe Konsistenz erforderlich ist, müssen die Workloads so geändert werden, dass sie die erforderliche Latenz des Datenspeichers, der synchron repliziert, tolerieren.

Grundlegendes 3 für mehrere Regionen: Verstehen Sie Ihre Workload-Abhängigkeiten

Ein bestimmter Workload kann in einer Region mehrere Abhängigkeiten aufweisen, z. B. verwendete AWS Dienste, interne Abhängigkeiten, Abhängigkeiten von Drittanbietern, Netzwerkabhängigkeiten, Zertifikate, Schlüssel, Geheimnisse und Parameter. Um den Betrieb des Workloads in einem Ausfallszenario sicherzustellen, sollten keine Abhängigkeiten zwischen der primären Region und der Standby-Region bestehen; beide sollten unabhängig voneinander arbeiten können. Um dies zu erreichen, müssen alle Abhängigkeiten im Workload überprüft werden, um sicherzustellen, dass sie in jeder Region verfügbar sind. Dies ist erforderlich, da ein Ausfall in der primären Region keine Auswirkungen auf die Standby-Region haben sollte. Darüber hinaus ist es unerlässlich zu wissen, wie der Workload funktioniert, wenn sich eine Abhängigkeit in einem heruntergestuften Zustand befindet oder vollständig nicht verfügbar ist, damit Lösungen entwickelt werden können, die diesem Problem angemessen gerecht werden.

3a: Dienstleistungen AWS

Beim Entwurf einer multiregionalen Architektur ist ein Verständnis der spezifischen AWS Dienste erforderlich, die verwendet werden. Der erste Aspekt besteht darin, zu verstehen, über welche Funktionen der Service verfügt, um mehrere Regionen zu ermöglichen, und ob eine Lösung entwickelt werden muss, um die multiregionalen Ziele zu erreichen. Bei Amazon Aurora und Amazon DynamoDB gibt es beispielsweise eine Funktion zum asynchronen Replizieren von Daten in eine Standby-Region. Alle AWS Serviceabhängigkeiten müssen in allen Regionen verfügbar sein, in denen ein Workload ausgeführt werden soll. Um sicherzustellen, dass die Dienste, die verwendet werden, in den gewünschten Regionen verfügbar sind, überprüfen Sie die Liste AWS-Region aller Dienste.

3b: Interne Abhängigkeiten und Abhängigkeiten von Drittanbietern

Stellen Sie bei allen internen Abhängigkeiten, die ein Workload hat, sicher, dass er in den Regionen verfügbar ist, in denen der Workload ausgeführt wird. Wenn der Workload beispielsweise aus vielen Microservices besteht, sollten Sie sich mit allen Microservices auskennen, die eine Geschäftsfähigkeit ausmachen. Stellen Sie anschließend sicher, dass all diese Microservices in jeder Region bereitgestellt werden, von der aus der Workload ausgeführt wird.

Von regionsübergreifenden Aufrufen zwischen Microservices innerhalb eines Workloads wird abgeraten, und die regionale Isolation sollte beibehalten werden. Dies liegt daran, dass die Schaffung regionsübergreifender Abhängigkeiten das Risiko eines korrelierten Fehlers erhöht, wodurch die Vorteile zunichte gemacht werden, die Sie mit isolierten regionalen Implementierungen des Workloads erzielen möchten. Lokale Abhängigkeiten können ebenfalls Teil der Arbeitslast sein. Daher ist es unerlässlich zu verstehen, wie sich die Merkmale dieser Integrationen ändern könnten, wenn sich die primäre Region ändern würde. Wenn sich die Standby-Region beispielsweise weiter von der lokalen Umgebung entfernt befindet, wird sich die erhöhte Latenz negativ auswirken.

Wenn Sie Software-as-a-Service (SaaS) -Lösungen, Software Development Kits (SDKs) und andere Abhängigkeiten von Drittanbieterprodukten verstehen und Szenarien ausführen können, in denen diese Abhängigkeiten entweder beeinträchtigt oder nicht verfügbar sind, erhalten Sie einen besseren Einblick in die Funktionsweise und das Verhalten der Systemkette unter verschiedenen Ausfallmodi. Diese Abhängigkeiten können innerhalb eines Anwendungscodes liegen, von der externen Verwaltung von Geheimnissen mithilfe von AWS Secrets Manager oder einer Tresorlösung eines Drittanbieters (wie Hashicorp) bis hin zu Authentifizierungssystemen, die für Verbundanmeldungen vom IAM Identity Center abhängig sind.

Redundanz in Bezug auf Abhängigkeiten kann zu einer erhöhten Resilienz beitragen. Es besteht auch die Möglichkeit, dass eine SaaS-Lösung oder eine Abhängigkeit von einem Drittanbieter dasselbe Primärsystem AWS-Region wie der Workload verwendet. In diesem Fall sollten Sie mit dem Anbieter zusammenarbeiten, um festzustellen, ob dessen Ausfallsicherheit den Anforderungen für den Workload entspricht.

Achten Sie außerdem darauf, dass der Workload und seine Abhängigkeiten, wie z. B. Anwendungen von Drittanbietern, gemeinsam genutzt werden. Wenn die Abhängigkeiten nach einem Failover in (oder von) einer sekundären Region nicht verfügbar sind, wird der Workload möglicherweise nicht vollständig wiederhergestellt.

3c: Failover-Mechanismus

Das Domain Name System (DNS) wird häufig als Failover-Mechanismus verwendet, um den Verkehr von der primären Region in eine Standby-Region zu verlagern. Prüfen und überprüfen Sie kritisch alle Abhängigkeiten, die der Failover-Mechanismus mit sich bringt. Wenn Ihr Workload beispielsweise Amazon Route 53 verwendet, bedeutet das Wissen, dass die Kontrollebene in US-East-1 gehostet wird, dass Sie von der Kontrollebene in dieser bestimmten Region abhängig sind. Dies wird als Teil eines Failover-Mechanismus nicht empfohlen, wenn die primäre Region ebenfalls US-East-1 ist. Wenn ein anderer Failover-Mechanismus verwendet wird, ist ein tiefes Verständnis jedes Szenarios

erforderlich, in dem dieser nicht wie erwartet funktionieren würde. Sobald dieses Verständnis hergestellt ist, sollten Sie für Notfälle planen oder, falls erforderlich, einen neuen Mechanismus entwickeln. Lesen Sie Creating Disaster Recovery-Mechanismen mithilfe von Amazon Route 53, um mehr über Methoden zu erfahren, die Sie für ein erfolgreiches Failover verwenden können.

Wie im Abschnitt zur internen Abhängigkeit beschrieben, müssen alle Microservices, die Teil einer Geschäftsfunktion sind, in jeder Region verfügbar sein, in der der Workload bereitgestellt wird. Im Rahmen der Failover-Strategie muss das Failover der Geschäftskapazitäten gemeinsam durchgeführt werden, um die Möglichkeit regionsübergreifender Anrufe auszuschließen. Wenn Microservices unabhängig voneinander ein Failover durchführen, kann dies zu unerwünschtem Verhalten führen, wenn Microservices möglicherweise regionsübergreifende Anrufe tätigen, was zu Latenz führt und dazu führen kann, dass der Workload im Falle von Client-Timeouts nicht verfügbar ist.

3d: Abhängigkeiten von der Konfiguration

Zertifikate, Schlüssel, Geheimnisse und Parameter sind Teil der Abhängigkeitsanalyse, die beim Design für mehrere Regionen erforderlich ist. Wann immer möglich, ist es am besten, diese Komponenten innerhalb der einzelnen Regionen zu lokalisieren, damit sie in Bezug auf diese Abhängigkeiten nicht von mehreren Regionen gemeinsam genutzt werden. Bei Zertifikaten sollte das Ablaufdatum je nach Region und wenn möglich in jeder Region unterschiedlich sein, um zu vermeiden, dass sich ein ablaufendes Zertifikat (mit Alarmen, die im Voraus benachrichtigen) auf mehrere Regionen auswirkt.

Verschlüsselungsschlüssel und -geheimnisse sollten ebenfalls regionsspezifisch sein. Auf diese Weise sind die Auswirkungen auf eine bestimmte Region beschränkt, wenn bei der Rotation eines Schlüssels oder Geheimnisses ein Fehler auftritt.

Schließlich sollten alle Workload-Parameter lokal gespeichert werden, damit der Workload in der jeweiligen Region abgerufen werden kann.

Wichtige Hinweise

- Eine Architektur mit mehreren Regionen profitiert von der physischen und logischen Trennung zwischen Regionen. Durch die Einführung regionsübergreifender Abhängigkeiten auf Anwendungsebene wird dieser Vorteil zunichte gemacht. Vermeiden Sie solche Abhängigkeiten.
- Die Failover-Steuerung sollte ohne Abhängigkeiten von der primären Region funktionieren.
- Der Failover muss an der Geschäftsstelle koordiniert werden, um die Möglichkeit einer erhöhten Latenz und Abhängigkeit regionsübergreifender Anrufe auszuschließen.

Grundlage 4 für mehrere Regionen: Einsatzbereitschaft

Der Betrieb eines Workloads mit mehreren Regionen ist eine komplexe Aufgabe, die mit betrieblichen Herausforderungen verbunden ist, die für mehrere Regionen spezifisch sind. Dazu gehören die AWS-Konto Verwaltung, die Anpassung der Bereitstellungsprozesse, die Entwicklung einer Beobachtungsstrategie für mehrere Regionen, die Erstellung und das Testen von Failover- und Failback-Runbooks und die anschließende Verwaltung der Kosten. Ein Operational Readiness Review (ORR) kann Teams dabei helfen, einen Workload für die Produktion vorzubereiten, unabhängig davon, ob er in einer einzelnen Region oder in mehreren Regionen ausgeführt wird.

4a: AWS-Konto Verwaltung

Um einen Workload flächenübergreifend bereitzustellenAWS-Regionen, stellen Sie sicher, dass alle AWSServicekontingenten innerhalb eines Kontos in allen Regionen paritätisch sind. Machen Sie sich zunächst mit allen AWS Diensten vertraut, die Teil der Architektur sind, schauen Sie sich die geplante Nutzung in den Standby-Regionen an und vergleichen Sie sie dann mit der aktuellen Nutzung. In einigen Fällen, wenn die Standby-Region noch nicht genutzt wurde, können Sie sich auf die Standard-Servicekontingente beziehen, um den Ausgangspunkt zu verstehen. Fordern Sie dann für alle Dienste, die verwendet werden, über die Service Quotas Quota-Konsole (Anmeldung erforderlich) oder über APIs eine Erhöhung des Kontingents an.

AWSIdentity and Access Management Zugriffsmanagement-Rollen (IAM) müssen in jeder Region konfiguriert werden, um sicherzustellen, dass Betreiber, Automatisierungstools und AWS Dienste über die entsprechenden Berechtigungen für Ressourcen in der Standby-Region verfügen. Durch die regionale Isolierung von Rollen wird die regionale Isolierung erreicht, die wir für Architekturen mit mehreren Regionen anstreben. Stellen Sie sicher, dass diese Berechtigungen vorhanden sind, bevor Sie eine Standby-Region live schalten.

4b: Bereitstellungspraktiken

Bei Kapazitäten für mehrere Regionen kann die Verteilung der Arbeitslast auf mehrere Regionen komplex sein. AWS CloudFormationhilft bei der Bereitstellung der Infrastruktur in einer oder mehreren Regionen und kann auf Ihre Bedürfnisse zugeschnitten werden. AWS CodePipelinehilft bei der Bereitstellung einer nahezu kontinuierlichen Integration/Continuous Delivery (CI/CD) - Pipeline mit regionenübergreifenden Aktionen, die den Einsatz in Regionen ermöglichen, die sich von der Region unterscheiden, in der sich die Pipeline befindet. In Kombination mit robusten

Implementierungsstrategien wie Blau/Grün ermöglicht dies eine Bereitstellung mit minimalen bis gar keinen Ausfallzeiten.

Die Bereitstellung von Stateful-Fähigkeiten kann jedoch komplexer sein, wenn der Status der Anwendung oder der Daten nicht in einen persistenten Speicher ausgelagert wird. Passen Sie in diesen Situationen den Bereitstellungsprozess sorgfältig an Ihre Bedürfnisse an. Entwerfen Sie die Bereitstellungspipeline und den Prozess so, dass die Bereitstellung jeweils in einer Region und nicht in mehreren Regionen gleichzeitig erfolgt. Dadurch wird die Wahrscheinlichkeit korrelierter Ausfälle zwischen den Regionen verringert. Um mehr über die Techniken zu erfahren, die Amazon zur Automatisierung von Softwarebereitstellungen verwendet, lesen Sie den Artikel Automatisieren sicherer Bereitstellungen in der Builder Library.

4c: Beobachtbarkeit

Bei der Planung für mehrere Regionen sollten Sie berücksichtigen, wie der Zustand aller Komponenten in jeder Region überwacht werden soll, um ein ganzheitliches Bild der regionalen Gesundheit zu erhalten. Dies könnte die Überwachung von Metriken im Hinblick auf Replikationsverzögerungen beinhalten, was bei einem Workload einer einzelnen Region nicht berücksichtigt wird.

Beim Aufbau einer Architektur mit mehreren Regionen sollten Sie erwägen, die Leistung des Workloads auch von den Standby-Regionen aus zu beobachten. Dazu gehören auch Integritätsprüfungen und Canaries (synthetische Tests), die von der Standby-Region aus durchgeführt werden, sodass ein externer Überblick über den Zustand der Primärregion möglich ist. Darüber hinaus können Sie Amazon CloudWatch Internet Monitor verwenden, um den Status des externen Netzwerks und die Leistung Ihrer Workloads aus der Sicht des Endbenutzers zu verstehen. In ähnlicher Weise sollte in der primären Region dieselbe Beobachtbarkeit für die Überwachung der Standby-Region vorhanden sein. Diese Kanarienvögel sollten die Kennzahlen zum Kundenerlebnis im Auge behalten, um sich einen Überblick über die Arbeitslast zu verschaffen. Dies ist erforderlich, da bei einem Problem in der primären Region die Beobachtbarkeit in der primären Region beeinträchtigt werden könnte, was sich auf die Fähigkeit auswirken würde, den Zustand der Arbeitslast zu beurteilen.

In diesem Fall kann eine Beobachtung außerhalb dieser Region Aufschluss geben. Diese Kennzahlen sollten in Dashboards zusammengefasst werden, die in jeder Region verfügbar sind, und in jeder Region sollten Alarme erstellt werden. Da CloudWatch es sich bei Amazon um einen regionalen Service handelt, ist es erforderlich, dass diese in beiden Regionen verfügbar sind. Diese

Überwachungsdaten werden verwendet, um den Anruf zum Failover von einer primären zu einer Standby-Region zu tätigen.

4d: Prozesse, Verfahren und Tests

Der beste Zeitpunkt, um die Frage „Wann sollte ich ein Failover durchführen?“ zu beantworten ist lange bevor Sie es müssen. Pläne zur Geschäftskontinuität, die Mitarbeiter, Prozesse und Technologien einbeziehen, sollten alle rechtzeitig vor einem Problem definiert und regelmäßig getestet werden. Entscheiden Sie sich für einen Rahmen für Rückforderungsentscheidungen. Wenn es einen gut geübten Wiederherstellungsprozess gibt und der Zeitpunkt bis zur Wiederherstellung genau bekannt ist, kann der Zeitpunkt gewählt werden, zu dem der Wiederherstellungsprozess gestartet wird, der das RTO-Ziel durch einen Failover erreicht. Dieser Zeitpunkt kann unmittelbar nach der Identifizierung eines Problems mit der Anwendung in der primären Region liegen, oder es könnte sich um einen weiteren Zeitpunkt handeln, bei dem die Wiederherstellungsoptionen innerhalb der Anwendung in der Region ausgeschöpft sind und nun ein Failover gestartet werden sollte, um die RTO-Anforderungen zu erfüllen.

Die Failover-Aktion selbst sollte zu 100% automatisiert sein, die Entscheidung, den Failover zu aktivieren, sollte jedoch von einem Menschen getroffen werden (in der Regel eine kleine Anzahl vorab festgelegter Personen in der Organisation). Außerdem müssen die Kriterien für die Entscheidung über einen Failover klar definiert und gemeinsam mit der Organisation geklärt werden. Diese Prozesse können mithilfe von AWSSystem Manager-Runbooks definiert und abgeschlossen werden, was eine vollständige end-to-end Automatisierung ermöglicht und die Konsistenz des Prozessablaufs während der Tests und des Failovers gewährleistet.

Diese Runbooks sollten in der Primär- und Standby-Region verfügbar sein, damit die Failover- oder Failback-Prozesse gestartet werden können. Sobald diese Automatisierung eingeführt ist, sollte ein regelmäßiger Testrhythmus festgelegt und eingehalten werden. Auf diese Weise wird sichergestellt, dass bei einem tatsächlichen Ereignis die Reaktion auf einem klar definierten, eingeübten Prozess erfolgt, in den das Unternehmen Vertrauen hat. Es ist auch wichtig, die festgelegten Toleranzen für Datenabgleichsprozesse zu berücksichtigen. Vergewissern Sie sich, dass die festgelegten RPO/RTO-Anforderungen mit dem vorgeschlagenen Prozess erfüllt werden.

4e: Kosten und Komplexität

Die Auswirkungen einer Architektur mit mehreren Regionen auf die Kosten sind auf eine höhere Infrastrukturnutzung, höhere Betriebskosten und mehr Ressourcenaufwand zurückzuführen. Wie

bereits erwähnt, entsprechen die Infrastrukturkosten in einer Standby-Region den Infrastrukturkosten in einer Primärregion bei der Vorbereitstellung, sodass die Kosten doppelt so hoch sind. Stellen Sie Kapazität so bereit, dass sie für den täglichen Betrieb ausreichend ist, reservieren Sie aber dennoch genügend Pufferkapazität, um Nachfragespitzen zu verkraften — und konfigurieren Sie in jeder Region dieselben Grenzwerte.

Darüber hinaus können Änderungen auf Anwendungsebene erforderlich sein, um in einer Architektur mit mehreren Regionen erfolgreich ausgeführt zu werden, wenn Sie eine Active-Active-Architektur verwenden, deren Entwicklung und Betrieb zeit- und ressourcenintensiv sein können. Unternehmen müssten zumindest Zeit damit verbringen, die technischen und geschäftlichen Abhängigkeiten in den einzelnen Regionen zu verstehen und Failover- und Failback-Prozesse zu entwerfen.

Die Teams sollten außerdem normale Failover- und Failback-Übungen durchführen, um sich mit Runbooks, die während einer Veranstaltung verwendet werden, vertraut zu machen. Obwohl diese Maßnahmen unglaublich wichtig und entscheidend sind, um das erwartete Ergebnis einer Investition in mehrere Regionen zu erzielen, sind sie mit Opportunitätskosten verbunden und nehmen Zeit und Ressourcen für andere Aktivitäten in Anspruch.

Wichtige Leitlinien

- AWSDie Servicequoten müssen überprüft werden, und zwar in allen Regionen, in denen die Arbeitslast eingesetzt werden soll, und zwar einheitlich.
- Der Bereitstellungsprozess sollte jeweils auf eine Region abzielen und nicht auf mehrere Regionen gleichzeitig.
- Zusätzliche Messwerte wie die Verzögerung bei der Replikation müssen überwacht werden und sind spezifisch für Szenarien mit mehreren Regionen.
- Erweitern Sie die Überwachung der Arbeitslast über die primäre Region hinaus. Die Kennzahlen zur Kundenzufriedenheit sollten für jede Region überwacht und außerhalb jeder Region, in der ein Workload ausgeführt wird, gemessen werden.
- Failover und Failback müssen regelmäßig getestet werden. Stellen Sie sicher, dass ein einziges Runbook für Failover- und Failback-Prozesse implementiert wird, das sowohl bei Tests als auch bei Live-Events verwendet wird. Runbooks für Tests und Live-Events können nicht unterschiedlich sein.

Schlussfolgerung

In diesem Whitepaper wurden allgemeine Anwendungsfälle für mehrere Regionen, Grundlagen zur Implementierung einer multiregionalen Architektur und die Auswirkungen dieses Ansatzes erörtert. Diese Grundlagen können auf jeden Workload angewendet werden und als Rahmen für die Entscheidungsfindung darüber dienen, ob eine multiregionale Architektur der richtige Ansatz für ein bestimmtes Unternehmen ist oder nicht.

Mitwirkende

Zu den Mitwirkenden an diesem Dokument gehören:

Technischer Mitwirkender:

- John Formento, Jr., leitender Lösungsarchitekt, regionsübergreifendes Team AWS

Redaktioneller Mitwirkender:

- Lisi Lewis, leitende Managerin, Produktmarketing

Weitere Informationen

Weitere Informationen finden Sie unter:

- Fortgeschrittene Multi-AZ-Resilienzmuster (AWSWhitepaper)
- Säule der Zuverlässigkeit — AWS Well-Architected Framework
- Verfügbarkeit und mehr: Die Widerstandsfähigkeit verteilter Systeme verstehen und verbessern AWS (AWSWhitepaper)
- AWSGrenzen der Fehlerisolierung (AWSWhitepaper)

Dokumentversionen

Abonnieren Sie den RSS-Feed, um über Aktualisierungen dieses Whitepapers informiert zu werden.

Änderung	Beschreibung	Datum
Dokument veröffentlicht	Erste Veröffentlichung.	20. Dezember 2022

Hinweise

Kunden sind dafür verantwortlich, Ihre eigene unabhängige Bewertung der Informationen in diesem Dokument vorzunehmen. Dieses Dokument: (a) dient nur zu Informationszwecken, (b) stellt die aktuellen AWS-Produktangebote und -praktiken dar, die ohne Vorankündigung geändert werden können, und (c) begründet keine Verpflichtungen oder Zusicherungen seitens AWS und der mit ihr verbundenen Unternehmen, Lieferanten oder Lizenzgeber. AWS-Produkte oder -Services werden ohne Mängelgewähr bereitgestellt, ohne ausdrückliche oder stillschweigende Garantien, Zusicherungen oder Bedingungen jeglicher Art. 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.

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

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.