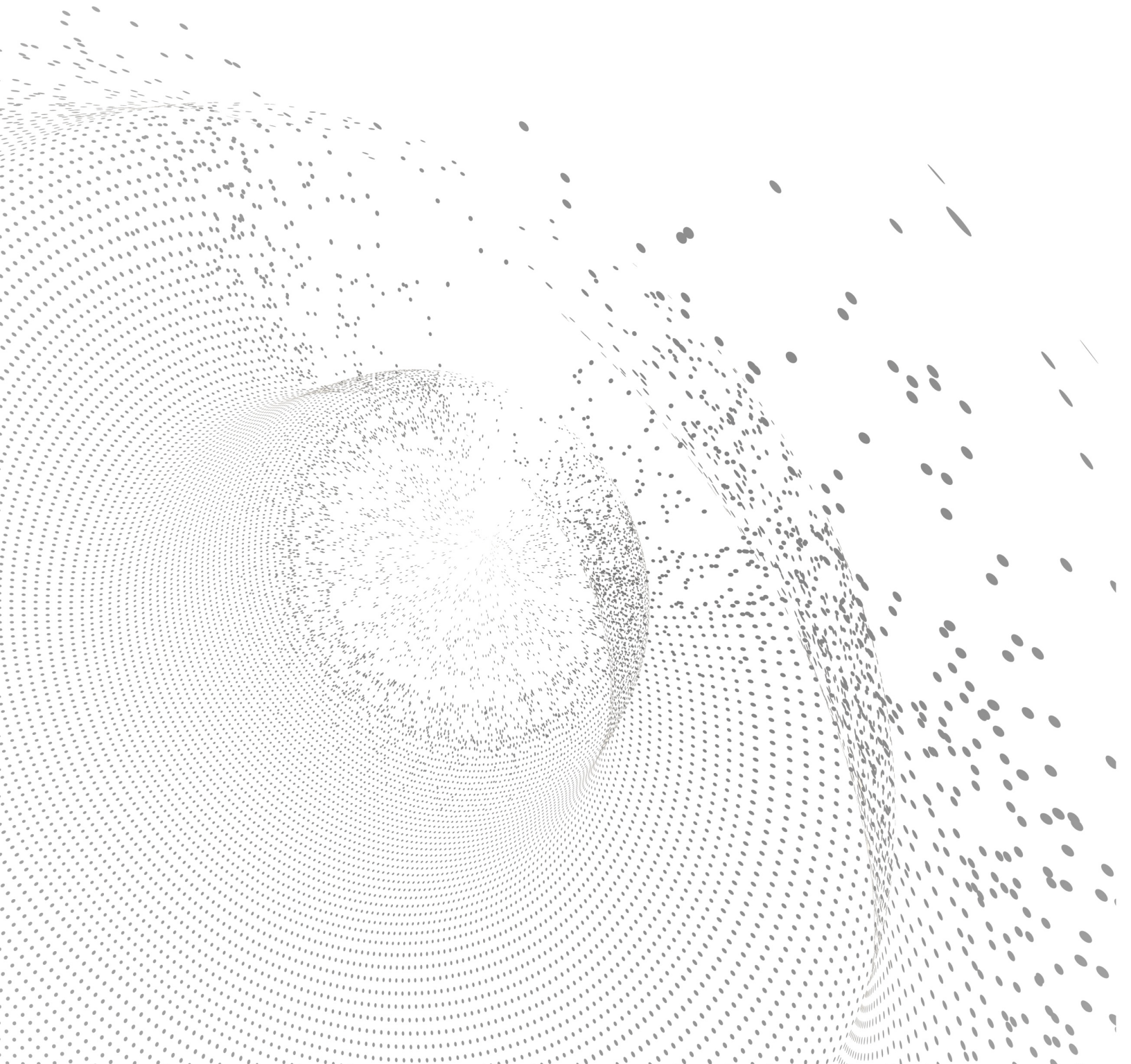
AEROSPIKE

Vorstellung der Architektur von Aerospike



Inhalt

Überblick

Der intensive Wettbewerb in der globalen Marktwirtschaft zwingt Unternehmen im Bank- und Finanzwesen, im Einzelhandel, in der Telekommunikation und in anderen Branchen dazu, hochgradig personalisierte Produkte und Dienstleistungen in Echtzeit und rund um die Uhr anzubieten. Das stellt enorme Anforderungen an die Datenmanagement-Infrastrukturen. Aus diesem Grund haben sich in den letzten zehn Jahren viele Unternehmen für die operative NoSQL-Datenbankplattform von Aerospike entschieden. Aerospike nutzt neueste Hardware-Innovationen für eine kosteneffiziente, hochskalierbare Transaktionsverarbeitung. Mit Aerospike können Anwender ihren Server-Footprint um bis zu 90 % reduzieren, die Betriebskosten senken und gleichzeitig aggressive Service Level Agreements (SLAs) einhalten.

Branchenführer wie Airtel, Banca D'Italia, Nielsen, PayPal, Snap, Verizon Media und Wayfair nutzen Aerospike für geschäftskritische Anwendungen wie Empfehlungsengines, personalisierte Customer Engagement-Services, Betrugserkennung, Zahlungsverarbeitung, maschinelles Lernen und vieles mehr. Aerospike muss dafür auf Geräten am Netzwerkrand, im Rechenzentrum und sogar über mehrere Rechenzentren oder Cloud-Regionen hinweg operieren. Solche Implementierungen benötigen einen vorhersehbaren, ultraschnellen und skalierbaren Datenzugriff von einer Datenbankplattform, die jederzeit betriebsbereit, verfügbar und konsistent ist.

In diesem Dokument stellen wir die Architektur von Aerospike vor. Wir erläutern, wie sie selbstreparierende Cluster bereitstellt, die Upgrades und Wartung ohne Ausfallzeiten unterstützen und gleichzeitig große Transaktionslasten mit extrem niedrigen Latenzzeiten und äußerster Konsistenz verarbeiten. Dies senkt die Betriebskosten und hilft Ihrem Unternehmen zu wachsen. Sehen wir uns die Technologie an, die all dies möglich macht.

Designziele

Aerospike hat eine verteilte NoSQL-Plattform entwickelt, um das zu erreichen, was andere Systeme nicht leisten können: einfache Skalierung, außergewöhnliche – und vorhersehbare – Leistung, nahezu 100-prozentige Verfügbarkeit, hohe Datenkonsistenz, Integration in bestehende IT-Infrastrukturen und konkurrenzlos niedrige Gesamtbetriebskosten (TCO). Das einzigartige Design von Aerospike, das sowohl lokal als auch in der Cloud eingesetzt werden kann, nutzt die neuesten Prozessor-, Speicher- und Netzwerktechnologien, um die anspruchsvollsten Geschäftsanforderungen zu erfüllen. Wie das funktioniert, erfahren Sie gleich. Zunächst aber betrachten wir einige bis dato übliche Optionen.

Mainframe-basierte Lösungen waren einst die einzig brauchbare Option für eine schnelle, zuverlässige und sichere Transaktionsverarbeitung. Leider erwiesen sie sich als teuer in der Wartung und schwierig in der Anpassung an neue Geschäftsanforderungen. Außerdem gab es nur selten freie Kapazitäten für neue Anwendungen. Einige Unternehmen wählten zweistufige Architekturen mit einer Caching-Schicht vor einem verteilten DBMS. Solche Infrastrukturen litten häufig unter unvorhersehbaren Latenzen beim Datenzugriff, benötigten eine große Menge an Servern und verursachten hohe Betriebskosten. Auch Probleme mit der Datenverfügbarkeit und -konsistenz machten diesen Architekturen zu schaffen. Infolgedessen entschieden sich eine Reihe von Unternehmen, die auf Mainframe- oder zweistufige Lösungen setzten, für Aerospike, um die Transaktionsverarbeitung zu optimieren. Einige ersetzten ihre älteren Infrastrukturen gleich ganz durch Aerospike.

Von Aerospike profitieren vor allem Anwendungen mit diesen Anforderungen:

- SLAs, die Datenbank-Reaktionszeiten von weniger als einer Millisekunde erfordern.
- Hoher Durchsatz für gemischte Workloads (z. B. 3 bis 5 Millionen Operationen pro Sekunde).

- Verwaltung von Hunderten von Milliarden von Datensätzen in Datenbanken, die mehrere 100 Terabyte bis hin zu mehreren Petabyte groß sind.
- Hohe Verfügbarkeit und Fehlertoleranz für unternehmenskritische Anwendungen.
- Hohe Skalierbarkeit zur Bewältigung unvorhersehbarer Spitzen bei Datenvolumen und Transaktionen.
- Anpassungsfähige Infrastruktur für die Verwaltung unterschiedlicher Datentypen mit minimalem Aufwand.
- Starke, sofortige Datenkonsistenz.
- Unterstützung für globale Transaktionen, die sich über mehrere Rechenzentren oder Cloud-Regionen erstrecken.
- Niedrige TCO.

Heute nutzen Firmen auf der ganzen Welt Aerospike als System of Engagement oder System of Record. In einigen Unternehmen hat Aerospike beide Funktionen übernommen. Aerospike kann sowohl am Rand als auch im Kern des Netzwerks eingesetzt werden, was es von anderen NoSQL-Systemen unterscheidet. Active/Active-Konfigurationen, die mehrere Rechenzentren oder Cloud-Regionen umfassen, unterstützen globale Verarbeitungsanforderungen. Abb. 1 zeigt verschiedene Möglichkeiten, wie Aerospike zur Unterstützung von operativen und analytischen Echtzeitanwendungen eingesetzt werden kann, indem es über Apache Spark, Apache Kafka und andere Konnektoren in die IT-Infrastrukturen von Unternehmen integriert wird.

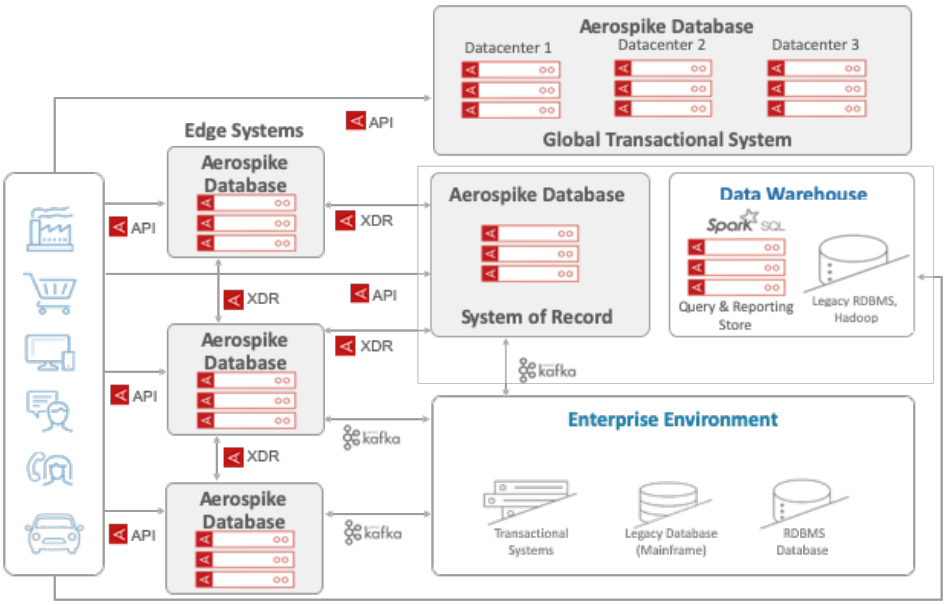


ABBILDUNG 1: Einsatzbeispiele für Aerospike

Die Architektur im Überblick

Aerospike ist eine verteilte Shared-Nothing-DBMS-Plattform, die für große operative Workloads mit einer Mischung aus Lese- und Schreiboperationen optimiert ist. Das schemafreie Key-Value-Datenmodell bietet beträchtliche Flexibilität, insbesondere im Vergleich zu relationalen DBMS, die im Vorfeld Schemadefinitionen erfordern. Aerospike ist in C geschrieben, leidet also nicht unter den Ineffizienzen der Java-Laufzeit, und nutzt moderne Hardware, sodass Unternehmen nicht laufend neue Server bereitstellen müssen, um wachsende Workloads oder Lastspitzen zu bewältigen. Die Aerospike Community Edition ist kostenlos. Sie

hat die gleiche Architektur wie die Enterprise Edition, die unbegrenzte Skalierbarkeit, standortübergreifenden Betrieb, zusätzliche Sicherheits- und Speicheroptionen und noch einiges mehr bietet. In diesem White Paper liegt der Fokus auf der Enterprise Edition.

Client- und Serverschichten

Die intelligente Clientschicht von Aerospike unterstützt gängige Programmiersprachen (C/C++, Java, PHP, Python und verschiedene andere) und eine Abfrageschnittstelle. Die Serverschicht (siehe Abb. 2) umfasst eine Echtzeit-Transaktionsengine, eine Datenverteilungskomponente, ein intelligentes Subsystem für die Clusterverwaltung, dynamische Datenumverteilung und eine leistungsstarke Speicherengine. Diese Komponenten bilden gemeinsam einen Aerospike-Server, der die Daten automatisch auf mehrere Rechenknoten verteilt und repliziert, um Verfügbarkeit und Fehlertoleranz zu gewährleisten.

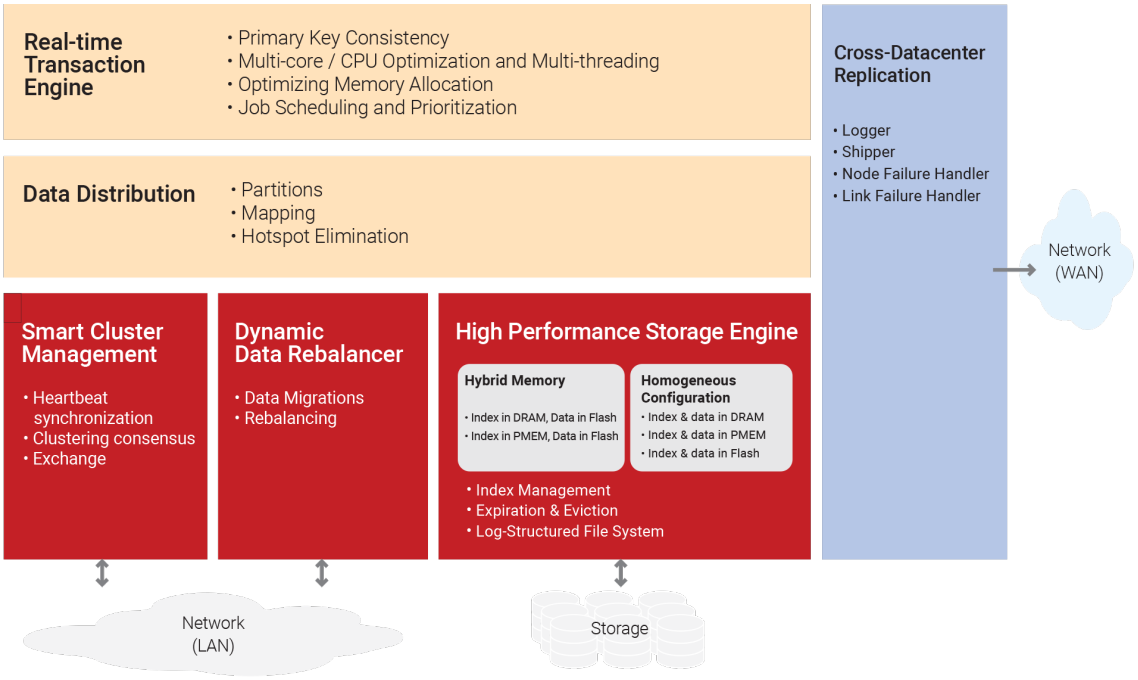


Abbildung 2: Architektur der Aerospike-Serverschicht

Viele Kunden nutzen Aerospike nur in einem einzigen Rechenzentrum bzw. einer einzigen Cloudregion. Unternehmen können einen Server für die globale Transaktionsverarbeitung aber auch über mehrere Standorte hinweg „elastisch“ einsetzen. Auf Multi-Site-Cluster werden wir später noch eingehen, aber im Wesentlichen ermöglichen sie Active/Active-Operationen über mehrere geografische Standorte hinweg mit synchroner Datenreplikation und starker, sofortiger Datenkonsistenz. Zurück zu Abb. 2: Die rechenzentrumsübergreifende Replikation (rechts) ermöglicht es Unternehmen, Daten asynchron zwischen mehreren Aerospike-Systemen zu replizieren.

Datenmodell

Aerospike-Anwender modellieren Daten in Datensätzen, die in Namespaces enthalten sind. (Im Vergleich dazu modellieren die Anwender relationaler DBMS Daten in Tabellen, die in Datenbanken enthalten sind.) Jeder in Aerospike gespeicherte Datensatz hat einen Schlüssel und benannte Felder („Bins“). Die Werte in den Bins sind stark typisiert, die Bins selbst jedoch nicht, sodass ein einzelner Bin verschiedene Datentypen enthalten kann. Darüber hinaus können Datensätze in ein und demselben Satz unterschiedliche Bins haben, was eine beträchtliche schematische Flexibilität ermöglicht. Fehlt in einem Datensatz ein bestimmtes Feld, so

wird dafür kein Bin geführt, was die effiziente Speicherung spärlich befüllter Datensätze ermöglicht. Aerospike unterstützt gängige skalare, komplexe und spezielle Datentypen: Ganzzahlen, Doubles, Strings, Byte-Arrays, BLOBs, Karten, Listen, Geodaten (GeoJSON) und probabilistische Daten (HyperLogLog). Neben der Suche nach Primärschlüsseln unterstützt Aerospike eine Vielzahl von Datenabruf- und -bearbeitungsoperationen, einschließlich paralleler Scans, sekundärer Indizierung, geospatialer Indizierung und Filterung sowie benutzerdefinierter Funktionen.

Benutzer, die von relationalen DBMS zu Aerospike migrieren, denormalisieren in der Regel ihre Daten, um kostspielige Join-Operationen zu vermeiden und schnelle Key-Lookups zu maximieren. Signifikante Leistungsverbesserungen, höhere Speichereffizienz, größere Flexibilität bei der Datenmodellierung und eine vergleichbar hohe Datenkonsistenz gehören zu den Vorteilen, von denen Anwender nach der Migration auf Aerospike berichten.

Betrachten wir die Architektur von Aerospike im Detail, beginnend mit einem der wichtigsten Merkmale: die Art und Weise, wie es Arbeits- und Massenspeicher verwaltet.

Speicherverwaltung

Um auch bei großen Datenmengen einen hohen Transaktionsdurchsatz und niedrige Datenzugriffslatenzen zu erreichen, nutzt Aerospike die neuesten Technologien in der Speichertechnik, darunter NVMe-Flash- oder Solid-State-Laufwerke (SSDs) und PMEM-Speicher (Non-Volatile Memory Extended). Darüber hinaus macht Aerospike effizienten Gebrauch von Dynamic-RAM (DRAM) und unterstützt auch herkömmliche Festplatten. Abb. 3 zeigt einige Beispielkonfigurationen. Weitere Konfigurationen sind möglich, einschließlich hybrider Architekturen, die DRAM oder PMEM mit Flash kombinieren.

In den nächsten Absätzen beleuchten wir kurz, wie Aerospike die Vorteile von Flash, PMEM und anderen Technologien nutzt. Weitere Einzelheiten finden Sie in der Dokumentation von Aerospike.

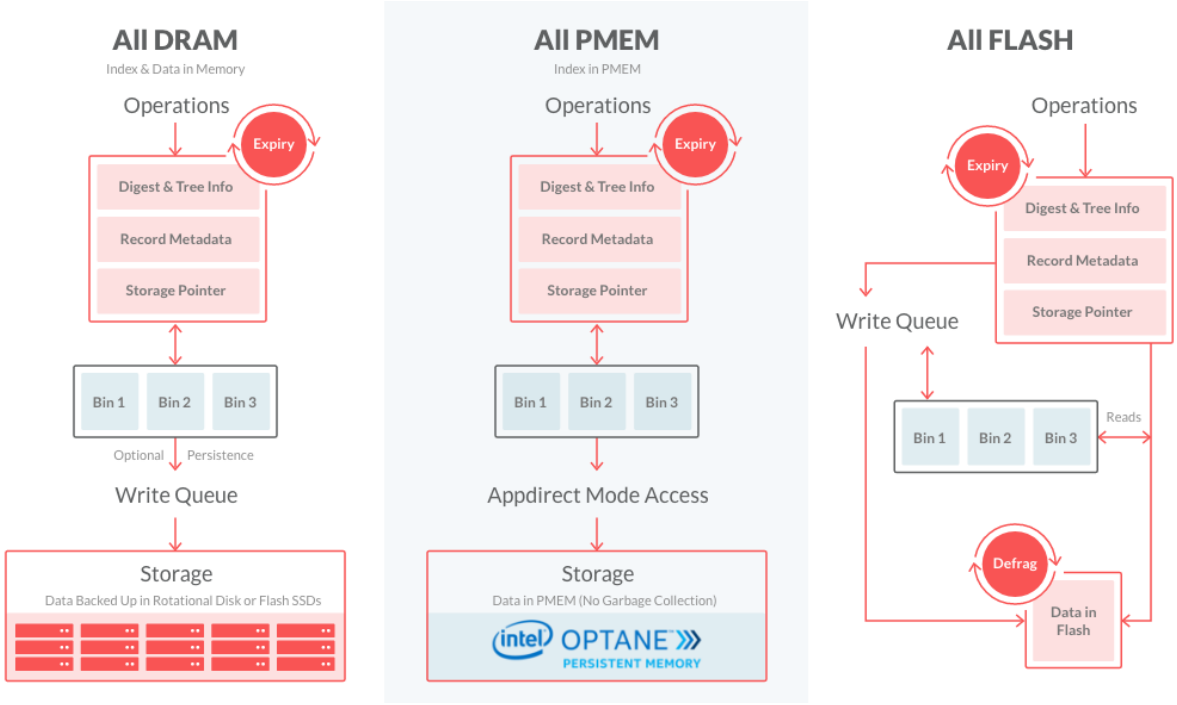


ABBILDUNG 3: Beispiele für homogene Speicherkonfigurationen

Homogene Konfigurationen

Aerospike kann als reines In-Memory-System (ohne Persistenz) oder als In-Memory-System konfiguriert werden, das alle Indizes und Benutzerdaten im DRAM verwaltet und die Daten nur als Backup auf der Festplatte speichert. Selbst bei der letztgenannten Konfiguration erfolgen alle Lesevorgänge aus dem DRAM. Schreibvorgänge werden auf einen In-Memory-Puffer angewendet und in Stapeln von 128-KB- oder 1-MB-Blöcken auf Flashspeicher oder Festplatten übertragen. Optional können Anwendungen jeden Schreibvorgang direkt im persistenten Speicher committen.

Aerospike ermöglicht es Anwendern auch, alle Index- und Benutzerdaten in PMEM zu verwalten. Aerospike war das erste kommerzielle offene DBMS, das Intel® Optane™ DC PMEM unterstützte und native Index- und Datenunterstützung nach dem Vorbild seiner früheren NAND-Optimierungen bot. Im Hinblick auf die Leistung, übertrifft Aerospike In-Memory-DBMS, die PMEM einfach auf bestehende Datenstrukturen abbilden, wie in diesem Blog dargelegt wird. Eine reine PMEM-Aerospike-Konfiguration bietet Skalierbarkeit, Persistenz, hohe Leistung (vergleichbar mit DRAM für Lesevorgänge) und schnelle Neustarts. Damit ist sie ideal für Unternehmen, die maschinelles Lernen und andere anspruchsvolle Anwendungen mit großen, PMEM-tauglichen Datenmengen beschleunigen wollen.

Anwender mit sehr großen Datensätzen können auf Wunsch alle Index- und Anwenderdaten auf Flash speichern. Eine solche Konfiguration eignet sich besonders für Unternehmen, deren Datenbanken so groß sind, dass sie von aktuellen Prozessoren nicht mit PMEM verarbeitet werden können. Die Speicherung von Indizes in Flash anstelle von DRAM ermöglicht erhebliche Kosteneinsparungen bei gleichzeitig hoher Leistung.

Hybride Speicherkonfigurationen

Viele Unternehmen konfigurieren Aerospike so, dass die Daten in Flash gespeichert werden und die Indizes in DRAM oder PMEM verbleiben. Dieser hybride Ansatz bietet erhebliche Preis-/Leistungsvorteile gegenüber anderen NoSQL-Lösungen.

Die Reduzierung von E/A-Vorgängen ist ein entscheidender Vorteil der hybriden Architektur von Aerospike. Beim Indexzugriff findet auf der Festplatte überhaupt kein E/A statt. Befindet sich der Zieldatensatz im lokalen Cache, wird er ohne E/A zurückgegeben. Andernfalls führt Aerospike eine einzige E/A-Operation aus. Die E/A-Leselatenz in NAND-Flash hat nur geringe Auswirkungen auf den wahlfreien Zugriff, sodass Aerospike eine schnelle, vorhersehbare Leistung bieten kann.

Zum Wear-Leveling schreibt Aerospike die Daten in großen Blöcken, was den internen Koaleszenzalgorithmus des Flash-Controllers entlastet. Zur weiteren Effizienzsteigerung behandelt Aerospike den Flashspeicher als Blockgerät und verwendet ein benutzerdefiniertes Datenlayout. Zugriffe auf den Flashspeicher erfolgen massiv parallel. Spezielle Hashing-Verfahren ordnen die Schlüssel gleichmäßig den Speichergeräten innerhalb der Knoten zu. Auf die Datenverteilung von Aerospike kommen wir später noch einmal zu sprechen.

Wird DRAM für Indizes verwendet, vermeidet Aerospike, wann immer möglich, den Neuaufbau von Indizes. Primärindizes speichert es in einem gemeinsamen Speicherbereich, der vom Speicherbereich des Serviceprozesses getrennt ist. Bei der routinemäßigen Wartung kann Aerospike so Datenscans zum Neuaufbau von Primärindizes vermeiden, da ein Aerospike-Prozess einfach an die aktuelle Kopie des Indexes im gemeinsamen Speicher anknüpfen und Transaktionen bedienen kann.

Eine hybride PMEM-/Flashkonfiguration bietet beträchtliche Skalierungsvorteile und ermöglicht es Aerospike-Knoten, jeweils 100 TB zu überschreiten, da die Indizes nicht mehr durch übliche DRAM-Größen beschränkt sind. Die Indexspeicherung im PMEM verkürzt die Kaltstartzeit, da die Indizes beim Neustart der Maschine nicht neu erstellt werden müssen.

Defragmentierung

Aerospike nutzt ein angepasstes Dateisystem mit einem Copy-on-Write-Mechanismus und führt in regelmäßigen Abständen einen Hintergrundprozess aus, um Speicherplatz zurückzugewinnen. Jedes Gerät verwaltet eine Map der Blöcke und überwacht die Speicherbelegung durch gültige Datensätze (Füllfaktor). Wenn der Füllfaktor eines Blocks unter einen konfigurierbaren Schwellenwert fällt, verschiebt Aerospike gültige Datensätze in einen Puffer, der auf Festplatte geschrieben wird, wenn er voll ist. Um zu vermeiden, dass neue und alte Schreibvorgänge vermischt werden, verwendet Aerospike zwei Schreibpuffer-Warteschlangen: eine für Client-Schreibvorgänge und eine für die Defragmentierung.

Datenverteilung

Aerospike verwendet ein ausgeklügeltes Hashing- und Partitionierungsverfahren, um Daten auf die Knoten zu verteilen. Insbesondere wird der Primärschlüssel eines Datensatzes mithilfe des RIPEMD-Algorithmus, der extrem kollisionssicher ist, in einen 160-Bit-Digest gehasht. Dieser Digest-Bereich wird in 4.096 nicht überlappende Partitionen gesplittet, die Aerospike automatisch über den Cluster verteilt. Eine Partition ist die kleinste Datenbesitzeinheit in Aerospike.

Aerospike verwendet ein ausgeklügeltes Hashing- und Partitionierungsverfahren, um Daten auf die Knoten zu verteilen. Insbesondere wird der Primärschlüssel eines Datensatzes mithilfe des RIPEMD-Algorithmus, der extrem kollisionssicher ist, in einen 160-Bit-Digest gehasht. Dieser Digest-Bereich wird in 4.096 nicht überlappende Partitionen gesplittet, die Aerospike automatisch über den Cluster verteilt. Eine Partition ist die kleinste Datenbesitzeinheit in Aerospike.

Aerospike ordnet die Datensätze den Partitionen auf der Grundlage des Primärschlüssel-Digests zu, wie in Abb. 4 dargestellt. Selbst wenn die Schlüsselverteilung im Schlüsselbereich ungleichmäßig ist, ist sie im Digestbereich und damit im Partitionsbereich gleichmäßig. Dieses einzigartige Datenpartitionierungsschema hilft Aerospike, Hotspots zu vermeiden, und fördert die Skalierbarkeit und Fehlertoleranz.

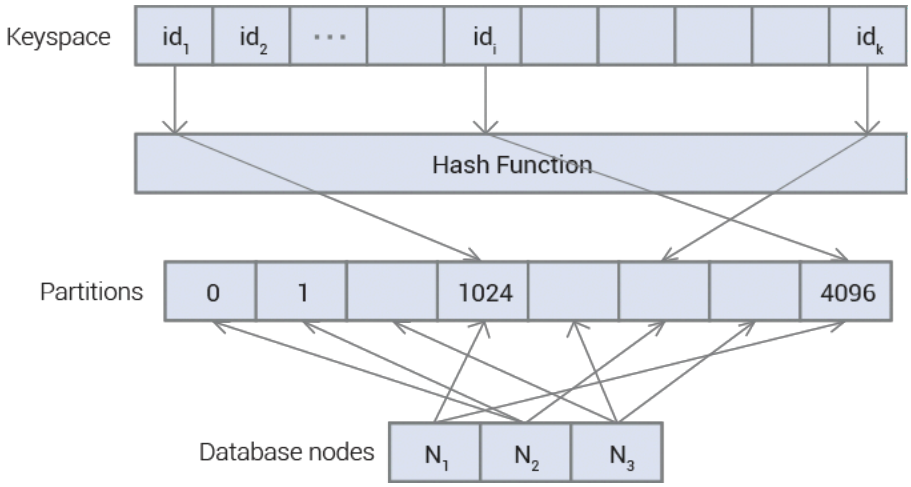


ABBILDUNG 4: Partitionierungsschema

Auch andere Aspekte des Datenverteilungsschemas von Aerospike sind erwähnenswert. Um knotenübergreifenden Datenverkehr bei Lesevorgängen zu vermeiden und eine einheitliche Datenverteilung zu fördern, partitioniert Aerospike die Indizes auf dieselbe Weise wie die Daten selbst und gewährleistet so Co-Location. Schreibvorgänge kommen für Index-Updates ohne Interknoten-Kommunikation aus, lediglich für die Datenreplikation fällt Interknoten-Aktivität an.

Der Ansatz von Aerospike bietet mehrere Vorteile: einheitliche Workloads über alle Knoten hinweg, vorhersehbare Lese- und Schreibperformance, Cluster-Elastizität und eine effiziente, unterbrechungsfreie Umverteilung der Daten. Da der intelligente Aerospike-Client das Schema der Datenpartitionierung kennt, sendet er außerdem Datenzugriffsanfragen an den entsprechenden Knoten und minimiert so kostspielige Netzwerk-Hops.

Clusterverwaltung

Das Subsystem für die Clusterverwaltung verwaltet die Knotenmitgliedschaft und stellt sicher, dass alle Knoten mit der aktuellen Mitgliedschaft übereinstimmen, die sich aufgrund von System-Upgrades, Knotenausfällen, Netzwerkunterbrechungen, Erweiterung (oder Entfernung) von Knoten usw. ändern kann. Da Änderungen am Cluster erhebliche Auswirkungen auf die Latenzzeit beim Datenzugriff haben können, ergreift Aerospike schnell entsprechende Maßnahmen.

Heartbeats und Knotenintegrität

Aerospike erkennt das Hinzufügen oder Entfernen von Knoten über Heartbeat-Meldungen, die zwischen den Knoten ausgetauscht werden, wobei jeder Knoten eine Liste seiner benachbarten Knoten führt, die kürzlich Heartbeats gesendet haben. Aus dem Cluster ausgeschiedene Knoten werden durch das Fehlen von

Heartbeats für ein konfigurierbares Timeout-Intervall erkannt und aus der Adjazenzliste entfernt. Da ein überlastetes Netzwerk zu zufälligen Paketverlusten führen kann, erlaubt Aerospike, dass andere regelmäßig ausgetauschte Meldungen (wie z. B. Replikatschreibvorgänge) als Ersatz für Heartbeats dienen. Ein instabiles Netzwerk kann sich so weniger stark auf die Clusteransicht auswirken, was wiederum unnötige Arbeit vermeidet.

Darüber hinaus bewertet jeder Knoten die Integrität seiner Nachbarknoten, indem er regelmäßig den gewichteten gleitenden Durchschnitt der erwarteten, pro Knoten empfangenen Meldungen berechnet und mit der Anzahl tatsächlich empfangener Meldungen vergleicht. Ein Knoten wird als „fehlerhaft“ eingestuft, wenn sein durchschnittlicher Meldungsverlust das Doppelte der Standardabweichung aller Knoten übersteigt. Wenn ein solcher Knoten dem Cluster angehört, wird er entfernt. Ist er noch kein Mitglied, darf er nicht beitreten, bis sein durchschnittlicher Meldungsverlust innerhalb akzeptabler Grenzen liegt.

Datenmigration und Datenumverteilung

Eine Änderung der Clusterzugehörigkeit bedeutet, dass Daten zwischen den Knoten migriert (verschoben) werden müssen, um das Gleichgewicht im Cluster wiederherzustellen und sicherzustellen, dass in jeder Master- und Replikatkopie im aktuellen Cluster die neueste Datenversion verfügbar ist. Sobald ein Konsens über eine neue Clusteransicht erreicht ist, führen alle Knoten einen deterministischen Algorithmus aus, der die Master- und Replikatknoten jeder Datenpartition zuweist.

Um die Auswirkungen von Datenmigrationen auf den Cluster zu minimieren, versioniert Aerospike die Partitionen und migriert nur die Datenabweichungen zwischen den Versionen. Ist eine Partition die Teilmenge einer Kopie auf einem anderen Knoten, vermeidet Aerospike die Migration ganz. Die Knoten mit den wenigsten Datensätzen in ihren Partitionsversionen werden zuerst migriert, sodass der Vorgang schnell abgeschlossen werden kann.

Damit nicht zu schnell auf Knotenzugänge und -abgänge reagiert wird, fasst Aerospike benachbarte Knotenereignisse zusammen und reagiert auf diese einmal pro Quantenintervall (das normalerweise auf einen etwas höheren Wert als das Heartbeat-Timeout-Intervall eingestellt ist). Aerospike entfernt oder fügt Knoten also in einem Schritt hinzu – ein klarer Vorteil bei der Clusterskalierung. Außerdem bietet Aerospike verschiedene Konfigurationsoptionen, die Administratoren in Echtzeit anpassen können, um Migrationen je nach aktueller Workload des Clusters zu beschleunigen oder zu verzögern.

Aerospike nutzt einen Uniform-Balancing-Algorithmus zur gleichmäßigen Datenverteilung, der die zufällige Zuweisung von Schlüsseln zu Partitionen und von Partitionen zu Knoten zusammenfasst und so sicherstellt, dass eine Umverteilung erst dann erforderlich ist, wenn sich die Clustermitgliedschaft ändert. Auf Schlüsselbereichen basierende Partitionierungsverfahren können ohne ständige Umverteilung keine gleichmäßige Datenverteilung auf die Knoten erzwingen, da verschiedene Schlüsselbereiche eine ungleichmäßige Verteilung von Datensätzen aufweisen können, die sich zudem ständig ändert, wenn Datensätze zur Datenbank hinzugefügt werden. Um eine gleichmäßige Datenverteilung über die Knoten hinweg zu erzwingen, werden die Schlüsselbereiche den Knoten häufig neu zugewiesen.

Der Umverteilungsmechanismus von Aerospike ist transparent und hat keine Auswirkungen auf das Clusterverhalten, was den Betrieb vereinfacht und zu einer vorhersehbaren Leistung beiträgt. Der Umverteilungsmechanismus bleibt zudem – selbst bei Ausfall eines Knotens während der Umverteilung – extrem stabil. Die Cluster können sich so selbst bei hoher Transaktionslast selbst reparieren. Ein Anwendereingriff ist nicht erforderlich.

Transaktionsengine

Die Echtzeit-Transaktionsengine von Aerospike liest und schreibt Daten auf Anforderung und bietet garantierte Datenkonsistenz und -isolierung. Um einen hohen Durchsatz bei geringer Latenz zu erreichen,

skaliert Aerospike vertikal *und* horizontal. Die Datenbankplattform nutzt die verfügbare Hardware voll aus, um die Effizienz zu maximieren, den Betrieb zu vereinfachen, die Gesamtbetriebskosten zu senken und gleichzeitig eine außergewöhnlich hohe Leistung und Verfügbarkeit zu bieten. Dabei ist Aerospike für Multi-Core-CPUs optimiert, lässt sich mit den neuesten Netzwerktechnologien integrieren und löst die Speicherfragmentierung auf einzigartige Weise.

Multi-Core-System

Standardprozessoren haben eine Multi-Core- und Multi-Socket-Architektur mit bis zu 64 Kernen. Ihre Caches verfügen über Non-Uniform Memory Access (NUMA) und ermöglichen eine Skalierung der Systemspeicherbandbreite mit der Anzahl der physischen Prozessoren. Das Latenz- und Durchsatzverhalten ist asymmetrisch und basiert auf dem Zugriff auf Daten im lokalen Speicher desselben Sockets (und nicht auf Daten, die von einem anderen Socket stammen). Latenzempfindliche Anwendungen benötigen lokalen Speicherverkehr und müssen ein Threading-Modell verwenden, das Socket-spezifische Zugriffslokalität aufweist, um mit der Anzahl der physischen Prozessoren zu skalieren.

Aerospike gruppiert mehrere Threads pro Socket statt pro Kern und passt sich damit einem NUMA-Knoten an. Diese Transaktions-Threads sind zudem bestimmten E/A-Geräten zugeordnet. Die Interrupt-Verarbeitung für die clientseitige Netzwerkkommunikation und die festplattenseitige E/A sind an den Kern gebunden, auf dem diese Threads ausgeführt werden. Dadurch müssen weniger Daten über verschiedene NUMA-Regionen hinweg ausgetauscht werden, was die Latenzzeiten verringert.

Netzwerkoptimierungen

Aerospike nutzt die Intel® Ethernet 800 Serie mit Application Device Queues (ADQ), um den Kontextwechsel weiter zu reduzieren und Daten in den lokalen Prozessorcaches zu halten.
Zu den wichtigsten Vorteilen dieser Netzwerktechnologie gehören: CPU-Entlastung durch die Ausführung von Warteschlangenentscheidungen auf der Netzwerkkarte (NIC), Verwendung von Busy Polling und anderen Techniken zur Verringerung der Latenzzeit und eine höhere Anzahl an Clients pro Knoten über ein 100-Gbit-Ethernet (10.000-fache Datenrate des ursprünglichen Protokolls). Aerospike war das erste kommerzielle DBMS, das ADQ nutzte, um geringere Latenzzeiten, hochgradig vorhersehbare Antwortzeiten und einen höheren Durchsatz zu erzielen.

Aerospike bearbeitet gleichzeitige Clientanfragen parallel, wobei der Verkehr gleichmäßig auf alle CPU-Kerne im System verteilt wird. Für jeden CPU-Kern gibt es eine Gerätewarteschlange, die jeweils so konfiguriert ist, dass sie Service-Interrupts auf einer bestimmten CPU erzeugt. Die gesamte Kernel- und Benutzermodus-Verarbeitung für Anfragen in einer bestimmten Warteschlange erfolgt im entsprechenden Kern, wodurch die Cachenutzung maximiert und nicht-lokale Speicherreferenzen auf Systemen mit mehreren Sockets minimiert werden. Um Overhead und Kontextwechsel weiter zu reduzieren und alle aufgelaufenen Pakete zu bearbeiten, nutzt Aerospike Linux-Netzwerkfunktionen (wie NAPI) und ein 50-ms-Polling-Intervall. In Kombination mit NUMA-Pinning erzielt Aerospike mit seiner ADQ-Unterstützung beeindruckende Leistungsvorteile. In diesem Blog wird die ADQ-Integration von Aerospike ausführlicher beschrieben, einschließlich Benchmark-Ergebnissen, die die Verbesserungen beim Transaktionsdurchsatz und bei der Vorhersagbarkeit der Antwortzeiten veranschaulichen.

Um Kontextwechsel zu vermeiden, führt Aerospike auch einige Operationen im Network-Listener-Thread aus. Dabei erstellt das System so viele Network-Listener-Threads, wie es Kerne gibt, um die Parallelität voll auszuschöpfen. Clientanfragen werden in der CPU empfangen, verarbeitet und erfüllt. Dies geschieht auf eine schnelle, nicht blockierende, und vorhersehbare Weise.

Speicherdefragmentierung

Aerospike teilt den gesamten für den residenten Index verwendeten Speicher nativ in Slabs auf, um maximale Kontrolle und Effizienz zu gewährleisten. Das ist wichtig, da wachsende Datenmengen, Transaktionsraten und DRAM-Kapazitäten die Speicherfragmentierung enorm erschweren. Um

Fragmentierung zu vermeiden, verwendet Aerospike den Speicherallokator `jemalloc` und benutzerdefinierte Erweiterungen, einschließlich spezieller Slab-Allokatoren (Arenen), um verschiedene Objekttypen innerhalb des Serverprozesses zu behandeln. Die Gruppierung von Datenobjekten nach Namensraum in derselben Arena optimiert die Erstellung, den Zugriff, die Änderung und die Löschung von Objekten.

Aufbau der Datenstruktur

Aerospike verwendet partitionierte Single-Threaded-Datenstrukturen mit detaillierten individuellen Sperren. Dadurch werden Speicherkonflikte zwischen den Partitionen reduziert und atomare Operationen werden auf modernen CPUs schnell verarbeitet. Im Gegensatz dazu erfordern verschachtelte Strukturen (z. B. B+-Bäume) Sperren auf jeder Ebene, was den gleichzeitigen Datenzugriff einschränken und den Durchsatz beeinträchtigen kann. Aerospike ermöglicht sichere und gleichzeitige Lese-, Schreib- und Löschzugriffe ohne Mehrfachsperren. Die Aerospike-Strukturen sind so konzipiert, dass Daten, auf die häufig zugegriffen wird, Lokalität haben und in eine einzige Cachezeile fallen, um Cache-Misses und Datenstaus zu reduzieren.

Scheduling und Priorisierung

Zusätzlich zu den Key-Value-Operationen unterstützt Aerospike Scans, Batch-Abfragen und sekundäre Indexabfragen. Um verschiedene Workloads effizient zu verwalten, verwendet Aerospike:

1. Typbasierte Jobpartitionierung. Jedem Jobtyp wird ein eigener Thread-Pool zugewiesen. Die Prioritäten werden in den Pools festgelegt. Jeder Job wird innerhalb seines eigenen Pools weiter priorisiert.
2. Aufwandsbezogene Arbeitseinheit. Die grundlegende Arbeitseinheit ist der Aufwand, der für die Verarbeitung eines einzelnen Datensatzes erforderlich ist, einschließlich Suche, E/A und Validierung. Jeder Job setzt sich aus mehreren Arbeitseinheiten zusammen, die seinen Aufwand definieren.
3. Kontrollierte Lastgenerierung. Der Thread-Pool verfügt über einen Lastgenerator, der die Geschwindigkeit der Arbeitserzeugung steuert. Die Threads im Pool verrichten die Arbeit.

Kooperatives Scheduling stellt sicher, dass Worker-Threads die CPU für andere Worker freigeben, damit diese ihre Arbeit nach X Arbeitseinheiten beenden können. Die Worker haben eine Affinität zu CPU-Kernen und Partitionen, um Datenkonflikte zu vermeiden, wenn parallele Worker auf bestimmte Daten zugreifen.

Gleichzeitige Workloads eines bestimmten Jobtyps werden in der Reihenfolge des Eingangs ausgeführt, um Latenzzeiten zu minimieren. Länger laufende Aufgaben, wie z. B. Scans und Abfragen, werden nach dem Round-Robin-Prinzip geplant, sodass viele Aufgaben eine Zeit lang parallel ausgeführt werden können, während andere angehalten und dynamisch neu geplant werden. Auf diese Weise werden langlaufende Jobs konstant vorangebracht, während kurzlaufende Jobs schnell abgeschlossen werden. Anwender können die Planungskonfiguration von Aerospike nach Bedarf anpassen.

Datenkonsistenz

Aerospike-Administratoren konfigurieren Namespaces oft für starke Konsistenz (Strong Consistency, SC), um Datenverluste und veraltete oder unsaubere Lesevorgänge wie bei herkömmlicher Unternehmensoftware zu verhindern. Alternativ können Namespaces im Verfügbarkeitsmodus betrieben werden. Dadurch werden zwar weniger Garantien geboten, aber bei Ausfällen so viele Daten wie möglich verfügbar gemacht. In beiden Modi sind die Garantien von Aerospike unmittelbar und der Transaktionsumfang besteht aus einem einzelnen Datensatz. Die Leistung ist vergleichbar, sodass bei Ausfällen hauptsächlich zwischen Konsistenz und Datenverfügbarkeit abgewägt werden muss.

Verfügbarkeitsmodus

Kurz gesagt: Der Verfügbarkeitsmodus ermöglicht gleichzeitige Lese- und Schreibvorgänge in mehreren Subclustern, wenn ein Netzwerk- oder sonstiger Ausfall dazu führt, dass einige Knoten nicht mehr miteinander kommunizieren können (dies wird auch als „Split Brain"-Szenario bezeichnet). Da jeder bestehen bleibende Subcluster Lese- und Schreibvorgänge verarbeiten kann, werden auf einem Subcluster gelesene Daten während der Trennung nicht auf einem anderen Subcluster aktualisiert. Ebenso führen gleichzeitige Schreibvorgänge auf Kopien desselben Datensatzes in verschiedenen Subclustern zu Konflikten bei der Wiederherstellung des Clusters, was zu Datenverlusten führen kann.

Strong-Consistency-Modus (SC-Modus): Überblick

Der SC-Modus hingegen verhindert Schreibkonflikte und stellt sicher, dass bei Lesevorgängen immer die zuletzt geschriebenen Daten zur Verfügung stehen. Dazu wird verhindert, dass bei Ausfällen mehrere Masterkopien derselben Daten aktiv sind. Um dies zu erreichen, muss der Zustand des Clusters bekannt sein. Dafür überwacht Aerospike die Heartbeats der Knoten und führt Abfragen in einem internen Verzeichnis durch. Das Verzeichnis listet die Knoten in einem gesunden Cluster auf und identifiziert die Master- und Replikatkopien der jeweiligen Datenpartitionen. Jeder Knoten im Cluster speichert eine Kopie dieses Verzeichnisses.

Im Gegensatz zu den meisten Systemen benötigt Aerospike nur zwei gespeicherte Kopien der Anwenderdaten, um für starke Konsistenz zu sorgen. Durch die Verwendung eines adaptiven Schemas, das bei Bedarf weitere Schreibkopien hinzufügt, optimiert Aerospike in den meisten Fällen die Leistung und verursacht nur selten einen geringen Overhead. Der Ansatz von Aerospike reduziert außerdem den Netzwerkdatenverkehr, den CPU-Overhead und die Speicherkosten. Die daraus entstehenden TCO-Einsparungen können beträchtlich sein, insbesondere bei großen Datenmengen.

Um Datenverluste zu vermeiden, bestätigt Aerospike die Schreibvorgänge in allen Replikaten, bevor es den erfolgreichen Vorgang an den Client zurückmeldet. Wenn ein Schreibvorgang auf einem Replikat fehlschlägt, sorgt der Master dafür, dass der Schreibvorgang auf der entsprechenden Anzahl von Replikaten durchgeführt wird. Aerospike verarbeitet alle Schreibvorgänge für einen Datensatz sequenziell, sodass sie nicht neu geordnet oder übersprungen werden. Unabhängige Tests des Jepsen-Workloads ergaben keine Fehler, wenn die empfohlenen Konfigurationseinstellungen von Aerospike verwendet wurden. Um die richtige Reihenfolge der Ereignisse im gesamten Cluster zu wahren, verwendet Aerospike eine angepasste Lamport-Uhr, die Zeitstempel mit weiteren Informationen kombiniert, einschließlich eines Zählers, der sich erhöht, wenn bestimmte Ereignisse auftreten. Dieser Ansatz bietet Vorteile gegenüber einer einfachen zeitstempelbasierten Architektur. Bei dieser können Probleme mit der Synchronisierung der Uhr zwischen den Knoten auftreten, die möglicherweise ungewollte Dateninkonsistenzen zur Folge haben.

SC-Modus: Fehlerbehandlung

Im SC-Modus nutzt Aerospike mehrere Regeln, um das System nach Ausfällen wiederherzustellen und neue Subcluster zu bilden. Wenn die Verzeichnis-Masterkopie einer Datenpartition nicht verfügbar ist, bestimmt Aerospike aus den verfügbaren Replikaten einen neuen Master und erstellt neue Replikate, um den eingestellten Replikationsfaktor durchzusetzen. Dies geschieht im Hintergrund und hat keine Auswirkungen auf die Verfügbarkeit. Während einer Clustertrennung können Anfragen für eine bestimmte Partition nur von einem Subcluster gemäß den folgenden Regeln angenommen werden:

1. Wenn ein Subcluster den Master und alle Replikate für eine Partition enthält, ist die Partition sowohl für Lese- als auch für Schreibzugriffe in diesem Subcluster verfügbar.
2. Wenn ein Subcluster die absolute Mehrheit der Knoten hat und entweder den Master oder ein Replikat für die Partition enthält, ist die Partition sowohl für Lese- als auch für Schreibzugriffe in diesem Subcluster verfügbar.
3. Wenn ein Subcluster genau die Hälfte der Knoten und den Master enthält, ist die Partition sowohl für Lese- als auch für Schreibvorgänge in diesem Subcluster verfügbar.

SC-Modus: Anwendungsbeispiele

Lassen Sie uns vor diesem Hintergrund einige Beispiele betrachten.. Abbildung 5 zeigt vier Betriebszustände eines 5-Knoten-Systems mit dem Replikationsfaktor 2. Knoten 5 verwaltet die Stammkopie einer bestimmten Datenpartition und Knoten 4 speichert das Replikat. Die oberste Zeile zeigt einen gesunden Cluster, bei dem alle Knoten in Betrieb und alle Daten verfügbar sind.

In der zweiten Zeile sind die Knoten 1–3 von den Knoten 4–5 getrennt. Lese-/Schreibanfragen für die Zieldaten sind nur auf den Knoten 4–5 zulässig, die sowohl die Master- als auch die Replikatkopien enthalten, wie im Verzeichnis angegeben (siehe Regel 1). In der dritten Zeile sind die Knoten 1–4 von Knoten 5 getrennt. Da Knoten 4 das Verzeichnis-Replikat der Daten enthält und Teil eines Subclusters mit den meisten Knoten ist, übernimmt er die Masterrolle für diese Daten und verarbeitet Lese-/Schreibanfragen (siehe Regel 2). Darüber hinaus repliziert Aerospike diese Daten automatisch auf einen anderen verfügbaren Knoten (in diesem Beispiel Knoten 3), um den eingestellten Replikationsfaktor einzuhalten. Knoten 5 nimmt keine Lese-/Schreibanfragen an, da er zu einem Minderheits-Subcluster gehört (in diesem Fall zu einem einzelnen Subcluster). Auf diese Weise werden Datenverluste und veraltete Lesevorgänge verhindert. Die letzte Zeile zeigt, was bei einer anschließenden Trennung zwischen Knoten 4 und den Knoten 1–3 passieren würde. Aerospike würde dann keine Lese-/Schreibanfragen für Datensätze in dieser Partition verarbeiten, da die Knoten 4 und 5 – die im Verzeichnis als Replikat- und Masterbesitzer der Daten angegeben sind – keinen eigenen Subcluster bilden und nicht Teil eines Subclusters sind, in dem sich die Mehrheit der im Verzeichnis aufgeführten Knoten befinden.

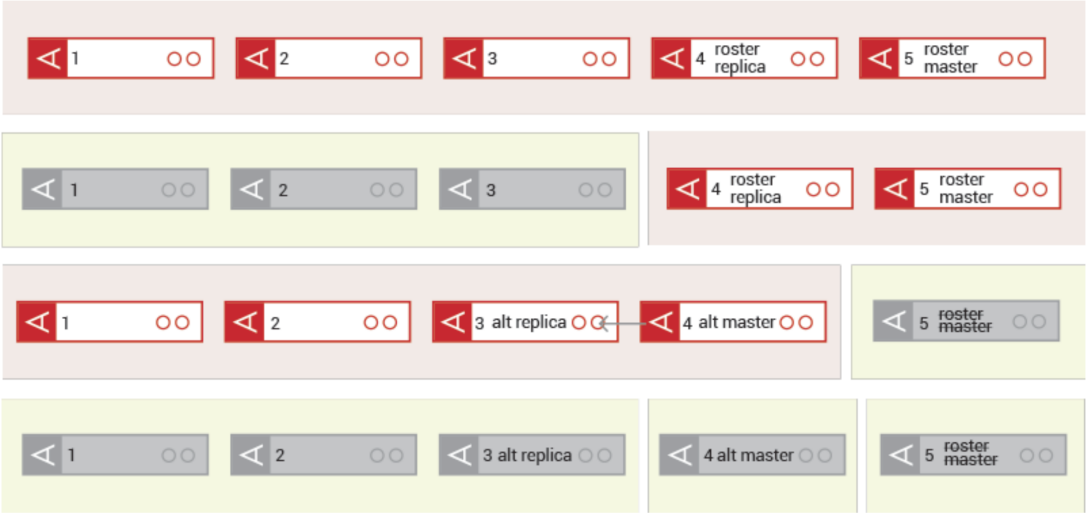


ABBILDUNG 5: Starke Konsistenz in verschiedenen Clusterzuständen erzwingen

Das Strong-Consistency-Modell von Aerospike berücksichtigt subtile Situationen, um ein angemessenes Verhalten zu gewährleisten. Betrachten wir zum Beispiel ein Fehlerszenario, in dem der Verzeichnis-Masterknoten für einen Datensatz ein Netzwerkproblem, das den Clusterzustand betrifft, noch nicht erkannt hat und somit den Masterbesitz seiner Daten noch nicht abgegeben hat – obwohl der neue Master (in einem separaten Subcluster) die Zustandsänderung des Clusters erkannt und übernommen hat. Während dieser Übergangsphase (manchmal auch als „Master Overhang“ bezeichnet) können die Clients sowohl auf den alten als auch auf den neuen Master schreiben, was zu Dateninkonsistenzen führen kann, wenn zur Sortierung von Ereignissen nur auf Zeitstempel gesetzt wurde. Um Inkonsistenzen dieser Art zu vermeiden, verknüpft Aerospike mit jeder Partition sogenannte „Regime“-Informationen. In Kombination mit der Zeit der letzten Aktualisierung und den Datensatzerstellungsdaten beinhalten diese Informationen die angepasste Lamport-Uhr von Aerospike, mit der Ereignisse im gesamten Cluster korrekt zugeordnet und Datenverluste vermieden werden können.

Aerospike bietet volle Datenverfügbarkeit während rollierender Upgrades. Wenn die Anzahl der fehlenden Knoten in einem Subcluster unter dem Replikationsfaktor liegt, dient er als Subcluster mit Supermehrheit und alle Partitionen sind für Lese-/Schreibvorgänge aktiv. Außerdem sind die Daten vollständig verfügbar, wenn sich das System in genau zwei Subcluster aufteilt. Alle Partitionen sind dann in dem einen oder dem anderen Subcluster für Lese-/Schreibvorgänge aktiv.

SC-Modus: Optionen für die Lesekonsistenz

Anwendungen können bei jedem Lesevorgang zwischen *linearem Zugriff* und *Sitzungskonsistenz* wählen. Ersteres ist am restriktivsten und erzwingt für alle zugreifenden Clients eine lineare Sicht auf die Daten. Wenn eine Anwendung einen Schreibvorgang abgeschlossen hat, ist der Wert dieses Vorgangs oder eines späteren Schreibvorgangs für alle späteren Lesevorgänge einer beliebigen Anwendung verfügbar. Anhand der Regime-Informationen für jede Datenpartition kann Aerospike feststellen, ob alle Kopien zum Zeitpunkt der Leseanfrage synchronisiert sind. Ist dies der Fall, kann der Lesevorgang fortgesetzt werden. Andernfalls ist möglicherweise eine Clusteränderung im Gange und der Client muss den Vorgang erneut anstoßen. Die Sitzungskonsistenz ist weniger restriktiv und garantiert lediglich, dass eine Anwendung ihre eigenen Schreibvorgänge liest. Wenn gelegentliche veraltete Lesevorgänge über verschiedene Anwendungsinstanzen hinweg kein Problem darstellen, kann die mit dem Verfügbarkeitsmodus verbundene deutlich höhere Leistung auch mit Sitzungskonsistenz erreicht werden.

Weitere Informationen zum Ansatz für Datenkonsistenz von Aerospike finden Sie in der Dokumentation von Aerospike.

Sicherheit

Aerospike sichert die Betriebsdaten der Anwender durch verschiedene Authentifizierungs- und Autorisierungsmechanismen, Verschlüsselung auf Daten- und Transportebene sowie Prüfprotokolle. Zusammen bieten diese Funktionen Unternehmensanwendern die nötige Sicherheit für die Verwaltung ihrer unternehmenskritischen Daten.

Anwender authentifizieren sich bei Aerospike mit intern verwalteten Passwörtern oder, falls gewünscht, über externe Dienste wie Lightweight Directory Access Protocol (LDAP) oder Kerberos. Jeder Anwender kann für eine oder mehrere Rollen autorisiert werden, die ihm bestimmte Berechtigungen verleihen. Um den Betrieb zu vereinfachen, bietet Aerospike mehrere vordefinierte Rollen, wobei Administratoren bei Bedarf neue Rollen erstellen können. Zu den vordefinierten Rollen gehören:

- **Lesen** aus einem Datensatz.
- **Schreiben** in einen Datensatz (ohne Leseberechtigung). Diese Schreibberechtigung ist besonders nützlich, wenn hochsensible Informationen erfasst, aber die zugehörige Berichterstellung eingeschränkt werden soll.
- **Lesen/Schreiben** in/aus einem Datensatz.
- **Lesen/Schreiben/Ausführen von UDFs** in einem Datensatz.
- **Datenverwaltung**, z. B. Objekte erstellen/löschen und Scans oder Abfragen abbrechen
- **Systemverwaltung**, alle Datenverwaltungsrechte sowie Berechtigungen für Vorgänge auf Systemebene, z. B. die dynamische Neukonfiguration des Servers und das Aktivieren/Deaktivieren von Audits.
- **Anwenderverwaltung**, z. B. Anwender erstellen/löschen und Rollen zuweisen/widerrufen.

Außerdem unterstützt Aerospike Zulassungslisten, um den Datenzugriff auf Domänen- und Anwenderebene zu beschränken.

Die Verschlüsselung wird für Daten während der Übertragung (TLS 1.2) und für ruhende Daten (AES-128 und AES-256) unterstützt. Um den Overhead zu minimieren, verschlüsselt Aerospike die Daten pro Datensatz und nicht pro Festplattensektor. Bei internen Tests konnten keine messbaren Leistungseinbußen bei Datensatzgrößen ab 1 KB festgestellt werden. Aerospike ermöglicht die Verwaltung und Speicherung der folgenden Sicherheitselemente in einem HashiCorp-Vault: LDAP-Anmeldeinformationen und TLS-Zertifikate, XDR-Passwörter für Remote-Ziele, Encryption-at-Rest-Schlüssel und Netzwerk-TLS- und -Schlüssel.

Intelligente Clientschicht und APIs

Die Clientschicht von Aerospike ist eine Open-Source-Bibliothek, die mehr als 10 Sprachen unterstützt und immer den aktuellen Zustand des Clusters kennt. Die API unterstützt das Erstellen, Lesen, Aktualisieren und Löschen von Daten sowie das Ausführen von Abfragen und benutzerdefinierten Funktionen.

Um die Leistung zu maximieren, speichert der Aerospike-Client Informationen darüber, wie die Daten im Cluster verteilt sind (Zuordnung von Datenpartitionen zu Knoten). Das Abrufen eines Datensatzes erfordert in der Regel nur einen Netzwerk-Hop: Der Aerospike-Client fordert die Daten vom Knoten an, auf dem sie sich befinden, und der Server gibt sie zurück. Scans oder Abfragen werden parallel auf mehreren Knoten verarbeitet.

Ohne korrektes Transaktionsrouting vom Client würde eine Anfrage mehr Netzwerk-Overhead erfordern. Dann müsste in der Mitte der Transaktion entweder ein Proxy platziert werden, was die Latenzzeit erhöhen und den Durchsatz verringern würde, oder die Transaktionen würden über die Clusterverbindung laufen, was einen zusätzlichen Hop zwischen den Servern bedeuten würde. Durch das Design von Aerospike wird dieser Overhead vermieden.

Globale Transaktionsverarbeitung

Die globale Wirtschaft hat die Voraussetzungen für viele Transaktionsanwendungen verändert, z. B. für Handelsabrechnungen, Lieferkettenmanagement, Währungsumtausch und Paketverfolgung. Die Forderung der Kunden nach einer schnellen Abwicklung kollidiert mit Geschäftsprozessen, die Stunden oder Tage in Anspruch nehmen. Dies hat viele Unternehmen dazu gezwungen, ihre zugrunde liegenden Infrastrukturen für die Transaktionsverarbeitung zu überdenken und eine hochbelastbare, geografisch verteilte Datenbankplattform mit hoher Datenkonsistenz und Leistung zu nutzen.

Active/Active-Datenbankinfrastrukturen sind bei diesen digitalen Transformationsprojekten oft ein Muss. Diese Plattformen umfassen zwei oder mehr Regionen und verarbeiten Anwendungsanfragen an allen Standorten. Aerospike bietet zwei Optionen: rechenzentrumsübergreifende Replikation (XDR) und standortübergreifendes Clustering. Lassen Sie uns beide Optionen nacheinander betrachten.

XDR

Bei der rechenzentrumsübergreifenden Replikation werden Daten transparent und asynchron zwischen Aerospike-Clustern repliziert. Unternehmen setzen oft auf XDR, um Daten von Aerospike-basierten Edge-Systemen in ein zentrales Aerospike-System zu replizieren. XDR ermöglicht Unternehmen auch, den Betrieb aufrechtzuerhalten, wenn ein ganzer Cluster in Folge eines Ausnahmezustands (z. B. bei einer Naturkatastrophe) ausfällt.

Die XDR-Replikation ist protokollbasiert. Anwendungen schreiben in ihren „lokalen“ Cluster und XDR zeichnet minimale Informationen zu den geänderten Werten auf (nicht den gesamten Datensatz). Zur Steigerung der Effizienz fasst XDR geänderte Daten in Batches zusammen und sendet nur die neueste Version eines Datensatzes. So erzeugen mehrere lokale Schreibvorgänge für einen Datensatz nur einen einzigen Remote-

Schreibvorgang – eine wichtige Funktion für häufig abgerufene und wichtige Daten. Außerdem hält XDR eine Auswahl offener Verbindungen für jeden Remote-Cluster bereit, sodass Replikationsanfragen nach dem Round-Robin-Verfahren verteilt werden.

Bei XDR wird ein Single Point of Failure vermieden. Solange mindestens ein Knoten in den Quell- und Zielclustern bestehen bleibt, werden Änderungen von Aerospike übertragen. Darüber hinaus passt sich die Lösung problemlos an neue Knoten in beliebigen Clustern an und kann alle verfügbaren Ressourcen gleichermaßen nutzen.

Da Anwendungen in einer Active/Active-Konfiguration Daten an jedem beliebigen Ort schreiben können, kann es auf Datensatz- oder Bin-Ebene zu Konflikten kommen, wenn dieselben Daten gleichzeitig an mehreren Standorten aktualisiert werden. Dies kann zu unterschiedlichen Werten an verschiedenen Orten oder zum Verlust einiger Daten führen. Einige Anwendungen können diese lockere Form der Datenkonsistenz tolerieren. Ist dies nicht der Fall, können Unternehmen den Schreibzugriff auf bestimmte Daten auf einen bestimmten Ort beschränken, um so das Risiko gleichzeitiger, in Konflikt stehender Aktualisierungen zu vermeiden. Die asynchrone Eigenschaft der Shipping-Algorithmen bedeutet, dass die neuesten Aktualisierungen eines Standorts bei einem Ausfall nicht an den anderen Standorten verfügbar sind. Die Geschwindigkeit und der Durchsatz von XDR eignen sich am besten für Anwendungsfälle, bei denen eine geringe Latenz unerlässlich ist. Die Technologie garantiert zwar keine starke Konsistenz zwischen den Standorten, bietet dafür aber Lese- und Schreibvorgänge mit niedrigeren Latenzen. Weitere Informationen zu XDR finden Sie in diesem Blog über Aerospike Database 5 und in diesem Blog über Active/Active-Funktionen.

Standortübergreifendes Clustering

Wie der Name schon andeutet, können sich durch die Multi-Site-Clustering-Unterstützung von Aerospike einzelne Cluster über mehrere geografische Regionen erstrecken (siehe Abbildung 6). Dadurch lassen sich gemeinsam genutzte Datenbanken in weit entfernten Rechenzentren und Cloud-Regionen einsetzen, ohne Datenverluste zu riskieren. Die Aerospike-Implementierung zeichnet sich durch automatisiertes Failover, hohe Ausfallsicherheit und starke Leistung aus. Zwei Merkmale liegen dem standortübergreifenden Clustering von Aerospike zugrunde: Rack Awareness und starke Konsistenz.

Rack Awareness ermöglicht es, Replikate von Datenpartitionen auf verschiedenen Hardware-Ausfallgruppen (verschiedenen Racks) zu speichern. Über die Einstellungen des Replikationsfaktors können Administratoren jedes Rack so konfigurieren, dass eine vollständige Kopie aller Daten gespeichert wird, um die Datenverfügbarkeit und die lokale Leseleistung zu maximieren. Um Hotspots zu vermeiden, verteilt Aerospike die Daten gleichmäßig auf alle Knoten innerhalb jedes Racks.

Dabei wird die Masterkopie einer bestimmten Datenpartition immer nur von einem Knoten verwaltet. Andere Knoten (in der Regel auf anderen Racks) speichern Replikate, die Aerospike automatisch mit dem Master synchronisiert. Wie bereits erwähnt, werden die Standorte der Master und Replikate im Verzeichnis erfasst. Auch die Racks und Knoten eines gesunden Clusters sind hier bekannt. In Abbildung 6 ist zu sehen, dass jedes Rechenzentrum über ein Rack mit drei Knoten verfügt. Auf jedem Knoten befindet sich dabei eine Kopie des Verzeichnisses (blau dargestellt). Dieses Beispiel (mit einem Replikationsfaktor von 3) zeigt die Verzeichnis-Masterkopie einer gelben Datenpartition auf Knoten 3 von Rack 2. Replikate sind auf Knoten 1 von Rack 1 und Knoten 2 von Rack 3 gespeichert.

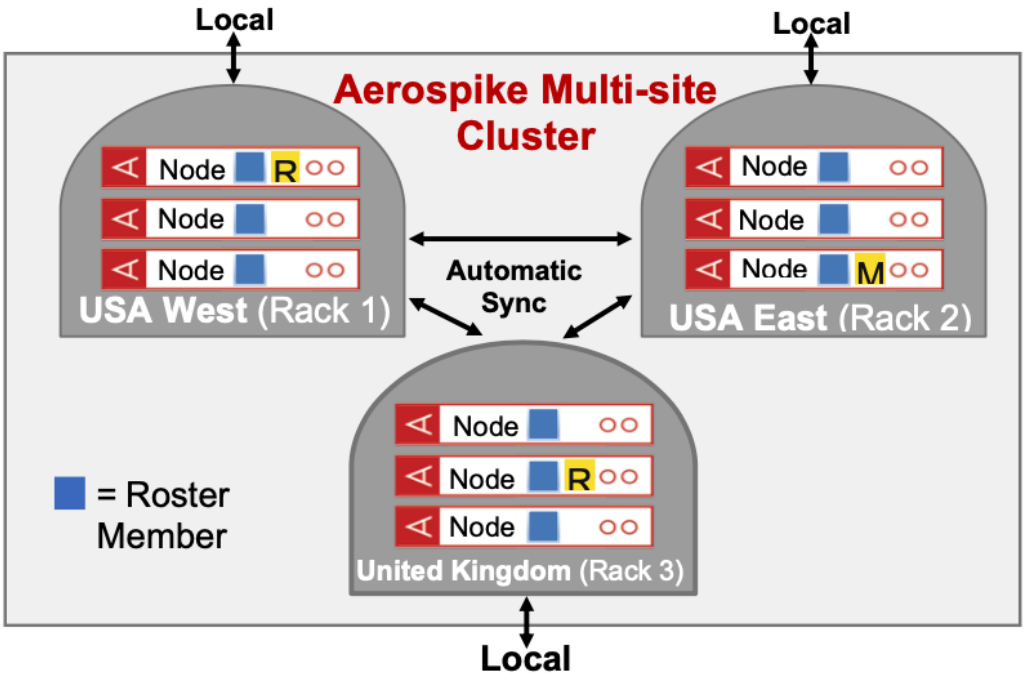


ABBILDUNG 6: Beispiel für ein Aerospike-System mit einem Replikationsfaktor von 3, das sich über 3 Standorte erstreckt.

Aerospike leitet Anfragen von Anwendungen zum Lesen von Datensätzen an das entsprechende Rack bzw. den entsprechenden Knoten im lokalen Rechenzentrum weiter. Abbildung 6 zeigt, dass eine Anwendung in der Region „USA East“, die Daten in der gelben Partition lesen möchte, mit einem Netzwerk-Hop auf die Masterkopie in Rack 2, Knoten 3 zugreifen kann. Eine Anwendung aus dem Vereinigten Königreich, die denselben Datensatz lesen möchte, kann ebenfalls mit nur einem Hop zugreifen, da Aerospike die Daten vom Replikat in Rack 3, Knoten 2 abruft. Durch die intelligente Verarbeitung von Leseanfragen kann Aerospike in Clustern mit mehreren Standorten Leselatenzen von unter einer Millisekunde erreichen.

Schreibvorgänge werden anders verarbeitet. Um die Konsistenz im gesamten Cluster zu gewährleisten, leitet Aerospike jeden Schreibvorgang an das Rack bzw. den Knoten mit der aktuellen Masterkopie der Daten weiter. Bevor der Vorgang bestätigt wird, stellt der Masterknoten sicher, dass der Schreibvorgang auf seine Kopie und alle Replikate angewendet wird. In Abbildung 6 würde ein Schreibvorgang auf die gelbe Partition an Rack 2, Knoten 3 weitergeleitet werden, unabhängig vom Ursprung der Anfrage. Die Weiterleitung von Schreibvorgängen und die Synchronisierung von Replikaten verursachen Overhead, weshalb Schreibvorgänge nicht so schnell ausgeführt werden wie Lesevorgänge. Die meisten Unternehmen, die das standortübergreifende Clustering von Aerospike verwenden, erzielen jedoch Schreiblatenzen von einigen hundert Millisekunden oder weniger, was durchaus im Rahmen ihrer Ziel-SLAs liegt.

Für die globale Transaktionsverarbeitung ist die Ausfallsicherheit von entscheidender Bedeutung. Standortübergreifende Cluster folgen denselben allgemeinen Regeln zur Durchsetzung einer starken Datenkonsistenz wie herkömmliche Cluster. Für gängige Szenarien werden automatisch Korrekturmaßnahmen ergriffen. Wenn der Verzeichnis-Master beispielsweise aufgrund eines Knoten- oder Netzwerkausfalls nicht mehr verfügbar ist, bestimmt Aerospike einen neuen Master aus den verfügbaren Replikaten und erstellt bei Bedarf neue Replikate, um den Replikationsfaktor zu erreichen. In einem standortübergreifenden Cluster befindet sich der neue Master normalerweise in einem anderen Rack.

Nehmen wir nun an, dass ein Rechenzentrum nach einem Netzwerk- oder Stromausfall nicht mehr verfügbar ist. In Abbildung 7 ist der britische Standort (Rack 3) für den Rest des Clusters nicht mehr erreichbar. In

einem solchen Fall, bildet Aerospike automatisch einen neuen Subcluster, bestehend aus USA West (Rack 1) und USA East (Rack 2), um Lese- und Schreibvorgänge ohne menschliches Eingreifen fortzusetzen.

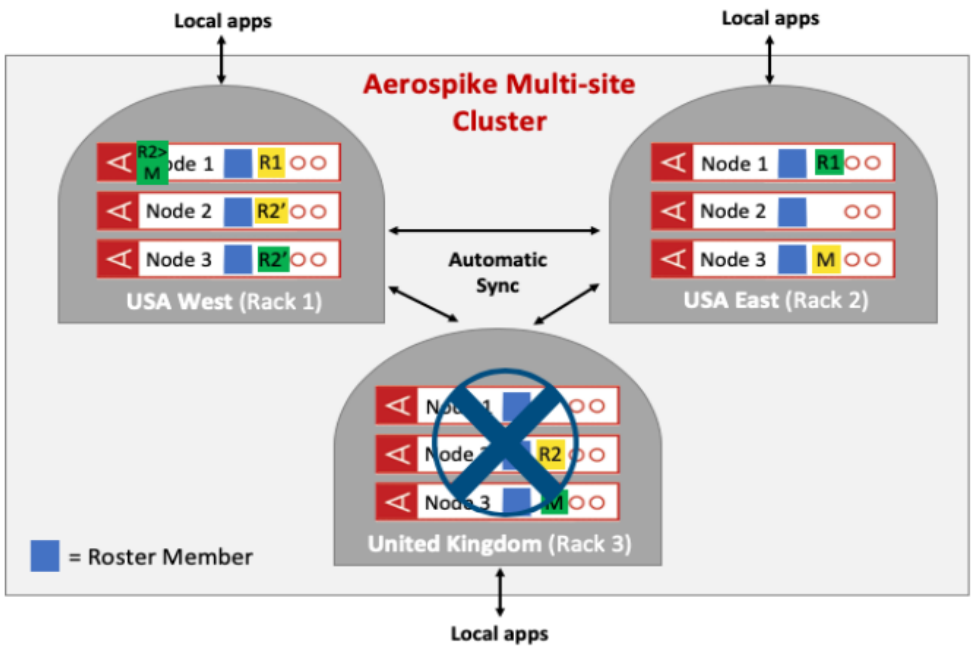


ABBILDUNG 7: Ausfallszenario, wenn ein Rechenzentrum unerreichbar wird

Betrachten wir zunächst den Zugriff auf die gelben Daten. USA East verarbeitet alle Schreibvorgänge, da sich hier der Master befindet. Lesevorgänge werden hingegen an einem beliebigen Ort im Subcluster verarbeitet. Um einen Replikationsfaktor von 3 beizubehalten, erstellt Aerospike automatisch ein neues gelbes Replikat (R2' auf Rack 1, Knoten 2). Werfen wir jetzt einen Blick auf die grünen Daten. Das Rack, in dem sich der Master befindet, ist nicht im aktiven Subcluster, weshalb Aerospike ein Replikat zum Master hochstuft. In diesem Beispiel wird R2 in USA West zum Master hochgestuft, und alle Schreibvorgänge, die diese Daten betreffen, werden dorthin geleitet. Lesevorgänge werden an jedem aktiven Standort im Subcluster ausgeführt. Um den Replikationsfaktor 3 beizubehalten, erstellt Aerospike automatisch ein neues grünes Replikat (R2' auf Rack 1, Knoten 3).

Wenn das Rechenzentrum im Vereinigten Königreich wieder verfügbar ist, werden seine Daten mit dem aktiven Subcluster synchronisiert. Nach der erneuten Verbindungsherstellung mit dem Cluster hostet der UK-Standort wieder die Masterkopie der grünen Daten und ein Replikat der gelben Daten. Aerospike entfernt die temporären Replikate auf anderen Knoten und wandelt alle temporären Masterkopien in Replikate um.

Eine ausführlichere Beschreibung des standortübergreifenden Clusterings finden Sie in einem separaten Whitepaper.

Integrationsmöglichkeiten

Um Unternehmen bei der Integration von Aerospike in ihre IT-Infrastruktur zu unterstützen, bietet Aerospike Konnektoren für Spark, Kafka und Java Message Service (JMS). Darüber hinaus sind mehr als ein Dutzend von der Community entwickelte Konnektoren für Apache Hadoop, das Spring-Framework, ASP.NET und andere verfügbar. Es würde den Rahmen dieses Dokuments sprengen, alle Aerospike-Konnektoren im Einzelnen zu beschreiben, deshalb gehen wir nur kurz auf Aerospike Connect für Spark ein.

Mit dem Spark-Konnektor von Aerospike können Spark-Entwickler den Umfang ihrer Echtzeitanalysen erweitern, indem sie Aerospike-Daten mit Daten kombinieren, die über Spark gestreamt werden. Insbesondere können Spark-Entwickler Aerospike-Daten in Datasets und DataFrames zur Analyse mit Spark SQL, Spark ML und anderen Spark-Bibliotheken laden. Darüber hinaus können Spark-Daten in Aerospike für die spätere Verwendung durch Spark-fremde Anwendungen persistiert werden, wodurch sich neue Nutzungsmöglichkeiten ergeben. Durch die effiziente Nutzung von flüchtigem, nicht-flüchtigem und persistentem Speicher in Aerospike erhalten Spark-Anwender einen kostengünstigen Backupspeicher für große Mengen an Echtzeitdaten.

Aerospike Connect für Spark optimiert die Leistung auf verschiedene Weisen. Beispielsweise unterstützt der Konnektor eine massive Parallelisierung, indem er bis zu 32.768 Spark-Partitionen nutzt, um Daten aus einem Aerospike-Namespace zu lesen, der bis zu 32 Milliarden Datensätze in 4.096 Partitionen speichert. Aerospike scannt seine Partitionen unabhängig und parallel. In Kombination mit Spark-Workern können Spark-Anwender große Aerospike-Datenmengen schnell verarbeiten (derzeit 100 TB innerhalb weniger Stunden).

Darüber hinaus können Spark-Programmierer Abfrageprädikate verwenden, um die Aerospike-Verarbeitung auf relevante Partitionen zu beschränken. Dies vermeidet kostspielige Scans des gesamten Datasets, beschleunigt den Datenzugriff und verhindert unnötige Datenbewegungen zwischen Aerospike und Spark. Die Filterung großer Datasets auf dem Aerospike-Server minimiert auch den Speicherbedarf auf Spark-Seite und vermeidet potenzielle Laufzeitfehler bzw. hohe Betriebskosten. Aersopike-Administratoren können die Anzahl der parallelen Scans über eine Konfigurationseinstellung steuern und die Anzahl der Scan-Threads an die Anforderungen ihrer Spark-Anwendungen anpassen. Darüber hinaus verwendet Connect für Spark das Worker-Thread-Pooling, um den Overhead beim Schreiben von Spark-Daten in Aerospike zu minimieren. Weitere Optimierungen umfassen Batch-Lesevorgänge für Abfragen mit Primärschlüsseln und die Partitionsfilterung für Sekundärschlüssel.

Gemeinsam ermöglichen Aerospike und Spark die Verarbeitung sehr umfangreicher Jobs, die sonst nicht durchführbar wären. Beispiele hierfür sind KI/ML-Training, 360-Grad-Profile von Millionen von Kunden, die Umwandlung großer Mengen an öffentlichen und privaten Datasets und vieles mehr.

Leistungsbenchmarks und TCO-Vergleiche

Eine Systemarchitektur ist nur so überzeugend, wie die greifbaren Vorteile, die sie bietet. Wie eingangs erwähnt, haben sich viele Unternehmen für Aerospike entschieden, nachdem sie festgestellt hatten, dass andere Lösungen ihre Anforderungen hinsichtlich Leistung, Verfügbarkeit, Skalierbarkeit oder Budget nicht erfüllen konnten. Im Folgenden haben wir für Sie einige der wesentlichen Vorteile zusammengestellt, die uns von Kunden und Partnern genannt wurden:

- Signal, Hersteller einer Identity-Resolution-Plattform, konnte durch die Migration auf Aerospike die Anzahl seiner Server von 450 auf 60 reduzieren und die Leistung im 99-Perzentil-Bereich um das 100-fache steigern. Geschäftsprozesse werden nun in einem Bruchteil der Zeit (10 % oder weniger) ausgeführt, und das Unternehmen kann die freigesetzten Ressourcen für neue strategische Projekte verwenden.
- Playtika, ein Online-Gaming-Unternehmen, konnte mit Aerospike seinen Server-Footprint um den Faktor 6 reduzieren und die Leistung um 300 % steigern. Das Unternehmen rechnet mit Einsparungen bei den Gesamtbetriebskosten von bis zu 4,2 Millionen Dollar über einen Zeitraum von 3 Jahren.
- Das Team für KI-basierte Big-Data-Lösungen von Hewlett Packard Enterprise hat mehrere PMEM-fähige NoSQL-Systeme einem Benchmarking unterzogen, um herauszufinden, wie gut diese ihre

Anforderungen an ultraschnelle Leistung und extreme Skalierbarkeit erfüllen. Aerospike erwies sich dabei als Sieger. Ein System konnte die Daten nicht einmal innerhalb des vorgegebenen Zeitraums vollständig laden. Bei der knotenbasierten Verarbeitung der Ziel-Workload war Aerospike 1.667 % schneller als ein anderes System.

- Eine große europäische Bank nutzt Aerospike für die Zahlungsverarbeitung von 2.000 Transaktionen pro Sekunde bzw. bis zu 43 Millionen Transaktionen pro Tag bei Kosten von 0,0020 € oder weniger pro Transaktion. Andere Lösungen konnten die Anforderungen der Bank in Bezug auf Ausfallsicherheit (100 % Verfügbarkeit), Konsistenz (keine Datenverluste und keine Schreib-Lese-Konflikte oder veralteten Lesevorgänge) sowie niedrige Transaktionskosten nicht erfüllen.
- Ein führendes Brokerunternehmen nutzt Aerospike, um seine Mainframe-Finanzhandelslösung zu entlasten. Ursprünglich wurde eine Caching-Schicht über dem Mainframe-DBMS verwendet, um die Antwortzeiten zu verbessern. Als das Unternehmen aber erkannte, dass der Cache von 150 auf 1.000 Knoten erweitert werden musste, um den zukünftigen Anforderungen gerecht zu werden, wandte es sich an Aerospike. Das Unternehmen steigerte die Verarbeitungsgeschwindigkeit um das Fünffache und verdreifachte gleichzeitig die Größe der Datenbank. Dank Aerospike benötigte das Unternehmen 90 % weniger Server, was zu Einsparungen von schätzungsweise 10.000 Dollar pro Handelstag führte. Im Zuge der Bereitstellung konnte das Unternehmen zudem mehr als ein Dutzend Anwendungen auf Aerospike ausführen, was seiner Schemaflexibilität und hohen Verfügbarkeit zu verdanken ist. Diese Anwendungen ermöglichen zum Beispiel die Überwachung des Kreditrisikos, die Verbesserung des Kundenservice und vieles mehr.

Weitere Informationen finden Sie in diesem Benchmarkvergleich und TCO-Bericht.

Fazit

Zeit ist in der Geschäftswelt immer kürzer bemessen, weshalb Unternehmen Datenverwaltungsinfrastrukturen benötigen, die mehr Daten und Transaktionen in Echtzeit immer schneller verarbeiten können. Ausfallzeiten sind nicht akzeptabel, ebenso wenig hohe Betriebskosten.

Bei der Entwicklung der Aerospike-Architektur wurden diese Anforderungen berücksichtigt. Mit den hier beschriebenen Technologien und Ansätzen macht sich Aerospike die Leistungsvorteile moderner Prozessoren, Speichergeräte und Netzwerkhardware zunutze, um eine Datenbankplattform mit hohem Durchsatz und geringer Latenz für Echtzeitanwendungen bereitzustellen. Auf diese Weise können große Mengen an Echtzeitdaten unverändert und effizient in verschiedenen Analyse-Pipelines (wie Spark, Kafka, JMS und anderen) verwendet werden. Im Gegensatz zu anderen Plattformen ist Aerospike skalierbar und bietet eine vorhersehbare, ultraschnelle Performance sowie vorhersehbare, niedrige TCO. Aus diesem Grund nutzen Unternehmen aus dem Bank- und Finanzwesen, der Telekommunikation, dem Einzelhandel, der IT-Branche sowie aus anderen Bereichen Aerospike am Netzwerkrand als System of Engagement und im Netzwerkkern als System of Record, um ihre Infrastrukturen für die globale Wirtschaft von heute zu digitalisieren.

Möchten Sie mehr über die Architektur von Aerospike erfahren? Kontaktieren Sie Aerospike, um ein Briefing zu vereinbaren oder mögliche Pilotprojekte zu besprechen.

Über Aerospike

Mit Aerospike können Unternehmen scheinbar unlösbare Datenengpässe überwinden und zu einem Bruchteil der Kosten und Infrastrukturkomplexität herkömmlicher NoSQL-Datenbanken wettbewerbsfähig bleiben und die Nase vorn behalten. Die patentierte Hybrid Memory Architecture™ von Aerospike schöpft das volle Potenzial moderner Hardware aus und unterstützt die hocheffiziente Verarbeitung großer Datenmengen am Rand und im Kern des Netzwerks sowie in der Cloud. Auf diese Weise bietet sie einen unschlagbaren Wettbewerbsvorteil. Mit Aerospike können Kunden Betrug effektiv bekämpfen, die Warenkorbgröße drastisch vergrößern, globale digitale Zahlungsnetzwerke implementieren und eine 1:1-Personalisierung für Millionen von Kunden realisieren. Zu den Kunden von Aerospike gehören Airtel, Banca d'Italia, Nielsen, PayPal, Snap, Verizon Media und Wayfair. Der Hauptsitz des Unternehmens befindet sich in Mountain View, Kalifornien, mit weiteren Standorten in London, Bengaluru, Indien, und Tel Aviv, Israel.