Technisches Whitepaper

Dell EMC ECS: Design für hohe Verfügbarkeit

Zusammenfassung

In diesem Whitepaper werden die architektonischen Details dazu beschrieben, wie die Dell EMC™ ECS-Plattform die Verfügbarkeit der Enterprise-Klasse bereitstellt.

June 2021

H16344.6

Überarbeitungen

Datum	Beschreibung
Juli 2017	Erstausgabe
August 2017	Aktualisiert um den Inhalt von ECS-Version 3.1
März 2019	Aktualisiert um den Inhalt von ECS-Version 3.3
April 2020	„TSO-Verhalten bei aktiviertem Zugriff während eines Ausfalls“ aktualisiert
Dezember 2020	Aktualisierung der Metadatenschutzmethode
Juni2021	Aktualisiert um den Inhalt von ECS-Version 3.6.1

Mitwirkung

Dieses Whitepaper wurde erstellt von:

Autor: Zhu, Jarvis

Inhaltsverzeichnis

Zusammenfassung

Unternehmen speichern immer größere Mengen an Daten, die unbedingt verfügbar sein müssen. Die Kosten und Komplexitäten der Wiederherstellung großer Datenmengen im Falle von System- oder Standortausfällen können für eine IT-Abteilung überwältigend sein.

Die Plattform Dell EMC™ ECS™ wurde entwickelt, um sowohl die Kapazitäts- als auch die Verfügbarkeitsanforderungen heutiger Unternehmen zu erfüllen. ECS bietet Exabyte-Skalierbarkeit mit Unterstützung für eine global verteilte Objektinfrastruktur. Es ist so konzipiert, dass es eine hohe Verfügbarkeit für Unternehmen bietet, mit automatischer Fehlererkennung und integrierten Recovery-Optionen.

In diesem Whitepaper werden die architektonischen Details beschrieben, wie ECS die Unternehmensverfügbarkeit gewährleistet. Es enthält Einzelheiten wie:

- Wie die verteilte Infrastruktur eine höhere Systemverfügbarkeit bietet
- Erweiterte Data-Protection-Methoden, die die Haltbarkeit der Daten gewährleisten
- Wie Daten für optimale Verfügbarkeit verteilt werden
- Automatische Fehlererkennung
- Integrierte Methoden zur automatischen Fehlerkorrektur
- Details zur Behebung von Festplatten-, Node- und Netzwerkfehlern
- Disaster Recovery:
  - Schutz von ECS vor standortweiten Ausfällen
  - Wie Konsistenz in einer Aktiv-Aktiv-Konfiguration mit mehreren Standorten aufrechterhalten wird
  - Wie standortweite Ausfälle erkannt werden
  - Zugriffsoptionen während eines Standortausfalls
  - Wie die Datenbeständigkeit nach einem dauerhaften standortweiten Ausfall wiederhergestellt wird

Terminologie

Virtuelles Rechenzentrum (VDC): In diesem Whitepaper wird der Begriff Virtuelles Rechenzentrum (VDC) synonym mit Standort oder Zone verwendet. ECS-Ressourcen in einem einzigen VDC müssen Teil desselben internen Managementnetzwerks sein.

Geoverbund: Sie können die ECS-Software in mehreren Rechenzentren einsetzen, um einen Geoverbund zu schaffen. In einem Geoverbund verhält sich ECS als lose gekoppelter Verbund autonomer VDCs. Verbundstandorte umfassen die Bereitstellung von Replikations- und Managementendpunkten für die Kommunikation zwischen Standorten. Sobald Standorte verbunden sind, können sie als eine einzige Infrastruktur von jedem Node im Verbund gemanagt werden.

Replikationsgruppen: Replikationsgruppen definieren, wo Daten geschützt werden. Eine lokale Replikationsgruppe enthält ein einziges VDC und schützt Daten im selben VDC vor Festplatten- oder Node-Ausfällen. Globale Replikationsgruppen enthalten mehr als ein VDC und schützen Daten vor Festplatten-, Node- und Systemausfällen am Standort. Replikationsgruppen werden auf Bucket-Ebene zugewiesen.

1 Übersicht über das Design für hohe Verfügbarkeit

Hohe Verfügbarkeit kann in zwei Hauptbereichen beschrieben werden: Systemverfügbarkeit und Datenbeständigkeit. Ein System ist verfügbar, wenn es auf eine Clientanfrage reagieren kann. Die Datenlebensdauer wird unabhängig von der Systemverfügbarkeit bereitgestellt und bietet Gewährleistungen für Daten, die ohne Verlust oder Beschädigung im System gespeichert werden. Das bedeutet, dass die Daten auch dann geschützt sind, wenn das ECS-System ausgefallen ist (z. B. ein Netzwerkausfall).

Die verteilte Beschaffenheit der ECS-Architektur bietet Systemverfügbarkeit, da jeder Node in einem virtuellen Rechenzentrum (VDC)/Standort auf Clientanforderungen reagieren kann. Wenn ein Node ausfällt, kann der Client entweder manuell oder automatisch (z. B. mithilfe von DNS oder einem Load Balancer) zu einem anderen Node umgeleitet werden, der die Anforderung bearbeiten kann.

ECS verwendet eine Kombination aus dreifacher Spiegelung und Erasure Coding, um Daten verteilt zu schreiben, um ausfallsicher gegen Festplatten- und Node-Ausfälle zu sein. ECS unterstützt die Replikation zwischen Standorten, um die Verfügbarkeit und Ausfallsicherheit durch den Schutz vor standortweiten Ausfällen zu erhöhen. ECS umfasst außerdem regelmäßige systematischen Datenintegritätsprüfungen mit automatischer Fehlerkorrektur.

Bei der hohen Verfügbarkeit ist es zunächst wichtig, die Architektur zu verstehen und zu verstehen, wie Daten für eine optimale Verfügbarkeit und Performance in ECS verteilt werden.

1.1 Blöcke

Ein Block ist ein logischer Container, den ECS verwendet, um alle Arten von Daten zu speichern, einschließlich Objektdaten, vom benutzerdefinierten Client bereitgestellte Metadaten und ECS-Systemmetadaten. Blöcke enthalten 128 MB Daten, die aus einem oder mehreren Objekten aus einem einzigen Bucket bestehen, wie in Abbildung 1 gezeigt.

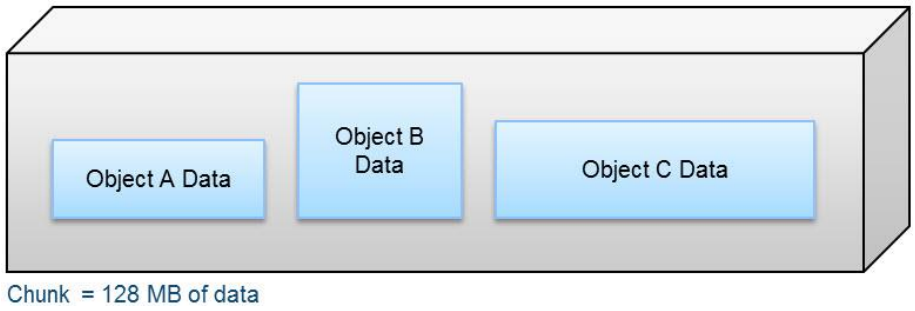

Abbildung 1 Logischer Block

ECS verwendet die Indexierung, um alle Daten in einem Block nachzuverfolgen. Weitere Details finden Sie im 1.2.

1.2 ECS-Metadaten

ECS verwaltet seine eigenen Metadaten, die nachverfolgen, wo Daten vorhanden sind, sowie den Transaktionsverlauf. Diese Metadaten werden in logischen Tabellen und Journalen verwaltet.

Die Tabellen enthalten Schlüsselwertpaare, um Informationen zu den Objekten zu speichern. Eine Hashfunktion wird verwendet, um schnelle Suchen nach Werten im Zusammenhang mit einem Schlüssel zu durchführen. Diese Schlüsselwertpaare werden in einer B+-Struktur gespeichert, um eine schnelle Indexierung der Datenpositionen zu ermöglichen. Durch die Speicherung des Schlüsselwertpaars in einem ausgeglichenen, durchsuchten Strukturformat wie einer B+-Struktur kann schnell auf den Speicherort der Daten und Metadaten zugegriffen werden. Um die Abfrageperformance dieser logischen Tabellen weiter zu verbessern, implementiert ECS außerdem eine LSM-Struktur (Log-Structured Merge) mit zwei Ebenen.

Es gibt also zwei Strukturen, wobei sich ein Teil der Struktur im Arbeitsspeicher befindet (Arbeitsspeichertabelle) und die Hauptstruktur von B+ auf der Festplatte liegt. Die Suche nach Schlüsselwertpaaren fragt zuerst die Arbeitsspeichertabelle ab. Wenn der Wert nicht im Arbeitsspeicher ist, wird er in der Hauptstruktur B+ auf der Festplatte angezeigt.

Der Transaktionsverlauf wird in Journalprotokollen aufgezeichnet und diese Protokolle werden auf Festplatten geschrieben. Die Journale verfolgen die Index-Transaktionen, die noch nicht in die B+-Struktur geschrieben wurden. Nach der Protokollierung der Transaktion in einem Journal wird die Tabelle im Arbeitsspeicher aktualisiert. Sobald die Tabelle im Arbeitsspeicher voll ist oder nach einem festgelegten Zeitraum, wird die Tabelle sortiert oder in die B+-Struktur auf der Festplatte kopiert und ein Prüfpunkt wird im Journal aufgezeichnet. Dieser Vorgang wird in Abbildung 2 dargestellt.

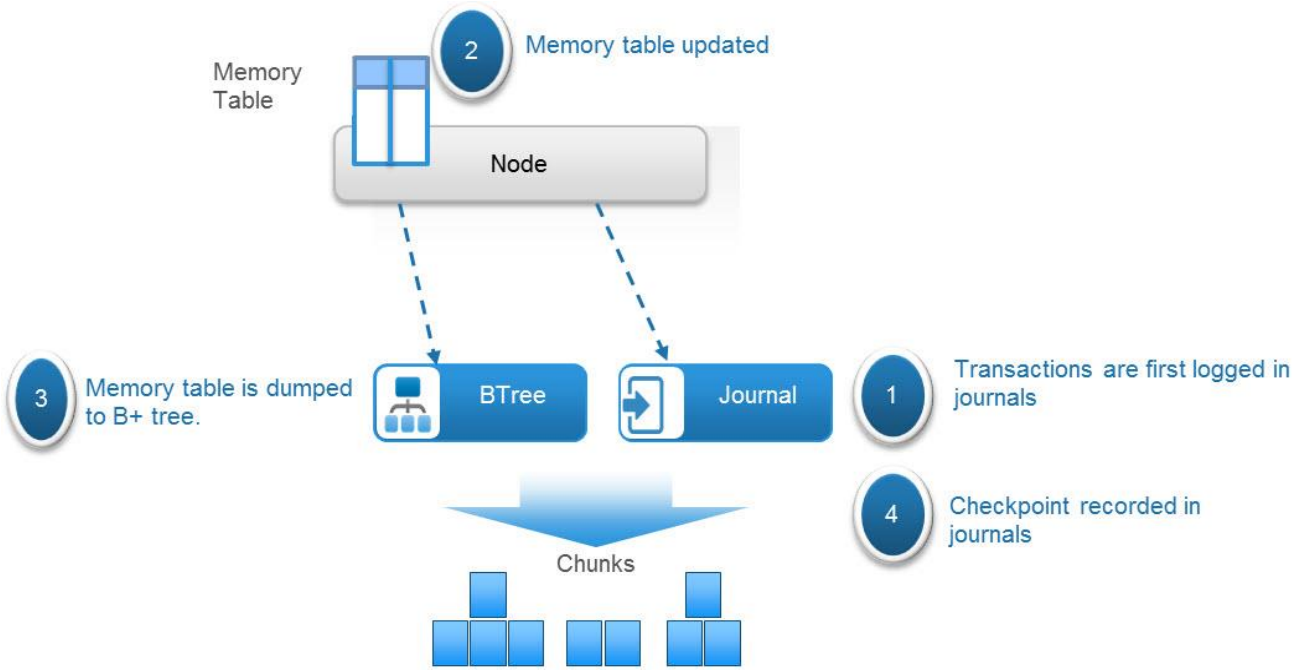

Abbildung 2 Workflow von Transaktionsaktualisierungen an ECS-Tabellen

Sowohl die Journale als auch die B+-Strukturen werden in Blöcke geschrieben.

ECS verwendet verschiedene Tabellen, von denen jede sehr groß werden kann. Um die Performance von Tabellenabfragen zu optimieren, wird jede Tabelle in Partitionen unterteilt, die über die Nodes in einem VDC/Standort verteilt sind. Der Node, auf den die Partition geschrieben wird, wird dann zum Eigentümer/zur Autorität dieser Partition oder des Abschnitts der Tabelle.

Eine solche Tabelle ist eine Blocktabelle, die den physischen Speicherort von Blockfragmenten und Replikatkopien auf der Festplatte verfolgt. Tabelle 1 zeigt ein Beispiel für eine Partition der Blocktabelle, die für jeden Block seinen physischen Speicherort identifiziert, indem die Festplatte innerhalb des Node, die Datei innerhalb der Festplatte, der Offset innerhalb dieser Datei und die Länge der Daten aufgelistet werden. Hier sehen wir, dass die Block-ID C1 löschungskodiert ist und die Block-ID C2 dreifach gespiegelt ist. Weitere Details zur dreifachen Spiegelung und Erasure Coding finden Sie im Abschnitt 1.4 dieses Dokuments.

Tabelle 1 Beispiel für eine Blocktabellenpartition

Block-ID	Blockspeicherort
C1	Node1:Disk1:file1:offset1:Length Node2:Disk1:File1:offset1:Length Node3:Disk1:File1:offset1:Length Node4:Disk1:File1:offset1:Length Node5:Disk1:File1:offset1:Length Node6:Disk1:File1:offset1:Length Node7:Disk1:File1:offset1:Length Node8:Disk1:File1:offset1:Length Node1:Disk2:File1:offset1:Length Node2:Disk2:File1:offset1:Length Node3:Disk2:File1:offset1:Length Node4:Disk2:File1:offset1:Length Node5:Disk2:File1:offset1:Length Node6:Disk2:File1:offset1:Length Node7:Disk2:File1:offset1:Length Node8:Disk2:File1:offset1:Length
C2	Node1:Disk3:File1:offset1:Length Node2:Disk3:File1:offset1:Length Node3:Disk3:File1:offset1:Length

Ein weiteres Beispiel ist eine Objekttabelle, die für die Zuordnung von Objektnamen zu Blöcken verwendet wird. Tabelle 2 zeigt ein Beispiel für eine Partition einer Objekttabelle, die angibt, welche Blöcke und wo sich ein Objekt innerhalb des Blocks befindet.

Tabelle 2 Beispielobjekttabelle

Objektname	Block-ID
ImgA	C1:offset:length
FileA	C4:offset:length C6:offset:length

Die Zuordnung von Tabellenpartitionseigentümern wird von einem Service namens vnest verwaltet, der auf allen Nodes ausgeführt wird. Tabelle 3 zeigt ein Beispiel für einen Teil einer vnest-Zuordnungstabelle.

Tabelle 3 Beispiel für vnest-Zuordnungstabelle

Tabellen-ID	Eigentümer der Tabellenpartition
Tabelle 0 P1	Node 1
Tabelle 0 P2	Node 2

1.3 Fehlerdomains

Im Allgemeinen beziehen sich Fehlerdomains auf ein Konzept des technischen Designs, das Komponenten innerhalb einer Lösung berücksichtigt, die ein Fehlerpotenzial haben. Die ECS-Software erkennt automatisch, welche Festplatten sich im selben Node befinden und welche Nodes sich im selben Rack befinden. Zum Schutz vor den meisten Ausfallszenarien ist die ECS-Software darauf ausgelegt, diese Informationen beim Schreiben von Daten zu nutzen. Die grundlegenden Richtlinien für Fehlerdomains, die ECS verwendet, umfassen Folgendes:

- ECS schreibt niemals Fragmente aus demselben Block auf dieselbe Festplatte auf einem Node.
- ECS verteilt Fragmente eines Blocks gleichmäßig über Nodes hinweg.
- Wenn ein VDC/Standort mehr als ein Rack enthält und ausreichend Platz vorhanden ist, bemüht sich ECS darum, die Fragmente eines Blocks gleichmäßig auf diese Racks zu verteilen.

1.4 Erweiterte Data-Protection-Methoden

Wenn ein Objekt in ECS erstellt wird, umfasst es das Schreiben von Daten, benutzerdefinierten Metadaten und ECS-Metadaten. ECS-Metadaten umfassen Journalblöcke und B-Struktur-Blöcke. Jeder wird in einen anderen logischen Block geschrieben, der ca. 128 MB Daten von einem oder mehreren Objekten enthält. ECS verwendet eine Kombination aus dreifacher Spiegelung und Erasure Coding, um die Daten innerhalb eines virtuellen Rechenzentrums (VDC)/Standorts zu schützen.

- Die dreifache Spiegelung sorgt dafür, dass drei Kopien von Daten geschrieben werden, wodurch der Schutz vor Ausfällen von zwei Nodes gewährleistet ist.
- Erasure Coding bietet erweiterte Data Protection vor Festplatten- und Node-Ausfällen. Es nutzt das Reed Solomon Erasure Coding-Schema, das Blöcke in Daten- und Codierungsfragmente aufteilt, die gleichmäßig über Nodes innerhalb eines VDC/Standorts verteilt sind.

Je nach Größe und Art der Daten werden sie mit einer der in Tabelle 4 dargestellten Data-Protection-Methoden geschrieben.

Tabelle 4 Bestimmen, welches Data-Protection-Level für verschiedene Arten von Daten verwendet wird

Datentyp	Verwendete Data-Protection-Methoden
Journalblöcke	Dreifache Spiegelung
B-Struktur-Blöcke/benutzerdefinierte Metadaten	Erasure Coding mit redundanten Datensegmenten
Objektdaten <128 MB	Dreifache Spiegelung plus Erasure Coding vor Ort
Objektdaten >128 MB	Inline Erasure Coding

Hinweis: In der All-Flash-Architektur wie EXF900 ist der Schutz von B-Struktur-Blöcken eine dreifache Spiegelung.

1.4.1 Dreifache Spiegelung

Die Schreibmethode der dreifachen Spiegelung gilt für die ECS-Journalblöcke, von denen ECS drei Replikatkopien erstellt. Jede Replikatkopie wird auf eine einzelne Festplatte auf verschiedenen Nodes über Fehlerdomains hinweg geschrieben. Diese Methode schützt die Blockdaten vor Ausfällen mit zwei Nodes oder zwei Festplatten.

Abbildung 3 zeigt ein Beispiel für die dreifache Spiegelung, bei der ein logischer Block, der 128 MB Metadaten enthält, drei Replikatkopien aufweist, die jeweils auf einen anderen Node geschrieben werden.

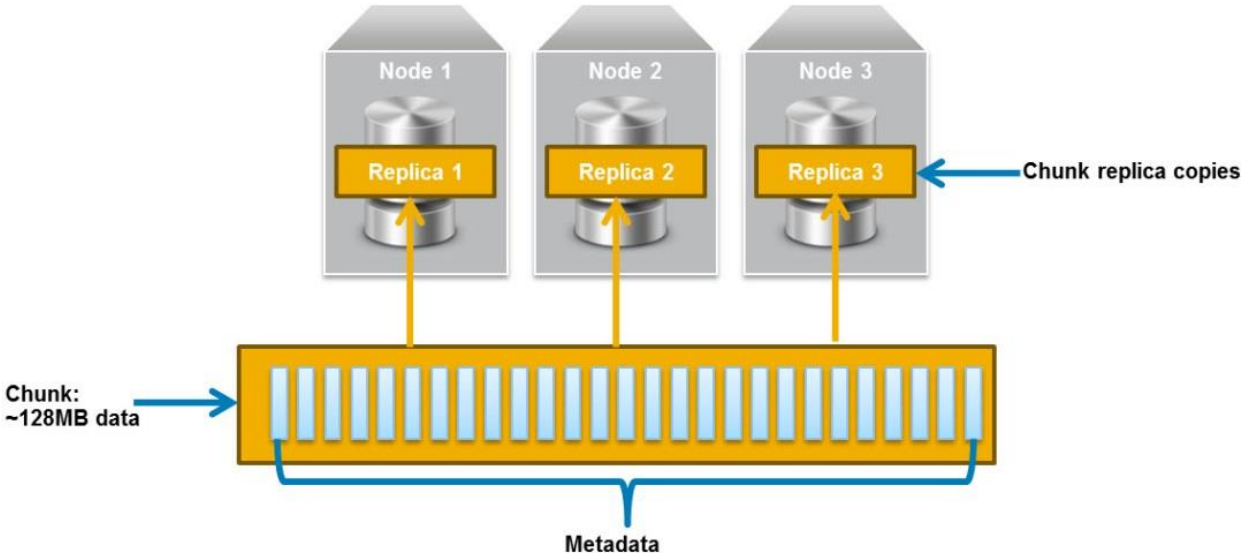

Abbildung 3 Dreifache Spiegelung

1.4.2 Erasure Coding mit redundanten Datensegmenten

Die Schreibmethode Erasure Coding mit redundanten Datensegmenten gilt für ECS-B-Struktur-Blöcke und benutzerdefinierte Objektmetadaten. Es umfasst 12 Datensegmente, 12 replizierte Datensegmente und 4 Paritätssegmente. Das neue EC-Schema für redundante Daten in B-Struktur-Blöcken, das den Metadatenschutzoverhead spart.

1.4.3 Dreifache Spiegelung plus Erasure Coding vor Ort

Diese Schreibmethode gilt für die Daten von jedem Objekt, das weniger als 128 MB groß ist.

Wenn ein Objekt erstellt wird, wird es in einen Block geschrieben, in dem ECS drei Replikatkopien wie folgt erstellt:

- Eine Kopie wird in Fragmente geschrieben, die über verschiedene Nodes und Festplatten verteilt sind. Die Verteilung verteilt die Fragmente über so viele Fehlerdomains wie möglich. Die Größe, die auf jede Festplatte geschrieben wird, hängt vom verwendeten Erasure-Coding-Schema ab.
  - Wenn das Erasure-Coding-Schema die Standardeinstellung (12+4) ist, erhält jede Festplatte maximal ca. 10,67 MB.
  - Wenn das Erasure-Coding-Schema eine Cold-Storage-Lösung (10+2) ist, erhält jede Festplatte maximal ca. 12,8 MB.
- Eine zweite Replikatkopie des Blocks wird auf eine einzelne Festplatte auf einem Node geschrieben.
- Eine dritte Replikatkopie des Blocks wird auf eine einzelne Festplatte auf einem anderen Node geschrieben.

Diese Methode bietet dreifache Spiegelung und schützt die Blockdaten vor Ausfällen mit zwei Nodes oder zwei Festplatten.

Zusätzliche Objekte werden in denselben Block geschrieben, bis sie ca. 128 MB Daten enthalten, oder nach einer vordefinierten Zeit, je nachdem, welcher Wert niedriger ist. Zu diesem Zeitpunkt berechnet das Reed Solomon Erasure Coding-Schema Codierungsfragmente (Parität) für den Block und schreibt diese auf verschiedene Festplatten. Dadurch wird sichergestellt, dass alle Fragmente in einem Block, einschließlich Codierungsfragmente, auf verschiedene Festplatten geschrieben und über Fehlerdomains verteilt werden.

Sobald die Codierungsfragmente auf die Festplatte geschrieben wurden, werden die zweite und dritte Replikatkopie von der Festplatte gelöscht. Nachdem dies abgeschlossen ist, wird der Block durch Erasure Coding geschützt, das eine höhere Verfügbarkeit als dreifache Spiegelung bietet.

1.4.4 Inline Erasure Coding

Diese Schreibmethode gilt für die Daten von jedem Objekt, das 128 MB oder größer ist. Objekte werden in 128-MB-Blöcke aufgeteilt. Das Reed Solomon Erasure Coding-Schema berechnet Coding-Fragmente (Parität) für jeden Block. Jedes Fragment wird auf verschiedene Festplatten geschrieben und über Fehlerdomains verteilt. Die Größe, die auf jede Festplatte geschrieben wird, hängt vom verwendeten Erasure-Coding-Schema ab.

- Wenn das Erasure-Coding-Schema die Standardeinstellung (12+4) ist, werden die Fragmente auf 16 Festplatten verteilt, wobei jedes Fragment ca. 10,67 MB beträgt.
- Wenn das Erasure Coding-Schema eine Cold-Storage-Lösung (10+2) ist, werden die Fragmente auf 12 Festplatten verteilt, wobei jedes Fragment ca. 12,8 MB beträgt.

Jeder verbleibende Teil eines Objekts mit weniger als 128 MB wird mithilfe des zuvor erwähnten dreifachen Spiegelung- und Erasure-Coding-Schemas vor Ort geschrieben. Beispiel: Wenn ein Objekt 150 MB beträgt, werden 128 MB mit Inline Erasure Coding geschrieben, die verbleibenden 22 MB werden mit der dreifachen Spiegelung plus Vor-Ort-Erasure Coding geschrieben.

Abbildung 4 zeigt ein Beispiel dafür, wie Blöcke über Fehlerdomains verteilt werden. Dieses Beispiel verfügt über ein einzelnes VDC/Standort, das sich über zwei Racks erstreckt, jedes Rack enthält vier Nodes.

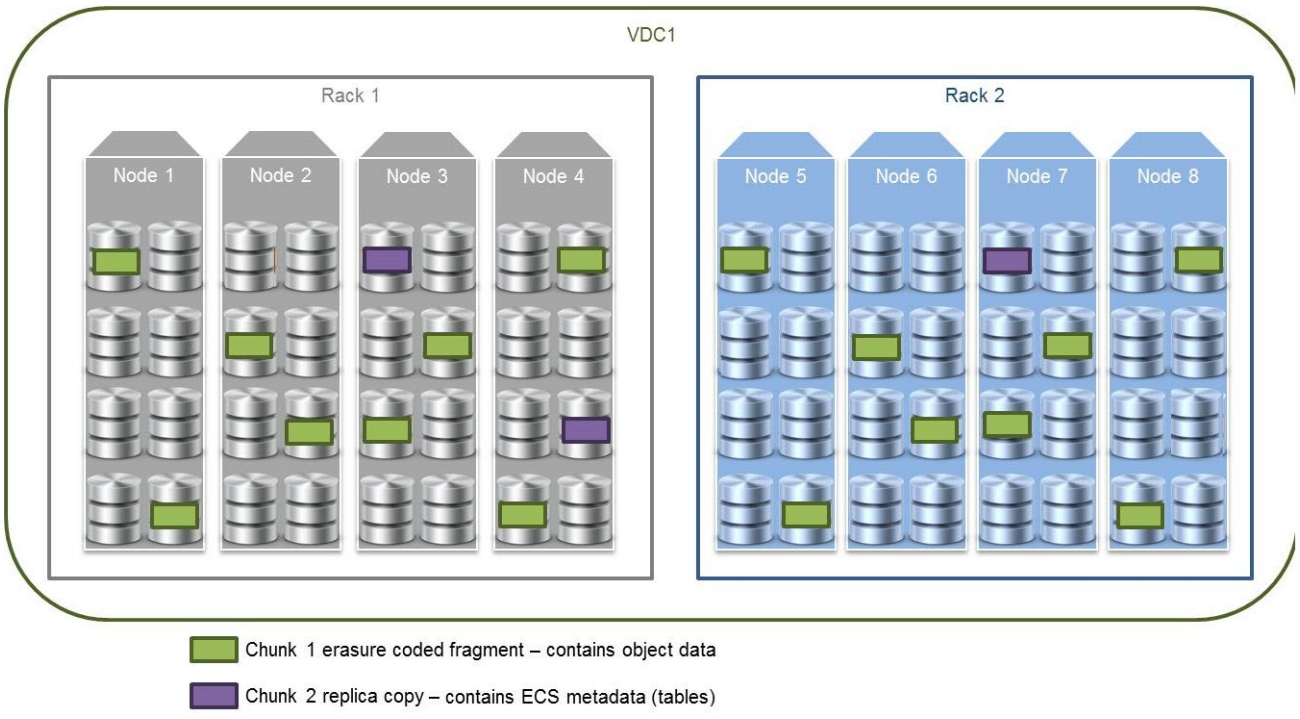

Abbildung 4 Wie Blöcke über Fehlerdomains verteilt werden

- Block 1 enthält Objektdaten, die mithilfe eines Erasure Coding von 12+4 gelöscht wurden. Fragmente sind gleichmäßig auf alle 8 Nodes verteilt, vier pro Rack. Jeder Node enthält 2 Fragmente, die es auf zwei verschiedene Festplatten schreibt.
- Block 2 enthält ECS-Metadaten (Tabellen) und wird daher dreifach gespiegelt. Jede Replikatkopie wird auf einen anderen Node geschrieben, jeweils auf einer einzelnen Festplatte. Die Kopien erstrecken sich über Racks, um die höchste Verfügbarkeit zu bieten.

1.5 Erasure-Coding-Schutzlevel

Abhängig vom Erasure-Coding-Schema, das während der Erstellung des Storage-Pools ausgewählt wurde, werden löschkodierte Daten vor den folgenden Fehlern geschützt.

1.5.1 Standard-Erasure-Coding-Schema (12 + 4):

ECS erfordert mindestens vier Nodes, um Erasure Coding mithilfe des standardmäßigen Erasure-Coding-Schemas durchführen zu können. Erasure Coding wird beendet, wenn ein Storage-Pool weniger als vier Nodes enthält, d. h., das Schutzlevel wird dreifach gespiegelt. Während dieser Zeit bleiben die drei Replikatkopien erhalten und die Parität wird nicht auf Blöcken berechnet. Sobald zusätzliche Nodes zum Storage-Pool hinzugefügt wurden und die minimale unterstützte Anzahl von Nodes erfüllt sind, wird Erasure Coding auf diesen sowie neuen Blöcken fortgesetzt.

Für jeden 128-MB-Block schreibt das standardmäßige Erasure-Coding-Schema 12 Datenfragmente und vier Codierungsfragmente, jeweils ca. 10,67 MB groß. Sie schützt die Blockdaten vor dem Verlust von bis zu vier Fragmenten eines Blocks, die die folgenden Fehlerszenarien umfassen können, die in Tabelle 5 dargestellt sind.

Tabelle 5 Standardmäßiger Erasure-Coding-Schutz

Anzahl der Nodes im VDC	Anzahl der Blockfragmente pro Node	Mit Löschcodes geschützte Daten vor
5 Nodes	4	• Ausfall von bis zu 4 Festplatten oder • Ausfall von 1 Node
6 oder 7 Nodes	3	• Ausfall von bis zu 4 Festplatten oder • Ausfall von 1 Node und 1 Festplatte aus einem 2. Node
8 oder mehr Nodes	2	• Ausfall von bis zu 4 Festplatten oder • Ausfall von 2 Nodes oder • Ausfall von 1 Node und 2 Festplatten
16 oder mehr Nodes	1	• Ausfall von 4 Nodes oder • Ausfall von 3 Nodes und Festplatten von 1 zusätzlichen Node oder • Ausfall von 2 Nodes und Festplatten von bis zu 2 anderen Nodes oder • Ausfall von 1 Node und Festplatten von bis zu 3 anderen Nodes oder • Ausfall von 4 Festplatten von 4 unterschiedlichen Nodes

Hinweis: Tabelle 5 spiegelt mögliche Schutzlevel mit vollständiger Verteilung von Blockfragmenten wider. Es kann Szenarien geben, in denen mehr Fragmente auf einem Node vorhanden sind, z. B. wenn auf einem Node nicht genügend Speicherplatz verfügbar ist. In diesem Fall können die Schutzlevel variieren.

1.5.2 Erasure-Coding-Schema für Cold-Storage-Lösung (10+2):

ECS erfordert mindestens sechs Nodes, um Erasure Coding mithilfe des Cold-Storage-Erasure-Coding-Schemas durchführen zu können. Erasure Coding wird beendet, wenn ein Storage-Pool weniger als sechs Nodes enthält, was bedeutet, dass die drei Replikatkopien verbleiben und die Parität nicht auf einem Block berechnet wird. Sobald zusätzliche Nodes zum Storage-Pool hinzugefügt wurden, wird Erasure Coding auf diesen sowie neuen Blöcken fortgesetzt.

Für jeden 128-MB-Block schreibt das Cold-Storage-Erasure-Coding-Schema zehn Datenfragmente und zwei Coding-Fragmente, jeweils ca. 12,8 MB. Sie schützt die Blockdaten vor dem Verlust von bis zu zwei Fragmenten eines Blocks, die die folgenden Fehlerszenarien umfassen können, die in Tabelle 6 dargestellt sind.

Tabelle 6 Schutz vor Erasure Coding für Cold-Storage

Anzahl der Nodes im VDC	Anzahl der Blockfragmente pro Node	Mit Löschcodes geschützte Daten vor
11 oder weniger Nodes	2	• Ausfall von bis zu 2 Festplatten oder • Ausfall von 1 Node
12 oder mehr Nodes	1	• Verlust einer beliebigen Anzahl von Festplatten von 2 verschiedenen Nodes oder • Ausfall von 2 Nodes

Hinweis: Diese Tabelle enthält mögliche Schutzlevel mit vollständiger Verteilung von Blockfragmenten. Es kann Szenarien geben, in denen mehr Fragmente auf einem Node vorhanden sind, z. B. wenn auf einem Node nicht genügend Speicherplatz verfügbar ist. In diesem Fall können die Schutzlevel variieren.

1.6 Prüfsummen

Ein weiterer Mechanismus, den ECS verwendet, um die Datenintegrität sicherzustellen, besteht darin, die Prüfsumme für geschriebene Daten zu speichern. Prüfsummen werden pro Schreibeinheit durchgeführt, bis zu 2 MB. Daher können Prüfsummen für ein Objektfragment für Schreibvorgänge großer Objekte oder auf Objektbasis für Schreibvorgänge kleiner Objekte von weniger als 2 MB auftreten. Während der Schreibvorgänge wird die Prüfsumme im Arbeitsspeicher berechnet und dann auf die Festplatte geschrieben. Bei Lesevorgängen werden die Daten zusammen mit der Prüfsumme gelesen und dann wird die Prüfsumme im Arbeitsspeicher aus dem Lesevorgang der Daten berechnet und mit der auf der Festplatte gespeicherten Prüfsumme verglichen, um die Datenintegrität zu bestimmen. Darüber hinaus führt die Storage-Engine regelmäßig eine Konsistenzprüfung im Hintergrund aus und führt eine Prüfsummenüberprüfung über das gesamte Datenvolumen durch.

1.7 Schreiben von Objekten

Wenn ein Schreibvorgang in ECS stattfindet, beginnt er mit dem Senden einer Anforderung an einen Node durch einen Client. ECS wurde als verteilte Architektur entwickelt, die es jedem Node in einem VDC/Standort ermöglicht, auf eine Lese- oder Schreibanforderung zu reagieren. Eine Schreibanforderung umfasst das Schreiben der Objektdaten, benutzerdefinierten Objektmetadaten und das Aufzeichnen der Transaktion in einem Journalprotokoll. Sobald dies abgeschlossen ist, wird dem Client eine Bestätigung gesendet.

Abbildung 5 und die oben genannten Schritte führen Sie durch eine allgemeine Übersicht über einen Schreibworkflow.

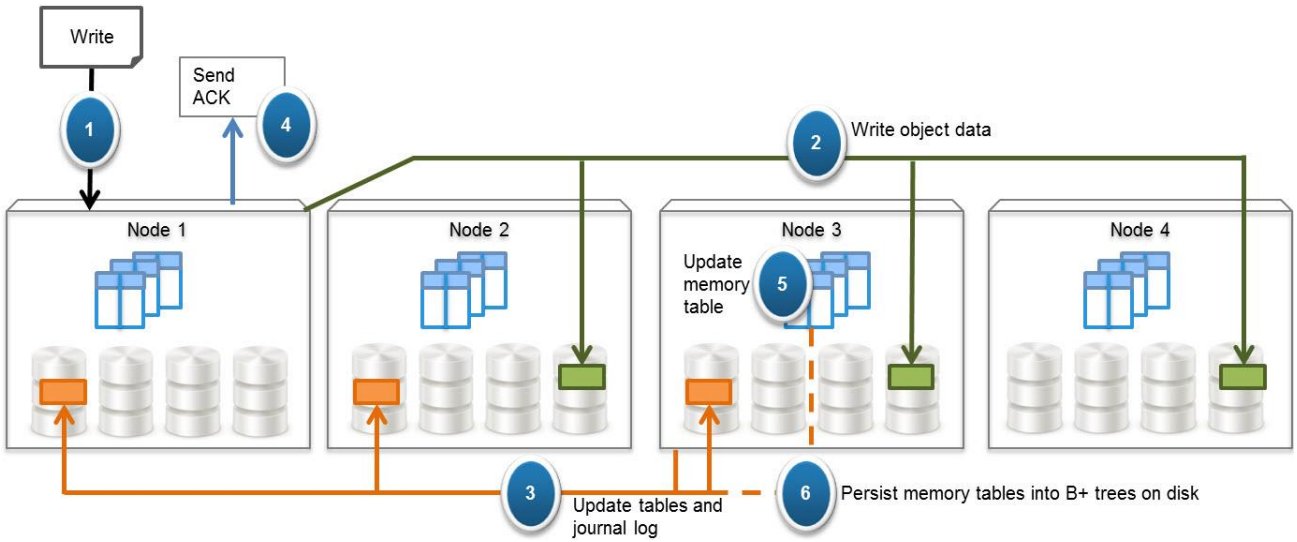

Abbildung 5 Objektschreibworkflow

1. Eine Schreibobjektanforderung wird empfangen. Jeder Node kann auf diese Anforderung reagieren, aber in diesem Beispiel verarbeitet Node 1 die Anforderung.
2. Je nach Größe des Objekts werden die Daten in einen oder mehrere Blöcke geschrieben. Jeder Block wird mithilfe erweiterter Data-Protection-Schemata wie dreifache Spiegelung und Erasure Coding geschützt. Vor dem Schreiben der Daten auf die Festplatte führt ECS eine Prüfsummenfunktion aus und speichert das Ergebnis.

   Die Daten werden zu einem Block hinzugefügt. Da dieses Objekt nur 10 MB groß ist, wird die dreifache Spiegelung zusammen mit dem Erasure-Coding-Schema verwendet. Dies führt zu Schreibvorgängen auf drei Festplatten auf drei verschiedenen Nodes, in diesem Beispiel Node 2, Node 3 und Node 4. Diese drei Nodes senden Bestätigungen zurück an Node 1.
3. Nachdem die Objektdaten erfolgreich geschrieben wurden, werden die Metadaten des Objekts gespeichert. In diesem Beispiel besitzt Node 3 die Partition der Objekttabelle, zu der dieses Objekt gehört. Als Eigentümer schreibt Node 3 den Objektnamen und die Block-ID in diese Partition der Journalprotokolle der Objekttabelle. Journalprotokolle werden dreifach gespiegelt, sodass Node 3 Replikatkopien parallel an drei verschiedene Nodes sendet, in diesem Beispiel Node 1, Node 2 und Node 3.
4. Die Bestätigung wird an den Client gesendet.
5. In einem Hintergrundprozess wird die Arbeitsspeichertabelle aktualisiert.
6. Sobald die Tabelle im Arbeitsspeicher voll ist oder nach einer bestimmten Zeit, wird die Tabelle zusammengeführt, sortiert oder in B+-Strukturen als Blöcke entsorgt, und ein Prüfpunkt wird im Journal aufgezeichnet.

1.8 Lesen von Objekten

ECS wurde als verteilte Architektur entwickelt, die es jedem Node in einem VDC/Standort ermöglicht, auf eine Lese- oder Schreibanforderung zu reagieren. Eine Leseanforderung umfasst das Auffinden des physischen Speicherorts der Daten mithilfe von Tabellenabfragen vom Eigentümer des Partitionsdatensatzes sowie Byte-Offsetlesevorgängen, Prüfsummenvalidierung und Rückgabe der Daten an den anfordernden Client.

Abbildung 6 und die vorgenannten Schritte geben eine Übersicht über den Leseworkflow.

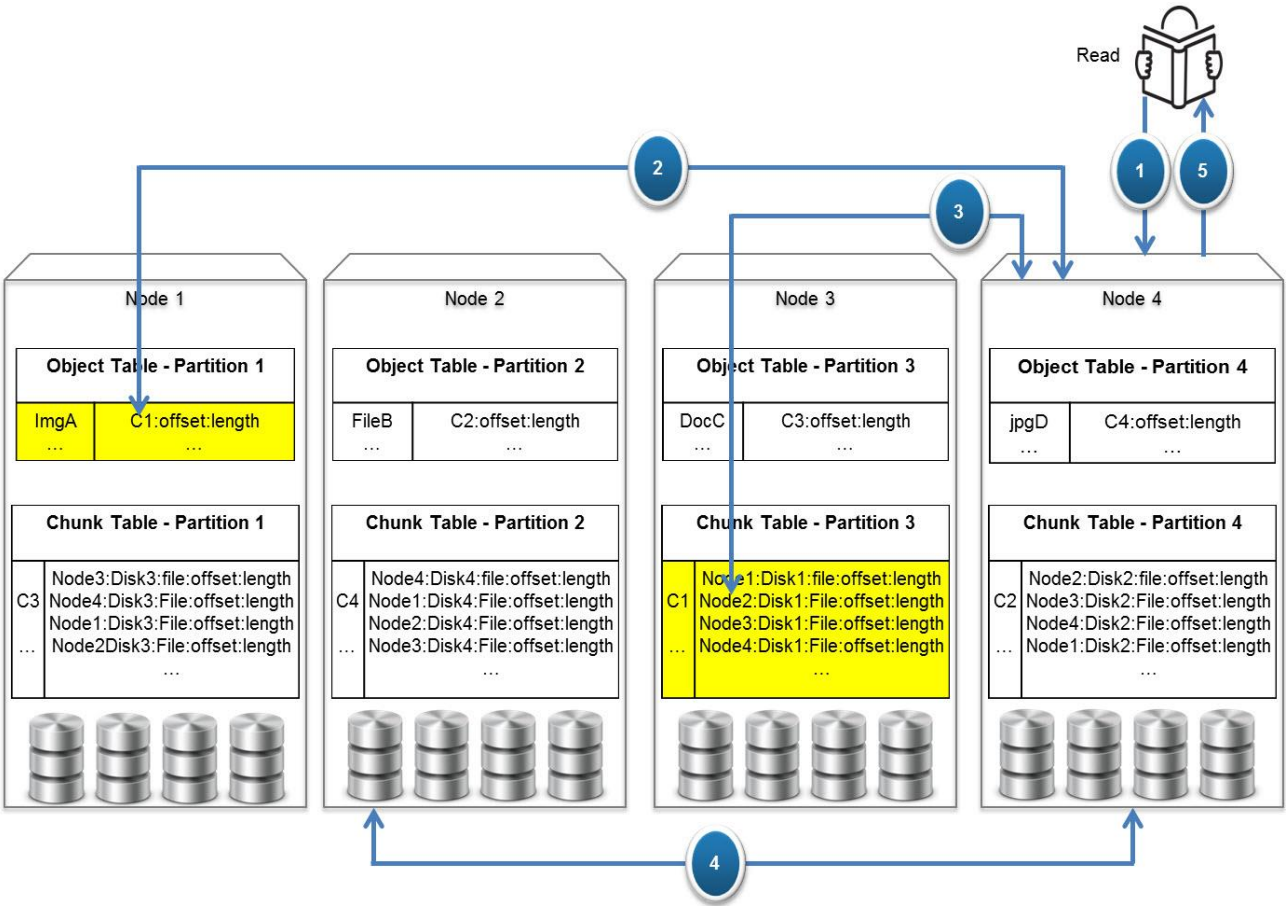

Abbildung 6 Objektleseworkflow

1. Eine Leseanforderung wird für ImgA empfangen. Jeder Node kann auf diese Anforderung reagieren, aber in diesem Beispiel verarbeitet Node 4 die Anforderung.
2. Node 4 fordert die Blockinformationen von Node 1 an (Eigentümer der Objekttabellenpartition für ImgA).
3. In dem Wissen, dass sich ImgA in C1 mit einem bestimmten Offset und einer bestimmten Länge befindet, fordert Node 4 den physischen Speicherort des Blocks von Node 3 an (Eigentümer der Blocktabellenpartition für C1).
4. Jetzt, da Node 4 den physischen Speicherort von ImgA kennt, werden diese Daten von den Nodes angefordert, die die Datenfragmente dieser Datei enthalten, in diesem Beispiel Node 2 Festplatte 1. Die Nodes führen einen Byte-Offsetlesevorgang durch und geben die Daten an Node 4 zurück.
5. Node 4 validiert die Prüfsumme und gibt dann die Daten an den anfordernden Client zurück.

Hinweis: In Schritt 4 kann jeder Node für All-Flash-Architekturen wie EXF900 Daten direkt von einem anderen Node lesen, abgesehen von der Festplattenlaufwerksarchitektur wie EX300, EX500 und EX3000, die jeder Node nur den Daten-Storage selbst lesen kann.

2 Verfügbarkeit des lokalen Standorts

Die verteilte Beschaffenheit der ECS-Architektur bietet hohe Verfügbarkeit in Form von Systemverfügbarkeit und Datenbeständigkeit gegen eine Vielzahl von Ausfällen. In diesem Abschnitt wird erläutert, wie die Verfügbarkeit bei Systemausfällen am lokalen Standort aufrechterhalten wird.

2.1 Festplattenausfall

Im Abschnitt über die Architektur wurde beschrieben, wie ECS eine Kombination aus dreifacher Spiegelung und Erasure Coding verwendet, um Daten auf verteilte Weise zu schreiben und so gegen verschiedene Ausfallszenarien abgesichert zu sein.

Um sicherzustellen, dass die Konsistenz der Datenprüfsummen nach Lesevorgängen und durch eine Konsistenzprüfung validiert wird. Die Konsistenzprüfung ist ein Hintergrundprozess, der regelmäßig Prüfsummenüberprüfungen für das gesamte Datenvolumen durchführt. Leseanforderungen führen auch eine Prüfsummenüberprüfung durch.

Wenn bei einer Leseanforderung ein Fragment fehlt, weil ein Laufwerk nicht reagiert oder die Prüfsummenüberprüfung fehlgeschlagen ist, wird eine Benachrichtigung an den Blockmanager gesendet. Der Blockmanager beginnt mit der Rekonstruktion des fehlenden Fragments bzw. der fehlenden Fragmente, wobei er entweder die verbleibenden löschungskodierten Daten- und Paritätsfragmente oder die Replikatkopien verwendet und anschließend die Blockinformationen aktualisiert. Sobald die Fragmente neu erstellt wurden, werden alle ausstehenden oder neuen Leseanforderungen die aktualisierten Blockinformationen verwenden, um die Datenfragmente anzufordern und die Leseanforderung zu bearbeiten.

ECS-Nodes führen ständig Integritätsprüfungen auf Festplatten durch, die direkt mit ihnen verbunden sind. Wenn ein Laufwerk nicht mehr reagiert, benachrichtigt der ECS-Node den Blockmanager, dass er die Aufnahme in neue Schreibvorgänge beendet. Wenn er nach einem bestimmten Zeitraum (Standard ist 60 Minuten) nicht mehr reagiert, wird eine Benachrichtigung an den Blockmanager gesendet, um die Daten vom ausgefallenen Laufwerk neu zu erstellen. Der ECS-Node erkennt, welche Blöcke sich auf dem ausgefallenen Laufwerk befinden und daher wiederhergestellt werden müssen. Diese Informationen werden an den Blockmanager gesendet, der die parallele Recovery aller auf dem ausgefallenen Laufwerk gespeicherten Blockfragmente startet. Die Blockfragmente werden mithilfe der verbleibenden fragmentierten Fragmente oder Replikatkopien auf anderen Festplatten wiederhergestellt. Wenn neue Fragmente geschrieben werden, werden die zugehörigen Blocktabellen mit diesen Informationen aktualisiert. Wenn möglich, löscht der Blockmanager auch die Fragmente des ausgefallenen Laufwerks. Wenn die Festplatte zu einem späteren Zeitpunkt wieder in Betrieb genommen wird, ist dies der Fall:

- Reagiert nicht für weniger als einen festgelegten Zeitraum (Standard ist 90 Min.), werden die verbleibenden Recovery-Vorgänge abgebrochen.
- Wenn ECS für einen festgelegten Zeitraum nicht reagiert (Standard ist 90 Minuten) oder vom Hardwaremanager als fehlgeschlagen gemeldet wird, entfernt ECS das Laufwerk. Sobald ein Laufwerk entfernt wurde, werden die verbleibenden Recovery-Vorgänge fortgesetzt, bis sie abgeschlossen sind. Wenn die Recovery abgeschlossen ist, entfernt der Blockmanager alle Verweise auf diese fehlerhafte Festplatte in der Blocktabelle.

  Wenn dieses Laufwerk nach dem Entfernen wieder online geht, wird es als neues Laufwerk hinzugefügt und der Blockmanager berücksichtigt es bei neuen Schreibvorgängen.

2.2 ECS-Node-Ausfall

ECS führt ständig Integritätsprüfungen auf Nodes durch. Um die Systemverfügbarkeit aufrechtzuerhalten, ermöglicht die verteilte ECS-Architektur jedem Node die Annahme von Clientanforderungen. Wenn ein Node ausgefallen ist, kann ein Client entweder manuell oder automatisch (z. B. mit Hilfe von DNS oder einem Load Balancer) zu einem anderen Node umgeleitet werden, der die Anforderung bearbeiten kann.

Um Rekonstruktionsvorgänge für falsche Ereignisse nicht auszulösen, wird kein vollständiger Rekonstruktionsvorgang ausgelöst, es sei denn, ein Node schlägt für eine festgelegte Anzahl sequenzieller Integritätsprüfungen fehl. Der Standardwert ist 60 Minuten. Wenn eine IO-Anforderung für einen Node eingeht, der nicht antwortet, aber bevor eine vollständige Rekonstruktion ausgelöst wird:

- Jede Anforderung für eine Partitionstabelle, die auf einem Node gehostet wird, der nicht reagiert, löst aus, dass die Eigentumsrechte der angeforderten Partitionstabellen über die verbleibenden Nodes am Standort verteilt werden. Sobald dies abgeschlossen ist, wird die Anfrage erfolgreich abgeschlossen.
- Alle IO-Anfragen für Daten, die auf Festplatten vom nicht reagierenden Node vorhanden sind, werden entweder mit den verbleibenden Erasure-Coded-Daten und Paritätsfragmenten oder den Replikatkopien rekonstruiert, danach werden die Blockinformationen aktualisiert. Sobald die Fragmente neu erstellt wurden, werden alle ausstehenden oder neuen Leseanforderungen die aktualisierten Blockinformationen verwenden, um die Datenfragmente anzufordern und die Leseanforderung zu bearbeiten.

Nachdem ein Node eine bestimmte Anzahl von aufeinanderfolgenden Integritätsprüfungen nicht bestanden hat (Standardwert: 60 Minuten), gilt ein Node als ausgefallen. Dies löst automatisch einen erneuten Erstellungsvorgang der Partitionstabellen und der Blockfragmente auf den Festplatten aus, die dem ausgefallenen Node gehören.

Im Rahmen des erneuten Erstellungsvorgangs wird eine Benachrichtigung an den Blockmanager gesendet, die eine parallele Recovery aller Blockfragmente startet, die auf den Festplatten der ausgefallenen Nodes gespeichert sind. Dies kann Blöcke umfassen, die Objektdaten, benutzerdefinierte vom Client bereitgestellte Metadaten und ECS-Metadaten enthalten. Wenn der fehlgeschlagene Node wieder online ist, wird ein aktualisierter Status an den Blockmanager gesendet und alle nicht abgeschlossenen Recovery-Vorgänge werden abgebrochen. Weitere Details zum Recovery von Blockfragmenten finden Sie oben im Abschnitt „Festplattenausfall“.

Neben der Hardwareüberwachung überwacht ECS auch alle Services und Datentabellen auf jedem Node.

- Wenn ein Tabellenfehler vorliegt, der Node aber immer noch aktiv ist, versucht er automatisch, die Tabelle auf demselben Node neu zu initialisieren.
- Wenn ein Servicefehler erkannt wird, versucht er zunächst, den Service neu zu starten.

Wenn dies fehlschlägt, verteilt es die Eigentumsrechte an den Tabellen, die dem ausgefallenen Node oder Service gehören, auf alle verbleibenden Nodes im VDC/Standort. Die Eigentumsänderungen beinhalten das Aktualisieren der vnest-Informationen und das erneute Erstellen der Arbeitsspeichertabellen, die dem ausgefallenen Node gehören. Die vnest-Informationen werden auf den verbleibenden Nodes mit neuen Eigentümerinformationen der Partitionstabelle aktualisiert.

Die Arbeitsspeichertabellen des ausgefallenen Node werden neu erstellt, indem Journaleinträge wiedergegeben werden, die nach dem letzten erfolgreichen Journalprüfpunkt geschrieben wurden.

2.2.1 Ausfälle mehrerer Nodes

Es gibt Szenarien, in denen mehrere Nodes an einem Standort ausfallen können. Mehrere Nodes können entweder einzeln oder gleichzeitig ausfallen.

- **Einzelausfall:** Wenn Nodes nacheinander ausfallen, bedeutet dies, dass ein Node ausfällt, alle Recovery-Vorgänge abgeschlossen sind und dann ein zweiter Node ausfällt. Dies kann mehrmals auftreten und ist analog zu einem VDC, das von etwa 4 Standorten → an 3 Standorten → 2 an → 1 Standort geht. Dies erfordert, dass die verbleibenden Nodes über ausreichend Speicherplatz verfügen, um Recovery-Vorgänge abzuschließen.
- **Gleichzeitiger Ausfall:** Wenn Nodes gleichzeitig ausfallen, bedeutet dies, dass Nodes fast zur gleichen Zeit ausfallen oder ein Node ausfällt, bevor die Recovery von einem vorherigen ausgefallenen Node abgeschlossen wird.

Die Auswirkungen des Ausfalls hängen davon ab, welche Nodes ausfallen. Tabelle 7 und Tabelle 8 beschreiben das Best-Case-Szenario für Fehlertoleranzen eines einzigen Standorts basierend auf dem Erasure-Coding-Schema und der Anzahl der Nodes in einem VDC.

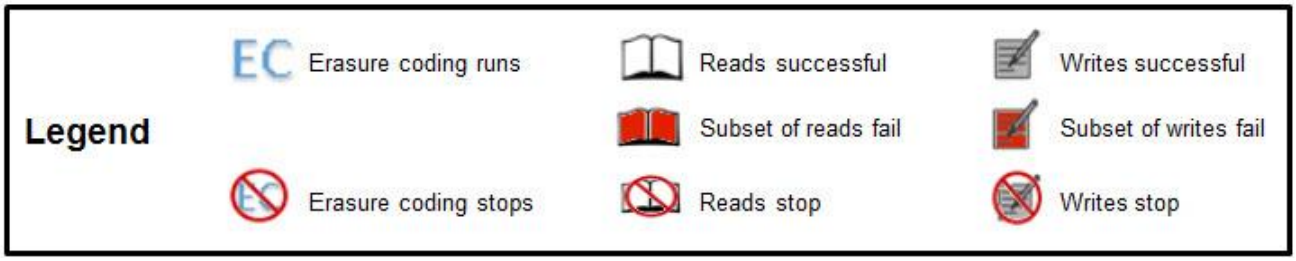

Tabelle 7 Best-Case-Szenario von mehreren Node-Ausfällen an einem Standort basierend auf dem standardmäßigen 12+4-Erasure-Coding-Schema

Anzahl der Nodes in VDC bei der Erstellung	Gesamtzahl der ausgefallenen Nodes seit VDC-Erstellung	Status nach gleichzeitigen Ausfällen	Status nach dem letzten Einzelausfall	Aktueller VDC-Status nach den vorherigen Einzelausfällen
5 Nodes	1	EC	EC	VDC mit 5 Nodes, wobei 1 Node ausfällt
	2			VDC ging zuvor von 5 → 4 Nodes, jetzt fällt 1 zusätzlicher Node aus
	3-4			VDC ging zuvor von 5 → 4 → 3 oder von 5 → 4 → 3 → 2 Nodes, jetzt fällt 1 zusätzlicher Node aus
6 Nodes	1	EC	EC	VDC mit 6 Nodes, wobei 1 Node ausfällt
	2	EC	EC	VDC ging zuvor von 6 → 5 Nodes, jetzt fällt 1 zusätzlicher Node aus
	3			VDC ging zuvor von 6 → 5 → 4 Nodes, jetzt fällt 1 zusätzlicher Node aus

Anzahl der Nodes in VDC bei der Erstellung	Gesamtzahl der ausgefallenen Nodes seit VDC-Erstellung	Status nach gleichzeitigen Ausfällen	Status nach dem letzten Einzelausfall	Aktueller VDC-Status nach den vorherigen Einzelausfällen
	4-5			VDC ging zuvor von 6 → 5 → 4 → 3 oder von 6 → 5 → 4 → 3 → 2 Nodes, jetzt fällt 1 zusätzlicher Node aus
8 Nodes	1-2			VDC mit 8 Nodes oder VDC, das von 8 → 7 Nodes ging, jetzt fällt 1 Node aus
	3-4			VDC ging zuvor von 8 → 7 → 6 oder von 8 → 7 → 6 → 5 Nodes, jetzt fällt 1 zusätzlicher Node aus
	5			VDC ging zuvor von 8 → 7 → 6 → 5 → 4 Nodes, jetzt fällt 1 zusätzlicher Node aus
	6-7			VDC ging zuvor von 8 → 7 → 6 → 5 → 4 → 3 Nodes oder von 8 → 7 → 6 → 5 → 4 → 3 → 2 Nodes, jetzt fällt 1 zusätzlicher Node aus

Tabelle 8 Best-Case-Szenario für mehrere Node-Ausfälle an einem Standort basierend auf dem 10+2-Erasure-Coding-Schema für Cold-Storage

Anzahl der Nodes in VDC bei der Erstellung	Gesamtzahl der ausgefallenen Nodes seit VDC-Erstellung	Status nach gleichzeitigen Ausfällen	Status nach dem letzten Einzelausfall	Aktueller VDC-Status nach dem vorherigen Einzelausfall
6 Nodes	1			VDC mit 6 Nodes, wobei 1 Node ausfällt
	2			VDC ging zuvor von 6 → 5 Nodes, jetzt fällt 1 zusätzlicher Node aus
	3			VDC ging zuvor von 6 → 5 → 4 Nodes, jetzt fällt 1 zusätzlicher Node aus
	4-5			VDC ging zuvor von 6 → 5 → 4 → 3 oder von 6 → 5 → 4 → 3 → 2 Nodes, jetzt fällt 1 zusätzlicher Node aus
8 Nodes	1			VDC mit 8 Nodes, wobei 1 Node ausfällt
	2			VDC ging zuvor von 8 → 7 Nodes, jetzt fällt 1 zusätzlicher Node aus

Anzahl der Nodes in VDC bei der Erstellung	Gesamtzahl der ausgefallenen Nodes seit VDC-Erstellung	Status nach gleichzeitigen Ausfällen	Status nach dem letzten Einzelausfall	Aktueller VDC-Status nach dem vorherigen Einzelausfall
	3-5			VDC ging zuvor von 8 → 7 → 6 oder von 8 → 7 → 6 → 5 oder von 8 → 7 → 6 → 5 → 4 Nodes, jetzt fällt 1 zusätzlicher Node aus
	6-7			VDC ging zuvor von 8 → 7 → 6 → 5 → 4 → 3 Nodes oder von 8 → 7 → 6 → 5 → 4 → 3 → 2 Nodes, jetzt fällt 1 zusätzlicher Node aus
12 Nodes	1-2			VDC mit 12 Nodes oder VDC ging zuvor von 12 → 11 Nodes, jetzt fällt 1 zusätzlicher Node aus
	3-6			VDC ging zuvor von 12→ 11 → 10 oder von 12→ 11 → 10 → 9 oder von 12→ 11 → 10 → 9 → 8 oder von 12→ 11 → 10 → 9 → 8 → 7 Nodes, jetzt fällt 1 zusätzlicher Node aus
	7-9			VDC ging zuvor von 12→ 11 → 10 → 9 → 8 → 7 → 6 oder von 12→ 11→ 10→ 9→ 8→ 7→ 6→ 5 oder von 12→ 11→ 10→ 9→ 8→ 7→ 6→ 5→ 4 Nodes, jetzt fällt 1 zusätzlicher Node aus
	10-11			VDC ging zuvor von 12→11→10→9→8→7→6→5→4→3 oder von 12→11→10→9→8→7→6→5→4→3→ 2 Nodes, jetzt fällt 1 zusätzlicher Node aus

Die grundlegenden Regeln für die Bestimmung, welche Vorgänge an einem einzigen Standort mit mehreren Node-Ausfällen fehlschlagen, sind:

- Wenn drei oder mehr gleichzeitige Node-Ausfälle auftreten, schlagen einige Lese- und Schreibvorgänge aufgrund des potenziellen Verlusts aller drei Replikatkopien der zugehörigen drei gespiegelten Metadatenblöcke fehl.
- Für Schreibvorgänge sind mindestens drei Nodes erforderlich.
- Erasure Coding wird beendet und Blöcke mit Erasure Coding werden in dreifachen Spiegelungsschutz konvertiert, wenn die Anzahl der Nodes kleiner als die für jedes Erasure-Coding-Schema erforderliche Mindestanzahl ist. Da für das standardmäßige Erasure-Coding-Schema 12+4 4 Nodes erforderlich sind, wird Erasure Coding beendet, wenn weniger als 4 Nodes vorhanden sind. Bei Cold-Storage-Erasure-Coding 10+2 wird Erasure Coding beendet, wenn weniger als 6 Nodes vorhanden sind.

- Wenn die Anzahl der Nodes unter das Minimum für das Erasure-Coding-Schema fällt, werden Blöcke mit Erasure Code in dreifachen Spiegelungsschutz konvertiert. Beispiel: In einem VDC mit standardmäßigen Erasure Coding und 4 Nodes würde nach einem Node-Ausfall Folgendes passieren:
  - Ein Node-Ausfall führt dazu, dass 4 Fragmente verloren gehen.
  - Fehlende Fragmente werden neu erstellt.
  - Block erstellt 3 Replikatkopien, eine auf jedem Node.
  - EC-Kopie wird gelöscht.
- Einzelausfälle fungieren wie Ausfälle einzelner Nodes. Wenn Sie beispielsweise 2 Nodes nacheinander verlieren, besteht jeder Ausfall nur darin, Daten nach dem Ausfall eines einzelnen Node wiederherzustellen.

  Beispiel: Mit 6 Nodes und standardseitigem Erasure Coding:

  - Erster Ausfall: Jeder der 6 Nodes hat bis zu drei Fragmente (16 Fragmente/6 Nodes). Die fehlenden drei Fragmente werden auf den verbleibenden Nodes neu erstellt. Nach Abschluss der Recovery endet das VDC mit 5 Nodes.
  - Zweiter Ausfall: Jeder der 5 Nodes hat bis zu 4 Fragmente (16 Fragmente/5 Nodes). Die fehlenden 4 Fragmente werden auf den verbleibenden Nodes neu erstellt. Nach Abschluss der Recovery endet das VDC mit 4 Nodes.
  - Dritter Ausfall: Jeder der 4 Nodes hat bis zu 4 Fragmente (16 Fragmente/4 Nodes). Die fehlenden 4 Fragmente werden auf den verbleibenden Nodes neu erstellt. Nach Abschluss der Recovery endet das VDC mit 3 Nodes und da dies unter dem Minimum für Erasure Coding liegt, wird der Erasure-Code-Block durch drei Replikatkopien ersetzt, die über die verbleibenden Nodes verteilt sind.
  - Vierter Ausfall: Jeder der 3 Nodes verfügt über eine Replikatkopie. Die fehlende Replikatkopie wird auf einem der verbleibenden Nodes neu erstellt. Nach Abschluss der Recovery endet das VDC mit 2 Nodes.
  - Fünfter Ausfall: Es gibt 3 Replikatkopien, 2 auf einem Node und eine auf dem anderen Node. Die fehlenden Replikatkopien werden auf dem verbleibenden Node neu erstellt. Nach Abschluss der Recovery endet das VDC mit 1 Node.

3 Übersicht über das Design an mehreren Standorten

Neben der Systemverfügbarkeit und Datenbeständigkeit, die an einem einzigen Standort entwickelt wurden, bietet ECS auch Schutz vor einem vollständigen standortweiten Ausfall. Dies wird in einer Bereitstellung mit mehreren Standorten erreicht, indem mehrere VDCs/Standorte miteinander verbunden und die Georeplikation konfiguriert wird.

Verbundstandorte umfassen die Bereitstellung von Replikations- und Managementendpunkten für die Kommunikation zwischen Standorten. Sobald Standorte verbunden sind, können sie als eine einzige Infrastruktur verwaltet werden.

Replikationsgruppenrichtlinien bestimmen, wie Daten geschützt werden und von wo aus darauf zugegriffen werden kann. ECS unterstützt sowohl die aktive Georeplikation als auch die passive Georeplikation. Die geoaktive Replikation bietet Aktiv-Aktiv-Zugriff auf Daten, sodass sie von jedem Standort innerhalb der definierten Replikationsgruppe gelesen und geschrieben werden können.

Bei der standortübergreifenden Replikation von All-Flash-Serien wie EXF900 sollten Sie die potenziellen Performanceauswirkungen auf das WAN berücksichtigen. Eine große Aufnahme kann eine hohe Last auf den Link verursachen, was zu Sättigung oder verzögerter RPO führt. Außerdem kann es bei Remotelesevorgängen und Remoteschreibvorgängen zu höheren Latenzzeiten für NutzerInnen/eine Anwendung im Vergleich zu lokalen Anforderungen kommen. Der andere sollte die partielle automatische Speicherbereinigung als fehlgeschlagen betrachten. Eine große Aufnahme sowohl vom lokalen als auch vom replizierten Standort kann dazu führen, dass das System bald 90 % erreicht, was dazu führt, dass das System Daten nicht mehr schreibt und Daten zurückgewinnt.

Hinweis: Partielle automatische Speicherbereinigung – Wenn ein Block zu 2/3 aus veralteten Objekten besteht, wird der Block zurückgewonnen, indem die gültigen Teile mit anderen teilweise gefüllten Blöcken zu einem neuen Block zusammengeführt werden, um Speicherplatz zurückzugewinnen.

Die Replikation kann auch als geo-passiver Standort konfiguriert werden, der zwei bis vier Quellstandorte und einen oder zwei Standorte angibt, die nur als Replikationsziel verwendet werden sollen. Die Replikationsziele werden nur für Recovery-Zwecke verwendet. Replikationsziele blockieren den direkten Client-Zugriff für Erstellungs-/Aktualisierungs-/Löschvorgänge.

Zu den Vorteilen der geo-passiven Replikation gehören:

- Es kann die Storage-Effizienz optimieren, indem die Wahrscheinlichkeit von XOR-Vorgängen erhöht wird, indem sichergestellt wird, dass Schreibvorgänge von beiden Quellstandorten zum gleichen Replikationsziel gehen.
- Es ermöglicht den AdministratorInnen zu steuern, wo die Replikationskopie von Daten vorhanden ist, z. B. in einem Backup-in-Cloud-Szenario.

ECS bietet Konfigurationsoptionen für die Georeplikation auf Bucket-Ebene, sodass die AdministratorInnen verschiedene Replikationsebenen für verschiedene Buckets konfigurieren können.

Abbildung 7 zeigt ein Beispiel, wie AdministratorInnen die Replikation von drei Buckets konfigurieren können:

- Bucket A: Engineering-Test-/Entwicklungsdaten – nicht replizieren, nur lokal behalten
- Bucket B: Europäische Vertriebsdaten – Replikation zwischen Standorten nur innerhalb Europas
- Bucket C: unternehmensweite Schulungsdaten – Replizieren an alle Standorte innerhalb des Unternehmens

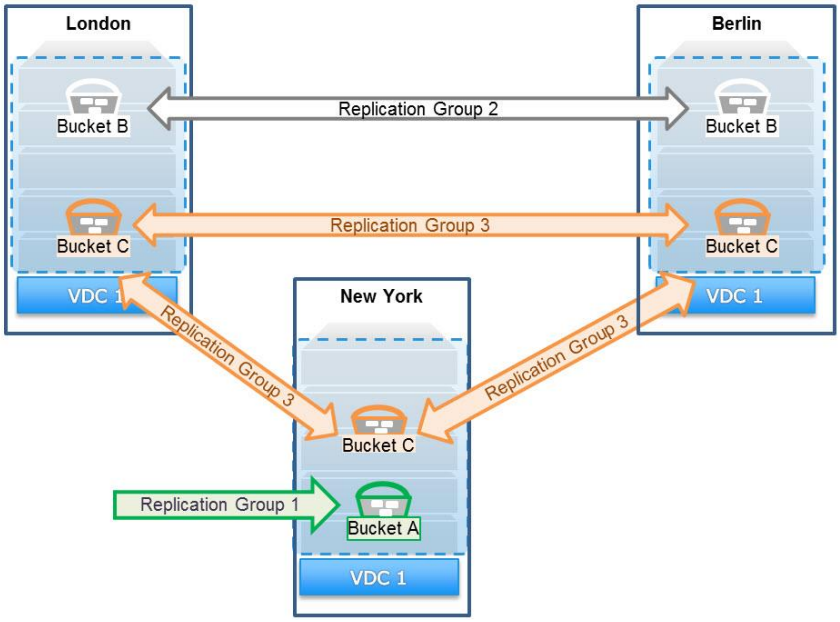

Abbildung 7 Beispiel, das zeigt, wie verschiedene Buckets unterschiedliche Replikationsrichtlinien verwenden können

Als Best Practice wird empfohlen, Replikationsgruppen für bestimmte Replikationspfade zu konfigurieren. Beispielsweise gibt es eine Abbildung 7 Replikationsgruppe, die Daten zwischen London und Berlin repliziert. Dies sollte für alle Buckets verwendet werden, die die Replikation nur zwischen London und Berlin benötigen.

Geografisch replizierte Daten werden geschützt, indem eine Primärkopie der Daten am lokalen Standort und eine sekundäre Kopie der Daten an einem oder mehreren Remotestandorten gespeichert werden. Die Anzahl der Kopien und die Menge des Speicherplatzes, den die sekundäre Kopie belegt, wird basierend auf der Anzahl der in der Replikationsgruppe konfigurierten Standorte, der Art und Weise, wie Daten über Standorte geschrieben werden, sowie davon bestimmt, ob **auf alle Standorte replizieren** aktiviert ist oder nicht.

Jeder Standort ist für die lokale Data Protection verantwortlich, was bedeutet, dass sowohl die lokalen als auch die sekundären Kopien die Daten einzeln mit Erasure Coding und/oder dreifacher Spiegelung schützen. Die Erasure-Coding-Schemata an jedem Standort müssen nicht identisch sein, d. h. ein Standort kann das standardmäßige Erasure-Coding-Schema von 12+4 und der andere Standort das Cold-Storage-Erasure-Coding-Schema von 10+2 verwenden.

Replizierte Daten werden verschlüsselt (AES256) und komprimiert, bevor sie über HTTP an den anderen Standort gesendet werden.

Um die Konsistenz zwischen Standorten aufrechtzuerhalten, muss es eine Autorität geben, die für die Pflege der neuesten Version von Metadaten verantwortlich ist. Die Autorität wird auf Standortebene definiert und bestimmt die Eigentumsrechte für Namespaces, Buckets und Objekte. Ownership-Informationen werden zuerst am Eigentümerstandort gespeichert, aber auch als Teil der ECS-Metadaten an die anderen Standorte repliziert.

- Autorisierende Version: Die maßgebliche Version ist immer der Eigentümer und wird verwendet, um eine starke Konsistenz zu gewährleisten.
- Replizierte Version: Die replizierte(n) Version(en) ist (sind) nicht unbedingt die Neueste(n), wird (werden) aber bei Fehleroperationen verwendet:
  - Wenn **der Zugriff während eines Ausfalls** aktiviert ist (letztendliche Konsistenz).
  - Und während Failover-Vorgängen an einem permanenten Standort.

Es gibt autorisierende Versionen der Bucket- und Objekteigentümer.

Namespace-Eigentümer:

- Der Standort, der den Namespace erstellt, ist der Namespace-Eigentümer.
- Es ist für die Pflege der maßgeblichen Version der Bucket-Liste verantwortlich.

Bucket-Eigentümer:

- Der Standort, an dem der Bucket erstellt wird, ist der Bucket-Eigentümer.
- Es ist für die Aufrechterhaltung der maßgeblichen Version verantwortlich:
  - Die Bucket-Liste, die die neuste Version enthält, deren Objekte sich in einem Bucket befinden.
  - Die Auflistung der Objekteigentumsrechte für Objekte innerhalb des Buckets

Objekteigentümer:

- Zunächst ist der Standort, an dem das Objekt erstmals erstellt wurde, der Objekteigentümer. Dies kann sich ändern. Weitere Informationen finden Sie im Abschnitt „Zugriff während eines Ausfalls“.
- Es ist für die Aufrechterhaltung der maßgeblichen Version der Objektmetadaten verantwortlich.

3.1 Blockmanagertabellen

Blockspeicherorte werden in der Blockmanagertabelle beibehalten, die unabhängig an allen Standorten gespeichert wird. Der Speicherort, an dem ein Block ursprünglich erstellt wurde, wird als primärer Standort bezeichnet. Der Speicherort, an dem er repliziert wird, wird als sekundärer Standort bezeichnet.

Wenn ein Block erstellt wird, werden der primäre und der sekundäre Standort bestimmt und der primäre Standort überträgt die Standortinformationen der Blöcke an die anderen Nodes in der Replikationsgruppe.

Da jeder Standort seine eigene Blockmanagertabelle verwaltet, enthält er außerdem Informationen über den Eigenschaftstyp für jeden Block. Eigenschaftstypen umfassen:

- **Lokal:** an dem Standort, an dem der Block erstellt wurde
- **Copy:** an dem Standort, an dem der Block repliziert wurde
- **Remote:** an den Standorten, an denen weder der Block noch das Replikat lokal gespeichert werden
- **Parität:** auf Blöcken, die das Ergebnis eines XOR-Vorgangs anderer Blöcke enthalten (weitere Informationen finden Sie im Abschnitt XOR unten)
- **Kodiert:** auf Blöcken, deren Daten lokal durch XOR-Daten ersetzt wurden (weitere Informationen finden Sie im Abschnitt XOR unten)

Tabelle 9 bis Tabelle 11 zum Anzeigen von Beispielabschnitten der Blockmanagertabellenauflistungen von jedem von drei Standorten.

Tabelle 9 Beispieltabelle für Blockmanager von Standort 1

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Lokale
C2	Standort 2	Standort 3	Remote
C3	Standort 1	Standort 3	Lokale
C4	Standort 2	Standort 1	Kopie

Tabelle 10 Beispieltabelle für Blockmanager von Standort 2

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Kopie
C2	Standort 2	Standort 3	Lokale
C3	Standort 1	Standort 3	Remote
C4	Standort 2	Standort 1	Lokale

Tabelle 11 Beispieltabelle für Blockmanager von Standort 3

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Remote
C2	Standort 2	Standort 3	Kopie
C3	Standort 1	Standort 3	Kopie
C4	Standort 2	Standort 1	Remote

3.2 XOR-Codierung

Um die Storage-Effizienz von Daten zu maximieren, die mit einer Replikationsgruppe mit drei oder mehr Standorten konfiguriert sind, verwendet ECS die XOR-Codierung. Wenn die Anzahl der Standorte in einer Replikationsgruppe zunimmt, ist der ECS-Algorithmus effizienter bei der Reduzierung des Overheads.

Die XOR-Codierung wird an jedem Standort durchgeführt. Es scannt seine Blockmanagertabelle und wenn er COPY-Blöcke findet, die von jedem der anderen Standorte in der Replikationsgruppe stammen, kann er XOR-Codierung für diese Blöcke durchführen. Beispiel: In Tabelle 12 wird Standort 3 einer Konfiguration mit drei Standorten mit den Blöcken **C2** und **C3** angezeigt, die jeweils vom Typ COPY mit einem anderen primären Standort sind. So kann Standort 3 sie miteinander XOR-verknüpfen und das Ergebnis speichern. Das Ergebnis ist ein neuer Block, **C5,** der ein XOR von **C2** und **C3** (rechnerisch C2 $\oplus$ C3) ist und einen Typ hat, der als **Parität** ohne einen sekundären Standort aufgeführt ist. Die Block-IDs von Paritätsblöcken werden nicht an andere Standorte übertragen.

Tabelle 12 zeigt ein Beispiel für eine Blockmanagertabelle an Standort 3, während XOR der Blöcke **C2** und **C3** zusammen in Block **C5** ausgeführt wird.

Tabelle 12 Standort 3 Blockmanagertabelle während der XOR-Operation

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Remote
C2	Standort 2	Standort 3	Kopie
C3	Standort 1	Standort 3	Kopie
C4	Standort 2	Standort 1	Remote
C5	Standort 3		Parität (C2 und C3)

Nachdem XOR abgeschlossen ist, werden die Datenkopien für **C2** und **C3** gelöscht, sodass Speicherplatz auf der Festplatte freigegeben wird, und der Blockmanagertabellentyp für diese Blöcke ändert sich in den Typ „Encoded“. Der XOR-Vorgang ist ein rein sekundärer Standortvorgang, der primäre Standort ist sich nicht bewusst, dass seine Blöcke kodiert wurden. Nachdem die XOR-Codierung abgeschlossen und die Datenkopie für **C2** und **C3** gelöscht wurde, wird die Blockmanagertabelle von Standort 3 aufgelistet, wie in Tabelle 13 gezeigt.

Tabelle 13 Standort 3 Blockmanagertabelle nach Abschluss der XOR-Kodierung von C2 und C3

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Remote
C2	Standort 2	Standort 3	Kodiert
C3	Standort 1	Standort 3	Kodiert
C4	Standort 2	Standort 1	Remote
C5	Standort 3		Parität (C2 und C3)

Anfragen nach Daten in einem kodierten Block werden von dem Standort bedient, der die Primärkopie enthält. Weitere Informationen zur Storage-Effizienz finden Sie im Whitepaper „ECS-Übersicht und -Architektur“.

3.3 Auf alle Standorte replizieren

„Auf alle Standorte replizieren“ ist eine Replikationsgruppenoption, die verwendet wird, wenn Sie drei oder mehr Standorte haben und möchten, dass alle Blöcke auf alle VDCs/Standorte repliziert werden, die in der Replikationsgruppe konfiguriert sind. Dies verhindert auch die Ausführung von XOR-Vorgängen. Wenn die Option **Auf alle Standorte replizieren** lautet:

- **Aktiviert:** die Anzahl der geschriebenen Kopien der Daten entspricht der Anzahl der Standorte in der Replikationsgruppe. Wenn Sie beispielsweise vier Standorte in der Replikationsgruppe haben, haben Sie eine Primärkopie plus drei sekundäre Kopien. eine an jedem Standort.
- **Deaktiviert:** die Anzahl der geschriebenen Datenkopien beträgt zwei. Sie haben die Primärkopie plus eine replizierte Kopie an einem Remotestandort, unabhängig von der Gesamtzahl der Standorte.

Hinweis: die Replikation auf alle Standorte kann bei einer als geopassiv konfigurierten Replikationsgruppe nicht aktiviert werden.

Diese Einstellung hat keine Auswirkungen auf Replikationsgruppen, die nur zwei Standorte enthalten, da im Wesentlichen bereits alle Daten auf beide Standorte repliziert werden. Die AdministratorInnen können auswählen, welche Buckets diese Replikationsgruppe verwenden.

Die Aktivierung der Replikation auf alle Standorte hat die folgenden Auswirkungen:

- Kann die Leseperformance verbessern, da nach Abschluss der Replikation nachfolgende Lesevorgänge lokal gewartet werden.
- Beseitigung von Performancebeeinträchtigungen durch XOR-Dekodierung.
- Erhöht die Datenlebensdauer.
- Verringert die Speicherauslastungseffizienz.
- Erhöht die WAN-Auslastung für die Georeplikation. Der Anstieg ist proportional zur Anzahl der VDCs/Standorte in der Replikationsgruppe.
- Verringert die WAN-Auslastung für Lesevorgänge replizierter Daten.

Aus diesen Gründen wird die Aktivierung dieser Option nur für bestimmte Buckets in Umgebungen empfohlen, die die folgenden Kriterien erfüllen:

- Deren Workload für dieselben Daten von geografisch verteilten Standorten leseintensiv ist.
- Deren Infrastruktur über eine ausreichende WAN-Bandbreite zwischen Standorten in der Replikationsgruppe verfügt, um einen erhöhten Geo-Replikationsdatenverkehr zu ermöglichen.
- Die mehr Wert auf die Leseperformance als auf die Effizienz der Speicherauslastung legen.

3.4 Schreibdatenfluss in geografisch replizierter Umgebung

Blöcke enthalten 128 MB Daten, die aus einem oder mehreren Objekten aus Buckets bestehen, die dieselben Replikationsgruppeneinstellungen gemeinsam nutzen. Die Replikation wird asynchron durchgeführt und vom Eigentümer der Blockpartition auf Blockebene initiiert. Wenn der Block mit Georeplikationsdaten konfiguriert ist, wird er einer Replikationswarteschlange hinzugefügt, während er in den Block des primären Standorts geschrieben wird, und wartet nicht, bis der Block versiegelt wird. Es gibt Worker-I/O-Threads, die die Warteschlange kontinuierlich verarbeiten.

Der Schreibvorgang erfolgt zuerst lokal, einschließlich hinzufügen von Data Protection, und dann wird er am Remotestandort repliziert und geschützt. Abbildung 8 zeigt ein Beispiel für den Schreibvorgang für ein 128-MB-Objekt in einen geografisch replizierten Bucket.

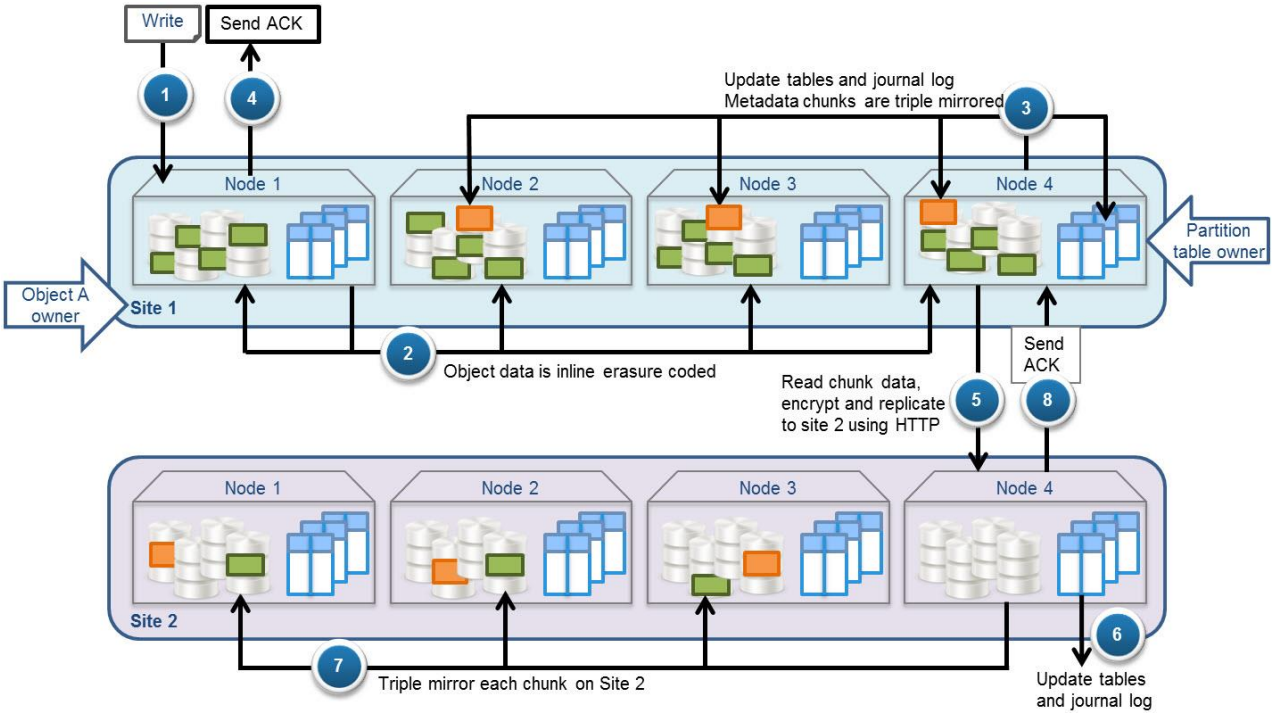

Abbildung 8 Datenworkflow für ein 128-MB-Objekt in einen geografisch replizierten Bucket schreiben

1. Die Schreibanforderung für Objekt A wird an einen Node gesendet, in diesem Beispiel Standort 1 Node 1. Standort 1 wird zum Eigentümer von Objekt A.
2. Daten werden mit Inline-Erasure-Code kodiert und in einen Block an Standort 1 geschrieben.
3. Eigentümer der Tabellenpartition, in diesem Beispiel Node 4, aktualisieren die entsprechenden Tabellen (z. B. Block-, Objekt- und Bucket-Auflistungstabellen) und schreiben die Transaktionen in die Journalprotokolle. Diese Metadaten werden in einen Metadatenblock geschrieben, der an Standort 1 dreifach gespiegelt wird.
4. Die Bestätigung des erfolgreichen Schreibens wird an den Client gesendet.
5. Für jeden Block muss der Eigentümer der Blockpartitionstabelle in diesem Beispiel Node 4:

a. die Daten innerhalb des Blocks der Replikationswarteschlange hinzufügen, nachdem sie lokal geschrieben wurden. Es wartet nicht, bis der Block versiegelt ist.
b. die Datenfragmente des Blocks lesen (Paritätsfragmente werden nur gelesen, wenn dies erforderlich ist, um ein fehlendes Datenfragment erneut zu erstellen).
c. die Daten über HTTP an Standort 2 verschlüsseln und replizieren.

6. Tabellenpartitionseigentümer für die replizierten Blöcke, in diesem Beispiel Standort 2 Node 4, aktualisieren die entsprechenden Tabellen und schreiben die Transaktionen in die Journalprotokolle, die dreifach gespiegelt werden.
7. Jeder replizierte Block wird anfänglich am zweiten Standort mithilfe von dreifacher Spiegelung geschrieben.
8. Die Bestätigung wird zurück an den Eigentümer der Blockpartitionstabelle des primären Standorts gesendet.

Hinweis: Daten, die auf den replizierten Standort geschrieben werden, werden nach einer Verzögerung mit Erasure Coding kodiert, sodass andere Prozesse, z. B. XOR-Vorgänge, zuerst abgeschlossen werden können.

3.5 Lesedatenfluss in geografisch replizierter Umgebung

Da ECS Daten asynchron auf mehrere VDCs innerhalb einer Replikationsgruppe repliziert, ist eine Methode erforderlich, um die Konsistenz der Daten über Standorte/VDCs hinweg zu gewährleisten. ECS sorgt für eine starke Konsistenz, indem die neueste Kopie der Metadaten von dem Standort abgerufen wird, der der Objekteigentümer ist. Wenn der anfordernde Standort eine Kopie (Blocktyp = lokal oder Kopie) des Objekts enthält, verwendet er dies, um die Leseanforderung zu bedienen, andernfalls werden die Daten vom Objekteigentümer abgerufen. Ein Beispiel, das den Lesedatenfluss in Abbildung 9 zeigt, die eine Leseanforderung für Objekt A von einem anderen Standort als dem Standort des Objekteigentümers darstellt.

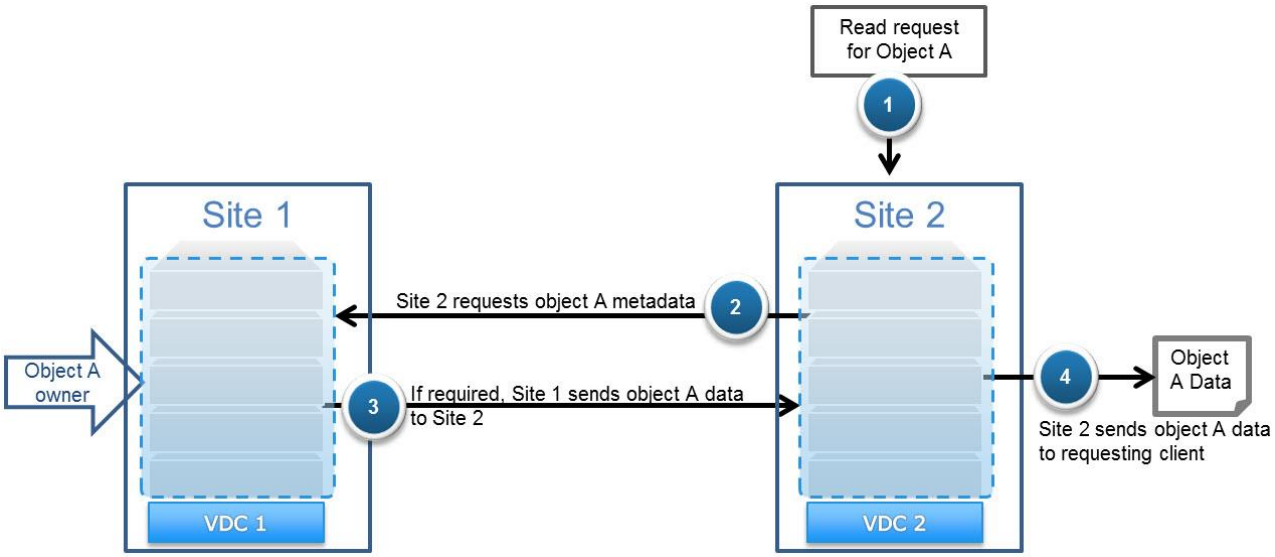

Abbildung 9 Lesedatenworkflow für ein geografisch repliziertes Objekt, das einem anderen Standort gehört

In diesem Beispiel wird der Ablauf des Lesevorgangs wie folgt dargestellt:

1. Standort 2 erhält eine Leseanforderung für Objekt A, das Standort 1 gehört.
2. Standort 2 kontaktiert den Bucket- und Objekteigentümer, in diesem Beispiel Standort 1, um die neueste Version der Metadaten zu erhalten.
   Objekteigentumsrechte:
   - Wenn **der Zugriff während eines Ausfalls** deaktiviert ist, überprüft es seine lokalen Informationen, um festzustellen, ob es der Objekteigentümer ist. Ist dies nicht der Fall, wird der Bucket-Eigentümer kontaktiert, um herauszufinden, wer der Eigentümer des Objekts ist.
   - Wenn **der Zugriff während eines Ausfalls** für den Bucket aktiviert ist, prüft der anfordernde Standort mit dem Bucket-Eigentümer, um festzustellen, wer der aktuelle Objekteigentümer ist.
3. Wenn Standort 2 keine Kopie des Objekts enthält (Blocktyp = lokal oder Kopie), sendet Standort 1 die Objekt-A-Daten an Standort 2.
4. Standort 2 sendet die Objekt-A-Daten an den anfordernden Client.

3.6 Aktualisieren des Datenflusses in geografisch replizierten Umgebungen

ECS ist so konzipiert, dass die Aktiv-Aktiv-Aktualisierung von Daten von Nodes innerhalb der zugehörigen Bucket-Replikationsgruppe ermöglicht wird. Dies kann erreicht werden, indem Standorte, die keine Objekteigentümer sind, synchron Informationen über eine Objektaktualisierung an den primären Eigentümerstandort senden und auf die Bestätigung warten müssen, bevor sie die Bestätigung zurück an den Client senden können. Die Daten im Zusammenhang mit dem aktualisierten Objekt werden als Teil der normalen asynchronen Blockreplikationsaktivitäten repliziert. Wenn die Daten noch nicht an den Eigentümerstandort repliziert wurden und sie eine Leseanforderung für die Daten erhalten, werden die Daten vom Remotestandort angefordert. Abbildung 10 zeigt ein Beispiel für eine Aktualisierungsanforderung für Objekt A von einem anderen Standort als dem Standort des Objekteigentümers.

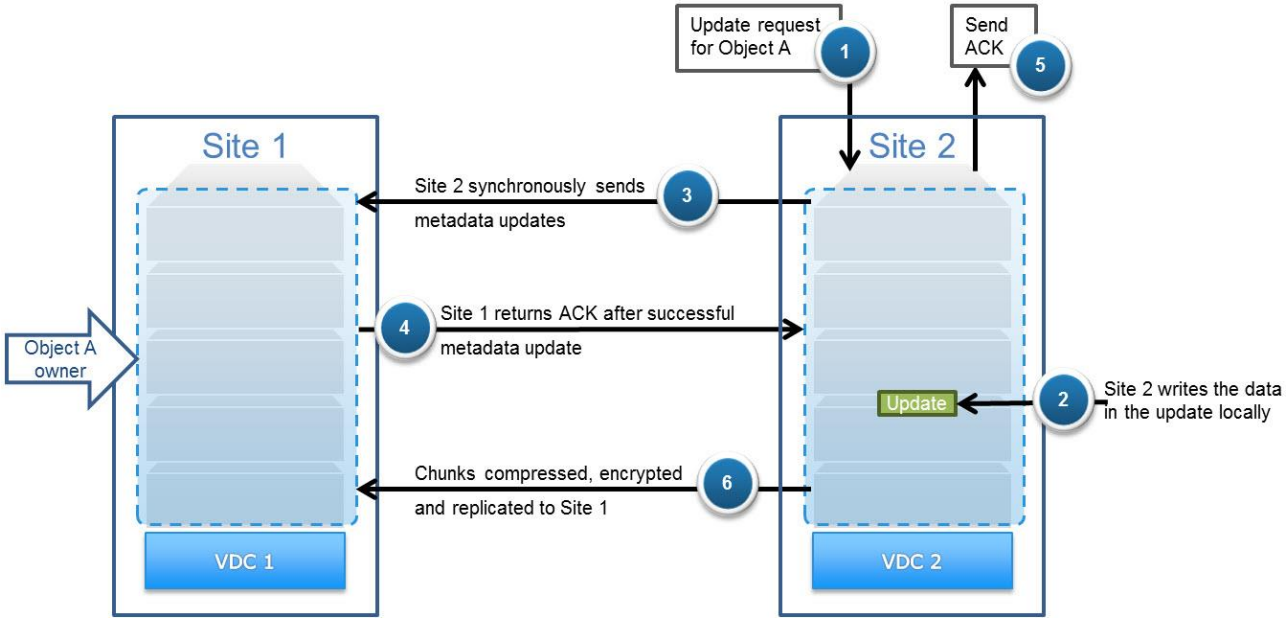

Abbildung 10 Aktualisieren des Objektworkflows für ein geografisch repliziertes Objekt, das einem anderen Standort gehört

In diesem Beispiel wird der Ablauf des Aktualisierungsvorgangs wie folgt angezeigt:

1. Standort 2 erhält eine Aktualisierungsanforderung für Objekt A, das Standort 1 gehört.
2. Standort 2 schreibt die Daten lokal in die Aktualisierung.
3. Standort 2 sendet Metadatenaktualisierungen synchron an den Objekteigentümer, in diesem Beispiel Standort 1.

Hinweis: Wenn **der Zugriff während eines Ausfalls** für den Bucket aktiviert ist oder Standort 2 nicht der Objekteigentümer ist, kontaktiert er zuerst den Bucket-Eigentümer, um zu bestimmen, wer der aktuelle Objekteigentümer ist.

---

4. Standort 1 sendet an Standort 2 die Bestätigung, dass die Metadatenaktualisierung erfolgreich war.
5. Standort 2 sendet die Bestätigung an den anfordernden Client, dass die Aktualisierung erfolgreich war.
6. Die Blöcke werden wie gewohnt zur Replikationswarteschlange hinzugefügt, verschlüsselt und asynchron an Standort 1 repliziert.

Unter normalen Bedingungen ändert sich der Objekteigentümer nach einer Aktualisierung nicht, unabhängig davon, von welchem Standort die Aktualisierung stammt. In diesem Beispiel bleibt der Objekteigentümer auch nach der erfolgreichen Aktualisierung, die von Standort 2 stammt, Standort 1. Die einzige Ausnahme gilt während eines vorübergehenden Standortausfalls, wenn **der Zugriff während eines Ausfalls** aktiviert ist. Weitere Informationen finden Sie im Abschnitt 4.1.2.

4 Verfügbarkeit für mehrere Standorte

ECS bietet eine starke Konsistenz, sodass I/O-Anforderungen mit dem Eigentümer überprüft werden müssen, bevor eine Antwort erfolgt. Aus diesem Grund kann der Zugriff auf einige Buckets und Objekte vorübergehend unterbrochen werden, wenn auf einen Standort nicht zugegriffen werden kann.

Standortausfälle können für unterschiedliche Dauer und aus vielen verschiedenen Gründen auftreten, z. B.:

- Vorübergehend, z. B. durch den Verlust der Netzwerkverbindung zwischen Verbundstandorten oder durch den Ausfall eines gesamten Standorts, z. B. durch einen Stromausfall am Gebäude.
- Dauerhaft, z. B. bei einer Naturkatastrophe.

Um temporäre Standortausfälle zu erkennen, erstellen Verbundstandorte einen Heartbeat zwischen Standorten. Wenn der Heartbeat zwischen Standorten über einen längeren Zeitraum verloren geht (der Standardwert beträgt 15 Minuten):

- In einer Konfiguration mit zwei Standorten werden die anderen als fehlgeschlagen markiert.
- In einer Konfiguration mit drei oder mehr Standorten wird ein Standort nur dann als fehlgeschlagen markiert, wenn beide:
  - Eine Mehrheit der Standorte verliert den Heartbeat für den längeren Zeitraum am selben ECS-Standort.
  - Und wenn alle verbleibenden Standorte derzeit als online markiert sind.

Beispiel: Wenn in einer Konfiguration mit drei Standorten die Netzwerkverbindung zu Standort 2 und 3 für einen längeren Zeitraum unterbrochen wird, markiert ECS Standort 1 als vorübergehend fehlgeschlagen.

Wenn ein Verbundstandort ausgefallen ist, kann die Systemverfügbarkeit aufrechterhalten werden, indem der Zugriff auf andere Verbundsysteme gesteuert wird. Während des standortweiten Ausfalls sind die geografisch replizierten Daten, die dem nicht verfügbaren Standort gehören, vorübergehend nicht verfügbar. Die Dauer, für die die Daten weiterhin nicht verfügbar sind, wird durch Folgendes bestimmt:

- Gibt an, ob der **Zugriff während eines Ausfalls** aktiviert ist
- Wie lange der vorübergehende Standortausfall bestehen bleibt
- Die Zeit, die für ein permanentes Standort-Failover benötigt wird, um Recovery-Vorgänge abzuschließen

Der Systemausfall am Standort kann entweder temporär oder dauerhaft sein. Ein vorübergehender Standortausfall bedeutet, dass der Standort wieder online geschaltet werden kann und in der Regel durch Stromausfälle oder den Verlust des Netzwerks zwischen Standorten verursacht wird. Ein permanenter Standortausfall tritt auf, wenn das gesamte System nicht wiederhergestellt werden kann, z. B. durch einen Laborbrand. Nur AdministratorInnen können feststellen, ob ein Standortausfall dauerhaft ist, und Recovery-Vorgänge starten.

4.1 Vorübergehender Standortausfall (TSO)

Temporäre Standortausfälle treten auf, wenn ein Standort vorübergehend nicht für andere Standorte in einer Replikationsgruppe zugänglich ist. ECS ermöglicht AdministratorInnen zwei Konfigurationsoptionen, die beeinflussen, wie während eines temporären Systemausfalls am Standort auf Objekte zugegriffen werden kann.

- Deaktivieren Sie den **Zugriff während eines Ausfalls** (ADO), der eine starke Konsistenz beibehält, indem Sie:
  - Es wird weiterhin der Zugriff auf Daten zugelassen, die Standorten gehören, auf die zugegriffen werden kann.
  - Verhindern des Zugriffs auf Daten, die einem unzugänglichen Standort gehören.

- Aktivieren Sie die Option **Zugriff während eines Ausfalls,** die Lese- und optional Schreibzugriff auf alle geografisch replizierten Daten ermöglicht, einschließlich der Daten, die im Besitz des als fehlgeschlagen markierten Standorts sind. Während eines TSO mit **Zugriff während des Ausfalls** schalten die Daten im Bucket vorübergehend auf eventuelle Konsistenz um. Sobald alle Standorte wieder online sind, wird wieder auf starke Konsistenz umgeschaltet.

Die Standardeinstellung ist, dass der **Zugriff während eines Ausfalls** deaktiviert wird.

Die Option **Zugriff während des Ausfalls** kann auf Bucket-Ebene eingestellt werden, d. h. Sie können diese Option für einige Buckets aktivieren und für andere nicht. Diese Bucket-Option kann jederzeit geändert werden, solange alle Standorte online sind, kann sie während eines Systemausfalls am Standort nicht geändert werden.

Während eines vorübergehenden Systemausfalls am Standort:

- Buckets, Namespaces, Objektnutzer, Authentifizierungsanbieter, Replikationsgruppen und NFS-Nutzer- und -Gruppenzuordnungen können nicht von jedem Standort aus erstellt, gelöscht oder aktualisiert werden (Replikationsgruppen können während eines permanenten Standort-Failovers aus einem VDC entfernt werden).
- Buckets für einen Namespace können nicht aufgeführt werden, wenn der Standort, der Eigentümer des Namespace ist, nicht erreichbar ist.
- Dateisysteme in HDFS/NFS-Buckets, die dem nicht verfügbaren Standort gehören, sind schreibgeschützt.
- Wenn Sie ein Objekt aus einem Bucket kopieren, der dem nicht verfügbaren Standort gehört, ist die Kopie eine vollständige Kopie des Quellobjekts. Das bedeutet, dass die Daten des gleichen Objekts mehr als einmal gespeichert werden. Unter normalen Umständen ohne TSO besteht die Objektkopie aus den Datenindizes des Objekts und ist kein volles Duplikat der Daten des Objekts.
- OpenStack Swift-NutzerInnen können sich während eines TSO nicht bei OpenStack anmelden, da ECS Swift-NutzerInnen sich während des TSO nicht authentifizieren können. Nach dem TSO müssen sich Swift-NutzerInnen erneut authentifizieren.

4.1.1 Standardmäßiges TSO-Verhalten

Da ECS eine starke Konsistenz bietet, müssen IO-Anfragen mit dem Eigentümer überprüft werden, bevor sie reagieren. Wenn ein Standort für andere Standorte innerhalb einer Replikationsgruppe nicht zugänglich ist, kann der Zugriff auf Buckets und Objekte unterbrochen werden.

Tabelle 14 zeigt an, welcher Zugriff erforderlich ist, damit ein Vorgang erfolgreich ist.

Tabelle 14 Zugriffsanforderungen

Vorgang	Voraussetzungen für den Erfolg
Objekt erstellen	Erfordert, dass der Bucket-Eigentümer zugänglich ist
Objekte auflisten	Erfordert, dass der Bucket-Eigentümer und alle Objekte im Bucket für den anfordernden Node zugänglich sind
Objekt lesen Objekt aktualisieren	Erfordert, dass der Anforderer: • Der Objekteigentümer und der Bucket-Eigentümer (Bucket-Eigentumsrechte sind nur erforderlich, wenn **der Zugriff während eines Ausfalls** für den Bucket aktiviert ist, der das Objekt enthält). • Oder dass sowohl der Objekteigentümer als auch der Bucket-Eigentümer für den anfordernden Node zugänglich sind

- Ein Erstellungsobjektvorgang umfasst die Aktualisierung der Bucket-Auflistung mit dem neuen Objektnamen. Dies erfordert Zugriff auf den Bucket-Eigentümer und schlägt daher fehl, wenn der anfordernde Standort keinen Zugriff auf den Bucket-Eigentümer hat.
- Zum Auflisten von Objekten in einem Bucket müssen sowohl Informationen vom Bucket-Eigentümer als auch Head-Informationen für jedes Objekt im Bucket aufgelistet werden. Aus diesem Grund schlagen die folgenden Anforderungen für die Bucket-Auflistung fehl, wenn **der Zugriff während eines Ausfalls** deaktiviert ist:
  - Anforderungen zum Auflisten von Buckets, die einem Standort gehören, auf den der Anforderer nicht zugreifen kann.
  - Bucket, der Objekte enthält, die einem Standort gehören, auf den der Anforderer nicht zugreifen kann.
- Leseobjekt erfordert zuerst das Lesen der Objektmetadaten vom Objekteigentümer.
  - Wenn der anfordernde Standort der Objekteigentümer ist und **der Zugriff während eines Ausfalls** deaktiviert ist, ist die Anforderung erfolgreich.
  - Wenn der anfordernde Standort der Objekt- und Bucket-Eigentümer ist, ist die Anforderung erfolgreich.
  - Wenn der Objekteigentümer nicht lokal ist, muss der Standort beim Bucket-Eigentümer nachsehen, um den Objekteigentümer zu finden. Wenn der Objekteigentümer oder der Bucket-Eigentümerstandort für den Anforderer nicht verfügbar ist, schlägt der Lesevorgang fehl.
- Aktualisierungen an Objekten erfordern eine erfolgreiche Aktualisierung der Objektmetadaten auf dem Objekteigentümer.
  - Wenn der anfordernde Standort der Objekteigentümer ist und **der Zugriff während eines Ausfalls** deaktiviert ist, ist die Anforderung erfolgreich.
  - Wenn der anfordernde Standort der Objekt- und Bucket-Eigentümer ist, ist die Anforderung erfolgreich.
  - Wenn der Objekteigentümer nicht lokal ist, muss der Standort beim Bucket-Eigentümer nachsehen, um den Objekteigentümer zu finden. Wenn der Objekteigentümer oder der Bucket-Eigentümerstandort für den Anforderer nicht verfügbar ist, schlägt der Lesevorgang fehl.

In einem Beispiel mit drei Standorten sind das Bucket- und Objektlayout in Abbildung 11 dargestellt.

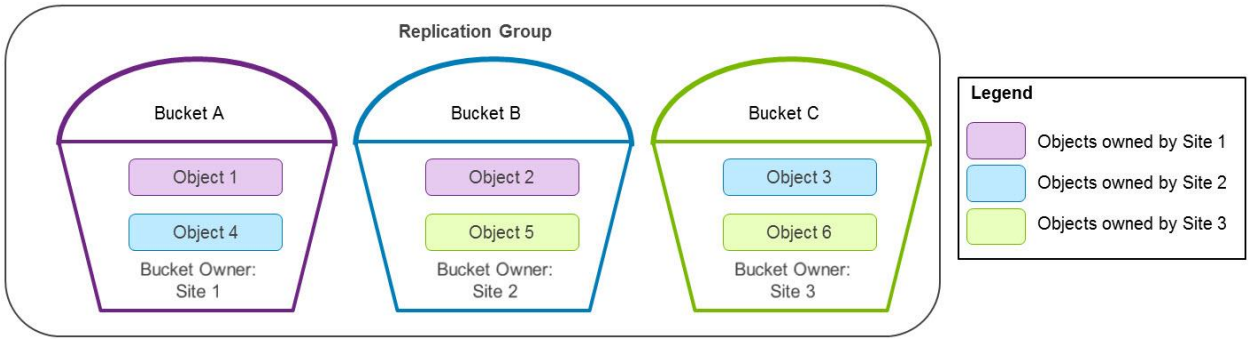

Abbildung 11 Beispiel für Bucket- und Objekteigentumsrechte

Tabelle 15 listet auf, welche Vorgänge im Beispiel für die Konfiguration mit drei Standorten erfolgreich sind oder fehlschlagen, Abbildung 11 wenn Standort 1 für die anderen Standorte in der Replikationsgruppe nicht zugänglich ist. Um die Interpretation der Tabelle zu vereinfachen, wird der nicht zugängliche Standort als fehlgeschlagen und die anderen beiden als online aufgeführt.

Tabelle 15 Vorgänge, die erfolgreich sind oder fehlschlagen, wenn Standort 1 für andere Standorte nicht zugänglich ist

Vorgang	Bucket/Objekt	Anfrage gesendet an		
		Standort 1 (fehlgeschlagen)	**Standort 2 (online)**	**Standort 3 (online)**
Erstellen von Objekten in	Bucket A	Success Bucket in lokalem Besitz	Fail Kein Zugriff auf Bucket-Eigentümer	Fail Kein Zugriff auf Bucket-Eigentümer
	Bucket B	Fail Kein Zugriff auf Bucket-Eigentümer	Success Bucket in lokalem Besitz	Success Bucket im Besitz eines Onlinestandortes
	Bucket C	Fail Kein Zugriff auf Bucket-Eigentümer	Success Bucket im Besitz eines Onlinestandortes	Success Bucket in lokalem Besitz
Auflisten von Objekten in	Bucket A	Fail Obwohl der Bucket im lokalen Besitz ist, enthält er ein Objekt, das im Besitz eines Standorts ist, auf den er nicht zugreifen kann.	Fail Kein Zugriff auf Bucket-Eigentümer	Fail Kein Zugriff auf Bucket-Eigentümer
	Bucket B	Fail Kein Zugriff auf Bucket-Eigentümer	Fail Obwohl der Bucket im lokalen Besitz ist, enthält er ein Objekt, das dem ausgefallenen Standorts gehört.	Fail Obwohl der Bucket-Eigentümer online ist, enthält der Bucket ein Objekt, das dem ausgefallenen Standort gehört.
	Bucket C	Fail Kein Zugriff auf Bucket-Eigentümer	Success Bucket-Eigentümer ist Onlinestandort und alle Objekte stammen von Onlinestandorten.	Success Bucket im Besitz lokal und alle Objekte sind von Onlinestandorten.
Objekt lesen oder aktualisieren	Objekt 1	Success Objekteigentum lokal	Fail Kein Zugriff auf Objekteigentümer	Fail Kein Zugriff auf Objekteigentümer
	Objekt 2	Success Objekteigentum lokal	Fail Kein Zugriff auf Objekteigentümer	Fail Kein Zugriff auf Objekteigentümer
	Objekt 3	Fail Kein Zugriff auf Objekteigentümer	Success Objekteigentum lokal	Success Das Objekt ist nicht im lokalen Besitz, daher wird der Objekteigentümer vom Bucket-Eigentümer abgerufen, der online ist.

Vorgang	Bucket/Objekt	Anfrage gesendet an		
		Standort 1 (fehlgeschlagen)	**Standort 2 (online)**	**Standort 3 (online)**
	Objekt 4	Fail Kein Zugriff auf Objekteigentümer	Success Objekteigentum lokal	Fail Das Objekt ist nicht lokal im Besitz, daher muss auf den Bucket-Eigentümer zugegriffen werden, der der ausgefallene Standort ist.
	Objekt 5	Fail Kein Zugriff auf Objekteigentümer	Success Das Objekt ist nicht im lokalen Besitz, daher wird der Eigentümer des Objekts vom Bucket-Eigentümer übernommen, der es ist.	Success Objekteigentum lokal
	Objekt 6	Fail Kein Zugriff auf Objekteigentümer	Success Das Objekt ist nicht im lokalen Besitz, daher wird der Objekteigentümer vom Bucket-Eigentümer abgerufen, der online ist.	Success Objekteigentum lokal

4.1.2 TSO-Verhalten bei aktiviertem Zugriff während eines Ausfalls

Wenn ein Standort zum ersten Mal für andere Standorte innerhalb einer Replikationsgruppe unzugänglich ist, verhält er sich wie im Abschnitt über das TSO-Standardverhalten beschrieben. Nachdem der Heartbeat zwischen Standorten für einen längeren Zeitraum verloren gegangen ist, markiert ECS einen Standort standardmäßig in 15 Minuten als fehlgeschlagen. Das Aktivieren des **Zugriffs während eines Ausfalls** (ADO) auf einem Bucket ändert das TSO-Verhalten, nachdem ein Standort als fehlgeschlagen markiert wurde, sodass Objekte in diesem Bucket letztendliche Konsistenz nutzen können. Das bedeutet, dass nach der Kennzeichnung eines Standorts als vorübergehend fehlgeschlagen alle Buckets mit aktiviertem **Zugriff während eines Ausfalls** Lese- und optional Schreibvorgänge von einem Standort unterstützen, der nicht Eigentümer ist. Dies wird erreicht, indem die Nutzung der replizierten Metadaten zugelassen wird, wenn die autorisierende Kopie am Eigentümerstandort nicht verfügbar ist. Sie können die Bucket-Option für den **Zugriff während eines Ausfalls** jederzeit ändern, außer während eines Systemausfalls am Standort.

Der Vorteil der Aktivierung des **Zugriffs während eines Ausfalls** besteht darin, dass der Zugriff auf Daten nach der Kennzeichnung eines Standorts als fehlgeschlagen ermöglicht wird. Der Nachteil besteht darin, dass die zurückgegebenen Daten möglicherweise veraltet sind.

Ab Version 3.1 wurde eine zusätzliche Bucket-Option für den **schreibgeschützten Zugriff während eines Ausfalls** hinzugefügt, die sicherstellt, dass sich die Eigentumsverhältnisse an den Objekten nie ändern und die Gefahr von Konflikten durch Objektaktualisierungen sowohl auf dem ausgefallenen als auch auf dem Onlinestandort während eines TSO beseitigt wird. Der Nachteil des **schreibgeschützten Zugriffs während des Ausfalls** besteht darin, dass nach der Kennzeichnung eines Standorts als ausgefallen keine neuen Objekte erstellt werden können und keine vorhandenen Objekte im Bucket aktualisiert werden können, bis alle Standorte wieder online sind. Die Option **schreibgeschützter Zugriff während des Ausfalls** ist nur während der Erstellung des Buckets verfügbar und kann danach nicht mehr geändert werden.

Wie bereits erwähnt, wird ein Standort als fehlgeschlagen markiert, wenn der Heartbeat zwischen Standorten über einen längeren Zeitraum verloren geht. Die Standardeinstellung beträgt 15 Minuten. Wenn der Heartbeat daher über einen längeren Zeitraum verloren geht:

- In einer Konfiguration mit zwei Standorten betrachtet sich jeder als online und markiert den anderen als fehlgeschlagen.
- In einer Konfiguration mit drei oder mehr Standorten wird ein Standort nur dann als fehlgeschlagen markiert, wenn beide:
  - Eine Mehrheit der Standorte verliert den Heartbeat für den längeren Zeitraum am selben ECS-Standort.
  - Und wenn alle verbleibenden Standorte derzeit als online markiert sind.

Ein ausgefallener Standort ist möglicherweise weiterhin für Clients und Anwendungen zugänglich, z. B. Wenn das interne Netzwerk eines Unternehmens die Verbindung zu einem einzigen Standort verliert, aber das Extranet-Netzwerk weiterhin betriebsbereit ist. Beispiel: Wenn in einer Konfiguration mit fünf Standorten die Netzwerkverbindung zu Standort 2 bis 5 für einen längeren Zeitraum unterbrochen wird, markiert ECS Standort 1 als vorübergehend fehlgeschlagen. Wenn Standort 1 immer noch für Clients und Anwendungen zugänglich ist, kann er Service-Requests für Buckets und Objekte im Besitz von lokal befindlichen Buckets und Objekten durchführen, da keine Suche an anderen Standorten erforderlich ist. Anforderungen an Standort 1 für Buckets und Objekte, die nicht im Besitz sind, schlagen jedoch fehl. Tabelle 16 zeigt an, welcher Zugriff erforderlich ist, nachdem ein Standort als fehlgeschlagen markiert wurde, damit ein Vorgang erfolgreich ist, wenn der **Zugriff während eines Ausfalls** auf „Aktiviert“ gesetzt ist.

Tabelle 16 Erfolgreiche Vorgänge, nachdem ein Standort als fehlgeschlagen markiert ist, wobei der Zugriff während eines Ausfalls aktiviert ist

Vorgang	**Anfrage an den fehlgeschlagenen Standort gesendet (in einem Verbund, der drei oder mehr Standorte enthält)**	**Die Anfrage wird an einen Onlinestandort gesendet, einschließlich:** • **Jeder Onlinestandort in einem Verbund, der drei oder mehr Standorte enthält** • **Oder einen Standort in einem Verbund, der nur zwei Standorte enthält**
Objekt erstellen	Erfolg für lokale Buckets, es sei denn, der **schreibgeschützte Zugriff während eines Ausfalls** ist für den Bucket aktiviert. Ausfall für Buckets im Remotebesitz	Erfolgreich, es sei denn, der **schreibgeschützte Zugriff während eines Ausfalls** ist für den Bucket aktiviert.
Objekte auflisten	Listet nur Objekte in seinen lokalen Buckets auf, wenn alle Objekte auch im lokalen Besitz sind	Successful Umfasst keine Objekte, die einem ausgefallenen Standort gehören, der noch nicht repliziert wurde
Objekt lesen	Erfolg für Objekte in lokalem Besitz in lokalen Buckets. (möglicherweise nicht die neueste Version) Ausfall bei Objekten in Remotebesitz	Successful Wenn sich das Objekt im Besitz des ausgefallenen Standorts befindet, muss das ursprüngliche Objekt die Replikation abgeschlossen haben, bevor der Fehler aufgetreten ist.
Objekt aktualisieren	Erfolg für Objekte in lokalem Besitz in lokalen Buckets, es sei denn, der **schreibgeschützte Zugriff während eines Ausfalls** ist für den Bucket aktiviert. Ausfall bei Objekten in Remotebesitz	Erfolgreich, es sei denn, der **schreibgeschützte Zugriff während eines Ausfalls** ist für den Bucket aktiviert. Erwerb von Eigentumsrechten an Objekten

- Objekt erstellen

  Nachdem ein Standort als fehlgeschlagen markiert wurde, ist die Erstellung von Objekten nicht erfolgreich, wenn **der schreibgeschützte Zugriff während eines Ausfalls** für den Bucket aktiviert ist. Wenn er deaktiviert ist:

  - In einem Verbund, der drei oder mehr Standorte enthält, kann der als ausgefallen markierte Standort, wenn er für Clients oder Anwendungen zugänglich ist, Objekte nur in seinen lokalen Buckets erstellen. Auf diese neuen Objekte kann nur von diesem Standort aus zugegriffen werden. Andere Standorte kennen diese Objekte erst, wenn der ausgefallene Standort wieder online ist und sie Zugriff auf den Bucket-Eigentümer erhalten.
  - Die Onlinestandorte können Objekte in jedem Bucket erstellen, einschließlich Buckets, die dem Standort gehören, der als fehlgeschlagen markiert ist. Zum Erstellen eines Objekts muss die Bucket-Liste mit dem neuen Objektnamen aktualisiert werden. Wenn der Bucket-Eigentümer inaktiv ist, erstellt er einen Objektverlauf, der das Objekt während der Recovery oder dem Zusammenführungsvorgang des Bucket-Eigentümers in die Bucket-Auflistungstabelle einfügt.

- Objekt auflisten

  - In einem Verbund, der drei oder mehr Standorte enthält, erfordert der als fehlgeschlagen markierte Standort lokale Eigentumsrechte für den Bucket und alle Objekte innerhalb des Buckets, um Objekte in einem Bucket erfolgreich aufzulisten. Die Auflistung vom ausgefallenen Standort enthält keine Objekte, die remote erstellt wurden, während sie als vorübergehend fehlgeschlagen markiert sind.
  - Die Onlinestandorte können Objekte in jedem Bucket auflisten, einschließlich eines Buckets, der dem Standort gehört, der als fehlgeschlagen markiert ist. Es listet die neueste Version des Bucket-Eintrags auf, der möglicherweise etwas veraltet ist.

- Objekt lesen

  - In einem Verbund, der drei oder mehr Standorte umfasst, kann der ausgefallene Standort, wenn er für Clients oder Anwendungen zugänglich ist, nur Objekte in lokalem Besitz in lokalem Besitz von Buckets lesen.
  - Die Leseanforderung an einen ausgefallenen Standort benötigt zunächst Zugriff auf den Bucket-Eigentümer, um den aktuellen Objekteigentümer zu validieren. Wenn er auf den Bucket-Eigentümer zugreifen kann und der aktuelle Objekteigentümer lokal ist, ist die Leseanforderung erfolgreich. Wenn entweder der Bucket-Eigentümer oder der aktuelle Objekteigentümer nicht zugänglich ist, schlägt die Leseanforderung fehl.
  - Die Onlinestandorte können alle Objekte lesen, einschließlich derjenigen, die im Besitz des als fehlgeschlagen markierten Standorts sind, solange das ursprüngliche Objekt die Replikation abgeschlossen hat. Es überprüft den Objektverlauf und antwortet mit der neuesten Version des Objekts, die verfügbar ist. Wenn ein Objekt später an dem als fehlgeschlagen markierten Standort aktualisiert wurde und die Georeplikation der aktualisierten Version nicht abgeschlossen wurde, wird die ältere Version verwendet, um die Leseanforderung zu bearbeiten.

Hinweis: Leseanforderungen, die an Onlinestandorte gesendet werden, an denen der Bucket-Eigentümer der ausgefallene Standort ist, verwenden die lokalen Bucket-Auflistungsinformationen und den Objektverlauf, um den Objekteigentümer zu bestimmen.

- Objekt aktualisieren

  - Nachdem ein Standort als ausgefallen markiert wurde, können Aktualisierungsobjekte nicht mehr erfolgreich aktualisiert werden, wenn für den Bucket der **schreibgeschützte Zugriff während des Ausfalls** aktiviert ist. Wenn er deaktiviert ist, kann der ausgefallene Standort in einem Verbund, der drei oder mehr Standorte enthält, nur lokal im Besitz von Buckets befindliche Objekte aktualisieren, wenn er für Clients oder Anwendungen zugänglich ist.

- Die Aktualisierungsanforderung benötigt zunächst Zugriff auf den Bucket-Eigentümer, um die aktuelle Objekteigentumsrechte zu validieren. Wenn er auf den Bucket-Eigentümer zugreifen kann und der aktuelle Objekteigentümer lokal ist, ist die Aktualisierungsanforderung erfolgreich. Wenn entweder der Bucket-Eigentümer oder der aktuelle Objekteigentümer nicht zugänglich ist, schlägt die Aktualisierungsanforderung fehl.
- Nach Abschluss der erneuten Zusammenführungsvorgänge ist diese Aktualisierung nicht in Lesevorgängen enthalten, wenn der Remotestandort auch dasselbe Objekt während desselben TSO aktualisiert hat.

Ein Onlinestandort kann sowohl Objekte, die Onlinestandorten gehören, als auch fehlgeschlagene Standorte aktualisieren. Wenn eine Objektaktualisierungsanforderung für ein Objekt, das dem als fehlgeschlagen markierten Standort gehört, an einen Onlinestandort gesendet wird, wird die neueste Version des Objekts aktualisiert, die auf einem System verfügbar ist, das als online markiert ist.

Der Standort, der die Aktualisierung durchführt, wird zum neuen Objekteigentümer und aktualisiert den Objektverlauf mit den neuen Eigentümerinformationen und der Sequenznummer. Dies wird für Recovery- oder erneute Zusammenführungsvorgänge des ursprünglichen Objekteigentümers verwendet, um den Objektverlauf mit dem neuen Eigentümer zu aktualisieren.

Hinweis: Aktualisierungsanforderungen, die an Onlinestandorte gesendet werden, an denen der Bucket-Eigentümer der ausgefallene Standort ist, verwenden die lokalen Bucket-Auflistungsinformationen und den Objektverlauf, um den Objekteigentümer zu bestimmen.

Dieses Beispiel zeigt, was mit dem Bucket- und Objektlayout für Namespace 1 in einer Konfiguration mit drei Standorten geschieht, wie in Abbildung 12 gezeigt.

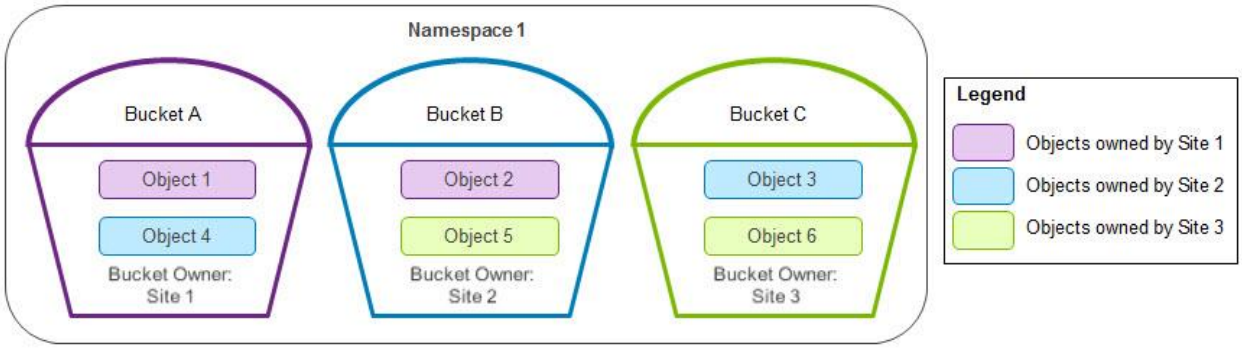

Abbildung 12 Bucket- und Objekteigentumsrechte für Namespace 1

Tabelle 17 zeigt ein Beispiel in dieser Konfiguration mit drei Standorten, wenn alle drei erfüllt sind:

- Der **Zugriff während eines Ausfalls** ist aktiviert.
- Und der **schreibgeschützte Zugriff während eines Ausfalls** ist deaktiviert.
- Und Standort 1 wird als fehlgeschlagen markiert.

Tabelle 17 Beispiel für Vorgänge, die erfolgreich sind oder fehlschlagen, wenn der **Zugriff während eines Ausfalls** und der **schreibgeschützte Zugriff während eines Ausfalls** deaktiviert sind, wobei Standort 1 in einer Konfiguration mit drei Standorten vorübergehend als fehlgeschlagen markiert ist

Vorgang	Bucket/Objekt	Anfrage gesendet an	
		Standort 1 (als fehlgeschlagen markiert)	**Standort 2 oder Standort 3 (online)**
Erstellen von Objekten in	Bucket A	Success	Success
	Bucket B	Fail Ausgefallene Standort kann nur Objekte in lokalen Buckets erstellen	Success
	Bucket C	Fail Ausgefallene Standort kann nur Objekte in lokalen Buckets erstellen	Success
Auflisten von Objekten in	Bucket A	Fail Obwohl der Bucket im lokalen Besitz ist, enthält er Objekte, die sich im Remotebesitz befinden	Success Umfasst keine Objekte, die einem ausgefallenen Standort gehören, der nicht repliziert wurde
	Bucket B	Fail Ausgefallener Standort kann nur Objekte in lokalen Buckets auflisten	Success
	Bucket C	Fail Ausgefallener Standort kann nur Objekte in lokalen Buckets auflisten	Success
Objekt lesen oder aktualisieren	Objekt 1	Erfolg, sowohl Objekt als auch Bucket sind lokal in Besitz	Success Erfordert, dass das Objekt die Replikation vor TSO abgeschlossen hat Aktualisierung überträgt den Besitz des Objekts
	Objekt 2	Fehler, Bucket ist nicht lokal im Besitz	
	Objekt 3, Objekt 4 Objekt 5 Objekt 6	Fail Ausgefallener Standort kann nur lokale Objekte in lokalen Buckets lesen und aktualisieren	Success

Sobald der Heartbeat zwischen Standorten wiederhergestellt wurde, markiert das System den Standort als online und der Zugriff auf diese Daten wird wie vor dem Ausfall fortgesetzt. Der Zusammenführungsvorgang wird:

- Bucket-Auflistungstabellen aktualisieren
- Objekteigentumsrechten aktualisieren, falls erforderlich
- Verarbeitung der Replikationswarteschlange des zuvor fehlgeschlagenen Standorts fortsetzen

Hinweis: ECS unterstützt den Zugriff nur während des temporären Ausfalls eines einzelnen Standorts.

In zwei weiteren Standortbeispielen, wie in Abbildung 13 gezeigt. Beide Standorte gehen davon aus, dass sie online sind, und markieren den anderen Standort als fehlgeschlagen, wenn ein TSO stattgefunden hat. Alle Erstellungs-, Listen-, Lese- und Aktualisierungsvorgänge sind erfolgreich.

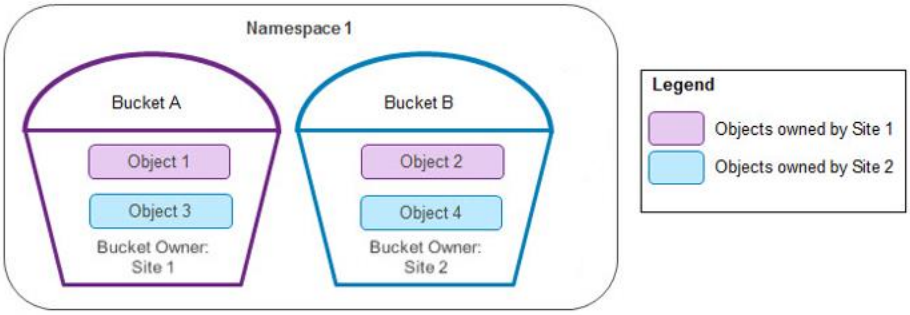

Abbildung 13 Bucket- und Objekteigentumsrechte an zwei Standorten

Während des TSO werden alle Objekte an jedem Standort aktualisiert. Tabelle 18 zeigt die endgültigen Daten am Standort an, sobald der Heartbeat zwischen den Standorten wiederhergestellt wurde.

Tabelle 18 Erfolgreicher Standort nach der Wiederherstellung des Standorts

Objekt	Bucket-Name	Bucket-Eigentümer	Objekteigentümer	Dann ist der erfolgreiche Standort...
Objekt 1	Bucket A	Standort 1	Standort 1	Standort 2
Objekt 2	Bucket B	Standort 2	Standort 1	Der Standort hat den neuesten Zeitstempel.
Objekt 3	Bucket A	Standort 1	Standort 2	Der Standort hat den neuesten Zeitstempel.
Objekt 4	Bucket B	Standort 2	Standort 2	Standort 1

Hinweis: In diesem Beispiel steht der neueste Zeitstempel für die zuletzt aktualisierte Zeit des Objekts am Standort.

4.1.2.1 XOR-Dekodierung mit drei oder mehr Standorten

Wie wir im Abschnitt „XOR-Codierung“ gesehen haben, maximiert ECS die Storage-Effizienz von Daten, die mit einer Replikationsgruppe konfiguriert sind, die drei oder mehr Standorte enthält. Die Daten in sekundären Kopien von Blöcken können nach einem XOR-Vorgang durch Daten in einem Paritätsblock ersetzt werden. Anforderungen für Daten in einem Block, der kodiert wurde, werden vom Standort bearbeitet, der die Primärkopie enthält. Wenn dieser Standort fehlgeschlagen ist, wird die Anforderung an den Standort mit der sekundären Kopie des Objekts gehen. Da diese Kopie jedoch kodiert wurde, muss der sekundäre Standort zunächst die Kopie der Blöcke abrufen, die für die Codierung von den primären Onlinestandorten verwendet wurden. Anschließend wird ein XOR-Vorgang durchgeführt, um das angeforderte Objekt zu rekonstruieren und auf die Anforderung zu reagieren. Nachdem die Blöcke rekonstruiert wurden, werden sie auch zwischengespeichert, sodass der Standort schneller auf nachfolgende Anforderungen reagieren kann.

Tabelle 19 zeigt ein Beispiel für einen Teil einer Blockmanagertabelle an Standort 4 in einer Konfiguration mit vier Standorten.

Tabelle 19 Standort 4 Blockmanagertabelle nach Abschluss der XOR-Codierung

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 4	Kodiert
C2	Standort 2	Standort 4	Kodiert
C3	Standort 3	Standort 4	Kodiert
C4	Standort 4		Parität (C1, C2 und C3)

Abbildung 14 zeigt die Anforderungen, die an der Neuerstellung eines Blocks beteiligt sind, um eine Leseanforderung während eines TSO zu bedienen.

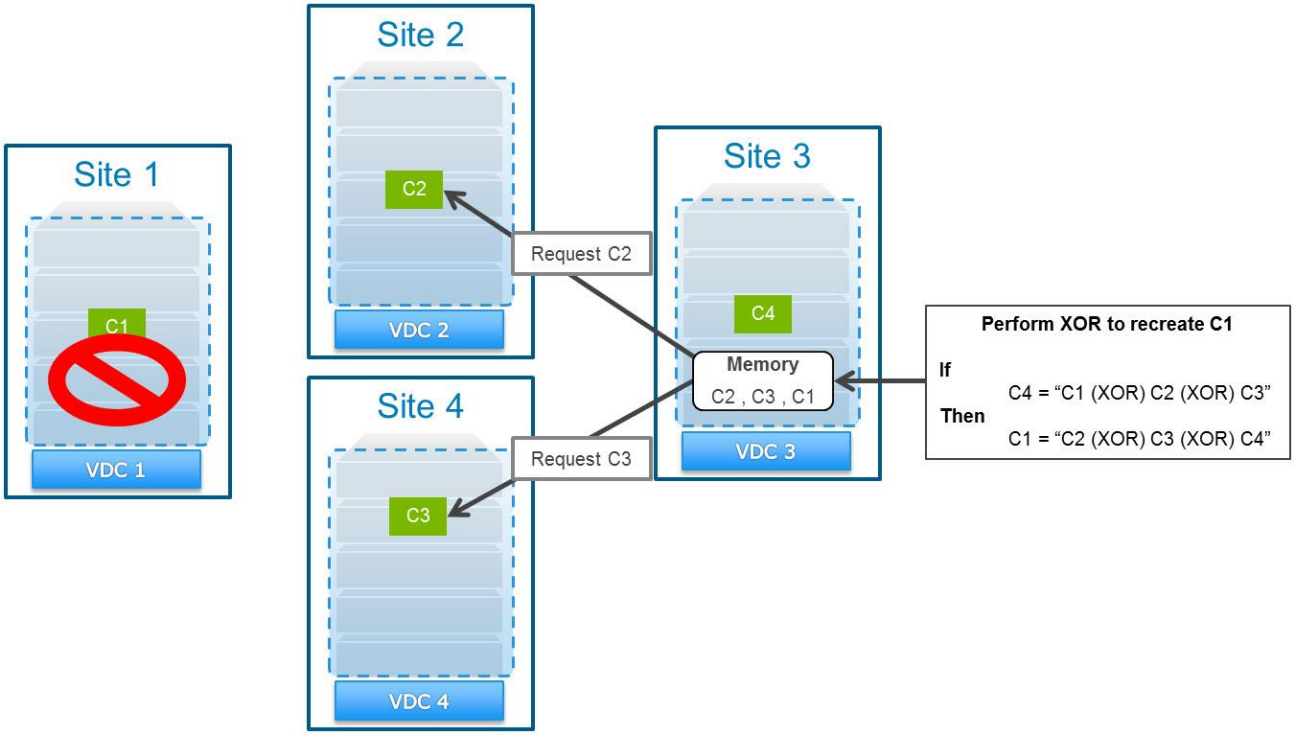

Abbildung 14 Bearbeiten einer Leseanforderung durch Rekonstruieren eines XOR-Blocks

Wenn in diesem Beispiel ein Lesevorgang für ein Objekt in Block **C1** eingeht, wenn **Standort 1** als fehlgeschlagen markiert ist, geschieht Folgendes:

- Da Standort 1 fehlgeschlagen ist, wird die Anforderung an den sekundären Standort von Block C1 gesendet, Standort 4
- Standort 4 hat bereits XORs für die Blöcke **C1**, **C2** und **C3** durchgeführt, d. h. es hat seine lokale Kopie der Daten aus diesen Blöcken durch Daten im Paritätsblock **C4** ersetzt.
- Standort 4 fordert eine Kopie von Block **C2** vom primären Standort (Standort 2) an und speichert diese lokal zwischen.
- Standort 4 fordert eine Kopie von Block **C3** vom primären Standort (Standort 3) an und speichert diese lokal zwischen.
- Standort 4 führt dann einen XOR-Vorgang zwischen den zwischengespeicherten Blöcken **C2** und **C3** mit dem Paritätsblock **C4** durch, um Block **C1** neu zu erstellen und lokal im Cache zu speichern.
- Standort 4 antwortet dann auf die Leseanforderung für das Objekt in Block **C1**.

Hinweis: Die Zeit für den Abschluss von Rekonstruktionsvorgängen erhöht sich linear basierend auf der Anzahl der Standorte in einer Replikationsgruppe.

4.1.2.2 Mit geo-passiver Replikation

Alle Daten in einem Bucket, der mit geo-passiver Replikation konfiguriert ist, haben zwei bis vier Quellstandorte und ein oder zwei dedizierte Replikationsziele. Auf die Replikationsziele geschriebene Daten können nach einem XOR-Vorgang durch Daten in einem Paritätsblock ersetzt werden. Anforderungen für geo-passiv replizierte Daten werden vom Standort bearbeitet, der die Primärkopie enthält. Wenn auf diesen Standort für den anfordernden Standort nicht zugegriffen werden kann, müssen die Daten von einem der Replikationszielstandorte wiederhergestellt werden.

Bei geo-passiver Replikation sind die Quellstandorte immer die Objekt- und Bucket-Eigentümer. Wenn ein Replikationszielstandort als vorübergehend fehlgeschlagen markiert ist, werden alle IO-Vorgänge wie gewohnt fortgesetzt. Die einzige Ausnahme ist die Replikation, die weiterhin in die Warteschlange stellt, bis der Replikationszielstandort wieder dem Verbund beitritt.

Wenn einer der Quellstandorte ausfällt, müssen Anforderungen an den Onlinequellstandort nicht lokal befindliche Daten von einem der Replikationszielstandorte wiederherstellen. Sehen wir uns ein Beispiel an, in dem Standort 1 und Standort 2 die Quellstandorte und Standort 3 der Replikationszielstandort sind. In diesem Beispiel ist die Primärkopie eines Objekts in Block C1 vorhanden, der Standort 1 gehört, und der Block wurde auf das Ziel, Standort 3, repliziert. Wenn Standort 1 fehlschlägt und eine Anforderung an Standort 2 zum Lesen dieses Objekts eingeht, muss Standort 2 eine Kopie von Standort 3 erhalten. Wenn die Kopie kodiert wurde, muss der sekundäre Standort zunächst die Kopie des anderen Blocks abrufen, der für die Codierung vom primären Onlinestandort verwendet wurde. Anschließend wird ein XOR-Vorgang durchgeführt, um das angeforderte Objekt zu rekonstruieren und auf die Anforderung zu reagieren. Nachdem die Blöcke rekonstruiert wurden, werden sie auch zwischengespeichert, sodass der Standort schneller auf nachfolgende Anforderungen reagieren kann.

Tabelle 20 zeigt ein Beispiel für einen Teil einer Blockmanagertabelle auf dem geo-passiven Replikationsziel.

Tabelle 20 Geo-passive Replikationszielblockmanagertabelle nach Abschluss der XOR-Kodierung

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Kodiert
C2	Standort 2	Standort 3	Kodiert
C3	Standort 3		Parität (C1 und C2)

Abbildung 15 zeigt die Anforderungen, die an der Neuerstellung eines Blocks beteiligt sind, um eine Leseanforderung während eines TSO zu bedienen.

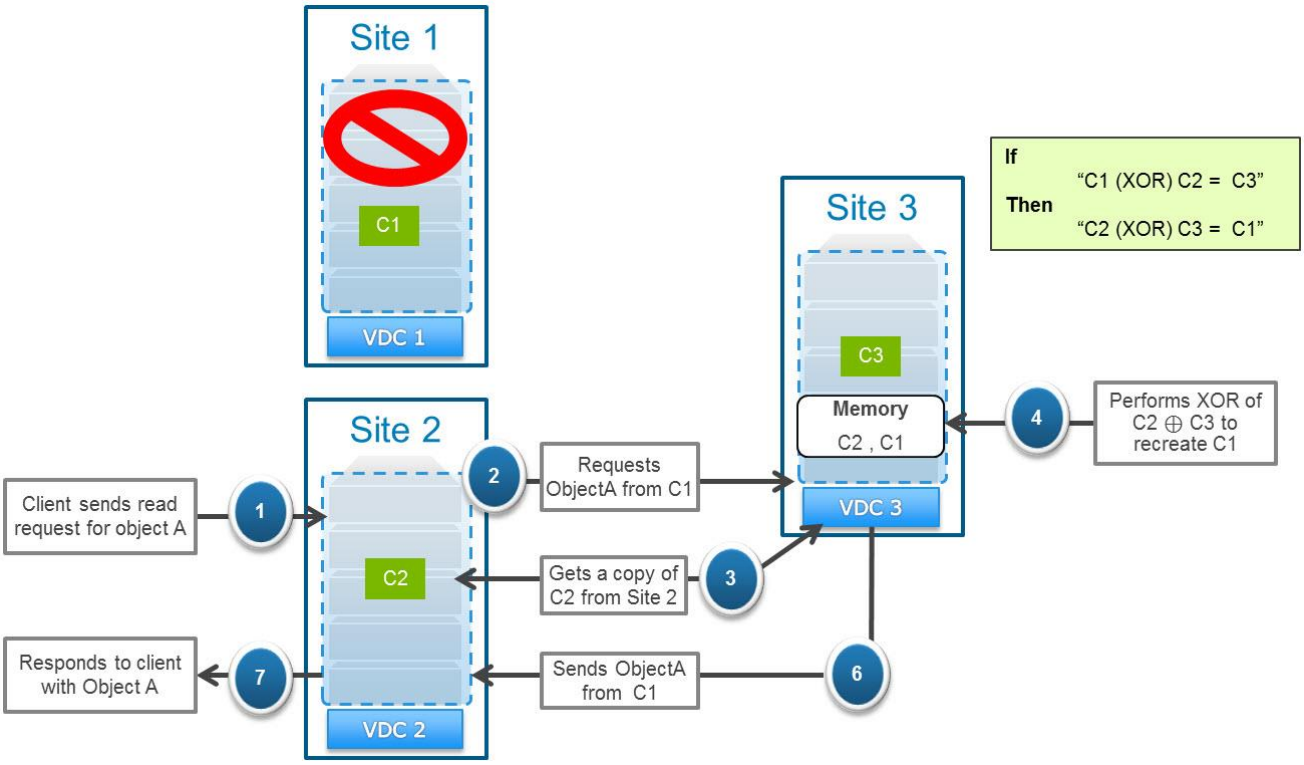

Abbildung 15 Bearbeiten einer Leseanforderung durch Rekonstruieren eines XOR-Blocks

Wenn in diesem Beispiel ein Lesevorgang für ein Objekt in Block **C1** eingeht, wenn **Standort 1** als vorübergehend fehlgeschlagen markiert ist, geschieht Folgendes:

- Da Standort 1 fehlgeschlagen ist, wird die Anforderung an den sekundären Standort von Block C1 gesendet, Standort 3.
- Standort 3 hat bereits XORs für Blöcke **C1** und **C2** durchgeführt, d. h. die lokale Kopie der Daten aus diesen Blöcken wurde durch Daten in Paritätsblock **C3** ersetzt.
- Standort 3 fordert eine Kopie von Block **C2** vom primären Standort (Standort 2) an und speichert diese lokal zwischen.
- Standort 3 führt dann einen XOR-Vorgang zwischen dem zwischengespeicherten Block **C2** mit dem Paritätsblock **C3** durch, um Block **C1** neu zu erstellen und lokal im Cache zu speichern.
- Standort 3 antwortet dann auf die Leseanforderung für das Objekt in Block **C1**.

4.1.2.3 Bei aktivierter Replikation an allen Standorten

Buckets, die mit den Optionen **Auf alle Standorte replizieren** und **Zugriff während eines Ausfalls** konfiguriert sind, können eine schnellere Leseperformance bieten. Die schnellere Leseperformance tritt sowohl während einer Zeit auf, in der alle Standorte online sind, als auch während eines vorübergehenden Standortausfalls, da kein XOR-Dekodierungsvorgang erforderlich ist und die Wahrscheinlichkeit höher ist, dass die Daten lokal gelesen werden.

Daten in Buckets mit der aktivierten Option **Auf alle Standorte replizieren** werden auf jeden Standort repliziert. Erstellungs- und Aktualisierungsobjekte werden genauso behandelt, als ob **Auf alle Standorte replizieren** deaktiviert wurde. Lese- und Listenobjekte werden jedoch etwas anders behandelt, da einige Daten möglicherweise erst die Replikation an einigen, aber nicht allen Standorten abgeschlossen haben, bevor der primäre Standort ausgefallen ist.

Während eines Lesevorgangs prüft der Node, der die Anforderung bearbeitet, zunächst die neueste Version der Metadaten des Objekteigentümers. Wenn der anfordernde Node:

- **der Objekteigentümer ist:**
  - Wenn eine lokale Kopie der angeforderten Daten vorhanden ist, wird diese verwendet, um die Anforderung zu bearbeiten.
  - Wenn das Objekt von einem anderen Standort aktualisiert wurde, der vor der Replikation der Daten fehlschlug, gibt es die Version zurück, die es lokal hat.
- **Nicht der Objekteigentümer ist**
  - Wenn der Standort des Objekteigentümers online ist und die Replikation der Objektdaten erfolgt:
    - Ist die Replikation an diesen Standort weitergeleitet worden, bedient er die Anfrage mit seiner lokalen Kopie der Daten.
    - Wurde die Replikation an diesen Standort noch nicht abgeschlossen, so wird eine Kopie vom Eigentümer des Objekts angefordert und zur Bearbeitung der Anfrage verwendet.
  - Wenn der Objekteigentümer inaktiv ist
    - Wenn die Replikation des Objekts an diesem Standort abgeschlossen ist, wird die lokale Kopie der Daten verwendet. Dies ist möglicherweise nicht die neueste Version.
    - Wenn die Replikation der Objektdaten an den anfordernden Standort nicht abgeschlossen wurde, fordert der anfordernde Standort eine Kopie vom sekundären Standort an, der zuerst in der Blockmanagertabelle aufgeführt ist. Wenn dieser Standort selbst betroffen ist, schlägt der Lesevorgang fehl.

Bei der Auflistung von Objekten in einem Bucket benötigt der Node sowohl Informationen vom Bucket-Eigentümer als auch Head-Informationen für jedes Objekt im Bucket. Wenn der Standort, der Objekteigentümer oder Bucket-Eigentümer ist, ausgefallen ist und der **Zugriff während eines Ausfalls** ebenfalls aktiviert ist, kann er die Anforderung weiterhin bearbeiten, wenn alle verbleibenden Standorte in der Replikationsgruppe online sind. Es wird die neueste Version der Bucket-Auflistung aufgelistet, die möglicherweise etwas veraltet ist und je nach Standort variieren kann.

4.1.3 Mehrere Systemausfälle am Standort

ECS unterstützt den Zugriff nur während eines temporären Ausfalls eines einzelnen Standorts innerhalb einer Replikationsgruppe. darüber hinaus kann nur ein Standort als fehlgeschlagen markiert werden. Dies bedeutet, dass einige Vorgänge fehlschlagen, wenn mehrere Standorte innerhalb einer Replikationsgruppe gleichzeitig ausfallen. Der erste Standort, für den ein Ausfall festgestellt wird (aufgrund eines dauerhaften Heartbeat-Verlusts), wird als fehlgeschlagen markiert. Alle verbleibenden Standorte, die auch einen dauerhaften Heartbeat-Verlust aufweisen, werden nicht als fehlgeschlagen markiert und werden daher als online betrachtet.

Wenn wir beispielsweise fünf Standorte in einer Replikationsgruppe haben und Standort 1 als mit einem dauerhaften Heartbeat-Verlust identifiziert wird, wird dies als fehlgeschlagen markiert. Wenn Standort 2 auch als mit einem dauerhaften Heartbeat-Verlust identifiziert wird, wird er als online aufgeführt. Folgendes geschieht:

- Wenn der **Zugriff während eines Ausfalls** aktiviert ist und der Bucket-Eigentümer Standort 2 ist, schlagen Lesevorgänge/Erstellungen/Aktualisierungen, die an andere Standorte gesendet werden, unabhängig vom Objekteigentümer fehl. Dies liegt daran, dass zunächst mit dem Bucket-Eigentümer geprüft wird, um den Objekteigentümer zu bestimmen. Wenn der Bucket-Eigentümer nicht als fehlgeschlagen markiert ist, sendet der Anforderer die Anforderung an Standort 2, was fehlschlägt.

- Lese- und Aktualisierungsanforderungen, die an Standort 2 gesendet werden, sind nur erfolgreich, wenn es sich um den Objekteigentümer handelt (und Bucket-Eigentümer, wenn der **Zugriff während eines Ausfalls** aktiviert ist).
- Lese- und Aktualisierungsanforderungen, die an andere Standorte als Standort 2 gesendet werden, sind nur dann erfolgreich, wenn der Objekteigentümer (und der Bucket-Eigentümer, wenn der **Zugriff während eines Ausfalls** aktiviert ist) nicht Standort 2 ist.
- Das Erstellen eines Objekts schlägt fehl, wenn der Bucket-Eigentümer Standort 1 oder Standort 2 ist. Dies liegt daran, dass zum Erstellen eines Objekts die Bucket-Auflistung mit dem neuen Objektnamen aktualisiert werden muss. Da dies an allen als online markierten Standorten nicht erfolgreich ist, schlägt der Erstellungsvorgang fehl.
- Anforderungen zum Auflisten von Objekten in einem Bucket sind nur dann erfolgreich, wenn der anfordernde Standort auf den Bucket-Eigentümer und alle Objekte zugreifen kann.
  - Wenn die Anforderung an Standort 2 gesendet wird, ist sie nur erfolgreich, wenn sie Eigentümer des Buckets und aller Objekte im Bucket ist.
  - Wenn die Anforderung an einen anderen Standort gesendet wird, ist sie nur erfolgreich, wenn weder der Bucket noch Objekte innerhalb des Bucket Standort 2 gehören.

4.2 Permanenter Standortausfall (PSO)

Wenn ein Ausfall an einem Standort auftritt und AdministratorInnen feststellen, dass der Standort nicht wiederhergestellt werden kann, können sie ein permanentes Standort-Failover initiieren (das VDC aus dem Verbund entfernen). Wenn ein permanentes Standort-Failover initiiert wird, werden alle Blöcke vom ausgefallenen Standort an den verbleibenden Standorten wiederhergestellt, um die Datenbeständigkeit wiederherzustellen.

Der Recovery-Prozess umfasst die verbleibenden Standorte, die ihre lokale Blockmanagertabelle durchsuchen und nach Referenzen auf Standorte suchen, die den ausgefallenen Standort enthalten. Alle, die es mit einem Blocktyp von findet:

- **Kodiert**
  a. Für Blöcke, deren Typ kodiert ist und deren primärer Standort online ist, werden die Daten lokal mithilfe der Daten vom primären Standort neu erstellt. Wenn der Vorgang abgeschlossen ist, wird dieser Block als Kopietyp markiert.
  b. Als Nächstes wird der kodierte Block neu erstellt, dessen primärer Standort der ausgefallene Standort ist, indem ein XOR-Vorgang der zuvor neu erstellten Kopietypen mit dem Paritätsblock durchgeführt wird. Dieser Standort wird jetzt zum primären Standort der Blöcke und verfügt über einen lokalen Typ.
  c. Diese Blöcke werden dann zur Replikationswarteschlange hinzugefügt, um sie an andere Standorte zu replizieren, die in der Replikationsgruppe aufgeführt sind.

- **Kopie** und ein primärer Standort, der als fehlgeschlagener Standort aufgeführt ist, wird zum neuen primären Standort. Anschließend wird der Block zur Replikationswarteschlange hinzugefügt, die an einen neuen sekundären Standort repliziert werden soll.
- **Lokal** und der sekundäre Standort ist der ausgefallene Standort. Es wird eine Aufgabe eingefügt, um den Block an einen neuen sekundären Standort zu replizieren.

Sobald das Failover des permanenten Standorts gestartet wurde, ist der Zugriff auf die Daten, die dem ausgefallenen Standort gehören, erst verfügbar, nachdem der Failover-Prozess für den permanenten Standort abgeschlossen ist. Die Replikation von Daten ist von Failover-Vorgängen getrennt und muss daher nicht abgeschlossen werden, damit der Zugriff auf die Daten des ausgefallenen Standorts wiederhergestellt werden kann.

In einem Beispiel mit drei Standorten schlägt Standort 1 fehl Tabelle 21 und Tabelle 22 ist die Blockmanagertabelle der beiden verbleibenden Standorte.

Tabelle 21 Standort 2 Blockmanagertabelle

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Kopie
C2	Standort 2	Standort 3	Lokale
C3	Standort 1	Standort 3	Remote
C4	Standort 2	Standort 1	Lokale

Standort 2 würde Folgendes ausführen:

- Block **C1** zur Replikationswarteschlange hinzufügen, die repliziert werden soll. Standort 2 wird zum neuen primären Standort und der Standort mit dem neuen Block wird zum neuen sekundären Standort.
- Block **C4** zur Replikationswarteschlange hinzufügen, die repliziert werden soll, und den sekundären Standort in der Tabelle aktualisieren.

Tabelle 22 Standort 3 Blockmanagertabelle

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 2	Remote
C2	Standort 2	Standort 3	Kodiert
C3	Standort 1	Standort 3	Kodiert
C4	Standort 2	Standort 1	Remote
C5	Standort 3		Parität (C2 und C3)

Standort 3 würde Folgendes ausführen:

1. Block-**C2**-Daten lokal mithilfe der Daten vom primären Standort neu erstellen (**Standort 2**). Blocktyp in „Kopiert“ ändern.
2. Block **C3** mithilfe von **C2**-Daten und **C5**-Paritätsdaten mithilfe des XOR-Vorgangs C2 ⊕ C5 rekonstruieren. Standort 3 wird zum neuen primären Standort.
3. Block **C5** löschen.
4. Block **C3** zur Replikationswarteschlange hinzufügen, um sie erneut zu replizieren. Der Standort mit dem neuen Block wird zum neuen sekundären Standort.

Nach Abschluss des permanenten Standort-Failovers des Blockmanagers sind die Tabellen der beiden verbleibenden Standorte wie in Tabelle 23 und Tabelle 24 dargestellt.

Tabelle 23 Standort 2 Blockmanagertabelle nach Abschluss des PSO

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 2	Standort 3	Lokale
C2	Standort 2	Standort 3	Lokale
C3	Standort 3	Standort 2	Kopie
C4	Standort 2	Standort 3	Lokale

Tabelle 24 Standort 3 Blockmanagertabelle nach Abschluss des PSO

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 2	Standort 3	Kopie
C2	Standort 2	Standort 3	Kopie
C3	Standort 3	Standort 2	Lokale
C4	Standort 2	Standort 3	Kopie

4.2.1 PSO mit geo-passiver Replikation

Ein permanenter Standortausfall wird bei Daten, die mithilfe der geo-passiven Replikation repliziert werden, etwas anders gehandhabt. Geo-passive replizierte Daten stellen während eines PSO keine Datenbeständigkeit wieder her. Stattdessen wird er wieder eingerichtet, nachdem ein neuer dritter Standort zur Replikationsgruppe hinzugefügt wurde. Die PSO-Vorgänge unterscheiden sich je nachdem, ob der Standort, der dauerhaft ausgefallen ist, einer der Quellstandorte oder der Replikationszielstandort ist.

Der Recovery-Prozess umfasst immer noch die verbleibenden Standorte, die ihre lokale Blockmanagertabelle durchsuchen und nach Referenzen auf Standorte suchen, die den ausgefallenen Standort enthalten. Alle, die es mit einem Blocktyp von findet:

- **Kodiert** (am Replikationszielstandort vorhanden)
  - Für Blöcke, deren Typ kodiert ist und deren primärer Standort online ist, werden die Daten lokal mithilfe der Daten vom primären Standort neu erstellt. Wenn der Vorgang abgeschlossen ist, wird dieser Block als Kopietyp markiert.
  - Als Nächstes wird der kodierte Block neu erstellt, dessen primärer Standort der ausgefallene Quellstandort ist, indem ein XOR-Vorgang der zuvor neu erstellten Kopietypen mit dem Paritätsblock durchgeführt wird. Dieser Standort wird jetzt zum primären Standort der Blöcke und verfügt über einen lokalen Typ. Es werden keine sekundären Standorte erstellt, bis ein dritter Standort zur Replikationsgruppe hinzugefügt wurde.
- **Kopie** und ein primärer Standort, der als fehlgeschlagener Standort aufgeführt ist, wird zum neuen primären Standort und sein Typ wird in lokal geändert. Es werden keine sekundären Standorte erstellt, bis ein dritter Standort zur Replikationsgruppe hinzugefügt wurde.
- **Lokal** und sein sekundärer Standort (das Replikationsziel) ist der ausgefallene Standort. Es werden keine neuen sekundären Standorte erstellt, bis ein dritter Standort zur Replikationsgruppe hinzugefügt wurde.

Nach dem PSO kann ein dritter Standort hinzugefügt werden, um die Datenbeständigkeit wiederherzustellen und vor standortweiten Ausfällen zu schützen. Nachdem ein dritter Standort zur Geo-Passiven-Replikationsgruppe hinzugefügt wurde, scannen die beiden vorherigen Standorte ihre lokale Blockmanagertabelle, um nach Blöcken zu suchen, für die kein sekundärer Block aufgeführt ist. Es wird dann Folgendes geben:

- Lokale Blöcke an einem Quellstandort ohne aufgeführten sekundären Block initiieren die Replikation eines sekundären Blocks zum neuen Replikationszielstandort. Die Blockmanagertabelle wird aktualisiert, um den neuen sekundären Blockspeicherort einzuschließen.
- Lokale Blöcke am Replikationszielstandort initiieren eine Replikation des Blocks an einen neuen Quellstandort. Nach Abschluss der Replikation ändert sich der Typ des Replikationszielstandorts von „Lokal“ zu „Kopie“ und der Typ des Quellstandorts von „Kopie“ zu „Lokal“. XOR-Vorgänge werden auf dem Ziel wie gewohnt fortgesetzt.

Sobald der PSO an einem Quellstandort gestartet wird, ist der Zugriff auf die Daten, die dem ausgefallenen Standort gehören, erst verfügbar, nachdem der Failover-Prozess für den permanenten Standort abgeschlossen ist. Sobald der PSO vollständigen Zugriff auf die Daten hat, wird wiederhergestellt. Bis ein dritter Standort zur Replikationsgruppe hinzugefügt wird, werden alle neuen Schreibvorgänge am Onlinequellstandort auf das Replikationsziel repliziert, aber XOR-Vorgänge werden nicht durchgeführt. Dies liegt daran, dass XOR nur auf Blöcken von zwei unterschiedlichen Quellstandorten ausgeführt wird, da alle neuen Quellstandorte identisch sind, kann XOR nicht ausgeführt werden.

Nach Abschluss des PSO kann der Replikationsgruppe ein dritter Standort hinzugefügt werden, um die Datenbeständigkeit wiederherzustellen und sich vor standortweiten Ausfällen zu schützen. Darüber hinaus kann das Replikationsziel die Ausführung von XOR-Vorgängen auf zwei beliebigen Blöcken von verschiedenen Quellstandorten fortsetzen, die als Typ „Kopie“ gekennzeichnet sind.

Sehen wir uns einige Beispiele an, bei denen Standort 1 und Standort 2 die Quellstandorte sind und Standort 3 der Zielstandort ist. Tabelle 25 und Tabelle 26 sind die Blockmanagertabellen der beiden Quellstandorte und Tabelle 27 ist die Blockmanagertabelle des Replikationszielstandorts.

Tabelle 25 Standort 1, Tabelle des Quellstandortblockmanagers

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Lokale
C2	Standort 2	Standort 3	Remote
C3	Standort 1	Standort 3	Lokale

Tabelle 26 Standort 2, Tabelle des Quellstandortblockmanagers

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Remote
C2	Standort 2	Standort 3	Lokale
C3	Standort 1	Standort 3	Remote

Tabelle 27 Standort 3, Tabelle des Replikationszielblockmanagers

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Kodiert
C2	Standort 2	Standort 3	Kodiert
C3	Standort 1	Standort 3	Kopie
C4	Standort 3		Parität (C1 und C2)

Beispiel 1: Wenn Standort 3 aufgrund eines PSO entfernt wird, werden die sekundären Standorte alle geleert, aber die primären Standorte und Typen bleiben. Bis ein neues Replikationsziel hinzugefügt wird, wird bei allen neuen Schreibvorgängen ein primärer Standort aufgelistet, aber kein sekundärer Standort.

Beispiel 2: Wenn Standort 1 aufgrund eines PSO entfernt wird, geschieht Folgendes:

- Standort 3 wird zum neuen primären Standort mit dem Typ „Lokal“ für Block **C3** und kein Standort wird als sekundärer Standort aufgeführt.
- Standort 3 erstellt blockbasierte **C2**-Daten mithilfe der Daten vom primären Standort (Standort 2) neu und ändert den Blocktyp in „Kopie“.
- Standort 3 rekonstruiert Block **C1** mithilfe von **C2**-Daten und den **C4**-Paritätsdaten mithilfe des XOR-Vorgangs C2 ⊕ C4. Standort 3 wird zum neuen primären Standort. Es wird kein sekundärer Standort aufgeführt.
- Standort 3 löscht Block **C4.**

Nach Abschluss eines permanenten Standort-Failovers von Standort 1 sind die Blockmanagertabellen der beiden verbleibenden Standorte wie in Tabelle 28 und Tabelle 29 dargestellt.

Tabelle 28 Standort 2, Tabelle des Quellstandortblockmanagers nach Abschluss des PSO

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 3		Remote
C2	Standort 2	Standort 3	Lokale
C3	Standort 3		Remote

Tabelle 29 Standort 3, Tabelle des Replikationszielstandort-Blockmanagers nach Abschluss des PSO

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 3		Lokale
C2	Standort 2	Standort 3	Kopie
C3	Standort 3		Lokale

Bis ein neuer Quellstandort hinzugefügt wird, haben alle neuen Schreibvorgänge einen primären Standort von Standort 2 mit dem Typ „Lokal“ und einen sekundären Standort von Standort 3 mit dem Typ „Kopie“.

Nachdem der Replikationsgruppe ein neuer Quellstandort hinzugefügt wurde, wird die Datenbeständigkeit zum Schutz vor standortweiten Ausfällen wiederhergestellt, indem ein sekundärer Standort hinzugefügt und die Daten darauf repliziert werden. XOR-Vorgänge werden auch auf dem Replikationsziel fortgesetzt.
Die neuen Blockmanagertabellen sind wie unter Tabelle 30 Tabelle 32 dargestellt.

- Die Blöcke C1 und C3 werden auf den neuen Quellstandort, Standort 1, repliziert. Nach Abschluss der Replikation wird der primäre Standort als Standort 1 und der sekundäre Standort als Standort 3 aufgeführt.
- Standort 3 führt die XOR-Kodierung auf den Blöcken C1 und C2 durch, was zu einem neuen C4-Block mit einem Paritätstyp führt und der Typ der Blöcke C1 und C2 in kodiert geändert wird.

Tabelle 30 Neue Tabelle „Standort 1 Blockmanager“ nach der Wiederherstellung der Datenbeständigkeit

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Lokale
C2	Standort 2	Standort 3	Remote
C3	Standort 1	Standort 3	Lokale

Tabelle 31 Standort 2 Blockmanagertabelle nach dem Hinzufügen des neuen Standorts 1 und Wiederherstellung der Datenbeständigkeit

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Remote
C2	Standort 2	Standort 3	Lokale
C3	Standort 1	Standort 3	Remote

Tabelle 32 Standort 3, Tabelle des Replikationszielstandort-Blockmanagers nach dem Hinzufügen des neuen Standorts 1 und Wiederherstellung der Datenbeständigkeit

Block-ID	Primärer Standort	Sekundärer Standort	Typ
C1	Standort 1	Standort 3	Kodiert
C2	Standort 2	Standort 3	Kodiert
C3	Standort 1	Standort 3	Kopie
C4	Standort 3		Parität (C1 und C2)

4.2.2 Wiederherstellbarkeit nach Systemausfällen an mehrerer Standorte

ECS unterstützt jeweils nur die Recovery nach einem Systemausfall an einem einzelnen Standort. ECS kann nach mehreren Systemausfällen am Standort wiederhergestellt werden, wenn sowohl PSO- als auch Data-Recovery-Vorgänge zwischen den Systemausfällen am Standort abgeschlossen sind. Wenn der zweite Standort vor Abschluss der Recovery fehlschlägt:

- Damit Recovery-Vorgänge für permanente Systemausfälle am Standort ausgeführt werden können, müssen alle anderen Standorte im System online sein. Wenn mehrere gleichzeitige Systemausfälle am Standort auftreten, muss bis auf einen vom TSO wiederhergestellt werden, bevor ein PSO an einem Standort ausgeführt werden kann.
- Bei einem zweiten Systemausfall am Standort, der nach Abschluss des PSO auftritt, aber bevor die Data Recovery abgeschlossen ist, gehen möglicherweise einige Daten verloren.

Wenn wir uns ein Szenario mit vier Standorten ansehen, stellen wir nach dem Verlust aller Standorte bis auf einen Standort erfolgreich wieder her (es wird davon ausgegangen, dass genügend Speicherplatz vorhanden ist, um alle Daten am verbleibenden Standort zu speichern):

- Standort 4 schlägt fehl
  - Die AdministratorInnen initiieren einen PSO-Vorgang zum Entfernen von Standort 4.
  - Daten werden an den verbleibenden Standorten wiederhergestellt, um die Datenbeständigkeit wiederherzustellen.

  Wir haben nun einen Verbund mit drei Standorten, der Standort 1, Standort 2 und Standort 3 enthält.
- Zweiter Standort schlägt fehl, Standort 2:

  Nach Abschluss des PSO und der Data Recovery kann ein anderer Standort fehlschlagen, z. B. Standort 2.

  - Die AdministratorInnen initiieren einen PSO-Vorgang zum Entfernen von Standort 2.
  - Daten werden an den verbleibenden Standorten wiederhergestellt, um die Datenbeständigkeit wiederherzustellen.

Wir haben nun einen Verbund mit zwei Standorten, der Standort 1 und Standort 3 enthält.

- Dritter Standort schlägt fehl, Standort 1:

  Nach Abschluss des PSO und der Data Recovery kann ein anderer Standort fehlschlagen, z. B. Standort 1.

  - Die AdministratorInnen initiieren einen PSO-Vorgang zum Entfernen von Standort 1.

  Wir haben jetzt einen Verbund mit einem einzigen Standort, der Standort 3 enthält.

In diesem Beispiel wurden mehrere Systemausfälle am Standort erläutert. Dies ist kein normales Szenario. permanente Systemausfälle am Standort werden in der Regel durch Katastrophenszenarien wie Erdbeben und Brände verursacht und treten daher nicht in kurzer Folge an mehreren Standorten auf. In der Regel wird nach einem dauerhaften Ausfall eines einzelnen Standorts ein neuer Standort hinzugefügt, bevor ein nachfolgender Standort ausfällt.

5 Fazit

Die ECS-Architektur wurde von Grund auf so konzipiert, dass sie sowohl Systemverfügbarkeit als auch Datenbeständigkeit bietet. Mit ECS können AdministratorInnen genau festlegen, wie sie die Verfügbarkeitsanforderungen mit den Gesamtbetriebskosten (TCO) in Einklang bringen. Funktionen wie automatische Fehlererkennung und Funktionen zur automatischen Fehlerkorrektur minimieren IT-administrative Workloads zu den kritischsten Zeiten, wenn es zu einem ungeplanten Ereignis wie einem Standortausfall kommt.

ECS schützt Daten innerhalb eines Standorts/VDC vor Festplattenausfällen mit einer Kombination aus dreifacher Spiegelung und Erasure Coding. ECS bietet zwei Ebenen des Erasure-Coding-Schutzes, die Standardeinstellung für typische Anwendungsbeispiele und Cold-Storage, der für Objekte mit seltenen Zugriffen effizienter ist. Außerdem werden die Daten über Fehlerdomains verteilt, um Schutz vor den meisten Ausfallszenarien zu bieten.

ECS sorgt für Datenintegrität, indem Prüfsummen im Rahmen eines Schreibvorgangs berechnet und geschrieben und diese Prüfsummen während eines Lesevorgangs validiert werden. Die Prüfsummenvalidierung wird auch proaktiv in einer Hintergrundaufgabe durchgeführt.

ECS ist darauf ausgelegt, die Systemverfügbarkeit weiterhin zu gewährleisten. Dies wird mit dem Design der verteilten Architektur erreicht, das es ermöglicht, Clientanfragen von jedem Node an einem Standort/VDC zu bearbeiten.

Das ECS-Design erweitert die Systemverfügbarkeit und den Schutz der Datenbeständigkeit, indem optionaler Schutz vor einem vollständigen standortweiten Ausfall hinzugefügt wird. Dies geschieht, indem Standorte verbunden werden und es den AdministratorInnen ermöglicht wird, eine Vielzahl von Replikationsgruppen-Policy-Optionen zu konfigurieren. Diese Optionen können auf Bucket-Ebene festgelegt werden und bestimmen, wo Daten repliziert werden sollen und wie die Daten an den Remotestandorten gespeichert werden sollen, sowie Zugriff während eines Ausfalls.

Darüber hinaus bietet ECS Kunden die Option „Zugriff während eines Ausfalls“, die Lese-, Listen- und optional Schreib- und Aktualisierungsvorgänge ermöglicht, die an einen Onlinestandort gesendet werden, wenn der Bucket und/oder das Objekt als fehlgeschlagen markiert ist.

Wenn AdministratorInnen feststellen, dass ein Standort nicht wiederhergestellt werden kann, können sie einen permanenten Standortausfall initiieren. Dadurch wird das VDC/der Standort aus der Replikationsgruppe entfernt und die Daten nach Bedarf neu erstellt, um die Datenbeständigkeitswiederherstellung zu sichern.

Zusammenfassend lässt sich sagen, dass ECS eine Cloud-Storage-Lösung der Enterprise-Klasse mit integrierter Ausfallsicherheit bietet, auf die Sie vertrauen können.

A Technischer Support und Ressourcen

Dell.com/support konzentriert sich auf die Erfüllung der Kundenanforderungen mit bewährtem Service und Support.

Technische Dokumentation und Videos zum Thema Speicher liefern das Know-how, das zum Kundenerfolg mit Dell EMC Storage-Plattformen beiträgt.

A.1 Zugehörige Ressourcen

- Whitepaper: ECS – Übersicht und Architektur
- ECS-Community
- ECS Test Drive
- ECS-Produktdokumentation auf der Support-Website oder auf der Community-Website
- SolVe-Desktop-PC (Verfahrensgenerator)