Fsas Technologies

P3AG-6032-03Z0

ETERNUS AX series オールフラッシュアレイ，ETERNUS HX series ハイブリッドアレイ

ONTAP での FlexCache ONTAP 9.11.1

目次

図目次

はじめに

FlexCache は、同じまたは異なる ONTAP クラスタ上にあるスパース (ボリュームのレプリカで、書き込み可能) を作成するキャッシュテクノロジーです。データやファイルをユーザーの近くに置くことで、より小さなリソースでより高速なスループットを実現できます。このドキュメントでは、FlexCache テクノロジーの仕組みについて詳しく説明し、設計と実装に関するベストプラクティス、制限、推奨事項、および考慮事項について説明します。

第 3 版
2025 年 3 月

登録商標

本製品に関連する他社商標については、以下のサイトを参照してください。
https://www.fujitsu.com/jp/products/computing/storage/trademark/
本書では、本文中の™、® などの記号は省略しています。

本書の読み方

対象読者

本書は、ETERNUS AX/HX の設定、運用管理を行うシステム管理者、または保守を行うフィールドエンジニアを対象としています。必要に応じてお読みください。

関連マニュアル

ETERNUS AX/HX に関連する最新の情報は、以下のサイトで公開されています。
https://www.fujitsu.com/jp/products/computing/storage/manual/

本書の表記について

本文中の記号

本文中では、以下の記号を使用しています。

注意	お使いになるときに注意していただきたいことを記述しています。必ずお読みください。
備考	本文を補足する内容や、参考情報を記述しています。

1. データはどこにでもある

情報のグローバル化に伴い、データは世界中に分散されます。大企業も中小企業も、世界中のデータを作成、保存、アクセスしています。データセンターは世界中で急増しており、すべてのデータを管理するためのデータ管理を強化する必要が生じています。

最近の IT ストレージの革新と、多くのアプリケーションのクラウドへの移行により、さらに多様なデータセットとネームスペースが作成されています。データ管理は大きな課題となっています。

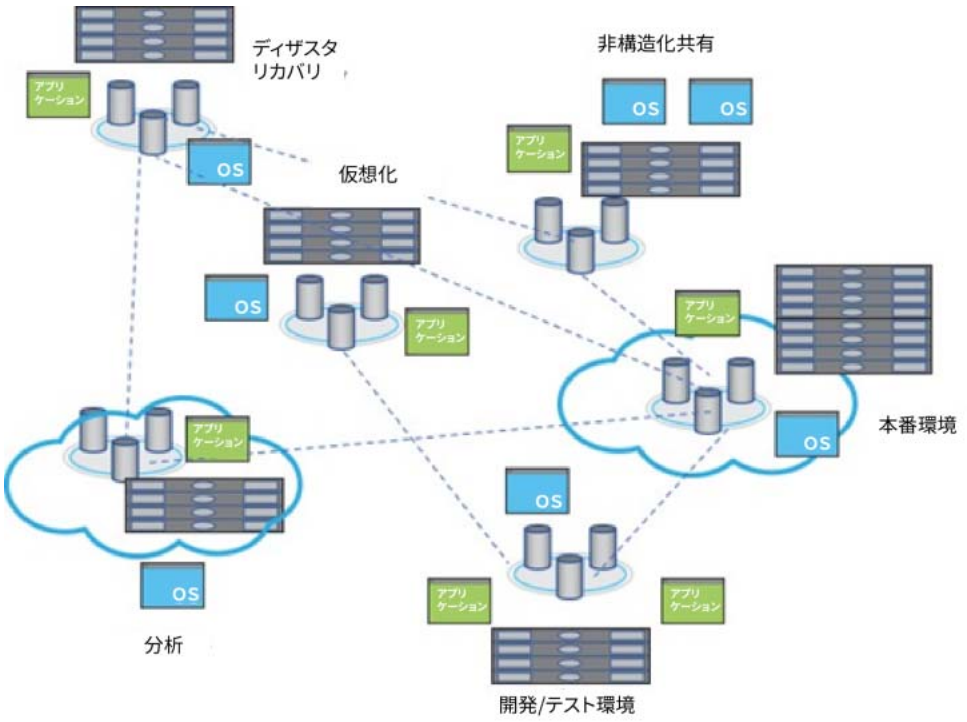
1.1 大規模なデータセットの管理が困難

人工知能 (AI)、機械学習 (ML)、ディープラーニング (DL) の登場により、データは一般的に多くのデバイスや場所から取り込まれるようになっています。結果として得られるデータセットは増加しており、さまざまな場所に適用する必要があります。データの量が増加しているだけでなく、データが存在する場所やデータにアクセスする場所も急激に増加しています。メディアレンダリング、財務アプリケーション、電子設計自動化 (EDA)、一般的な非構造化ファイル共有などの使用例では、ワークフローによってデータが 1 つの場所に作成され、別の場所でアクセスまたは分析されます。これらのワークフローは、アプリケーション、ユーザー、またはその他のリソースであるデータコンシューマに問題を引き起こす可能性があります。

問題は 2 つあります。まず、データが複数の場所に存在する場合、ネームスペースの問題があります。インフラストラクチャを統合して単一のネームスペースを作成しない限り、単一のネームスペースからデータにアクセスすることはできません。このインフラストラクチャでは、データを 1 つのファイルシステムとして認識し、すべての場所でデータの書き込み性、可読性、即時性を実現する必要があります。次に、遅延の問題があります。世界中、クラウドプロバイダ、または物理的に離れた場所からデータにアクセスすると、速度が低下し、問題が発生しやすくなります。

図 1.1 は、大規模なデータセットによって孤立したストレージとデータのサイロがどのように形成されるかを示しています。このようなサイロでは、連携、効率性、時間、その他の主要なデータ戦略を企業が利用できません。

図 1.1 ストレージサイロ



1.2 データレプリケーション

多くの機関が、データの爆発的増加に関連する問題の解決をデータレプリケーションに求めています。ただし、2 つのサイト間でデータをレプリケートすると、多くの点でコストが高くなります。

- **機器の二重化**
  サイト A に 100TB のデータがあり、サイト B からアクセスする場合は、100TB のデータを格納するスペースが必要です。つまり、サイト B のデータ利用者がサイト A のデータ利用者と同じような経験をするためには、ストレージプラットフォームが少なくとも類似している必要があります。
- **余分な装置が必要になる可能性**
  レプリケートされたデータを処理するためにインフラストラクチャを複製するだけでなく、レプリケーション構成を処理して監視するために新しいインフラストラクチャを導入する必要もあります。
- **遅延、遅延、そしてまた遅延**
  レプリケーションスキーマでは、変更されたデータのみを移動できますが、スケジュールに基づいてデータを移動すると、コストと遅延が発生します。オンデマンドレプリケーションは存在しますが、データ構造によっては、不必要な帯域幅の使用による遅延が発生する可能性があります。どちらにしても、あなたは遅延の代償を払い、結果として古いデータを提供する可能性があります。また、「現在のデータで作業しているか ?」または「最後にデータが同期されたのはいつか ?」といった問題にも直面しなければなりません。さらに、レプリケーションスキームが停止すると、すべてのダウンストリームも停止します。
- **複雑化**
  複数のリレーションシップを管理する必要があるため、重複する機器や追加のインフラストラクチャを管理するための追加の作業が必要になり、データ管理がより複雑になります。
- **ライタビリティ**
  複製によって、デスティネーションデータセットへの書き込みが許可されない場合があります。

1.3 データの同期

データの同期 (sync) によって、デスティネーションデータが書き込み可能であることを確認できます。双方向同期を構成することもできますが、レプリケーションで説明したコストに加えて、より多くのコストが発生する可能性があります。
データの同期を維持すると、レプリケーションの競合が発生する可能性があります。サイト A とサイト B で同じファイルへの書き込みが発生する可能性があります。これらのレプリケーション競合の調整には時間とコストがかかり、データが危険にさらされる可能性があります。

2. ONTAP の FlexCache: 進化

ONTAP の FlexCache は、リモートサイトにある矛盾なく一貫した最新のボリュームの書き込み可能な永続キャッシュを提供することで、これらの問題を解決します。

キャッシュは、ホストとデータソースの間にある一時的な保存場所です。キャッシュの目的は、ソースデータの頻繁にアクセスされる部分を、ソースからデータをフェッチするよりも高速にデータを処理できる方法で格納することです。キャッシュは、データが複数回アクセスされ、複数のホストによって共有される読み取り集中型の環境で最も効果的です。キャッシュは、次のいずれかの方法でデータを高速に処理できます。

- データソースがあるシステムよりも高速なキャッシュシステム。これは、キャッシュを提供するプラットフォームで、ストレージの高速化 (たとえば、HDD に対するソリッドステートドライブ (SSD))、処理能力の向上、またはメモリの増設 (またはより高速なメモリへの交換) によって実現できます。
- キャッシュのストレージ領域は物理的にホストに近いため、データに到達するまでにそれほど時間はかかりません。

キャッシュはさまざまなアーキテクチャ、ポリシー、セマンティクスで実装されるため、データがキャッシュに格納されてホストに提供されるときにデータの整合性が保護されます。

FlexCache テクノロジーには次のような利点があります。

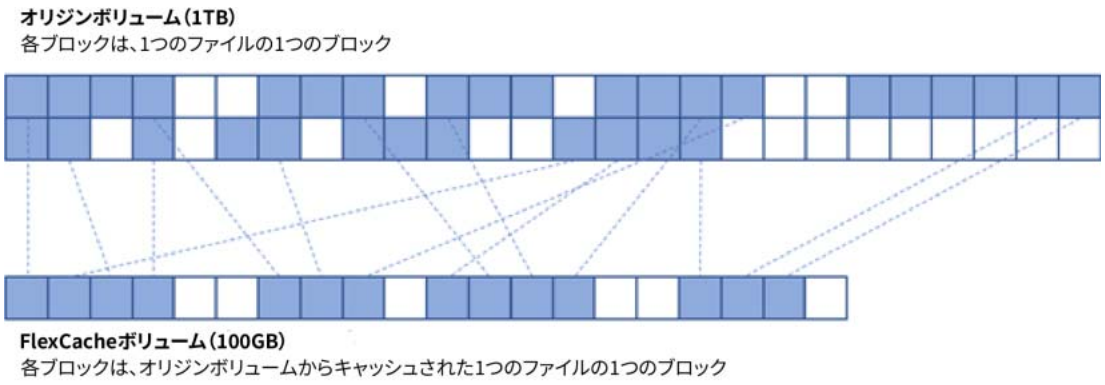
- 負荷分散によるパフォーマンスの向上
- クライアントアクセスのポイントにより近い場所にデータを配置することにより、レイテンシを削減
- ネットワークが切断された状況でも、キャッシュデータを提供することによる可用性の向上

FlexCache テクノロジーは、キャッシュの一貫性、データの一貫性、データの即時性、ストレージの効率的な使用を維持しながら、拡張性とパフォーマンスに優れた方法で、上記のすべてのメリットを提供します。

FlexCache ボリュームはスパースのコンテナです。オリジンデータセットのすべてのファイルがキャッシュされるわけではなく、キャッシュされた inode のすべてのデータブロックがキャッシュに存在できるわけでもありません。作業データセット (最近使用したデータ) の保存に優先順位を付けることで、ストレージを効率的に使用できます。

FlexCache テクノロジーを使用すれば、ディザスタリカバリの管理とその他の企業データ戦略をオリジンで実装するだけで済みます。データ管理はソース上でのみ行われるため、FlexCache テクノロジーを使用すると、リソースをより効率的に使用でき、データ管理とディザスタリカバリの戦略がシンプルになります。

図 2.1　スパースボリュームの詳細



2.1 用語

このセクションでは、FlexCache 固有の用語について説明します。

- オリジン
  FlexCache リレーションシップのソースボリュームです。
- FlexCache ボリューム (キャッシュボリューム)
  オリジンのスパースキャッシュであるデスティネーションボリュームです。

- FlexGroup ボリューム
  FlexGroup ボリュームは、複数のメンバーボリュームで構成される単一のネームスペースであり、ストレージ管理者にとっては FlexVol ボリュームのように管理および動作します。FlexGroup ボリューム内のファイルは、個々のメンバーボリュームに割り当てられ、ボリュームまたはノード間でストライプされません。これは、キャッシュボリュームのデフォルトのボリュームスタイルです。
- 読み取り負荷の高いワークロード
  ほとんどの操作が読み取りと書き込みである場合、データアクセスの読み取り負荷が高くなります。
- ライトバック (ライトビハインド)
  書き込み操作は、その操作が発生したキャッシュボリュームにのみ適用されます。この書き込みは、後でキャッシュライトバックポリシーに基づいてオリジンに適用されます。
- ライトスルー
  書き込み操作は、クライアントに書き込み確認 (ACK) が送信される前に、操作が発生したキャッシュボリュームとオリジンの両方に適用されます。
- ライトアラウンド
  書き込み操作は、キャッシュをバイパスしてオリジンに直接適用されます。キャッシュで書き込みを確認するには、キャッシュがオリジンから情報を引き出す必要があります。
- 作業データセット
  FlexCache でキャッシュされるオリジンに保存されるデータの合計のサブセットです。このデータセットの内容は、クライアントが FlexCache ボリューム要求にマウントした内容によって異なります。ほとんどのアプリケーション (EDA、AI、メディアレンダリング) では、FlexCache で読み込まれる、明確に定義されたファイルとディレクトリのセットを指します。
- Remote Access Layer (RAL)
  RAL は、WAFL システムの機能の 1 つであり、オリジンからキャッシュに対して inode で FlexCache が取消し可能な読み取り / 書き込みキャッシュ、または読み取り専用キャッシュを設定できるようにします。これは、FlexCache 機能を有効にする機能です。
- リモートエントリメタファイル (REM)
  FlexCache でアクティブにキャッシュされているすべてのファイルのオリジン情報を保持するデリゲーションのファイルです。
- リモートインデックスメタファイル (RIM)
  FlexCache でキャッシュされるすべてのファイルのデリゲーション情報を保持するキャッシュのファイルです。
- リモートロックエントリーメタファイル (RLEM)
  オリジンとキャッシュの両方でロック権限を保持するファイルです。
- FCMSID
  キャッシュ FlexGroup ボリュームの FlexGroup マスターセット ID (MSID) です。
- ファンアウト
  1 つのオリジンにアタッチできるキャッシュの総数です。
- 切断モード
  オリジンをホストしている ONTAP クラスタが、FlexCache ボリュームをホストしている ONTAP クラスタと通信できない状態です。
- ネットワークロックマネージャ (NLM)
  NFS クライアントが NFS サーバにロックコマンドとロック解除コマンドを送信できるようにする、NFSv3 のサイドバンドプロトコルです。
- Universal Naming Convention (UNC) パス
  ドライブをマップするためのパスです。最初の部分はホスト、2 番目の部分はマップするための共有とパスです (例： `\\svm1\share1`)。

2.2 FlexCache と同様の ONTAP 機能の違い

FlexCache ソフトウェアと同様の機能を提供する ONTAP 機能がいくつかあります。このセクションでは、これらの機能を高いレベルで比較します。

2.2.1 SnapMirror

SnapMirror テクノロジーは、リモートサイトのボリュームの読み取り専用コピーを提供します。このコピーは、非同期 (設定したスケジュールに従う) または同期 (SnapMirror 同期) に提供できます。リレーションシップが壊れていない限り、読み取り専用コピーにしかなりません。また、非同期はスケジュールに従ってのみ動作するため、データの即時性が遅延します。SnapMirror セカンダリを作成すると、データセット全体が新しいボリュームにコピーされ、スケジュールされた同期時間が経過すると、すべての変更がプライマリからセカンダリにコピーされます。この処理中のいずれかの時点で、SnapMirror セカンダリボリュームには、プライマリにあるすべてのデータの 100% が物理的に格納されます。この方法では、データサイズの 100% の容量が必要です。

3. 使用例

ONTAP 設計の FlexCache は、特定の状況で最もメリットがあり、それらの特定の状況は「理想的な例」として列挙されています。FlexCache ボリュームの他の使用例も可能ですが、その利点は十分に検証されていません。ほとんどの場合、使用状況はサポートされている機能セットに限定されます。理想的でない状況での使用は推奨されないことはありませんが、FlexCache テクノロジーの利点を理想的でない状況に関連するコストと比較した方が良いでしょう。

3.1 理想的な使用例

FlexCache テクノロジーはライトアラウンドモデルに限定されているため、読み取り負荷が高いワークロードでもうまく動作します。書き込みにはレイテンシが発生します。性能ペナルティは、キャッシュとオリジナル間のラウンドトリップタイム (RTT) に依存します。書き込みが少なくても、レイテンシはデータセットにアクセスするアプリケーションの全体的なパフォーマンスに影響しません。以下の例があります。

- 電子設計の自動化
- メディアレンダリング
- AI、ML、および DL のワークロード
- ホームディレクトリなどの非構造化 NAS データ
- Git などのソフトウェア構築環境
- 共通ツール配布
- ホットボリュームの性能分散
- クラウドのバースト、アクセラレーション、キャッシュ
- MetroCluster 構成をまたぐストレッチ NAS ボリューム

3.2 サポートされる機能

最新のサポート機能一覧については、「FlexCache ボリュームでサポートされる機能とサポートされない機能」を参照してください。

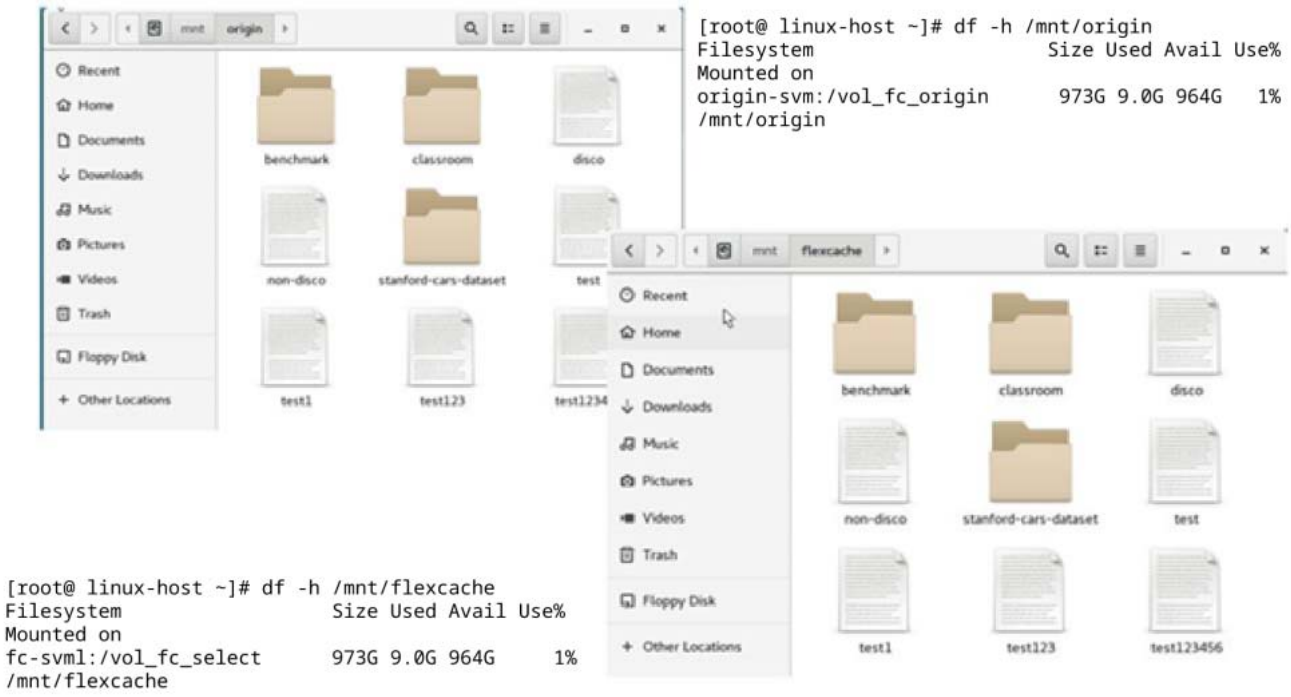
4. ONTAP の FlexCache 技術概要

このセクションでは、FlexCache テクノロジーの動作の技術的な概要について説明します。FlexCache は、通常「set and forget」機能なので、設定はあまり必要ありません。

4.1 スパースデータの詳細

FlexCache テクノロジーは、クライアント要求で読み取られたデータのみをキャッシュします。FlexCache ボリュームが作成され、クライアントが最初にそれをマウントするときには、そのボリュームの領域を占有するデータはありません。ただし、FlexCache ボリュームをマウントしたクライアントから見ると、キャッシュされているかどうかに関係なく、ボリュームはオリジンボリュームと同じように見えます。図 4.1 のオリジンは 1TB のボリュームですが、キャッシュのサイズはわずか 100GB です。ボリュームは作成されたばかりなので、キャッシュされたデータはありません。この図は、オリジンと FlexCache ボリュームのサイズとデータ内容がボリュームごとに異なるにもかかわらず、まったく同じに見えることを示しています。

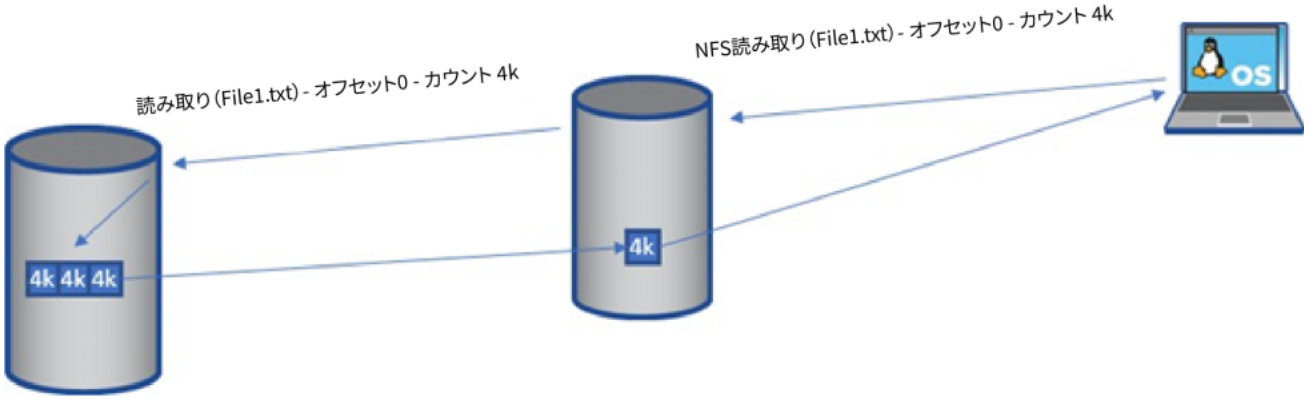
図 4.1 オリジンボリュームと FlexCache ボリュームの見え方



接続されているクライアントがデータを読み取ると、キャッシュが作成されます。詳細については、図 2.1 を参照してください。クライアントがファイルのブロックを読み取ると、それらのブロックもキャッシュボリュームに書き込まれ、FlexCache ボリュームに接続されたクライアントにサービスが提供されます。その結果、すべてのキャッシュが一意であり、すべて同じオリジンを指しているにもかかわらず、2 つのキャッシュが同じであることはありません。データの異なるブロックは、様々なキャッシュにキャッシュされます。

図 4.2 は、キャッシュ 1 のファイル 1 から 4096 バイトのデータを読み取るクライアントを示しています。ファイル 1 は 12KB のファイルなので、ファイルの 3 分の 1 だけがオリジンから取得され、FlexCache ボリュームにキャッシュされます。クライアントが残りのファイルを要求すると、オリジンからも取得されてキャッシュされます。その後、ファイルの任意の部分が読み取られると、キャッシュから直接処理されます。

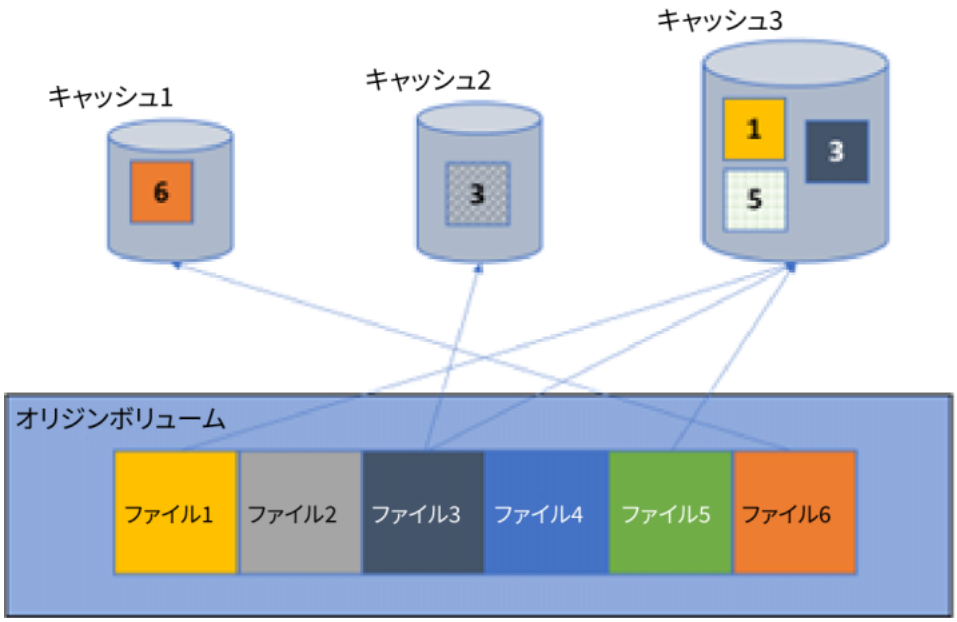
図 4.2　　スパースデータの詳細



FlexCache ボリュームには、マウントされているクライアントから要求されたデータしかないため、各キャッシュのデータは異なります。図 4.3 は、1 つのオリジンに対する複数のキャッシュが、どのように異なって見えるかを示しています。オリジンには 6 つのファイルがあります。キャッシュ 1 は最小のキャッシュボリュームで、ファイル 6 全体をキャッシュします。キャッシュ 2 はキャッシュ 1 よりもわずかに大きく、ファイル 3 の数ブロックだけをキャッシュします。キャッシュ 3 は最大のキャッシュで、ファイル 1 の全体 、ファイル 5 の数ブロック、ファイル 3 の全体をキャッシュします。

これらの違いは、キャッシュでのクライアント要求の直接的な結果です。キャッシュ 1 にマウントされたクライアントは、1 つの要求または複数の要求 (複数のクライアント間) を通じて、ファイル 6 全体を要求しました。キャッシュ 2 にマウントされたクライアントはファイル 3 の一部のみを要求し、キャッシュ 3 にマウントされたクライアントはファイル 1 のすべて、ファイル 5 のいくつかのブロック、ファイル 3 のすべてを要求しました。この場合も、キャッシュに接続されたクライアント間での要求の順列を使用して実行できます。

図 4.3　　キャッシュの変化



4.2 ポリシーと FlexCache のエクスポート

NFS エクスポートをマウントするには、アクセスするボリュームにルールを設定したエクスポートポリシーを適用する必要があります。オリジンボリュームが存在し、そこに NFS クライアントが接続されている場合は、そのボリュームに関連付けられた NFS アクセス用のエクスポートポリシーが既に存在します。

FlexCache ボリュームを作成しても、エクスポートポリシーは複製されません。同じ SVM で FlexCache ボリュームを作成する場合でも、これらは独立して存在しています。独立したエクスポートポリシーとは、各キャッシュに異なるエクスポートポリシーを関連付け、異なるクライアントに異なるアクセスを提供できることを意味します。特定のサブネットは別のキャッシュまたはオリジンに接続する必要があるため、この機能を使用して、それらのサブネットから特定のキャッシュへのアクセスを拒否できます。また、クライアントがキャッシュに書き込むことができないように、読み取り専用ルールを持つキャッシュを作成することもできます。

新しく作成された FlexCache ボリュームへのアクセスを許可するには、FlexCache ボリュームに関連付けられたエクスポートポリシーに正しいルールが関連付けられている必要があります。異なるクラスタ内の SVM 間で ONTAP のポリシーを複製する方法はありません。FlexCache ボリュームがオリジンと同じクラスタにあるが、異なる SVM にある場合は、`vserver export-policy copy` コマンドを使用して、ある SVM から別の SVM にエクスポートポリシーをコピーできます。FlexCache ボリュームとオリジンが同じ SVM にある場合は、同じエクスポートポリシーを両方のボリュームに適用できます。

4.3 SMB 共有と FlexCache

ONTAP 9.8 から、SMB プロトコルがキャッシュボリュームでサポートされるようになり、FlexCache ボリュームを指す SMB 共有を作成できるようになりました。エクスポートポリシーと同様に、FlexCache ボリュームの作成時に SMB 共有は複製されません。同じ SVM で FlexCache ボリュームを作成する場合でも、これらは独立して存在しています。これは、異なる共有アクセス許可をキャッシュに実装できることも意味します。キャッシュデータアクセスの詳細な制御が可能であり、必要に応じてキャッシュを読み取り専用に制限することもできます。

SMB 共有を使用して FlexCache のデータにアクセスする場合、オリジンのボリュームセキュリティスタイルはほとんどの場合 NTFS になります。NTFS ACL(アクセスコントロールリスト) および共有アクセス許可アプリケーションは、ONTAP 内の SMB サーバーの構成に大きく依存するため、アクセス許可がオリジンとキャッシュで同じように適用されるようにするには、いくつかの要件があります。

4.3.1 SMB/CIFS サーバ

オリジンとは別の SVM で FlexCache ボリュームを作成する場合、その SVM は別の論理エンティティになります。そのキャッシュで SMB プロトコルを使用するには、その SVM 上に SMB サーバーを作成する必要があります。オリジンボリュームのセキュリティスタイルが NTFS である場合、そのオリジンボリュームのデータ内には既に NTFS ACL が存在します。最も低いレベルでは、権限は SID 形式で格納され、キャッシュ SVM は、オリジンの SVM と同じドメインに作成された SMB サーバに必要な権限の解釈方法を理解する必要があります。SMB サーバーの信頼できるドメインへの作成がサポートされていますが、オリジン SVM と同じドメインに SMB サーバーを作成することを強くお勧めします。

- **ベストプラクティス 1: キャッシュとオリジン SVM は同じドメイン内にある必要がある**
  ACL と権限を正しく適用するには、キャッシュとオリジン SVM を同じドメインに配置する必要があります。

ワークグループモード

ACL は SID 形式であるため、SVM がワークグループモードで動作している場合、SID のドメイン部分は SVM ごとに異なるため、ある SVM は別の SVM のユーザ SID を解釈できません。このため、NTFS 形式のボリュームでキャッシュのワークグループモードを使用することはサポートされていません。両方の SVM 間で UID と GID を一致させることができるため、UNIX セキュリティスタイルのボリュームの使用がサポートされます。

4.4 RAL の概要

RAL(リモートアクセスレイヤー) は、FlexCache ソフトウェアがクラスタの内部または外部の任意のボリュームからオリジンへのコンテンツのロードと操作を可能にするために使用する機能です。RAL は FlexCache ボリュームとオリジン間のリクエスト、データ、デリゲーション、ロックリクエスト、その他の情報を同期させるトラフィック監視役です。FlexCache ボリュームとオリジンが別々のクラスタにある場合もあります。RAL が I/O 処理をオリジンボリュームを所有するノードに転送するには、クラスタが異なるとクラスタピアリングが必要です。

RAL 機能が呼び出されると、オリジンに転送する準備ができている操作はすべて、ノード間のクラスタインターコネクト LIF を介してリダイレクトされます。要求を開始および受信するノードは、発信元ボリュームの場所とオリジンボリュームの場所に依存します。運用効率を上げるために、ONTAP は FlexCache とオリジン間の通信を開始するノードが、FlexCache ボリュームが作成されるアグリゲートを所有するノードと同じになるように設計されています。デスティネーションノードは、アグリゲートボリュームが存在するオリジンを所有するノードです。

FlexCache テクノロジーはオリジンとキャッシュの両方でメタファイルを使用して、何がキャッシュされ、データまたはロックデリゲーションが有効かを追跡します。データデリゲーションはオリジン用の REM ファイルとキャッシュ用の RIM ファイルで追跡されます。ロックデリゲーションは、オリジンとキャッシュの両方に存在する RLEM メタファイルで追跡されます。これらはストレージ管理者には表示されず、照会もできません。

4.5 読み取り処理

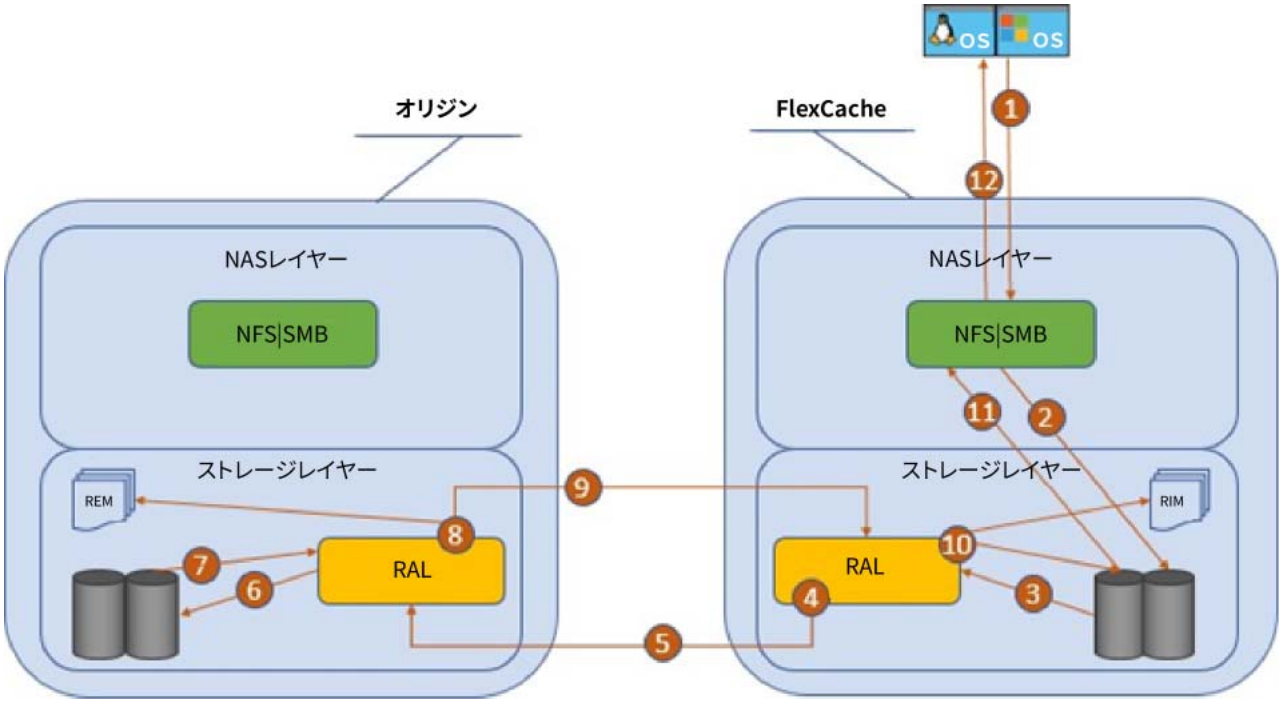
クライアントがファイルの読み取りを発行する場合、FlexCache ボリュームが標準の読み取りプロセスからフォークする方法はいくつかあります。まず、ファイルがローカルで見つからない場合、読み取り要求がオリジンに転送されます。このプロセスは、オリジンが ENOENTRY(または「ファイルが見つかりません」) エラーを返すことに責任を持つことを意味します。次に、ファイルがローカルで見つかった場合、FlexCache ボリュームのローカル RIM ファイルを調べてデリゲーションが取り消されていないことを確認する必要があります。デリゲーションエントリが削除されている場合は、要求されたブロックをオリジンから再度読み取る必要があります。以下に、各読み取りシナリオの詳細を示します。

4.5.1 File Not Cached

これは、FlexCache ボリュームを最初に作成するときに発生するシナリオです。すべての読み取りはキャッシュされた inode を持たず、データを得るためにオリジンに転送されなければなりません。
図 4.4 は、次の手順を示しています。

1. クライアントからのプロトコル要求が NAS レイヤーに到達します。
2. NAS レイヤーはオペレーションを解析し、最適化されたオペレーションをストレージレイヤーに渡します。
3. 次に、ストレージレイヤーは、操作が FlexCache(リモート) inode 上にあるかどうかを判断します。inode をロードしようとすると、キャッシュされていないことを検出して RAL を起動します。この検出によって、ストレージ操作も一時停止されます。
4. RAL は、オリジンから inode を取得するためのリモートストレージ操作を生成します。
5. リモート検索操作は、クラスタのクラスタ間 LIF を介してオリジンノードに送信されます。
6. オリジン上の RAL モニタは、要求を受信し、ドライブのストレージ操作を生成します。
7. inode がドライブから取得されます。(適切なファイルのロックが確認されます)
8. 次に、RAL はデリゲーション用の REM ファイルへのエントリーを作成し、FlexCache への応答を生成します。
9. 応答は、クラスタのクラスタ間 LIF を介して FlexCache ノードに送り返されます。
10. RAL は応答を受け取り、inode のデータをローカルドライブに保管し、この inode に関するエントリーを RIM ファイルに作成します。
11. 元のストレージオペレーションが再開され、データが取得され、応答が NAS レイヤーに送り返されます。
12. NAS レイヤーは、プロトコル応答をクライアントに返します。

図 4.4 File Not Cached の読み取り手順

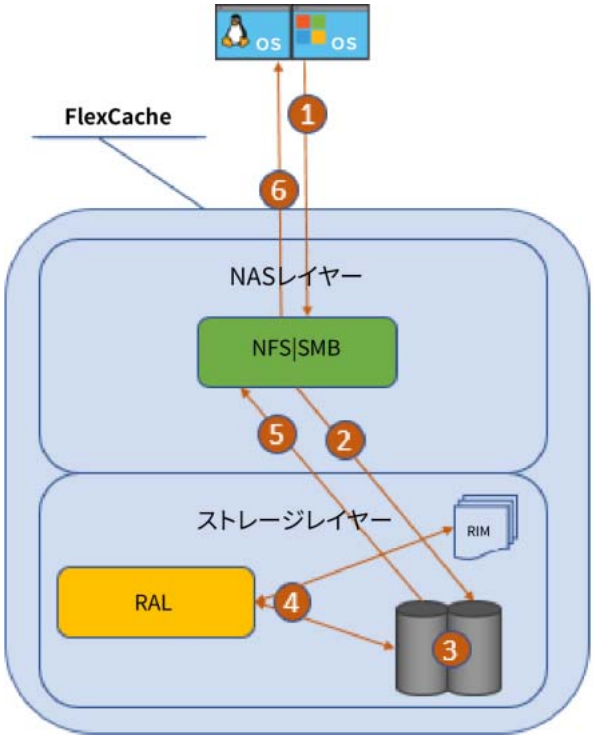


4.5.2 File Cached and Valid

File Cached and Valid は、ファイルが FlexCache ボリュームにキャッシュされ、デリゲーションがまだ有効である理想的なシナリオです。オリジンとの通信がないため、キャッシュファイルからのデータの提供に遅延が生じないことに注意してください。このシナリオは、FlexCache 以外のボリュームからの読み取りとほぼ同じように機能します。図 4.5 は、次の手順を示しています。

1. クライアントからのプロトコル要求が NAS レイヤーに到達します。
2. NAS レイヤーはオペレーションを解析し、最適化されたオペレーションをストレージレイヤーに渡します。
3. 次に、ストレージレイヤーは、操作が FlexCache(リモート)inode 上にあるかどうかを判断します。inode をロードしようとすると、それを見つけます。
4. これは FlexCache inode なので、RAL が介入し、RIM inode ファイルにデリゲーションエントリーがあるかどうかを判断します。
5. デリゲーションエントリーが見つかったため、データが検索され、応答が NAS レイヤーに送り返されます。
6. 次に、NAS レイヤーはプロトコル応答をクライアントに送り返します。

図 4.5 File Cached and Valid の手順



4.5.3 File Cached but Not Valid

File Cached but Not Valid は、元のファイルはキャッシュされたが、オリジンで何かが変更され、キャッシュされたデリゲーションが無効になったというシナリオです。このシナリオは、ファイルがキャッシュされていても最新ではなく、オリジンから再フェッチする必要があることを意味します。ONTAP 9.8 より前のバージョンでは、デリゲーションの無効化はファイル単位でのみ行われていたため、オリジンでの変更はファイル全体を無効化します。ONTAP 9.8 には、キャッシュされたファイルデータをブロックレベルで無効にする、ブロックレベルの無効化を有効にするオプションが追加されています。図 4.6 に、このシナリオの手順の概要を示します。

1. クライアントからのプロトコル要求が NAS レイヤーに到達します。
2. NAS レイヤーはオペレーションを解析し、最適化されたオペレーションをストレージレイヤーに渡します。
3. 次に、ストレージレイヤーは、操作が FlexCache(リモート)inode 上にあるかどうかを判断します。inode をロードしようとすると、それを見つけます。
4. これは FlexCache inode なので、RAL が介入し、RIM inode ファイルにデリゲーションエントリーがあるかどうかを判断します。
5. デリゲーションエントリーが有効でないため、ストレージ操作が一時停止され、RAL はオリジンから inode を検索するためのリモートストレージ操作を生成します。
6. リモート検索操作は、クラスタのクラスタ間 LIF を介してオリジンノードに送信されます。
7. オリジン上の RAL モニタが要求を受信し、ドライブのストレージオペレーションを生成します。
8. inode がドライブから取得されます。

9 次に、RAL はデリゲーション用の REM ファイルへのエントリーを更新し、FlexCache ノードへの応答を生成します。

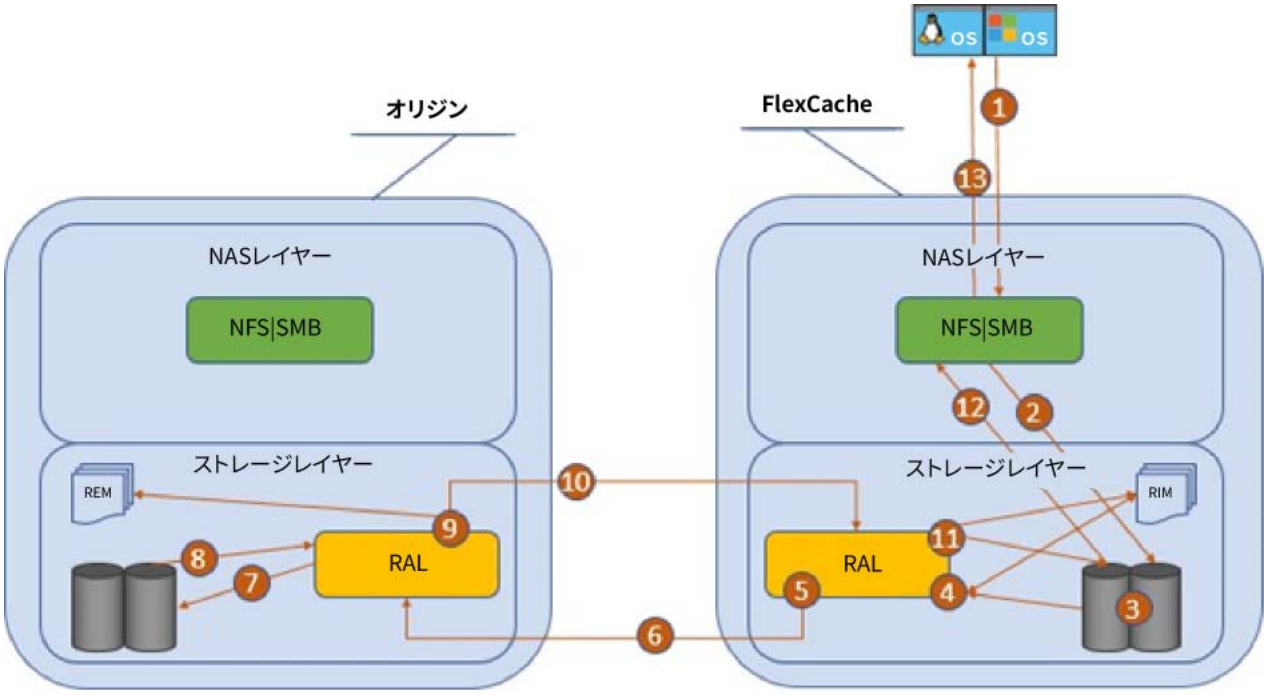
10 応答は、クラスタのクラスタ間 LIF を介して FlexCache ノードに送り返されます。

11 RAL は応答を受け取り、inode のデータをローカルドライブに保管し、この inode に関する項目を更新するか、RIM ファイルに項目を作成します。

12 元のストレージオペレーションが再開され、データが取得され、応答が NAS レイヤーに送り返されます。

13 NAS レイヤーは、プロトコル応答をクライアントに返します。

図 4.6 File Cached but Not Valid の手順



このように、読み取り要求を送信する方法はいくつかあり、取得されるパスはファイルがキャッシュされているか、デリゲーションが有効かどうかによって異なります。オリジンがキャッシュファイルを無効にできるシナリオはいくつかあります。1 つは、ファイルへの書き込みがある場合です。この書き込みは、キャッシュから実行することも、オリジンで直接実行することもできます。次のシナリオは、オリジンでファイルの読み取りがある場合です。デフォルトでは、ボリュームの atime-update プロパティは true に設定されているため、オリジンでの読み取りが多いと、FlexCache ボリュームがオリジンから継続的に読み取られる原因となり、多くのキャッシュファイルが無効になる可能性があります。この設定は最適ではありません。したがって、ONTAP 9.10.1 以前ではオリジンボリュームで `atime-updates` を無効にすることを推奨します。FlexCache ボリュームが作成されると、`atime-updates` はすでに無効になっているため、FlexCache ボリュームに対してこのアクションを実行する必要はありません。ONTAP 9.11.1 以降では、ファイルアクセス時間の更新の動作が改善されました。オリジンで `-atime-update-period` を設定する際にオプションが使用可能です。`atime-update-period` に 0 より大きい値を設定した場合、読み込み操作では atime 値が変更できなくなったため、キャッシュしたデータを削除しなくてもよくなりました。`atime-update-period` が経過した後は、次の読み込み操作で atime 値が更新されるため、キャッシュが無効になります。

- **ベストプラクティス 2a ONTAP 9.10.1 以前の場合：オリジンの `atime-updates` を `False` に設定**

  オリジンで読み取りが行われただけのときにキャッシュされたファイルが無効になるのを防ぐには、オリジンボリュームの最終アクセス時刻の更新を無効にします。

```
origin_cluster::*> volume modify -vserver origin-svm -volume vol_fc_origin
-atime-update false
```

- **ベストプラクティス 2b ONTAP 9.11.1 以降の場合：オリジンの `atime-updates` を `True` に設定し、`atime-update-period` を指定**

  キャッシュされたファイルが強制的に無効化されるのを防ぎつつファイルアクセス時間の更新機能は維持したい場合、オリジンボリュームで `atime update` を有効にして `atime-update-period` を指定します。

```
origin_cluster::*> volume modify -vserver <origin-svm> -volume <vol-fc-origin>
-atime-update true -atime-update-period <seconds>
```

4.6 ネガティブルックアップキャッシュ

ONTAP 9.9.1 以前では、キャッシュに存在しないファイルを検索すると、常にオリジンにアクセスするための RTT が発生します。ONTAP 9.9.1 では、ネガティブルックアップキャッシュ (NLC) が導入されました。オリジンから返された ENOENT は、FlexCache ボリュームにキャッシュされます。その後、同じファイルを検索すると、FlexCache ボリュームのキャッシュから ENOENT が返されるため、オリジンにアクセスするための RTT が発生しません。NLC は、親ディレクトリがオリジンで変更されるまで有効です。NLC は、ONTAP 9.9.1 以降ではデフォルトで有効になっています。

4.7 ライトアラウンド処理

ライトアラウンド処理は、書き込みプロセスを単純にフォークするものです。ONTAP はこれが FlexCache ボリュームであることを認識しているため、ドライブにローカルに書き込む代わりに、最適化された書き込み要求が直接オリジンに転送されます。
その後、オリジンは、オリジンに直接接続されているクライアントから書き込み要求が要求された場合とまったく同じように、書き込みを処理します。このプロセスにより、オリジンは同時書き込み、ロック、およびキャッシュの無効化を調整できます。書き込みはキャッシュに記録されません。図 4.7 は、このプロセスを示しています。変更中のファイルが FlexCache ボリュームでキャッシュされていても、書き込みはオリジンによって実行されます。その後、キャッシュされたファイルは無効になります。

書き込みの手順は次のとおりです。

1. クライアントからのプロトコル要求が NAS レイヤーに到達します。
2. NAS レイヤーはオペレーションを解析し、最適化されたオペレーションをストレージレイヤーに渡します。
3. 次に、ストレージレイヤーは、操作が FlexCache(リモート)inode 上にあるかどうかを判断します。書き込み要求を RAL に転送します。
4. RAL は、データをオリジンに書き込むためにリモート書き込み要求を生成します。
5. リモート書き込み要求は、クラスタ間 LIF を介してオリジンノードに送信されます。

6 オリジン上の RAL モニタが要求を受信し、ドライブのストレージオペレーションを生成します。

7 その後、データがドライブに書き込まれます。

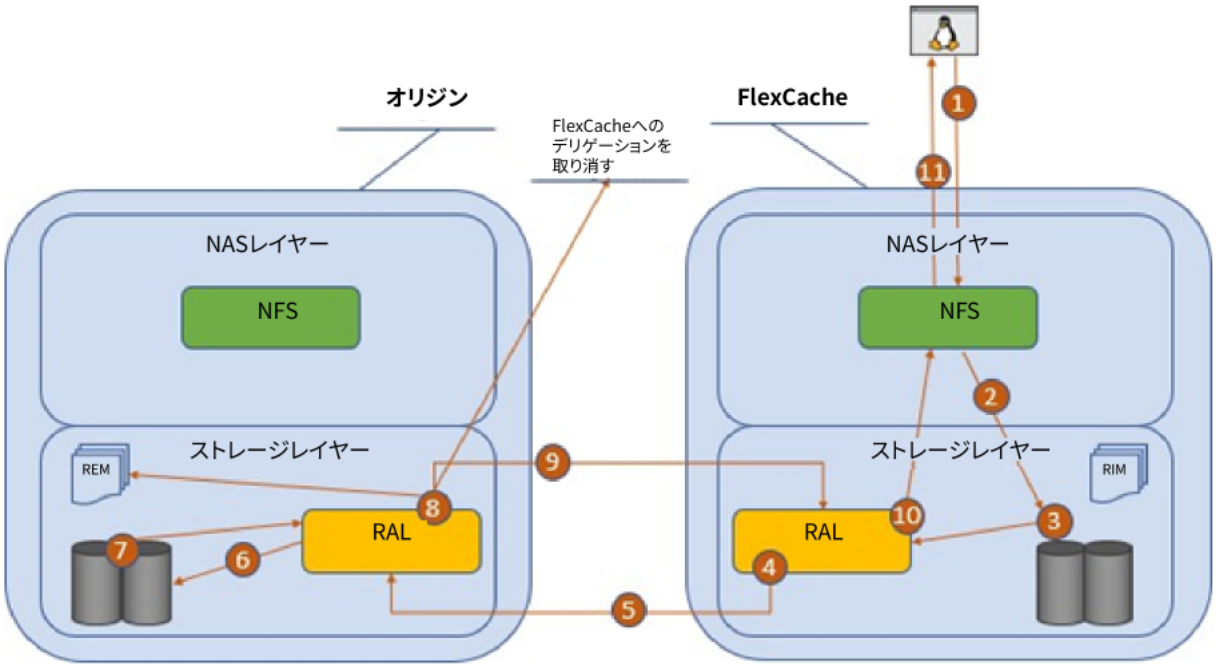
8 既存のファイルである場合、RAL は REM ファイルにデリゲーションがあるかどうかをチェックします。REM ファイルに有効なデリゲーションが存在するファイルエントリは、各 FlexCache ボリュームと通信して、そのファイルのデリゲーションを取り消します。この無効化は、ファイルを書き込んでいる FlexCache ボリュームがすでにキャッシュされている場合に発生する可能性があります。

9 応答は、クラスタのクラスタ間 LIF を介して FlexCache ノードに送り返されます。

10 RAL が応答を受信し、応答が NAS レイヤーに送り返されます。

11 NAS レイヤーは、プロトコル応答をクライアントに返します。

図 4.7 ライトアラウンド



キャッシュには書き込みがないため、書き込み後読み取り処理を実行するアプリケーションではパフォーマンスの問題が発生します。これらは通常、書き込まれた内容を確認する設定を持つアプリケーションです。この設定は FlexCache には適していません。FlexCache がそのアプリケーションのインフラストラクチャの一部である場合は、パフォーマンスが十分かどうかを幅広くテストする必要があります。

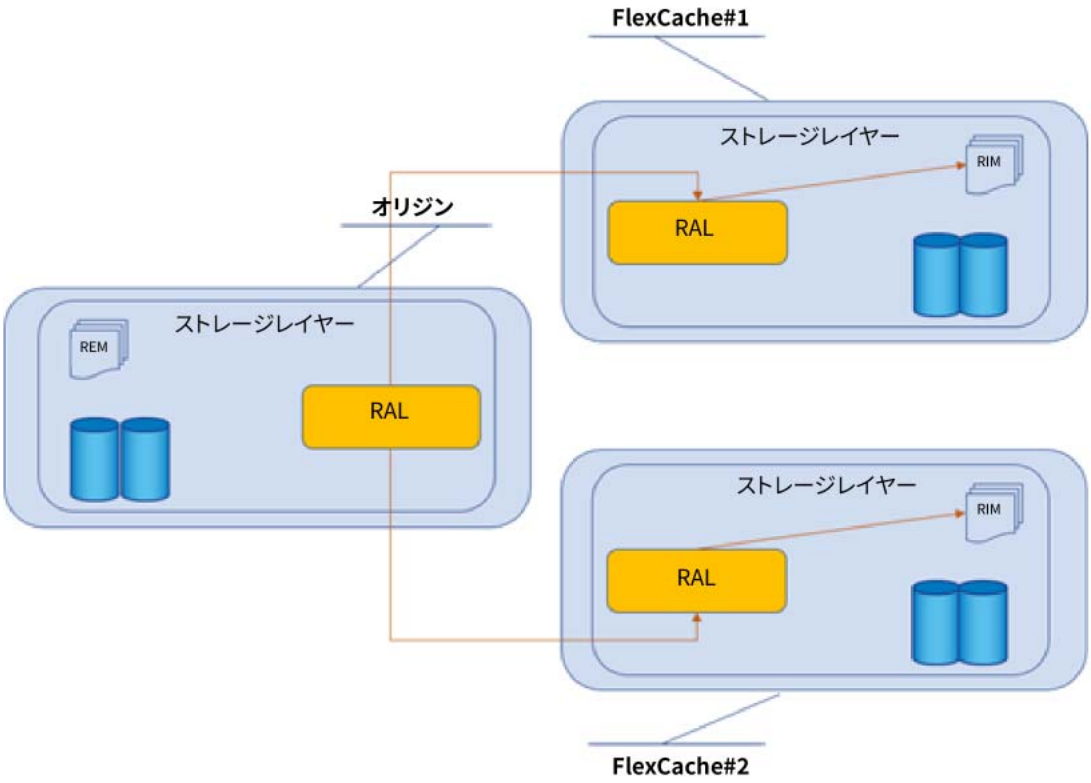
- **ベストプラクティス 3: リードアフターライトを使用しない**

  read-after-write で書き込みを確認するアプリケーションは使用しないようにしてください。FlexCache テクノロジーのライトアラウンド特性により、このようなアプリケーションで遅延が発生する可能性があります。

4.7.1 キャッシュの無効化

オリジンの REM ファイルとキャッシュの RIM ファイルはすべての同期を保っています。これらのファイルには、キャッシュされるファイル、ファイルがキャッシュされる場所、およびデリゲーションエントリーが有効かどうか、または除去されたかどうかが記録されます。
書き込みが発生すると、オリジンの RAL レイヤが REM ファイルをチェックします。前述のように、REM ファイルには、ファイルが FlexCache ボリュームにキャッシュされたかどうか、およびキャッシュされたキャッシュに関する情報が保持されます。書き込まれるファイルが REM ファイル内にエントリを有する場合、RAL は、inode のデリゲーションエントリを無効にするメッセージを作成し、クラスタ間 LIF を介してキャッシュされたファイルを有するすべてのキャッシュに送信する。キャッシュはそのメッセージを受け取り、RIM にあるその inode のデリゲーションを無効にするか削除します。次に任意のキャッシュでファイルが読み取られると、デリゲーションは無効にされているため、オリジンから読み取られます。
図 4.8 は、キャッシュの無効化を示しています。

図 4.8 キャッシュの無効化



ブロックレベルの無効化

ONTAP 9.8 はブロックレベルの無効化を導入しており、書き込みがあったときにキャッシュでファイル全体を無効にするのではなく、変更されたブロックだけをキャッシュで無効にします。このプロセスは、大きなファイルがあり、ファイルの数ブロックしか変更されていない場合に役立ちます。変更後にキャッシュで読み取る場合、ファイル全体ではなく、変更されたブロックのみをオリジンから再読み取りする必要があります。注意すべき点は、書き込み時には、各書き込み要求は最終的に各キャッシュに無効化要求が送信されるということです。これにより、書き込み遅延が発生する可能性があります。ワークフローがブロックレベル無効化機能の使用に適しており、書き込みに不必要な遅延が発生しないことを確認してください。ブロックレベル無効化機能を有効にすると書き込み遅延が許容できなくなる場合は、ブロックレベル無効化機能を無効にすることをお勧めします。

ブロックレベルの無効化機能を有効にするためには、オリジンのクラスタで以下の Advanced コマンドを実行してください。

```
origin_cluster::*> flexcache origin config modify -origin-vserver <svm>
-origin-volume <origin_volume> -is-bli-enabled true
```

4.8 ロック

データの取得と無効化は一元的に管理されますが、ロックは一元的に調整されます。ロックは主にパフォーマンスのために分散されていますが、集中管理されたプロパティもあります。ONTAP 9.8 以前は、NLM がキャッシュでサポートされる唯一のロック機構であり、すべてのロックはオリジンに転送されて一元的に追跡されていました。NLM ロックはアドバイザリであるため、グローバル NLM ロック機能を提供するには、この調整で十分です。キャッシュで SMB および NFSv4.x をサポートしたことにより、強制ロック、およびキャッシュの読み取り / 書き込み / ファイル制御のためにファイルを開いてロックを要求するコンセプトが導入されました。このプロセスによって、クライアントがより効率的に操作をキャッシュし、それらをまとめてサーバーに送り返すことができます。また、ONTAP は、キャッシュがオリジンに接続しなくても特定のタイプのロックを許可できるように、これらのロックを効率的に処理する必要があります。この概念は、前に説明した RLEM メタファイルを使用して、データデリゲーションを処理するのと非常によく似た方法で管理されます。ONTAP for FlexCache のこの分散ロックアーキテクチャは、すべてのロックを順番に維持し、すべてのロックが確実に考慮され、容易に強制されるようにします。

読み取りロックデリゲーションと書き込みロックデリゲーションは、どちらも RLEM メタファイルで追跡されます。この分散パラダイムでは、すべての読み取りロックがデリゲートされ、キャッシュに分散されます。書き込みロックにはハイブリッドアーキテクチャがあり、NLM 書き込みロックは引き続きオリジンでのみ転送および付与されますが、SMB および NFSv4.x の書き込みロックは要求されるとキャッシュにデリゲートされます。ONTAP がロックを要求するタイミングと方法の詳細については、次のセクションで説明します。

4.8.1 最初の読み取りでの読み取りロックデリゲーションキャッシュ

ファイルの最初の読み取りが行われ、ファイルのデータがキャッシュされていない場合、FlexCache ノードは、クライアントがロックを要求しているかどうかに関係なく、データの要求時にそのファイルに対して適切なロックデリゲーションを要求します。このプロセスでは、トラフィックが最適化されて、1 つの操作だけがオリジンに送信され、データとともにロックデリゲーションが取得されます。その結果、ファイルに対して特定のタイプのロックを要求する後続のクライアント操作は、キャッシュがオリジンにアクセスする必要なく許可されます。要求される初期ロックデリゲーションは、読み取りキャッシュロック、拒否キャッシュロック、または処理キャッシュロックです。つまり、他のクライアントがキャッシュロックの読み取り、拒否、または処理を要求した場合でも、キャッシュクラスタは、オリジンにアクセスしてキャッシュファイルのロックタイプを許可する必要がありません。

4.8.2 オープン操作のための読み取りロック処理

キャッシュでの SMB および NFSv4.x 読み取り操作も、クライアントがファイル内のデータを読み取る前にファイルを開くように要求します。この要求には、クライアントが操作を合理化して特定の操作をキャッシュするために要求するロック操作が含まれます。ほとんどの場合、クライアントは読み取りロック、拒否ロック、およびキャッシュロックの処理のみを要求します。これらは、特定のファイルに対する初期キャッシュ操作時に要求されるので、ロックデリゲーションを無効にする操作がない限り、キャッシュは、オリジンにアクセスすることなく、これらのロックを付与することができます。ロックデリゲーションが無効になっている場合は、クライアントのオープンコールに応答する前に、キャッシュがデータ読み取りと同じ接続を介してオリジンにアクセスし、ロックを要求する必要があります。データデリゲーションとロックデリゲーションは互いに独立しているため、無効なデータデリゲーションと有効なロックデリゲーション、または有効なデータデリゲーションと無効なロックデリゲーションの両方を持つことができます。ONTAP は、データとまったく同じ方法と転送方法で、ロックデリゲーションをオリジンに要求します。

4.8.3 グローバルファイルロックが有効の場合の読み取りロック処理

グローバルファイルロックは ONTAP 9.10.1 で導入された機能です。グローバルファイルロックでは、ファイルの読み取り拒否およびファイルの特定バイト領域のロックが、すべてのキャッシュとオリジンに適用されます。ワークフローでファイルの読み取り拒否またはファイルの特定バイト領域のロックを必要とする場合は、グローバルファイルロック機能を有効にしてください。有効にする場合は、Advanced モードで以下のコマンドを実行します。

```
origin_cluster::*> flexcache origin config modify -origin-vserver <svm>
-origin-volume <origin_vol> -is-global-file-locking-enabled true
```

注意

グローバルファイルロックを有効にすると性能に影響が出ます。グローバルファイルロックを行うために、キャッシュでの読み取り時に毎回オリジンにアクセスすることで、RTT が増加します。

4.8.4 NLM 書き込みロック

前述のとおり、NLM ロックはオリジンで調整されます。そのため、すべてのロックを適切に保持し、すべてのロックが考慮されていることを確認するのは簡単です。クライアントが NLM を通じてファイルロックを要求すると、その要求はオリジンに転送され、オリジンで FlexCache 対応 NLM ロックが発生します。FlexCache クラスタには NLM ロックに関する情報がないため、これらのロックを確認するには、オリジンから確認する必要があります。オリジンでの FlexCache NLM ロックの例を次に示します。

```
origin_cluster::> vserver locks show
Notice: Using this command can impact system performance. It is recommended
that you specify both the vserver and the volume when issuing this command to
minimize the scope of the command's operation. To abort the command, press Ctrl-C.

Vserver: origin-svm
Volume Object Path          LIF    Protocol Lock Type Client
-------- ------------------------ ----------- --------- ----------- ----------
vol_fc_origin
     /vol_fc_origin/test     -     nlm    byte-range 198.51.100.193
        Bytelock Offset(Length): 0 (18446744073709551615)
        FlexCache Lock: true
```

クライアント 198.51.100.193 は、/vol_fc_origin/test のファイルに NLM バイト範囲ロックを所有します。
オリジン上でファイルを直接ロックするクライアントと FlexCache ボリュームからロックするクライアントの唯一の違いは、FlexCache Lock フラグがあることです。

4.8.5 NLM ロックリカバリ

NLM ロックが発行されると、クライアントにコールバックしてロックを解放するか、プロトコル標準を介してサーバーイベントの後にロックを再要求する方法があります。NFSv3 は、サイドバンドの Network Status Monitoring プロトコルによって実現可能です。ロック要求はオリジンに転送されるため、FlexCache ボリュームにロックは保持されません。これを確認するには、FlexCache ボリュームに表示されているロックを確認します。
ONTAP では、FlexCache ボリュームのロックは追跡されません。したがって、FlexCache のイベント (takeover、giveback、または LIF migrate など) にはロックの回復は必要ありません。

ロック回復が必要なのは、オリジンボリュームを所有するノードで takeover または giveback イベントが発生した場合だけです。オリジンで takeover または giveback イベントが発生すると、オリジンのクライアントによって生成されたロックのすべてのロック再要求が正常に処理されます。FlexCache ロックフラグが設定されているロックの場合、ロックの再要求は同じ方法で行われます。ただし、ロック再要求メッセージは、そのロックを発信した FlexCache ノードに転送されます。次に、FlexCache ノードが NSM メッセージをクライアントに送信します。

4.8.6 SMB 書き込みロック

クライアントが SMB を通じてキャッシュにあるファイルの書き込みロックを要求すると、そのロック要求は、すでに説明した他の読み取り / 書き込みデータコールと同じ方法および転送方法でオリジンに転送されます。次に、オリジンはそのファイルに他の排他的書き込みロックがないかどうかをチェックし、単一のクライアント書き込みロックを付与する代わりに、キャッシュへの書き込みデリゲーションを付与します。この書き込みデリゲーションは、キャッシュとオリジンの両方で RLEM ファイルを介して追跡されます。この書き込みロックが正常にデリゲートされると、そのキャッシュは、そのキャッシュを要求しているすべてのクライアントにバイト範囲の書き込みロックを付与できます。その書き込みデリゲーションの存続期間は、別のクライアントがオリジンまたは別のキャッシュで書き込みロックを要求するか、オリジンが別の理由でその書き込みロックデリゲーションを取り消すまでです。書き込みロックは実際にはキャッシュで許可および追跡されるため、`vserver locks show` コマンドを使用してキャッシュクラスタで表示できます (以下を参照)。

```
cache_cluster::> vserver locks show

Notice: Using this command can impact system performance. It is recommended
that you specify both the vserver and the volume when issuing this command to
minimize the scope of the command's operation. To abort the command, press Ctrl-C.

Vserver: cache_svm
Volume   Object Path               LIF         Protocol  Lock Type   Client
-------- ------------------------- ----------- --------- ----------- ----------
cache_of_vol_fc_origin/file.doc cache_svm_lif2 cifs share-level 198.51.100.194
                Sharelock Mode: all-deny_read_write
                                                                    op-lock 198.51.100.194
                Oplock Level: null
                                                                     byte-range
198.51.100.194
                Bytelock Offset(Length): 0 (2147483647)
```

クライアント 198.51.100.194 は、キャッシュで複数のロックを所有します。共有ロックは読み取りと書き込みを拒否します。さらに、オフセット 0 から始まるバイト範囲ロックがあります。書き込みロックデリゲーションは、RLEM メタファイルを介して ONTAP によって排他的に追跡され、オリジンまたはキャッシュで `vserver locks show` コマンドを使用しても表示できません。

■ 書き込みロック競合のチェックと調整

オリジンは書き込みロックの調整と管理を行っています。1 つのクライアントにつき 1 つの書き込みロックをキャッシュに与える代わりに、任意のクライアントに書き込みロックを許可するデリゲーションをそのキャッシュに与えて、効率を向上します。そのキャッシュがファイルに対して有効な書き込みロックデリゲーションを持っていれば、オリジンにアクセスすることなく、キャッシュのどのクライアントにも書き込みロックを付与できます。この書き込みロックデリゲーションを持つキャッシュは 1 つしかないため、オリジンもその他のキャッシュも書き込みロックを付与できません。

キャッシュに書き込みロックデリゲーションを付与されており、オリジンにファイルを書き込みたいクライアントがいる場合、オリジンはキャッシュにアクセスしてその書き込みデリゲーションを取り消します。このデリゲーション失効のセマンティクスは、クライアントが既にロックを持っているファイルに書き込みたい時と似ています。現在、キャッシュに未解決の書き込みロックがない場合、キャッシュはデリゲーションを取り消し、オリジンに応答します。オリジンは、新しいクライアントに書き込みロックを付与できます。未処理の書き込みロックがある場合、キャッシュはロックを保持しているクライアントにアクセスし、それらのロックの取り消しを要求します。これは、ロックを所有しているクライアントへのコールバックと同じです。このコールバックは、クライアントが書き込みを行おうとして、同じボリュームからアクセスした場合に発生します。キャッシュはクライアントからの応答を受け入れ、適切なアクションを実行します。現在の書き込みロックを所有するクライアントがダウングレードを許可する場合、キャッシュは書き込みロックを解除し、書き込みロックのデリゲーションをオリジンに応答します。オリジンは、新しいクライアントに書き込みロックを付与できます。クライアントが書き込みロックのダウングレードを拒否した場合、キャッシュはオリジンに拒否を返します。次に、オリジンは新しいクライアントからの書き込みロックの要求を拒否します。

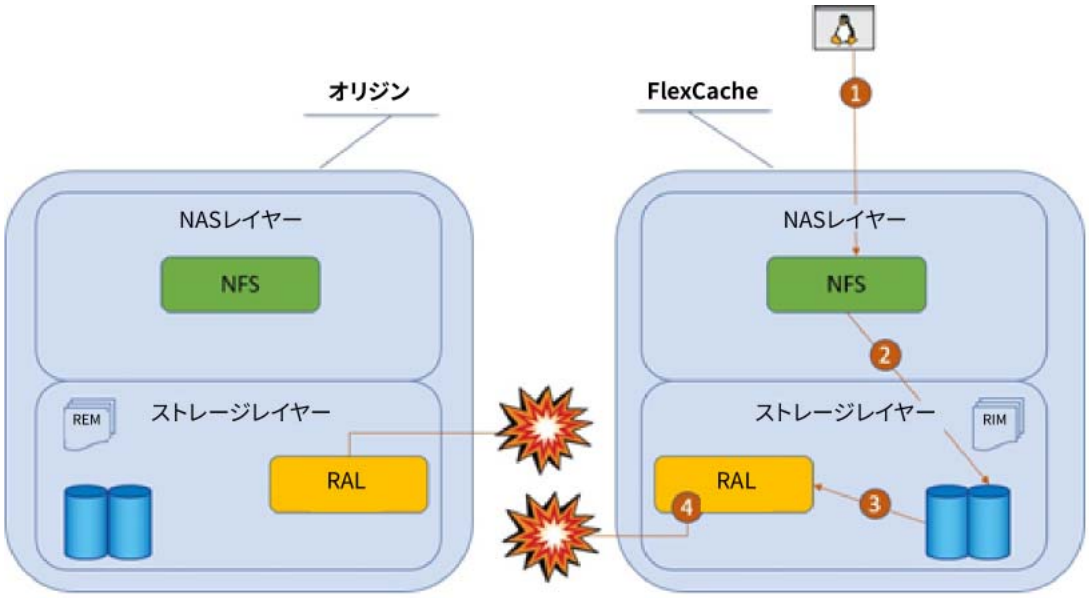
新しい書き込みロック要求が、書き込みロックデリゲーションを保持しているキャッシュとは異なるキャッシュで発生した場合、上記と同じプロセスが発生しますが、キャッシュはオリジンを介して通信し、相互には直接通信しません。書き込みロックデリゲーションの取り消しが成功した場合、ONTAP は、書き込みロックを要求する新しいキャッシュで書き込みロックデリゲーションを付与します。そのキャッシュが書き込みロックのデリゲーションを獲得すると、期待したとおりにクライアントに書き込みロックを付与できます。

4.9 切断モード

オリジンを含むクラスタと FlexCache ボリュームを含むクラスタが、WAN の問題またはその他の問題のためにクラスタ間 LIF を介して相互に通信できない場合があります。この状態を切断モードと呼びます。双方向トラフィックがあるために切断が発生した場合は、オリジンと FlexCache ボリュームの両方で検出できます。切断モードはどちらの方向にも使用できます。クラスタピア内の他のノードと通信できないノードがある場合は、切断モードになります。

図 4.9 に、FlexCache の切断モードを示します。

図 4.9 FlexCache の切断モード



切断モードになると、FlexCache ボリュームでイベント管理システム (EMS) メッセージが生成されます。これは FlexCache 固有の EMS メッセージです。クラスタと SVM ピアの関係に関する EMS メッセージもありますが、切断モードのメッセージはオリジンへの接続を試行している FlexCache ボリュームに固有のものです。以下にその EMS メッセージの例を示します。

```
10/7/2018 19:15:28 fc_cluster-01   EMERGENCY   Nblade.fcVolDisconnected: Attempt to access
FlexCache volume with MSID 2150829978 on Vserver ID 2 failed because FlexCache
origin volume with MSID 2163227163 is not reachable.
```

FlexCache が切断されると、オリジンは FlexCache に特定の EMS メッセージを送信しませんが、クラスタと SVM のピアリング関係の問題については通常の EMS メッセージを送信します。

4.9.1 切断モードでできること

切断モードでの操作時には、制限事項があります。このセクションでは、切断モードで実行できる操作と実行できない操作について説明します。

注意

次の情報は、このドキュメントで説明されているすべてのベストプラクティスが実行されていることを前提としています。

■ オリジン側

- すべての読み取りは通常どおりに行われます。
- 新しいファイルへの書き込みはすべて通常どおりに行われます。
- FlexCache でキャッシュされていない既存のファイルへの書き込みは、通常どおりに処理されます。
- ファイルに対してアクティブなデータデリゲーションがキャッシュに存在する場合、キャッシュが REM によって、切断された FlexCache まで追跡されていると、ファイルへの書き込みは許可されずにハングします。設定されたタイムアウト後にこの書き込みが許可されます。詳細については、「4.9.2 切断モード TTL と再同期」(P.29) を参照してください。

■ 切断された FlexCache ボリューム側

- すでにキャッシュされているデータの読み取りは、通常どおりに進行します。

  注意

  一部のアプリケーションは、ファイルを開こうとするときに書き込みロックやその他の排他ロックを要求します。読み取りロックデリゲーションがない場合、キャッシュは読み取りロックを付与できません。また、書き込みロックを付与することもできません。切断されたキャッシュでの読み取りは失敗する可能性があります。

- キャッシュされていないデータの読み取りはハングします。
- FlexCache への書き込みはハングします。
- `ls` コマンドまたは `dir` コマンドは、接続を切断する前に、そのディレクトリで ls コマンドまたは同等のコマンドが実行されていた場合にのみ機能します。

  注意

  ディレクトリは、もう 1 つの inode です。キャッシュされている場合は、サービスが提供されます。キャッシュされていない場合は提供されません。

■ 切断されていない FlexCache ボリューム側

切断された FlexCache ボリュームが 1 つありますが、その他の切断されていない FlexCache ボリュームは正常に動作します。ただし、「■ オリジン側」(P.28) の項で説明した制限と同じ制限があります。

READDIRPLUS、FlexGroup ボリューム、および切断モード

ONTAP FlexGroup への `ls` コマンドの処理速度を上げるために、ディレクトリエントリを一度に 1 つずつ読み込むのではなく、一括で読み込むように FlexGroup コードに変更が加えられました。これは `fast-readdir` と呼ばれます。FlexGroup ボリューム内のディレクトリー覧は高速化されていますが、FlexCache でのキャッシュは促進されません。ONTAP 9.8 P2 以前を使用していて、切断モードで `ls` またはその他 `dir-type` コマンド実行時にハングアップした場合は、必要に応じて `fast-readdir` を無効にしてください。

4.9.2 切断モード TTL と再同期

ONTAP の FlexCache には、オリジンノードとキャッシュノード間の接続が機能しているかどうかを確認するハートビート機構が備わっています。キャッシュが切断されると、オリジンまたはその他の切断されていないキャッシュからそのキャッシュに送信されたファイルへのすべての書き込みは、キャッシュがオリジンに再接続するまで停止されます。ハートビート機構は、接続の状態を判別し、オリジンによる取り消し時に詳細を提供します。接続状態は、以下の Advanced モードの CLI コマンドで利用できます。

```
cluster1::*> volume flexcache connection-status show

Node: cluster1-01

              Remote      Remote       Remote       Remote                Connection
+Vserver      Volume      Vserver      Volume       Cluster     Endpoint  Status
+------------ ----------  ----------   ----------   ----------  --------  -----------
cache-svm1    vol_cache1  origin-svm1  vol_origin1  cluster2    origin    connected

Node: cluster1-02

              Remote      Remote       Remote             Remote                Connection
+Vserver      Volume      Vserver      Volume             Cluster     Endpoint  Status
+------------ ----------  ----------   ----------         ----------  --------  -----------
origin-svm2   vol_origin2 cache-svm2   vol_cache2__0001   cluster2    cache     connected
origin-svm2   vol_origin2 cache-svm2   vol_cache2__0002   cluster2    cache     connected
origin-svm2   vol_origin2 cache-svm2   vol_cache2__0003   cluster2    cache     connected
origin-svm2   vol_origin2 cache-svm2   vol_cache2__0004   cluster2    cache     connected
5 entries were displayed.
```

上記の CLI 出力は、cluster 1 に 2 つの FlexCache 関係があることを示しています。

(1) 1 つ目はキャッシュで、SVM `cache-svm1` に cluster2 の SVM `origin-svm1` 内にあるオリジン `vol_origin1` とリンクした FlexCache ボリュームを持っています。

(2) もう 1 つはオリジンで、SVM `origin-svm2` のボリューム `vol_origin2` が cluster2 上の SVM `cache-svm2` にある FlexCache ボリューム `vol_cache2` にサービスを提供します。

クラスタがオリジンとして動作している場合、リストされている FlexCache ボリュームに複数のエントリーがあることがあります。この重複は、FlexCache ボリュームが複数のボリュームを持つ FlexGroup であるために発生します。FlexGroup ボリュームの詳細については、FTI (Fsas Technologies Inc.) のマニュアルサイトの「ETERNUS AX/HX Series FlexGroup ボリュームを使用した拡張性とパフォーマンスパワーガイド」を参照してください。

オリジンとキャッシュの間の接続が切断されたものとしてマークされると、切断されたキャッシュにキャッシュされたファイルへの変更は、有効期間 (TTL) に達した後に続行されます。TTL は約 120 秒です。`volume flexcache connection-status show` コマンドの出力に含まれるステータスも、connected から disconnected に変わります。

オリジンノードとキャッシュノードが通信を再確立すると、キャッシュは RIM ファイル内のエントリを持つすべてのファイルをソフト削除としてマークします。ソフト削除は、キャッシュの内容が無効とマークされており、そのキャッシュがサービスを提供しても安全であることがオリジンによって証明されるまでサービスが提供されないことを証明します。証明は、キャッシュされたコンテンツに関連付けられたメタデータを取得し、

それを比較してキャッシュが切断されている間の変更を判別することによって行われます。キャッシュ自体は再検証を開始しません。

注意

オリジンとキャッシュの間の接続リンクが切断されると、オリジンがキャッシュを切断済みとして宣言するのに約 2 分かかります。ただし、オリジンが切断されたことを示すキャッシュの記録には約 1 分かかります。オリジンの切断時間は、キャッシュが同じ結果に至らなかった場合の誤検出を除外するため、キャッシュの切断時間より長くなっています。

4.9.3 ファイルの細分性の再検証

ONTAP 9.8 以降は、切断イベント後にキャッシュファイルを再検証する、より効率的な方法を導入しています。この方法には、ボリューム内のオリジンで変更されたファイルを識別するオリジンからのデータのペイロードが含まれます。この拡張機能により、切断イベント後にソフト的に削除されたすべてのファイルについて、キャッシュがオリジンに問い合わせる必要がなくなりました。オリジンから送信されたペイロードを調べることができ、オリジンに何度も戻ることを回避できます。その結果、オリジンへのトラフィックが減り、切断イベント後のキャッシュでのファイルアクセス時間が短縮されます。

4.9.4 最終アクセス時刻

既定では、最終アクセス時刻 (または `atime`) は、読み取りが行われるたびに更新されるファイルプロパティです。本質的に、読み取りは書き込みとして機能します。FlexCache ボリュームの作成中は、FlexCache ボリュームで読み取りが実行されるたびに、オリジンへのこの値の過剰なライトバックが発生するのを防ぐために、`atime` トラッキングをオフにしています。オリジン側ではオフになっていないので、切断中は `atime` の影響でオリジンへのアクセスの一部が阻害されることがあります。以下は、オリジンボリュームで `atime` の更新が無効になるその他の理由です。

- 切断されている FlexCache ボリュームでキャッシュされたファイルは、オリジンで読み取れない可能性があります。
- 場合によっては、`ls` コマンドをオリジンで実行できないことがあります。

場合によっては、「● ベストプラクティス 2a ONTAP 9.10.1 以前の場合：オリジンの atime-updates を False に設定」で説明しているように切断モードがオリジンボリュームへのアクセスに影響します。

4.10 MetroCluster クラスタのサポート

ONTAP 9.7 以降、FlexCache オリジンボリュームとキャッシュボリュームを ONTAP MetroCluster システムでホストできるようになりました。これは、MetroCluster のミラーリングされたアグリゲートまたはミラー解除されたアグリゲートのいずれかに設定できます。

MetroCluster システムで FlexCache ボリュームがミラーリングされたアグリゲート上にある場合、FlexCache テクノロジーを使用する際には追加の考慮事項があります。FlexCache の操作に必要な REM と RIM のメタファイルは、SyncMirror を使用して一方から他方にミラーリングされます。このため、FlexCache の処理にさらに遅延が発生し、これらのファイルに書き込みを行って SyncMirror の処理が完了するまで待機することになります。これは、オリジンがミラーリングされたアグリゲート上にあり、ファイルが 1 つ以上のキャッシュにキャッシュされているファイルへの書き込み操作で最も一般的です。REM ファイルは変更されているため、書き込みが正常に終了するには、もう一方のプレックスにミラーリングする必要があります。これが影響を及ぼす可能性のある別の一般的な操作としては、ミラーリングされたアグリゲートにホストされたキャッシュにおける最初のファイル読み取りを含みます。

4.10.1 切替後または切替後に手動操作を要する場合

場合によっては、MetroCluster スイッチオーバーまたはスイッチバック後に FlexCache ボリュームが正常に機能するために、手動操作が必要になることがあります。これは、スイッチオーバー動作中に SVM リピアリングが発生しなかったり、失敗したりする場合があるためです。SnapMirror でも同様の制限が適用されます。2 つのシナリオを次に示します。

- オリジンと FlexCache ボリュームが同じ MetroCluster クラスタおよびサイト上の異なる SVM にある場合。
- オリジンまたは FlexCache ボリュームが別のクラスタ上にあり、クラスタピアトラフィックが利用できない間にスイッチオーバーまたはスイッチバックが発生した場合。

オリジン SVM とキャッシュ SVM が同じ MetroCluster サイトにある場合

スイッチオーバーの後もう一方のサイトの SVM をピアリングする必要があります。これは、すべてのスイッチオーバーの後に起こらなければなりません。プライマリボリュームとセカンダリボリュームが同じ MetroCluster サイト上の異なる SVM にある場合、同様の SnapMirror 関係にも同じ制限が適用されます。ピアが確立されるまで、FlexCache はスイッチオーバー中は切断モードで動作します。オリジンボリュームでも、切断されたキャッシュが検出されるため、影響がある可能性があります。

スイッチオーバーまたはスイッチバック中に他のクラスタ上のオリジンまたは FlexCache ボリュームが動作し、クラスタ間通信がない場合

スイッチオーバーまたはスイッチバック操作を実行している MetroCluster クラスタの外部のクラスタにクラスタおよび SVM ピアがある場合、ONTAP はイベントの完了後に自動的に SVM をリピアリングします。MetroCluster クラスタがリピアリングを試行しているピアクラスタと通信できない場合、リピアリングコマンドが正常に発行されるまで、操作は失敗し、FlexCache は切断モードで動作します。MetroCluster サイトスイッチのログには、イベント後に手動での中継が必要であることが明確に示されています。オリジンボリュームでも、切断されたキャッシュが検出されるため、影響がある可能性があります。

5. FlexCache によるマルチプロトコルグローバルファイルシステムのネームスペースの作成

FlexCache テクノロジーなら、単一の ONTAP クラスタに限定されることなく、世界中のデータに対応する真のマルチプロトコルグローバルネームスペースを作成できます。キャッシュの作成は、グローバルなネームスペースを作成するための最初のステップにすぎません。完全な自律型グローバルネームスペースを作成するためには、クライアントもキャッシュについて知っていて、キャッシュにデータをマウントまたはマッピングする必要があります。これは、手動で行うことも、いくつかの方法で自律的に行うこともできます。

5.1 ネームサービスの複製

グローバルネームスペースを作成するための最初の手順は、オリジンで設定されたネームサービス構成がキャッシュで供給されることを確認することです。アクセス許可と監査を正しく適用し、オリジンと同じキャッシュアクセスを保証するには、これらの設定を同じように構成する必要があります。

5.1.1 DNS サーバー

まず、キャッシュ SVM で DNS サーバーを構成して、オリジンで構成されているサーバーと類似した DNS サーバーを指すようにします。この手順の主な目的は、ONTAP がキャッシュの DNS サーバを介して名前解決を実行する必要がある場合に、クエリの結果がオリジンで実行された場合と同じになるようにすることです。このプロセスにより、ホスト名が検出され、ONTAP がホスト名をキャッシュの IP アドレスに変換する必要がある場合、クエリ結果はオリジンとまったく同じになります。

5.1.2 ユーザー名マッピング

次に同じ構成が必要なネームサービス項目は、ユーザー名マッピングです。これらは、オリジンで構成された UNIX から Windows または Windows から UNIX へのマッピング構成です。
`vserver name-mapping show` の出力が一致していることを確認すれば十分です。

```
Vserver: cache-svm1
Direction: win-unix
Position Hostname         IP Address/Mask
-------- ---------------- ----------------
1       -                -                   Pattern: (.+)\\testerdwu(.+)
                                          Replacement: win_ldap_user\2
Vserver:   cache-svm1
Direction: unix-win
Position Hostname         IP Address/Mask
-------- ---------------- ----------------
1       -                -                   Pattern: root
                                               Replacement: CACHE-SVM1\\Administrator
2       -                -                   Pattern: testerdlu0003
                                               Replacement: DOMAIN\\win_ldap_user0003
```

5.1.3 ns-switch

オリジンと一致する次のネームサービス設定項目は、ns-switch 設定です。ns-switch は、ホスト名、UNIX ユーザー、UNIX グループなどを照会するときに ONTAP が通信するネームサーバのタイプを構成します。このプロセスは、オリジン SVM のキャッシュでの正確な出力と一致させるだけです。

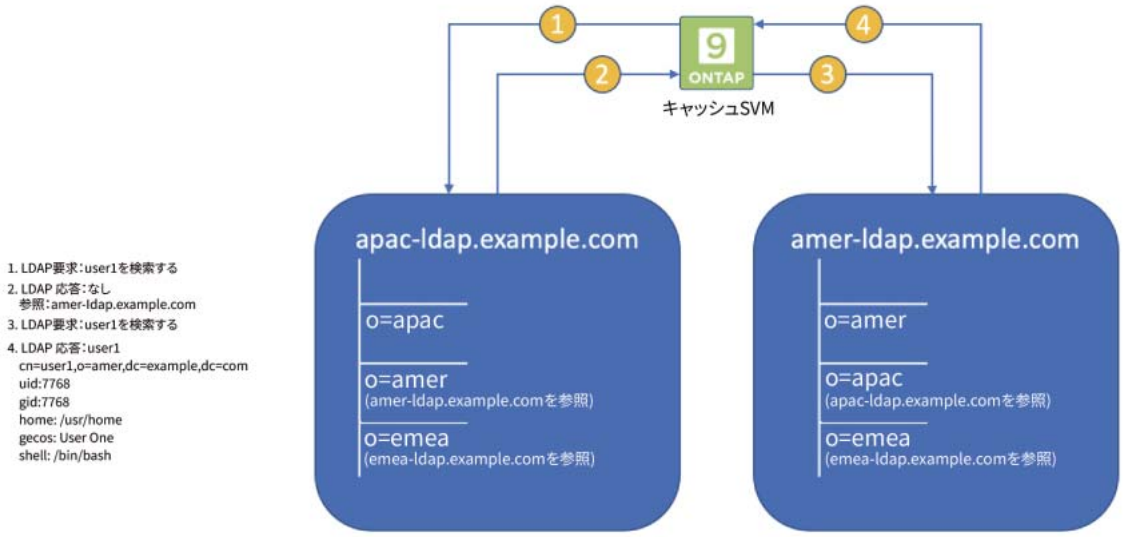
```
cluster1::> vserver services name-service ns-switch show
                                Source
Vserver         Database        Order
--------------- ------------    ---------
cache-svm1      hosts           files,
                                dns
cache-svm1      group           files,
                                ldap
cache-svm1      passwd          files,
                                ldap
cache-svm1      netgroup        files,
                                ldap
cache-svm1      namemap         files
```

5.1.4 Lightweight Directory Access Protocol

LDAP (Lightweight Directory Access Protocol) 構成は、キャッシュとオリジンで同様である必要がありますが、キャッシュ側での LDAP クエリの結果がオリジン側での LDAP クエリの出力と同じになればそれで構いません。広域分散型で複数レルムの LDAP 構成の場合、オリジンと同じ LDAP レルム内の LDAP サーバをキャッシュと同じ場所で使用できないことがあります。この問題は、LDAP 参照が機能していることを確認するか、キャッシュで構成される LDAP サーバーのオリジンで構成されたレルムをカバーするグローバルカタログが使用可能であることによって解決されます。参照またはグローバルカタログを設定するには、LDAP ベンダーのマニュアルを参照してください。単一のレルム構成では参照やグローバルカタログは必要ありません。キャッシュと同じ場所にある LDAP サーバーで十分です。

図 5.1 は、LDAP 検索の実行に参照が必要な一般的な広域 LDAP 構成を示しています。レプリカ、グローバルカタログ、およびその他の LDAP 構成を作成すると、WAN リンクにまたがる場合の LDAP 検索のパフォーマンスが向上します。複数レルム構成で LDAP 検索のパフォーマンスを向上させる方法については、LDAP ベンダーにお問い合わせください。

図 5.1 広域 LDAP 要求



必要な LDAP 設定のタイプを決定したら、ONTAP LDAP クライアントを設定する必要があります。LDAP クライアントの設定は、上記の実装によって異なります。LDAP サーバーとすべての DN 設定が実装に適しており、LDAP クエリの結果がオリジンとキャッシュで同じであることを確認してください。

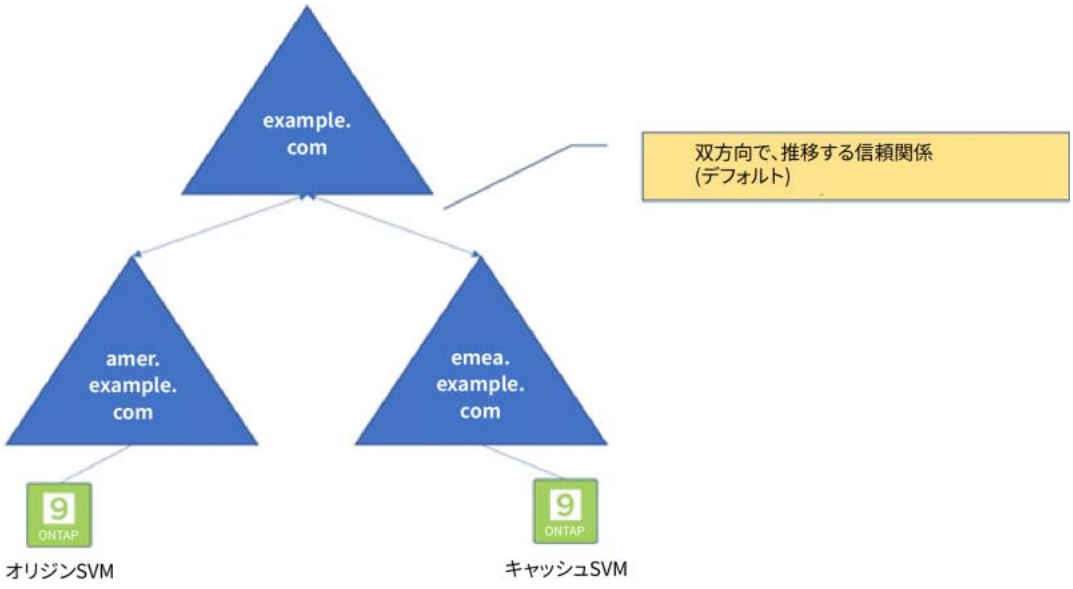
LDAP スキーマ

キャッシュで考慮すべきもう 1 つの項目は、使用されている LDAP スキーマです。オリジンでカスタムスキーマが使用されている場合は、おそらくキャッシュでカスタムスキーマも構成する必要があります。FlexCache SVM の LDAP 設定で設定されている LDAP スキーマが適切であることを確認します。

5.2 Active Directory

NTFS セキュリティスタイルを使用する場合、または SMB 共有を作成する場合は、アクセス許可を正しく適用するように Active Directory を構成する必要があります。これをオリジンのようにキャッシュで行うには、キャッシュ SMB サーバーをオリジンと同じドメイン、またはオリジンが設定されているドメインと双方向の信頼を持つドメインのいずれかに作成する必要があります。これが必要な理由は 2 つあります。まず、NTFS ACL は、実際にはユーザー名形式ではなく SID 形式で保存されます。これは、ONTAP がアクセス許可を正しく適用するには SID を検索する必要があるということです。次に、ONTAP はクライアントを正しく認証する必要もあります。SMB クライアントとサーバーが同じドメイン内にない場合、または SMB クライアントとサーバーの間に双方向の信頼関係があるドメイン内にない場合、認証は課題となり、データにアクセスできなくなる可能性があります。クライアントがグローバルに分散している場合は、複数のドメインを作成してクライアントに対応できます。この構成の最も一般的な種類は、単一フォレスト複数ドメイン構成です。図 5.2 は、複数のドメインを持つ Active Directory フォレストの例を示しています。この例では、オリジンとキャッシュは別々のドメインにありますが、同じフォレスト内に推移する信頼関係があります。

図 5.2 信頼されている別のドメイン内のオリジンとキャッシュ



5.2.1 ワークグループモード

ONTAP では、SMB サーバを Active Directory ドメインではなくワークグループに設定して、クライアントに共有を提供できます。ビジネス上の理由からワークグループが好まれる特定の使用例があります。FlexCache ボリュームを共有していて、オリジンボリュームのセキュリティスタイルが unix の場合に限り、FlexCache テクノロジーはキャッシュのワークグループモードで動作します。前述のように、ACL は SID 形式で保存されます。ワークグループモードでは、SID は SMB サーバーに固有で、キャッシュ SVM はオリジン SVM と同じ SMB サーバーではないため、保存されている NTFS ACL をユーザー名に変換できません。さらに、ワークグループモードでは、SMB サーバーを接続して検索する方法はありません。

- **ベストプラクティス 4: Unix セキュリティスタイルのワークグループモード**
  オリジン SVM がワークグループモードを使用している場合、FlexCache ボリュームを作成できるボリュームは、セキュリティスタイルが UNIX に設定されているボリュームだけです。

5.3 SMB クライアント用のグローバルネームスペースの作成

前述のセクションで説明した必要な構成を作成した後、クライアントがオリジンであるかキャッシュであるかにかかわらず、クライアントの場所と最も近いターゲットに従ってドライブをマップする手段をクライアントに設定できます。SMB クライアントと NFS クライアントは動作が異なるため、それぞれ独自の方法でターゲットを構成する必要があります。この項では、FlexCache テクノロジーを使用する際に、最も近い共有ターゲットをクライアントにマッピングするために SMB クライアント情報を提供する方法について説明します。

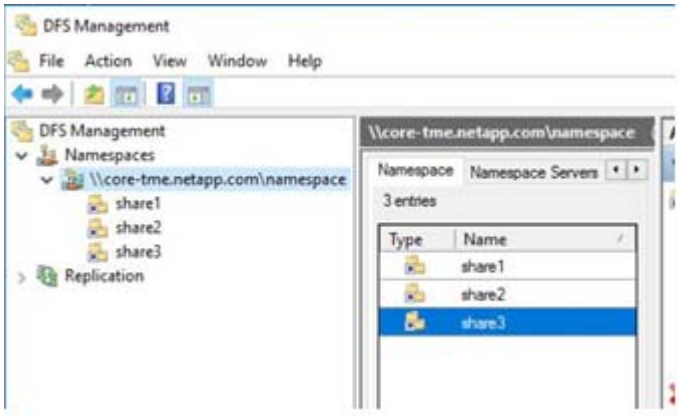
5.3.1 Windows DFS ネームスペース

論理的にクライアントに最も近い共有ターゲットを Windows クライアントにマッピングさせるには、Windows DFS ネームスペースサーバーを使用する必要があります。Windows DFS を使用すると、複数のサーバ上の共有を論理的にグループ化し、共有を 1 つの階層ネームスペースに透過的にリンクできます。DFS は、ネットワーク上の共有リソースをツリー構造で編成します。Windows DFS の機能には DFS ネームスペースと DFS レプリケーションの 2 つのコンポーネントがあります。
ONTAP が必要とするのは、FlexCache テクノロジーでグローバルネームスペースを作成するための DFS ネームスペース機能だけです。DFS レプリケーションサービスは、Windows のみの DFS 構造にのみ必要です。DFS ネームスペースサーバーのベストプラクティス、または Windows DFS ネームスペースとレプリケーションの詳細情報が必要な場合は、Microsoft のドキュメントを参照してください。

ネームスペースの作成

Windows DFS ネームスペースサーバーをインストールしたら、ネームスペースを構成する必要があります。これは、すべてのクライアントがマップされたドライブを開始するために必要な初期パスです。パスの最初の部分は、主に DFS ネームスペースサーバーを解決するためのものです。ネームスペースサーバーは、UNC パスの最初の部分としてドメイン名を使用してドメインベースのネームスペースをホストすることも、別のホスト名を使用してネームスペースサーバーを開始することもできます。ネームスペースがドメインベースのネームスペースであることは非常に一般的です。このため、UNC パスの最初の部分は\\ domain.local となります。SVM の SMB サーバー名をネームスペースのルートとして使用することはできません。図 5.3 に、ドメインベースのネームスペースを示します。パスの 2 番目の部分は、ネームスペースのルートです。これは、特定の DFS ネームスペースツリーを入力するための任意の名前です。共有名やその他の SMB サーバー設定と一致している必要はありません。

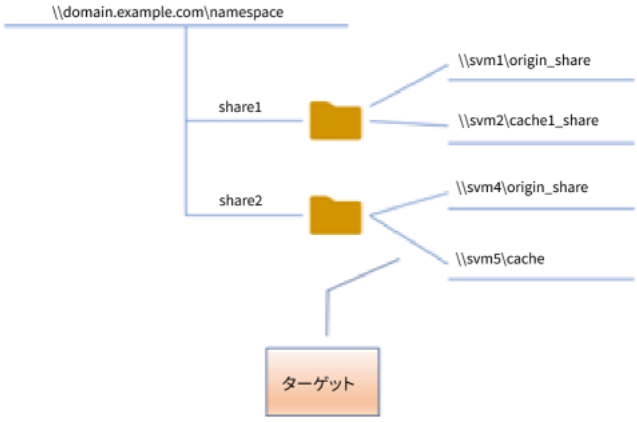
図 5.3 ドメインベースの DFS ネームスペース



5.3.2 ターゲットの構成

ネームスペースを作成したら、グローバルに使用可能な共有の作成を開始します。リストに追加するターゲットの数とターゲット、オリジンおよびキャッシュを決定します。ターゲットのリストは、クライアントがドライブをマップしようとしたときにアクセスする実際の共有です。図 5.4 は、ネームスペースとフォルダツリーおよびターゲットとの関係を示しています。すべてのキャッシュまたはオリジンをターゲットのリストに含める必要はありません。また、DFS ネームスペースサーバーのターゲットを、クライアントに影響を与えることなく自由に無効にできます。SMB サーバーとのセッションが終了するまで、最初に接続したターゲットに接続されたままになります。DFS 検索キャッシュの有効期限が切れても、現在のセッションが終了し、クライアントが共有への再接続を試みるまで、ターゲットは再評価されません。

図 5.4 ネームスペースツリーの関係



ネームスペースと共有へのアクセス

クライアントが DFS ネームスペース内の共有にアクセスすると、いくつかのことが発生します。まず、パスの最初の部分の DNS エントリを検索します。接続する IP アドレスを取得すると、クライアントはサーバーに接続します。ドメインベースのネームスペースの場合、DC にアクセスします。次に、DFS 照会を通じて、ドメインベースのネームスペースのネームスペースサーバにクライアントを照会します。ネームスペースサーバにアクセスした後、サーバはターゲットをチェックし、構成されたアルゴリズムを使用して、クライアントを最も近いターゲットに参照します。使用可能なアルゴリズムは、「ランダム順序」、「最低コスト」、および「クライアントサイト外のターゲットを除外」です。

Active Directory サイトとサービス

Active Directory サイトとサービスは、共有にアクセスしようとしているクライアントに提示されるターゲットを決定する上で重要な役割を果たします。「最低コスト」または「クライアントサイト外のターゲットを除外」のいずれかを使用するには、ターゲット SMB サーバーに関連するクライアントの場所を決定するために、Active Directory サイトとサービスをある程度の粒度で設定する必要があります。これらの関係は、Active Directory サイトとサービス MMC スナップインに格納されている IP アドレスの範囲によって決まります。インフラストラクチャに最適な Active Directory サイトおよびサービスの構成方法の詳細については、Microsoft のドキュメントを参照してください。

5.4 NFS クライアント用のグローバルネームスペースの作成

前述のように、NFS クライアントでは、ターゲットを構成するために別の方法が必要です。必要なネームサーバー構成が作成されると、NFS クライアントは Linux/UNIX の自動マウントサービスを使用してグローバルネームスペースを提供できます。自動マウントサービスは、マウントポイントのターゲットのマップを NFS クライアントに提供します。オートマウンタを使用すると、クライアントは MOUNT 呼び出しを調整し、マップにリストされているどの NFS サーバーがクライアントに最も近いかを判断します。次の節では、オートマウンタの使用方法について詳しく説明します。

5.4.1 autofs のインストール

ほとんどの標準リポジトリには、`autofs` がパッケージとして用意されています。これを行うには、次のいずれかのコマンドを使用します (Linux ディストリビューションに依存)。

```
yum install autofs
apt-get install autofs
dnf install autofs
```

5.4.2 マップファイルを作成

NFS の場合、1 回のエクスポートで複数のターゲットが存在することをクライアントに通知する唯一の方法は、マップファイルを使用することです。マップファイルは、マウントされる任意のパスに対してどのターゲットが有効かを示します。つまり、ローカルパス `/mnt/vol1` をマウントする場合、ターゲットは `svm1:/vol1`、`cache-svm1:/cache_of_vol1`、および `cache-svm2:/cache_of_vol1` にあります。

オートマウンタマップファイルを使用するための高度な方法は数多くあり、詳細を知りたい場合に役立つオンラインリソースも多数あります。この節では、1 つのマウントに対する複数の NFS サーバーエクスポートターゲットにクライアントを誘導するマップファイルの簡単な例を示します。O'Reilly 社が発行する『NFS & NIS』は優れたリソースです。O'Reilly 社の他の本でも、自動マウントマップについて説明しており、一元管理と分配のためにマップを LDAP に追加する方法についても説明しています。

次のマップファイルの例では、マウントパスは `/mnt/data` です。クライアントの場所に応じて、マウントするターゲットとなる NFS サーバーエクスポートには、次の 4 種類があります。

```
svm1:/data, cache-svm1:/cache_of_data, cache-svm2:/data, svm3:/cache_of_vol1.
/mnt/data -rw,hard  svm1:/data \
            cache-svm1:/cache_of_data \
            cache_svm2:/data \
            svm3:/cache_of_vol1
```

上記の例からわかるように、SVM(NFS サーバー) 名またはボリュームジャンクションパスの名称は重要ではなく、オリジンとキャッシュの間の一貫性は必要ありません。唯一の要件は、すべてのエントリーがオリジンであるか、そのオリジンのキャッシュであることです。

5.4.3 NFS クライアントが最も近いマウントを決定する方法

これで autofs パッケージがインストールされ、automount マップが作成され、autofs サービスが開始されました。次のステップは、データに実際にアクセスすることだけです。マウントディレクトリはマップファイルに定義されているため、まだ存在していない時は UNIX ファイルシステムに自動的に作成されます。さらに、autofs は必要に応じてマウントを行います。NFS エクスポートは、マウントパスディレクトリに移動するか、そのパス内のファイルにアクセスするまで、実際にはマウントされません。

クライアントがマウントパスにアクセスすると、いくつかの処理が行われます。最初に、そのマウントポイントのマップファイル内のホスト名が解決されます。次に、クライアントは、ターゲットのいずれかがクライアントと同じサブネットにあるかどうかを確認します。これは、クライアントがマルチホーム化されている場合、クライアントで設定されているすべての IP アドレスに対して実行されます。クライアントのサブネットにターゲットが 1 つある場合は、そのターゲットが排他的に使用され、MOUNT コマンドがその NFS サーバーに送信されます。クライアントと同じサブネット内に複数のターゲットがある場合は、サブネット内のすべてのターゲットに MOUNT コマンドが送信され、最初に応答するのが使用される NFS サーバーです。クライアントのローカルサブネットにターゲットが存在しない場合、またはどのターゲットも応答しない場合は、MOUNT コマンドがリスト内の残りのターゲットに設定されます。クライアントが受信した MOUNT 呼び出しに最初に応答した NFS サーバが使用されます。

クライアントは MOUNT コマンドへの最初の応答を NFS サーバーとして使用するため、このサーバーがこのマウントパスで最も高速で最適であると想定されます。これは、MOUNT コマンドの実行時のネットワークパフォーマンスとスループットに大きく依存します。

注意

NFSv3 のファイルハンドルは、オリジンとキャッシュの間で一致しません。つまり、オリジンからキャッシュへ、またはあるキャッシュから別のキャッシュへマウントポイントを移動する場合は、「古いファイルハンドル」エラーを回避するために再マウントが必要です。

6. カウンタ、統計

FlexCache の運用効率とパフォーマンスの分析に役立つさまざまなカウンタと統計情報があります。これらのカウンタは以下の判断の助けになります。

- オリジンに対していくつの FlexCache ボリューム操作が予定されているか。
- オリジンから情報を取得するのにかかる時間はどのくらいか。
- デリゲーションの無効化に何時間必要か。
- 通信に障害があるかどうか。

カタログのエントリは以下のようになります。

```
Object: waflremote
  Counter              Description
  -------------------------- ------------------------------------------------
  cancel_flexcache_ops Number of times flexcache remote ops were
                canceled due to ARL or HA
  cancel_flexcache_ops_aborted
                Number of times cancelling of flexcache ops
                during HA or ARL was aborted because run time
                was over limit
  fc_buff_hole_size_hist Buffer size histogram for retreived hole
                buffers for flexcache
  fc_buff_raw_size_hist Buffer size histogram for retreived raw
                buffers for flexcache
  fc_buff_zero_size_hist Buffer size histogram for retreived zero
                buffers for flexcache
  fc_bulk_attr_latency Bulk Attribute Latency
  fc_bulk_attr_ops Number of times bulk_attr has been performed
                for flexcache
  fc_bulk_attribute_hist Latency histogram for bulk attribute for
                flexcache
  fc_evict_local_hist Latency histogram for evicts started by cache
                to itself for flexcache
  fc_evict_local_latency Evict local latency, when a cache starts an
                evict message to itself
  fc_evict_local_ops Number of times evict has been sent localy
                (from cache to iteself) for flexcache
  fc_evict_remote_hist Latency histogram for evict origin-cache for
                flexcache
  fc_evict_remote_latency Evict remote latency, when an origin sends an
                evict message to a cache
  fc_evict_remote_ops Number of times evict has been sent from
                origin to cache for flexcache
  fc_retrieve_hist Latency histogram for remote retrieve for
                flexcache
  fc_retrieve_latency Remote Retrieve Latency
  fc_retrieve_message_hist Latency histogram for retrieve at origin for
                flexcache
  fc_retrieve_message_latency retrieve message latency at origin for
                flexcache
  fc_retrieve_message_ops Number of times retrieve messages that has
                been performed at the origin of a flexcache
  fc_retrieve_ops Number of times remote_retrieve has been
                performed for flexcache
```

```
  fc_store_message_hist Latency histogram for store_message for
                flexcache
  fc_store_message_latency store message latency Latency
  fc_store_message_ops Number of times store message has been
                performed for flexcache
  retrieve_flexcache_full Number of inbound retrieve messages that
                started revoke because flexcache cache list
                was full
Object: spinhi
  Counter Description
  -------------------------- ----------------------------------------------
  spinhi_flexcache_forward_base
                Array of select FlexCache spinhi
                op-forwarding file operations count for
                latency calculation
  spinhi_flexcache_forward_csm_errors
                Array of select FlexCache spinhi
                op-forwarding file operations csm error count
  spinhi_flexcache_forward_fileops
                Array of select FlexCache spinhi
                op-forwarding file operations count
  spinhi_flexcache_forward_latency
                Array of latencies of select FlexCache spinhi
                op-forwarding file operations
  spinhi_flexcache_forward_latency_histogram
                Histogram of FlexCache spinhi op-forwarding
                latency
  spinhi_flexcache_forward_max_time
                Array of select FlexCache spinhi
                op-forwardring file operations representing
                maximum time taken in receiving response from
                the origin
  spinhi_flexcache_forward_sent
                Array of select FlexCache spinhi
                op-forwarding file operations that are
                forwaded
  spinhi_flexcache_forward_total_errors
                Array of select FlexCache spinhi
                op-forwarding file operations error count
  spinhi_flexcache_forward_write_size_histogram
                Histogram of FlexCache spinhi op-forwarding
                write size
```

6.1 FlexCache ミス

FlexCache ミスは workload_volume object と read_io_type カウンタ内で検出されます。このカウンタは、FlexCache ボリュームにキャッシュされなかったファイルが要求された回数を示します。次の例を参照してください。

```
fc_cluster::> statistics show -sample-id workload_cache -counter read_io_type

Object: workload_volume
Instance: vol_fc_cache1-wid48920
Start-time: 10/2/2018 00:29:02
End-time: 10/2/2018 00:38:31
Elapsed-time: 569s
Scope: fc_cluster
Number of Constituents: 2 (complete_aggregation)
  Counter                         Value
  ------------------------------- --------------------------------
  read_io_type                          -
                cache                  17
                pmem                   0
              ext_cache                   0
                disk                   0
              bamboo_ssd                  0
               hya_hdd                  0
              hya_cache                   0
            hya_non_cache                    0
                cloud                  0
               fc_miss                  82
              cloud_s2c                   0

Object: workload_volume
Instance: vol_fc_cache1__0001-wid45107
Start-time: 10/2/2018 00:29:02
End-time: 10/2/2018 00:38:31
Elapsed-time: 569s
Scope: fc_cluster
Number of Constituents: 2 (complete_aggregation)
  Counter                         Value
  ------------------------------- --------------------------------
  read_io_type                          -
                cache                  0
                pmem                   0
              ext_cache                   0
                disk                   0
              bamboo_ssd                  0
                 hya_hdd                  0
                hya_cache                   0
              hya_non_cache                    0
                  cloud                  0
                fc_miss                  100
              cloud_s2c                   0
```

6.2 ボリュームワークロード統計での FlexCache の遅延

FlexCache の遅延に関する統計情報は、ボリュームワークロードの詳細でも確認できます。
例：

```
fc_cluster::> statistics show -sample-id workload_detail -instance *FLEXCACHE*

Object: workload_detail_volume
Instance: vol_fc_cache1-wid48920.DELAY_CENTER_FLEXCACHE_RAL
Start-time: 10/2/2018 00:29:22
End-time: 10/2/2018 00:42:43
Elapsed-time: 801s
Scope: fc_cluster-01

  Counter                           Value
  -------------------------------- --------------------------------
  in_latency_path                         1
  process_namer                          -
  resource_name             DELAY_CENTER_FLEXCACHE_RAL
  wait_time                        4786us

Object: workload_detail_volume
Instance: vol_fc_cache1-wid48920.DELAY_CENTER_FLEXCACHE_RAL
Start-time: 10/2/2018 00:29:22
End-time: 10/2/2018 00:42:43
Elapsed-time: 801s
Scope: fc_cluster-02

  Counter                            Value
  -------------------------------- --------------------------------
  in_latency_path                          1
  process_name                            -
  resource_name              DELAY_CENTER_FLEXCACHE_RAL

Object: workload_detail_volume
Instance: vol_fc_cache1-wid48920.DELAY_CENTER_FLEXCACHE_SPINHI
Start-time: 10/2/2018 00:29:22
End-time: 10/2/2018 00:42:43
Elapsed-time: 801s
Scope: fc_cluster-01

  Counter                           Value
  -------------------------------- --------------------------------
  in_latency_path                         1
  process_name                           -
  resource_name               DELAY_CENTER_FLEXCACHE_SPINHI
  wait_time                         13138us
```

FlexCache の I/O オペレーションごとの平均レイテンシの集約ビューもあります。このビューは、次の Quality of Service (QoS) 統計にあります。

```
fc_cluster::*> qos statistics workload latency show -iterations 100
Workload     ID  Latency  Network  Cluster   Data     Disk   QoS    VRAM  Cloud  FlexCache
SM
Sync
-------------- ------ ---------- ---------- ---------- ---------- ---------- ---------- ---
------ ---------- ---------- ----------
-total-        -     0ms    0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms
-total-        -  543.00us  228.00us    0ms  315.00us   0ms    0ms    0ms    0ms    0ms    0ms
User-Default     2  543.00us  228.00us   0ms   315.00us 0ms    0ms    0ms    0ms    0ms    0ms
-total-        -    0ms     0ms     0ms     0ms     0ms        0ms     0ms     0ms     0ms     0ms
-total-        - 389.00us   213.00us    0ms  176.00us        0ms    0ms    0ms    0ms    0ms
0ms
User-Default    2  389.00us  213.00us     0ms  176.00us      0ms    0ms    0ms    0ms    0ms
0ms
-total-        - 394.00us   211.00us    0ms  183.00us     0ms    0ms    0ms    0ms    0ms
0ms
User-Default    2  394.00us  211.00us   0ms  183.00us       0ms    0ms    0ms    0ms    0ms
0ms
-total-        - 1160.00us  236.00us   0ms 711.00us       0ms    0ms    0ms    0ms   213.00us
0ms
User-Default    2 1160.00us  236.00us    0ms  711.00us     0ms    0ms    0ms    0ms
213.00us
0ms
-total-        -     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms
-total-        -     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms     0ms
-total-        -   9.02ms   630.00us  234.00us   4.35ms   0ms    0ms  30.00us   0ms    3.77ms
0ms
User-Default     2   9.02ms  630.00us   234.00us   4.35ms   0ms   0ms   30.00us   0ms
3.77ms
0ms
-total-        -   5.09ms   318.00us   488.00us    0ms   0ms   0ms   0ms   0ms   4.29ms   0ms
User-Default     2   5.09ms   318.00us   488.00us  0ms   0ms   0ms   0ms   0ms   4.29ms
0ms
```

7. パフォーマンス

FlexCache ボリュームで、すでにキャッシュされたファイルを処理する際のパフォーマンスについては、FTI のマニュアルサイトの「ETERNUS AX/HX series ONTAP FlexGroup ボリュームベストプラクティス / インプリメンテーションガイド」を参照してください。
FlexCache ボリュームは FlexGroup ボリュームであるため、FlexCache ボリュームですでにキャッシュされているファイルを処理する方法と、FlexCache ボリューム以外の FlexGroup ボリュームからファイルを処理する方法には、ほとんど違いがありません。FlexCache ボリュームのパフォーマンス情報は、FlexGroup ボリュームの情報と同じです。

FlexCache ボリュームが潜在的な WAN を介して接続されている場合、ファイルの初回読み取り時のパフォーマンスの影響を受けやすくなります。この感度は、前述の統計で確認できます。

```
statistics show -object waflremote -counter fc_retrieve_hist
```

このカウンタは、FlexCache ボリュームがオリジンからの読み取りで遭遇した余分な待機時間のヒストグラムを示します。遅延はいくつかの方法で減らすことができます。

まず、FlexCache クラスタとオリジンクラスタ間の実際のネットワーク遅延を短縮できます。WAN オプティマイザが役立つ場合もありますが、WAN オプティマイザはパブリックプロトコルではないため、トラフィックを高速化できない場合があり、結果はさまざまです。
次に、データを先読み検索します。ユースケースに応じて、データを先読みする方法はいくつかあります。

次のセクションでは、いくつかの可能性について説明します。

キャッシュのプリウォーム

ONTAP 9.8 は、データを使用してキャッシュを事前にウォームアップするオンボックス方式を導入しています。コマンドは `flexcache prepopulate start` です。このコマンドは、ボリューム、qtree、ディレクトリ、または単一のファイルレベルで事前設定ができるように、パス変数を取ります。これは、このコマンドでキャッシュされたファイルが削除されないこと (ピン留め) を保証するものではありません。事前設定されたファイルは、標準の削除規則に基づいて削除の対象となります。

事前準備はクライアント側でも実行できます。Linux/Unix クライアントには、特定のディレクトリにあるすべてのファイルをプリロードする bash コマンドがあります。このコマンドは、<dir> にドット (.) を使ってカレントディレクトリから実行したり、ディレクトリを指定して実行したりできます。コマンドは次のとおりです。

```
[linux-host ~]# find <dir> -type f -print -exec sh -c "cat {} > /dev/null" \;
```

通常、このコマンドはディレクトリ一覧もウォームアップします。ウォームアップされない場合は、<dir> を上記と同じ情報に置き換えてから `ls -R <dir>` を実行してください。

Windows クライアントでは、コマンドを使用して、特定のディレクトリ内のすべてのファイルを事前にウォームアップさせることもできます。このコマンドは、カレントディレクトリから <dir> にドット (.) を使ってカレントディレクトリから実行したり、ディレクトリを指定して実行したりできます。キャッシュをウォームアップさせる Windows コマンドは次のとおりです。

```
C:\>for /f %i in (‘dir /b /s <dir>’) do @echo %i >NUL
```

通常、Linux の場合と同様に、このコマンドはディレクトリ一覧もウォームアップします。ウォームアップされない場合は、<dir> を上記と同じ情報に置き換えてから `dir/s <dir>` を実行してください。

8. ベストプラクティス集

このセクションでは、FlexCache ボリュームに関連するベストプラクティスについて説明します。

8.1 FlexGroup 構成要素

FlexCache ボリュームは FlexGroup ボリュームです。つまり、FlexCache ボリュームの総容量は、より小さな構成ボリュームで構成されています。FlexCache には、FlexGroup を作成するための 2 つのオプションがあります。

- 構成要素のアグリゲートをリストアップします (`-aggr-list`)
- FlexGroup による自動選択を有効にします (`-auto-provision-as`)

構成ボリュームを作成するためのアグリゲートを指定するには、`-aggr-list` オプションを使用することを推奨します。`-aggr-list-multiplier` オプションは、`-aggr-list` オプションでリストアップされたアグリゲートごとに使用されている構成ボリュームの数を決定します。構成要素の総数に関する推奨事項は、FlexCache ボリュームのサイズに比例します。

- FlexCache ボリュームが 100GB 未満 = 1 メンバーボリューム
- FlexCache ボリュームが 100GB ～ 1TB = 2 メンバーボリューム
- FlexCache ボリュームが 1TB ～ 10TB = 4 メンバーボリューム
- FlexCache ボリュームが 10TB ～ 20TB = 8 メンバーボリューム
- FlexCache ボリュームが 20TB 以上 = デフォルトのメンバーボリューム数 (`-auto-provision-as flexgroup` を使用)

● **ベストプラクティス 5: FlexCache ボリュームにはサイズに基づいた構成要素番号が必要**

`-aggr-list` オプションだけを指定して FlexCache を作成し、指定した数の構成要素を作成します。

```
fc_cluster::> flexcache create -vserver fc-svm1 -volume vol_fc_cache -origin-vserver
origin- svm -origin-volume vol_fc_origin -aggr-list aggr1_node1 -size 5TB -aggr-list-
multiplier 4 - junction_path /vol_fc_cache1
```

8.2 読み取りと書き込み

大まかな目安として、FlexCache は少なくとも 80% の読み取りと 20% の書き込みがキャッシュで行われます。この比率は、FlexCache のライトアラウンド特性のために機能します。書き込み操作をオリジンに転送すると、書き込みに遅延ペナルティが発生します。FlexCache では書き込みの割合を高くすることができますが、ONTAP の FlexCache で書き込みを処理する方法としては最適ではありません。

8.3 アクセス制御リスト、パーミッション、および適用

FlexCache は、FlexCache ボリュームで NTFS と UNIX の両方のセキュリティスタイルをサポートします。qtree サポートでは、親ボリュームとは異なる qtree セキュリティスタイルもサポートされています。つまり、FlexCache はオリジンボリュームで設定されたアクセス権を適用します。ただし、権限を適用するには、FlexCache を所有する SVM にオリジン上と同じディレクトリ設定を適用する必要があります。マルチプロトコルアクセスは、現在、オリジンとキャッシュの両方に分散されています。つまり、オリジンでは SMB アクセスしかなかったが、FlexCache で NFSv3 アクセスが発生している場合、そのボリュームはグローバルなマルチプロトコルになります。マルチプロトコル構成は、オリジンボリュームと FlexCache ボリュームの両方に適用する必要があります。

8.3.1 オリジンでの NTFS スタイルボリューム

オリジンが NTFS セキュリティスタイルの場合、FlexCache は NTFS セキュリティスタイルにもなります。FlexCache SVM には、CIFS サーバが構成されている必要があります。CIFS サーバは、同じドメインまたは信頼される側のドメイン内に存在する必要があります。NTFS アクセス許可は、ユーザー名やグループ名ではなく SID とともに保存されるため、FlexCache を提供する SVM は SID を検索できる必要があります。SVM が同じドメイン、フォレスト、または双方向の信頼される側のドメインにある場合は、権限を適切に適用できます。CIFS サーバが設定されていない場合、または CIFS サーバが所属するドメインがオリジンで構成されたドメインを信頼していない場合、権限は意図しない結果となる可能性があります。

8.3.2 オリジンでの UNIX スタイルのボリューム

オリジンが UNIX セキュリティスタイルのボリュームである場合、いくつかのオプションがあります。オリジンの SVM に適用される Lightweight Directory Access Protocol(LDAP サーバー) クライアントがない場合は、FlexCache に設定を作成する必要はありません。NFSv3 ユーザー ID (UID) とグループ ID (GID) は、LDAP で情報を検索する必要なく、ボリューム内のファイルとフォルダに対して UNIX モードビットを完全に使用できます。

LDAP クライアントが設定済みでオリジンの SVM に適用されている場合、ベストプラクティスは FlexCache に同じ LDAP クライアント設定を設定することです。UNIX スタイルのボリュームでは、2 つの LDAP ドメインで UID と GID が競合する可能性があるため、同じ LDAP ドメインが必要です。LDAP 参照は正しい情報を見つけることができますが、オリジンと同じ LDAP ドメインに直接接続するようにキャッシュの ONTAP LDAP クライアントを構成するのではなく、参照に依存して同じ情報を提供するように設定すると、UID および GID の競合のリスクが高くなります。ローカル UNIX ユーザーおよびグループについても同様です。オリジンの SVM でアクセスするように設定されたユーザーまたはグループがある場合は、このアクセス権限を FlexCache SVM でレプリケートする必要があります。

NFSv4 アクセス制御リスト (ACL) がオリジンのボリュームに存在する場合は、LDAP クライアント構成が一致している必要があります。NFSv4 プロトコルは、NFSv3 プロトコルのように UID と GID を使用しませんが、グループ名とユーザー名を使用します。ACL を FlexCache で適切に適用するには、ONTAP がこれらのグループ名とユーザ名を検出できる必要があります。

8.3.3 マルチプロトコルアクセス

オリジンボリュームへのマルチプロトコルアクセスが許可され、設定されている場合は、前の 2 つの項を FlexCache SVM にも適用する必要があります。同じドメインに対する CIFS サーバと LDAP クライアントの構成を作成し、FlexCache に適用する必要があります。さらに、ユーザー名マッピング構成を複製する必要があります。

8.3.4 ユーザー名マッピング

オリジンにユーザー名マッピング設定がある場合は、設定を複製して FlexCache ボリュームに適用する必要があります。これは、確実に権限が適用され、認証情報がオリジンと同じ方法で FlexCache に作成されるようにするためです。

- **ベストプラクティス 6: オリジンと同じ方法で CIFS サーバ、LDAP クライアント、ユーザー名マッピングを構成する**

  FlexCache SVM の CIFS サーバは、オリジン CIFS サーバと同じドメイン、フォレスト、または信頼される側のドメインに配置する必要があります。FlexCache SVM の LDAP 設定もオリジンと同じである必要があります。LDAP サーバー名は、バインドユーザーと同じく異なる場合があります。そのホスト名がオリジンと同じドメインを提供し、バインド DN(識別名) がまったく同じアクセス権を持つ限り、それは許可されます。
  ユーザー名マッピング設定も複製し、FlexCache SVM に適用する必要があります。

8.4 監査、FPolicy、およびウイルス対策スキャン

ONTAP 9.7 以降では、オリジンで監査、ウイルス対策スキャン、および FPolicy 操作を実行する機能が導入されています。オリジンは書き込みができる場所であるため、ウイルス対策の保護が必要なのはそこだけです。オリジンがウイルス対策ソフトウェアで保護されると、すべてのキャッシュが準拠し、キャッシュでウイルス対策設定を行う必要がなくなります。
ONTAP のウイルス対策は、SMB をウイルス対策サーバーへのプロトコルとして使用する場合にのみ構成できます。したがって、FlexCache オリジンのウイルス対策機能を使用する場合は、そこで CIFS サーバを構成する必要があります。また、ONTAP は SMB 操作のオンデマンドスキャンのみをサポートしています。FlexCache 操作は SMB 操作ではないため、ボリュームと SVM のオンデマンドスキャンスケジュールを使用する必要があります。

- **ベストプラクティス 7: CIFS サーバを構成し、ウイルス対策のためにオンデマンドスキャンを使用**

  FlexCache の関係があるボリュームのウイルス対策機能を使用したい場合は、オリジンで CIFS サーバとオンデマンドのウイルス対策スキャンを設定します。

監査はオリジンボリュームに対してのみ実行されます。すべての書き込みと変更は最終的にオリジン側で行われるため、書き込みと変更を監査する必要性はすべてのボリュームでカバーされます。読み取りは、読み取りがキャッシュではなくオリジンにある場合にのみ監査されます。
また、FPolicy はオリジンボリュームでのみ実行されます。つまり、FPolicy で使用できるのはオリジンでの読み取りと書き込みだけです。書き込みはキャッシュに格納されるため、ネイティブの FPolicy ファイルブロックはすべての書き込みに使用できます。その他のサードパーティ製 FPolicy サーバも使用できますが、イベントを FPolicy サーバに転送するのはオリジンだけであることに注意してください。キャッシュはイベントを転送しません。

8.5 キャッシュボリュームサイズ

最適なキャッシュボリュームサイズはデータによります。ワーキングセットは、特定の操作またはジョブに対して FlexCache ボリュームで使用されるデータ量です。ワーキングセットによって、FlexCache ボリュームのサイズ設定が決まります。たとえば、オリジンボリュームに 1TB のデータが含まれているが、特定のジョブで 75GB のデータしか必要としない場合、FlexCache ボリュームの最適なサイズは、ワーキングセットサイズ (75GB) とオーバーヘッド (約 25%) の合計です。この場合、75GB + 25% × 75 = 93.75GB または 94GB になります。これは、読み取り中のすべてのデータが FlexCache でキャッシュされるようにするための最も積極的なサイズ設定です。このセクションで説明しているこの規則には、いくつかの例外があります。

最適なキャッシュボリュームサイズを決定するもう 1 つの方法は、オリジンボリュームサイズの 10 ～ 15% を取得してキャッシュに適用することです。オリジンのボリュームサイズが 50TB の場合、キャッシュのサイズは 5 ～ 7.5TB になります。この方法は、ワーキングセットが明確に理解されていない場合に使用し、統計、使用サイズ、その他の指標を使用して、全体的な最適キャッシュサイズを決定することができます。

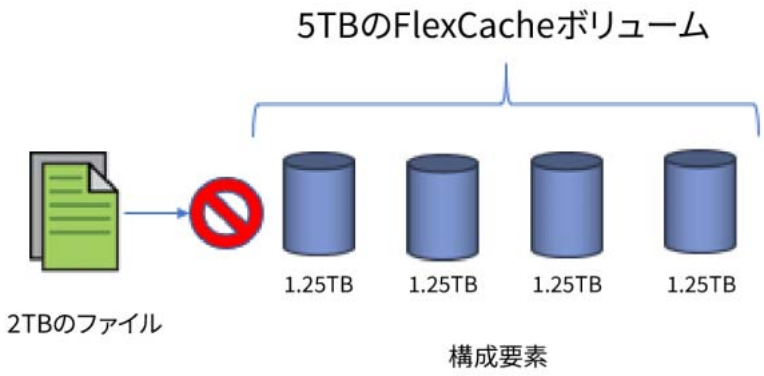
- **ベストプラクティス 8: キャッシュサイズを具体的に定義する**
  FlexCache のボリュームサイズを指定するには、必ず FlexCache create の -size オプションを使用してください。

8.5.1 ファイルサイズ

FlexCache ボリュームは FlexGroup ボリュームであるため、最適なキャッシュのために FlexCache ボリュームのサイズを決定する際には、他にも考慮事項があります。FlexGroup ボリュームは、FlexGroup コンテナを作成する構成ボリュームを作成します。データセットにキャッシュしなければならないファイルが構成要素のどれよりも大きい場合、そのファイルは常にオリジンから提供されます。構成するボリュームにそのファイルのための十分な領域がないため、キャッシュされることはありません。

前述の 50TB のオリジンと 5 ～ 7.5TB の FlexCache ボリュームの例では、構成要素に関する前述のセクションに基づくと、推奨される構成要素の数は 4 になります。1.25 ～ 1.8TB を超えるファイルは、構成要素がキャッシュするのに十分なサイズでないためキャッシュされません。エラスティックサイジングと呼ばれる FlexGroup 機能は、ある構成要素から空きスペースを取り出し、それを別の構成要素に追加して、より大きなファイルをキャッシュできるようにします。ただし、この機能に依存する必要がないように、構成ボリュームの適切なサイズ設定を推奨しています。エラスティックサイジングの詳細については、FTI のマニュアルサイトの「ETERNUS AX/HX series ONTAP FlexGroup ボリューム技術概要」を参照してください。

図 8.1 大きすぎてキャッシュできないファイル



これが、FlexCache ボリューム内にボリュームサイズに基づいて正しい数の構成要素を作成することを推奨する主な理由です。オリジン上の各ファイルをキャッシュするのに必要な容量を最適化します。

- **ベストプラクティス 9: キャッシュサイズは最大サイズのファイルよりも大きくする必要があります。**
  FlexCache ボリュームは FlexGroup ボリュームであるため、単一の構成要素が、キャッシュする必要がある最大のファイルよりも小さくなることはありません。デフォルトで構成要素が 1 つあるため、FlexCache ボリュームのサイズは、少なくともキャッシュする最大のファイルと同じ大きさにする必要があります。

> **注意**
>
> 構成要素のサイズがキャッシュされるファイルサイズより小さい場合でも、ONTAP はファイルのキャッシュを試みます。そのため、サイズが原因でキャッシュから削除されます。

8.5.2 削除

FlexCache ボリュームでは、容量の制約により、ファイルはキャッシュからのみ削除できます。スクラバーは、いずれかの構成要素の使用率が 90% を超えると開始します。ONTAP には、スペースが原因でファイルを削除するためにスクラバーが実行された回数を示すカウンターがあります。

```
fc_cluster::*> statistics show waflremote -counter scrub_need_freespace

Object: waflremote
Instance: 0
Start-time: 11/8/2018 21:46:39
End-time: 11/8/2018 21:46:39
Scope: fc_cluster-01

 Counter                           Value
 -------------------------------- --------------------------------
 scrub_need_freespace                   172984
```

定常状態では、FlexCache ボリュームの使用率の値が高いことは珍しくありません。使用率の値が大きいのは定常状態では正常であり、FlexCache のボリューム使用率が常に 80 ～ 90% であれば問題はありません。

8.5.3 自動拡張

FlexCache ボリュームの容量を節約するには、自動拡張を使用するとよい場合があります。ワーキングセットサイズがわからない場合や、FlexCache クラスタの容量を確保する必要がある場合は、自動拡張の使用を検討してください。

FlexCache ボリュームで自動拡張をセットアップするには、初期サイズで FlexCache ボリュームを作成する必要があります。FlexCache ボリュームの初期サイズをオリジンの 1 ～ 5% の間に設定することを推奨します。ボリュームの自動拡張が有効の場合、サイズが大きいファイルがあると、空き容量への影響が構成要素のサイズによるものよりも大きくなるため、データのファイルサイズも考慮しなければなりません。

前述の例の 50TB のオリジンボリュームを再利用しましょう。FlexCache の初期ボリュームサイズは 500GB ～ 2.5TB に設定されています。最適な構成要素のサイズを維持するには、ファイルがそのサイズを超えないようにする必要があります。これらのファイルサイズは、前述の 1.25 ～ 1.8TB のサイズよりもオリジンに存在する可能性が高くなります。
そのため、キャッシュサイズや構成要素サイズが小さい自動拡張を使用する場合よりも効果があります。自動拡張の最大サイズをオリジンの 10 ～ 15% の間に設定することも推奨します。これにより、キャッシュするデータが増えるにつれて FlexCache が最大値を超えないようにします。

```
fc_cluster::*> volume autosize -volume vol_fc_cache1 -vserver fc-svm1 -maximum-size 5TB -
mode grow
```

自動拡張は、特定のしきい値でのみトリガーされます。デフォルトでは、このしきい値は 85% です。ある構成要素の使用率が 85% に達すると、ONTAP によって計算された特定の数まで拡張します。また、削除のしきい値は 90% です。そのため、取り込み率 (オリジンから最初に読み取られ、キャッシュに書き込まれる) が 5% を超えると、拡張と削除のループによって望ましくない動作が発生する可能性があります。

自動拡張が発生すると、EMS メッセージが生成されます。これらのメッセージは、構成要素が何であるか、および構成要素がどれだけ大きくなるかを示します。次の例を参照してください。

```
fc_cluster::*> event log show
Time        Node       Severity   Event
------------------- --------------- ------------ --------------------------
11/12/2018 19:45:15 fc_cluster-01 NOTICE wafl.vol.autoSize.done: Volume autosize: Automatic
grow of volume 'vol_fc_cache1__0001@vserver:6cbc44db-cd6f-11e8-a1ce-00a0b8a0e70d' by 110MB
is complete.
```

8.6 LS ミラーの交換

データボリューム用の LS ミラーは、推奨されていません。FlexCache テクノロジーは、LS ミラーがクラスタ内の複数のドライブおよびノードに読み取りを分散する機能の代わりに使用できます。クラスタ内の複数のドライブおよびノードに負荷を分散すると、ホットボリューム、ホットファイル、ホットディレクトリを軽減したり、データの負荷分散を向上させることができます。この機能をクライアントが使用できるようにする唯一の方法は、オートマウンタを使用することです。

NFS クライアントは、高度な自動マウントマップトリックを使用して、キャッシュの 1 つまたはオリジンなどの異なるボリュームをマウントし、複数の LIF および場合によっては SVM に接続を分散させることができます。この方法では、複数の SVM、ノード、またはボリューム間で負荷が均等に分散されるわけではありませんが、複数の LIF、ノード、および SVM 間で特定のデータボリュームへの接続を平準化させることができます。

8.6.1 autofs のインストール

ほとんどの標準リポジトリには、autofs がパッケージとして用意されています。これは、次のコマンドを使用して実行できます。

```
yum install autofs
apt-get install autofs
dnf install autofs
```

8.6.2 FlexCache ボリュームの作成

次の手順では、FlexCache ボリュームを作成します。FlexCache ボリュームがオリジンと同じ SVM に作成されている場合は、ピア接続は必要ありません。FlexCache ボリュームを別の SVM で作成する場合は、SVM をピア接続し、FlexCache アプリケーションをピアに追加する必要があります。どちらの方法でも利点は変わりませんが、FlexCache ボリュームを別の SVM に配置すると管理はしやすくなります。マウントポイントは、すべての FlexCache ボリュームとオリジンで同じにすることができます。

```
fc_cluster::> flexcache create -vserver fc-svm1 -volume data_volume -aggr-list aggr1_node1
-size 100G -origin-vserver origin-svm -origin-volume data_volume -junction-path /data_volume
```

8.6.3 自動マウントマップファイルを作成し、Auto.Master ファイルに追加する

FlexCache ボリュームを自動マウントマップファイルに定義するには、いくつかの方法があります。キャッシュボリュームが同じ SVM にある場合、またはオリジンと同じマウントポイントを持っていない場合は、マップファイルで次のように定義できます。この例では、次のように表示されます。

- data_volume にマウントされている SVM origin-svm 上のオリジンボリューム
- /data_volume_cache にマウントされた同じ SVM 上のキャッシュ
- /data_volume_cache1 にマウントされた SVM cache1-svm 上のキャッシュ
- /data_volume_cache2 にマウントされた SVM cache1-svm 上のキャッシュ

```
/mnt/data -rw,hard origin-svm:/data_volume \
            origin-svm:/data_volume_cache \
            fc-svm1:/data_volume_cache1 \
            fc-svm1:/data_volume_cache2
```

各 FlexCache ボリュームが別々の SVM にあり、同じマウントパスを使用する場合は、マップファイルに各 FlexCache ボリュームを定義するためのショートカットがあります。次に、SVM origin-svm 上のオリジンが /data_volume にマウントされ、cache1-svm、cache2-svm、および cache3-svm 上のキャッシュを持つマップファイルの例を示します。これらのキャッシュも、各 SVM の /data_volume にマウントされます。

```
/mnt/data -rw,hard  origin-svm,fc-svm1, fc-svm2, fc-svm3:/data_volume
```

処理が完了したら、autofs サービスを再起動すると、データは /mnt/data で利用できるようになります。クライアントからのマウントがどの SVM に適用されるかは不明ですが、オートマウンタが順序を維持するには以下のルールがあります。
まず、サブネットマスクの近さを調べます。マップファイル内の 1 つのエントリがクライアントと同じサブネット内にある場合は、排他的に使用されます。同じサブネット内のマップファイルに複数のエントリがある場合、NFS クライアントは各 NFS サービスに NULL 呼び出しを発行して、それぞれに ping を発行します。応答時間が最も短いサービスがマウントされます。

9. トラブルシューティング

このセクションでは、FlexCache ボリュームで発生する問題について説明します。

ファイルが提供されていません。

FlexCache ボリュームがファイルを提供しない理由はいくつか考えられます。一番の理由は、ファイルがキャッシュされず、ファイルを取得するためのオリジンへの接続がないことです。接続を確認するために実行できるコマンドがいくつかあります。FlexCache ボリュームがオリジンから切断された場合に予想される動作の詳細については、「4.9 切断モード」(P.27) を参照してください。FlexCache ボリュームが切断されたときに実行できる操作と実行できない操作を理解することが重要です。

以下のコマンドは、クラスタ間 LIF を介した適切な通信があることを確認します。

- クラスタ間 ping
- クラスタ間接続の表示
- ネットワーク ping

ファイルの提供が遅い。

ファイルの提供速度が遅い場合は、オリジンまたはオリジンノードへの低速リンクからファイルがロードされている可能性があります。このエラーは、次のコマンドの組み合わせで見つかります。

- `waflremote fc_retrieve_ops`
  統計によると、本コマンドでオリジンにアクセスが必要な操作が増加していることを示します。
- `waflremote fc_remote_latecy_histo`
  統計によると、本コマンドでオリジンからデータを取得する時間の幅が増加していることを示します。

ETERNUS AX series オールフラッシュアレイ , ETERNUS HX series ハイブリッドアレイ
ONTAP での FlexCache ONTAP 9.11.1

P3AG-6032-03Z0

発行年月　2025 年 3 月
発行責任　エフサステクノロジーズ株式会社

Fsas Technologies