パラレル**NFS**

ONTAP 9

NetApp
January 23, 2026

目次

パラレルNFS

はじめに

ONTAPでの並列NFS（pNFS）について

パラレルNFSは、メタデータとデータパスを分離することで、クライアントがNFSv4.1サーバ上のファイルデータに直接アクセスできるようにするために、2010年1月にRFC-5661でRFC標準として導入されました。この直接アクセスにより、データのローカライゼーション、CPU効率、処理の並列化といったパフォーマンス上のメリットが得られます。その後、2018年にpNFSレイアウトタイプ（RFC-8434）を網羅したRFCが作成され、ファイル、ブロック、およびオブジェクトレイアウトの標準が定義されています。ONTAPはpNFS操作にこのファイルレイアウトタイプを活用します。

2024年7月より、これまでPDF形式で公開されていたテクニカルレポートの内容がONTAP製品ドキュメントに統合されました。ONTAP NFSストレージ管理ドキュメントには、_TR-4063：NetApp ONTAPにおけるパラレルネットワークファイルシステム（pNFS）_の内容が含まれるようになりました。

NFSv3は長年にわたり、ほぼすべてのユースケースで使用されてきたNFSプロトコルの標準バージョンでした。しかし、このプロトコルには、ステートフル性の欠如、基本的な権限モデル、基本的なロック機能といった制限がありました。NFSv4.0（RFC 7530）では、NFSv3に対する一連の改良が導入され、その後のNFSv4.1（RFC 5661）およびNFSv4.2（RFC 7862）バージョンでさらに改良が加えられ、パラレルNFS（pNFS）などの機能が追加されました。

NFSv4.xの利点

NFSv4.x は NFSv3 に比べて次のような利点があります：

- ファイアウォールとの親和性。NFSv4では1つのポート（2049）しか使用しません。
- 高度でアグレッシブなキャッシュ管理。NFSv4.xの委譲機能など。
- 強固なRPCセキュリティ種別。暗号化を実装します。
- 文字の国際化
- 複合操作。
- TCPでのみ動作。
- ステートフル プロトコル（NFSv3はステートレス）。
- 効率的な認証メカニズムのための完全なKerberos統合
- NFSリファーラル
- UNIXおよびWindowsと互換性のあるアクセス制御のサポート。
- 文字列ベースのユーザ識別子とグループ識別子。
- pNFS（NFSv4.1）
- 拡張属性（NFSv4.2）

- セキュリティラベル（NFSv4.2）
- スパース ファイル操作（FALLOCATE）（NFSv4.2）

ベスト プラクティスや機能の詳細など、NFSv4.x 全般の詳細については、 "NetAppテクニカル レポート4067：『NFS Best Practice and Implementation Guide』"を参照してください。

関連情報

- "NFS 構成の概要"
- "NFSの管理 - 概要"
- "FlexGroupボリューム管理"
- "NFSトランキングの概要"
- https://www.netapp.com/pdf.html?item=/media/19370-tr-4523.pdf
- "NetApp テクニカルレポート 4616：ONTAP における NFS Kerberos と Microsoft Active Directory"

ONTAPのpNFSアーキテクチャについて

pNFSアーキテクチャは、pNFSをサポートするNFSクライアント、メタデータ操作専用のパスを提供するメタデータサーバ、およびファイルへのローカライズされたパスを提供するデータサーバという3つの主要コンポーネントで構成されています。

pNFSへのクライアントアクセスには、NFSサーバで使用可能なデータパスとメタデータパスへのネットワーク接続が必要です。NFSサーバにクライアントがアクセスできないネットワークインターフェイスが含まれている場合、サーバはクライアントにアクセスできないデータパスをアドバタイズし、サービス停止を引き起こす可能性があります。

メタデータサーバ

pNFSのメタデータサーバは、NFSサーバでpNFSが有効になっているときに、クライアントがNFSv4.1以降を使用してマウントを開始すると確立されます。これが完了すると、すべてのメタデータトラフィックはこの接続を介して送信され、インターフェースが別のノードに移行された場合でも、マウント中はこの接続上に維持されます。

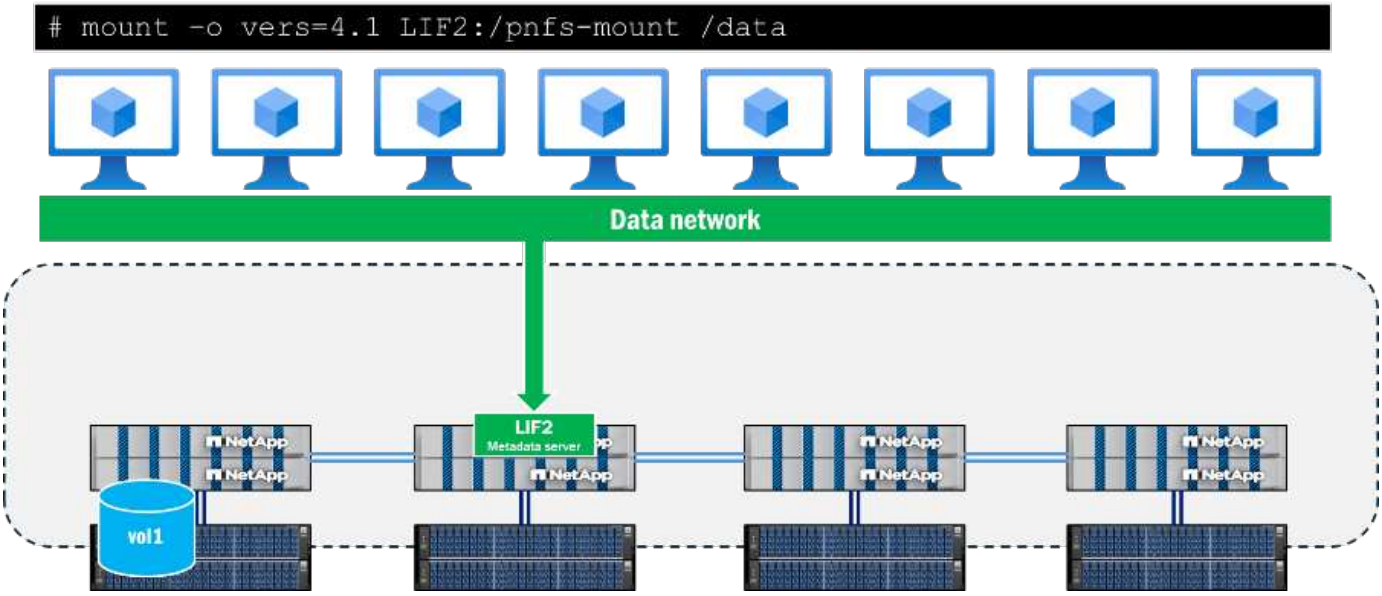


図 1. ONTAPでpNFSのメタデータサーバを確立

pNFSのサポートは、マウント呼び出し、具体的にはEXCHANGE_ID呼び出し時に決定されます。これは、NFS操作の下にあるパケットキャプチャでフラグとして確認できます。pNFSフラグ `EXCHGID4_FLAG_USE_PNFS_DS`と `EXCHGID4_FLAG_USE_PNFS_MDS`が1に設定されている場合、インターフェースはpNFSのデータ操作とメタデータ操作の両方に対応します。

```
Operations (count: 1)
  Opcode: EXCHANGE_ID (42)
    Status: NFS4_OK (0)
    clientid: 0x004050a97100001c
    seqid: 0x00000001
    flags: 0x00060100, EXCHGID4_FLAG_USE_PNFS_DS, EXCHGID4_FLAG_USE_PNFS_MDS, EXCHGID4_FLAG_BIND_PRINC
      0... .... .... .... .... .... .... .... = EXCHGID4_FLAG_CONFIRMED_R: Not set
      .0.. .... .... .... .... .... .... .... = EXCHGID4_FLAG_UPD_CONFIRMED_REC_A: Not set
      .... .... .... .1.. .... .... .... .... = EXCHGID4_FLAG_USE_PNFS_DS: Set
      .... .... .... ..1. .... .... .... .... = EXCHGID4_FLAG_USE_PNFS_MDS: Set
      .... .... .... ...0 .... .... .... .... = EXCHGID4_FLAG_USE_NON_PNFS: Not set
      .... .... .... .... .... ...1 .... .... = EXCHGID4_FLAG_BIND_PRINC_STATEID: Set
      .... .... .... .... .... .... .... ..0. = EXCHGID4_FLAG_SUPP_MOVED_MIGR: Not set
      .... .... .... .... .... .... .... ...0 = EXCHGID4_FLAG_SUPP_MOVED_REFER: Not set
```

図 2. pNFSマウントのパケットキャプチャ

NFSのメタデータは通常、ファイルハンドル、権限、アクセス時刻と変更時刻、所有権情報などのファイルとフォルダの属性で構成されます。メタデータには、作成と削除の呼び出し、リンクとリンク解除の呼び出し、名前の変更などが含まれる場合もあります。

pNFSには、pNFS機能に固有のメタデータ呼び出しのサブセットも存在します。これらについては"RFC 5661"で詳しく説明します。これらの呼び出しは、pNFS対応デバイス、デバイスとデータセットのマッピング、その他の必要な情報を決定するのに役立ちます。次の表は、これらのpNFS固有のメタデータ操作の一覧です。

処理	概要
LAYOUTGET	メタデータ サーバからデータ サーバ マップを取得します。
LAYOUTCOMMIT	サーバはレイアウトをコミットし、メタデータ マップを更新します。

処理	概要
LAYOUTRETURN	レイアウト、またはデータが変更された場合は新しいレイアウトを返します。
GETDEVICEINFO	クライアントは、ストレージ クラスタ内のデータ サーバの更新情報を取得します。
デバイスリスト取得	クライアントは、ストレージ クラスタに参加しているすべてのデータ サーバのリストを要求します。
CB_LAYOUTRECALL	競合が検出された場合、サーバはクライアントからデータ レイアウトを再呼び出しします。
CB_RECALL_ANY	すべてのレイアウトをメタデータ サーバに返します。
CB_NOTIFY_DEVICEID	デバイス ID の変更を通知します。

データパス情報

メタデータサーバが確立され、データ操作が開始されると、ONTAPはpNFSの読み取りおよび書き込み操作に有効なデバイスIDと、クラスタ内のボリュームをローカル ネットワーク インターフェースに関連付けるデバイス マッピングの追跡を開始します。このプロセスは、マウント内で読み取りまたは書き込み操作が実行されたときに発生します。 `GETATTR`などのメタデータ呼び出しでは、これらのデバイス マッピングはトリガーされません。そのため、マウント ポイント内で `ls`コマンドを実行しても、マッピングは更新されません。

デバイスとマッピングは、次に示すように、ONTAP CLI の高度な権限を使用して表示できます。

```
::*> pnfs devices show -vserver DEMO
  (vserver nfs pnfs devices show)
Vserver Name     Mapping ID      Volume MSID     Mapping Status
Generation
---------------  --------------- --------------- ---------------
-------------
DEMO             16              2157024470      available       1

::*> pnfs devices mappings show -vserver SVM
  (vserver nfs pnfs devices mappings show)
Vserver Name    Mapping ID      Dsid            LIF IP
--------------  --------------- --------------- --------------------
DEMO            16              2488            10.193.67.211
```

これらのコマンドでは、ボリューム名は使用されません。代わりに、ボリュームに関連付けられた数値ID（マスター セット ID（MSID）とデータ セット ID（DSID））が使用されます。マッピングに関連付けられたボリュームを確認するには、ONTAP CLIのadvanced権限で `volume show -dsid [dsid_numeric]`または `volume show -msid [msid_numeric]`を使用します。

クライアントがメタデータ サーバ接続から離れたノードにあるファイルの読み取りまたは書き込みを試みると、pNFSは適切なアクセス パスをネゴシエートし、これらの操作におけるデータの局所性を確保します。クライアントは、ファイルにアクセスするためにクラスタ ネットワークを経由するのではなく、アドバタイズされたpNFSデバイスにリダイレクトします。これにより、CPUオーバーヘッドとネットワークレイテンシが削減されます。

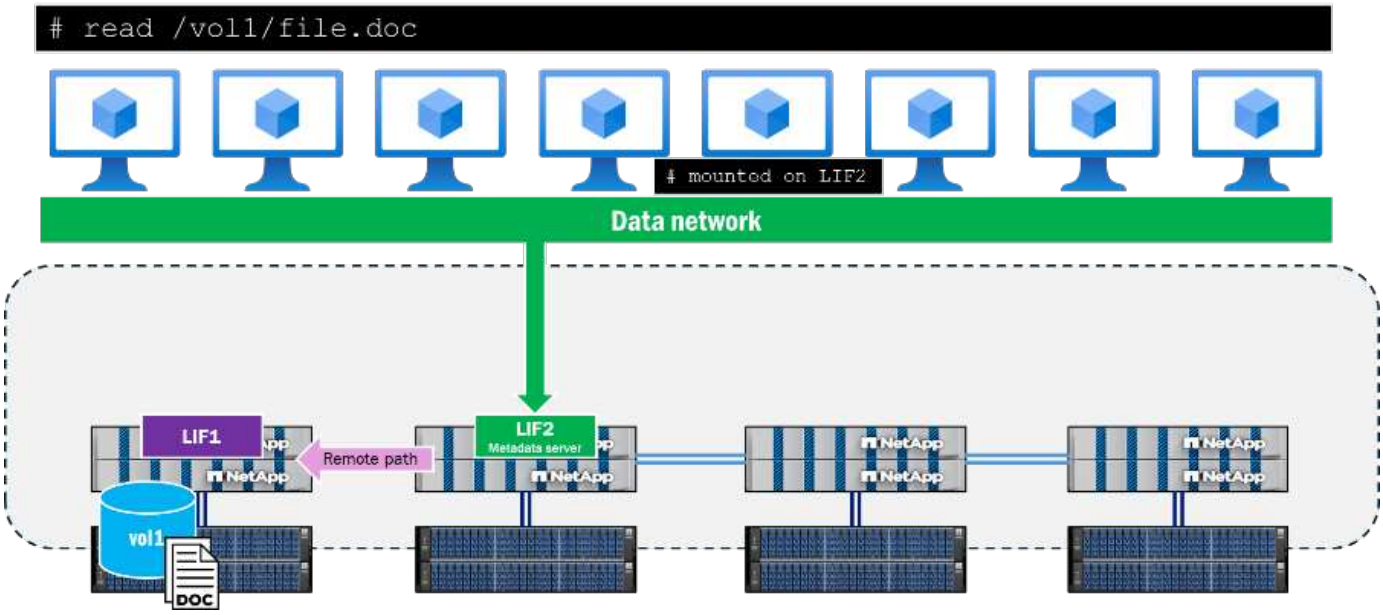


図 **3. pNFS**を使用しない**NFSv4.1**を使用したリモート読み取りパス

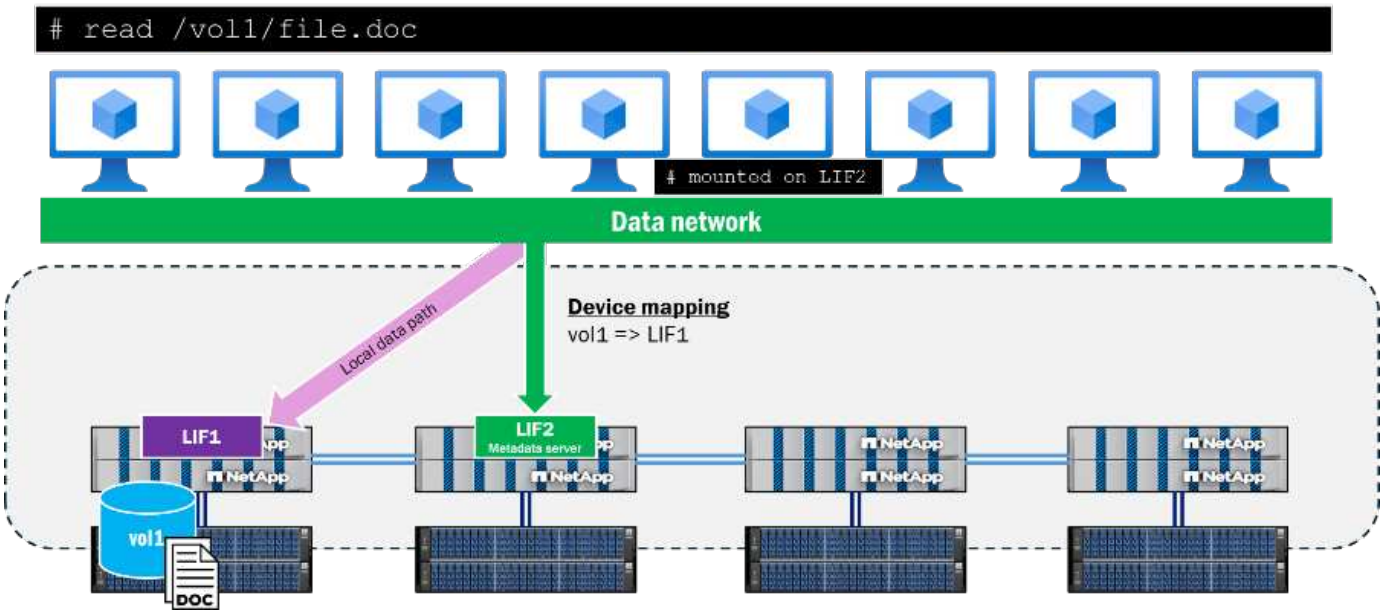


図 **4. pNFS**を使用したローカライズされた読み取りパス

pNFS制御パス

pNFSには、メタデータとデータ部分に加えて、pNFS制御パスも存在します。この制御パスは、NFSサーバがファイルシステム情報を同期するために使用されます。ONTAPクラスタでは、バックエンドクラスタネットワークが定期的にレプリケーションを実行し、すべてのpNFSデバイスとデバイスマッピングが同期されていることを確認します。

pNFSデバイス設定ワークフロー

以下では、クライアントがボリューム内のファイルの読み取りまたは書き込みを要求した後に、pNFSデバイスがONTAPにどのように設定されるかについて説明します。

1. クライアントが読み取りまたは書き込みを要求すると、OPENが実行され、ファイル ハンドルが取得されます。

2. OPENが実行されると、クライアントはメタデータ サーバ接続を介したLAYOUTGET呼び出しでファイル ハンドルをストレージに送信します。
3. LAYOUTGET は、状態 ID、ストライプ サイズ、ファイル セグメント、デバイス ID など、ファイルのレイアウトに関する情報をクライアントに返します。
4. 次に、クライアントはデバイスIDを取得し、GETDEVINFO呼び出しをサーバに送信して、デバイスに関連付けられたIPアドレスを取得します。
5. ストレージは、デバイスへのローカル アクセス用に関連付けられたIPアドレスのリストを含む応答を送信します。
6. クライアントは、ストレージから返されたローカル IP アドレスを介してNFS会話を継続します。

pNFSとFlexGroupボリュームの相互作用

FlexGroupボリュームは、ONTAPでストレージをクラスタ内の複数のノードにまたがるFlexVolボリューム構成要素として提示します。これにより、ワークロードは単一のマウントポイントを維持しながら、複数のハードウェアリソースを活用できます。複数のネットワークインターフェイスを持つ複数のノードがワークロードとやり取りするため、リモートトラフィックがONTAPのバックエンドクラスタネットワークを通過するのは自然な結果です。

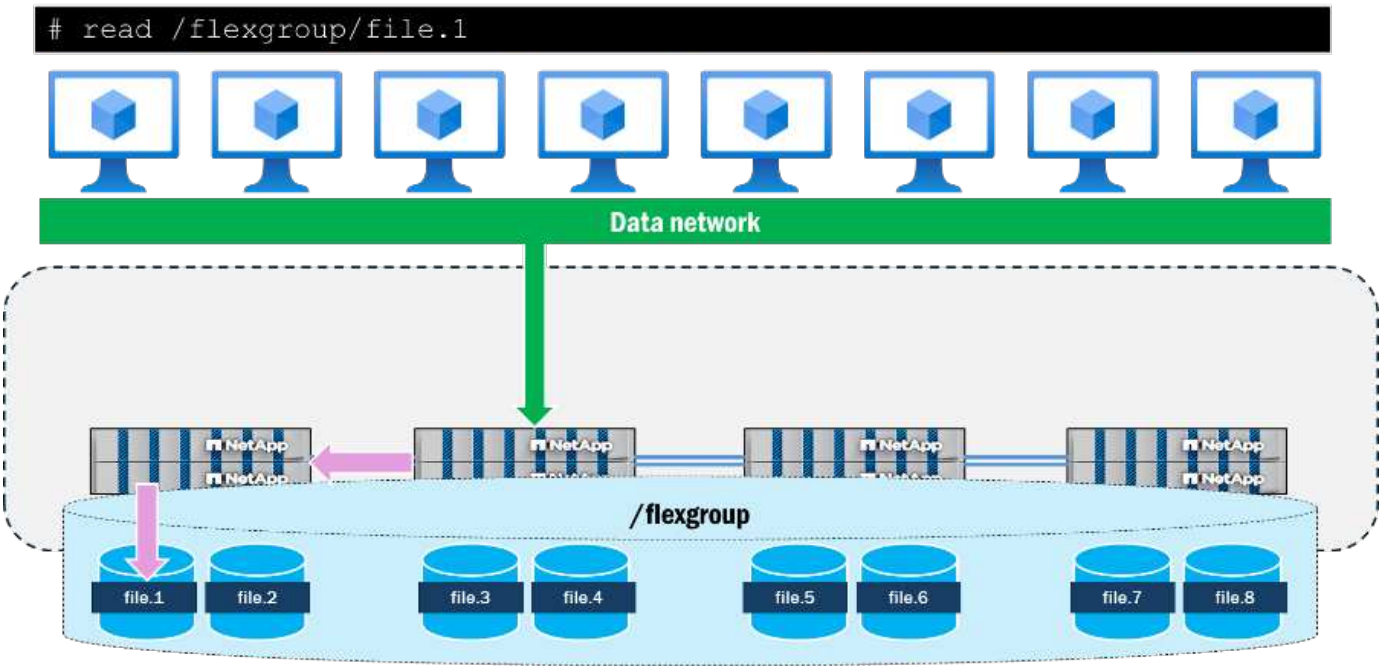


図 **5. pNFS** を使用しない**FlexGroup**ボリューム内の単一ファイル アクセス

pNFSを利用する場合、ONTAPはFlexGroupボリュームのファイルとボリュームのレイアウトを追跡し、それらをクラスタ内のローカル データ インターフェースにマッピングします。例えば、アクセス対象のファイルを含む構成ボリュームがノード1に存在する場合、ONTAPはクライアントにネットワーク トラフィックをノード1のデータ インターフェースにリダイレクトするよう通知します。

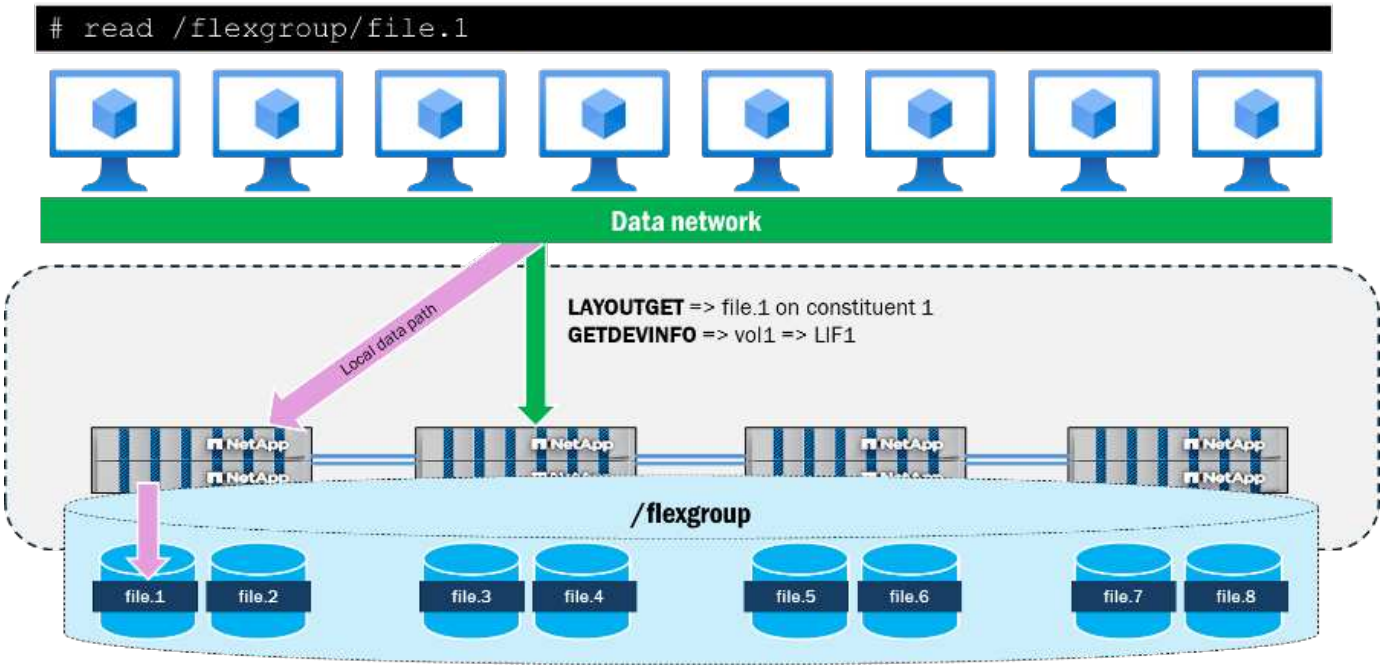


図 **6. pNFS**を使用した**FlexGroup**ボリューム内の単一ファイル アクセス

pNFSは、pNFSのないNFSv4.1では提供されない、単一のクライアントからファイルへの並列ネットワーク パスのプレゼンテーションも提供します。たとえば、クライアントがpNFSのないNFSv4.1を使用して同じマウントから4つのファイルに同時にアクセスする場合、すべてのファイルに同じネットワーク パスが使用され、ONTAPクラスタは代わりにそれらのファイルへのリモート要求を送信します。マウント パスは、すべての操作が単一のパスをたどって単一のノードに到達し、データ操作とともにメタデータ操作も処理するため、操作のボトルネックになる可能性があります。

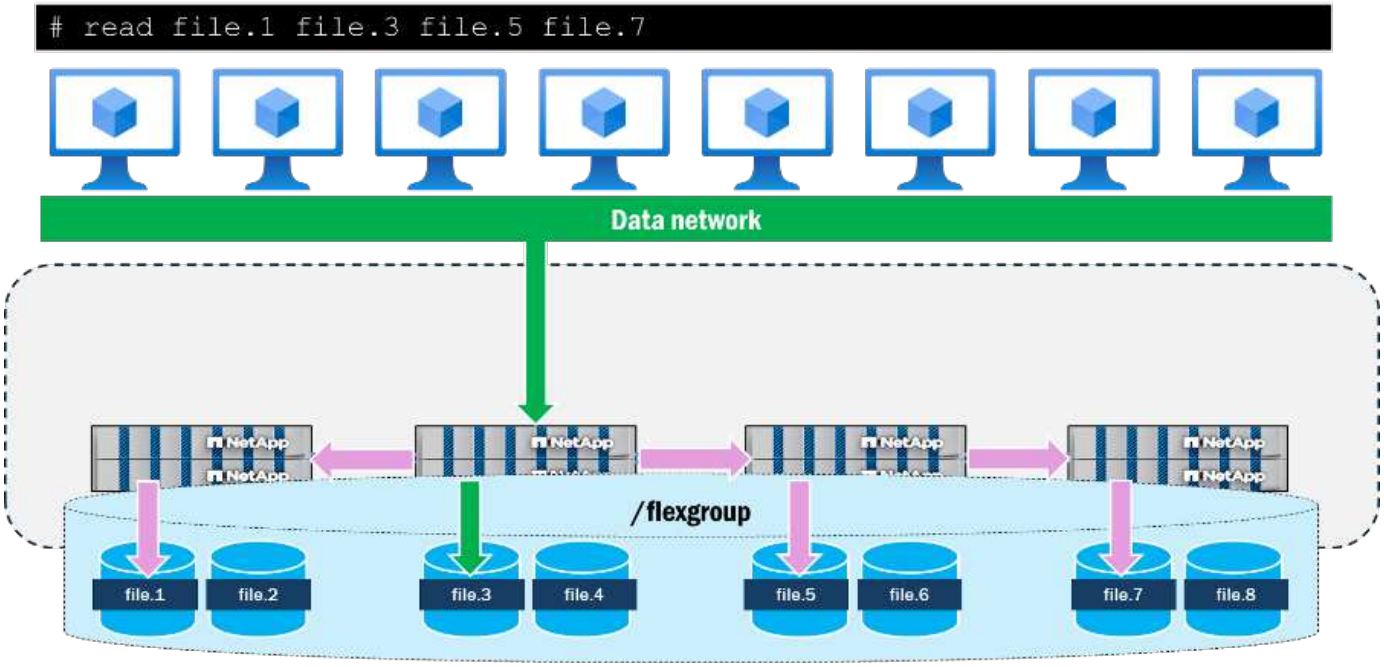


図 **7. pNFS** を使用しない**FlexGroup**ボリュームでの複数の同時ファイル アクセス

pNFSを使用して単一のクライアントから同じ4つのファイルに同時にアクセスする場合、クライアントとサーバはファイルが存在する各ノードへのローカル パスをネゴシエートし、データ操作には複数のTCP接続を使用します。一方、マウント パスはすべてのメタデータ操作の場所として機能します。これにより、ファイルへのローカル パスを使用することでレイテンシが低減されるだけでなく、複数のネットワーク インターフ

ェイスを使用することでスループットも向上します（ただし、クライアントがネットワークを飽和させるのに十分なデータを送信できる場合）。

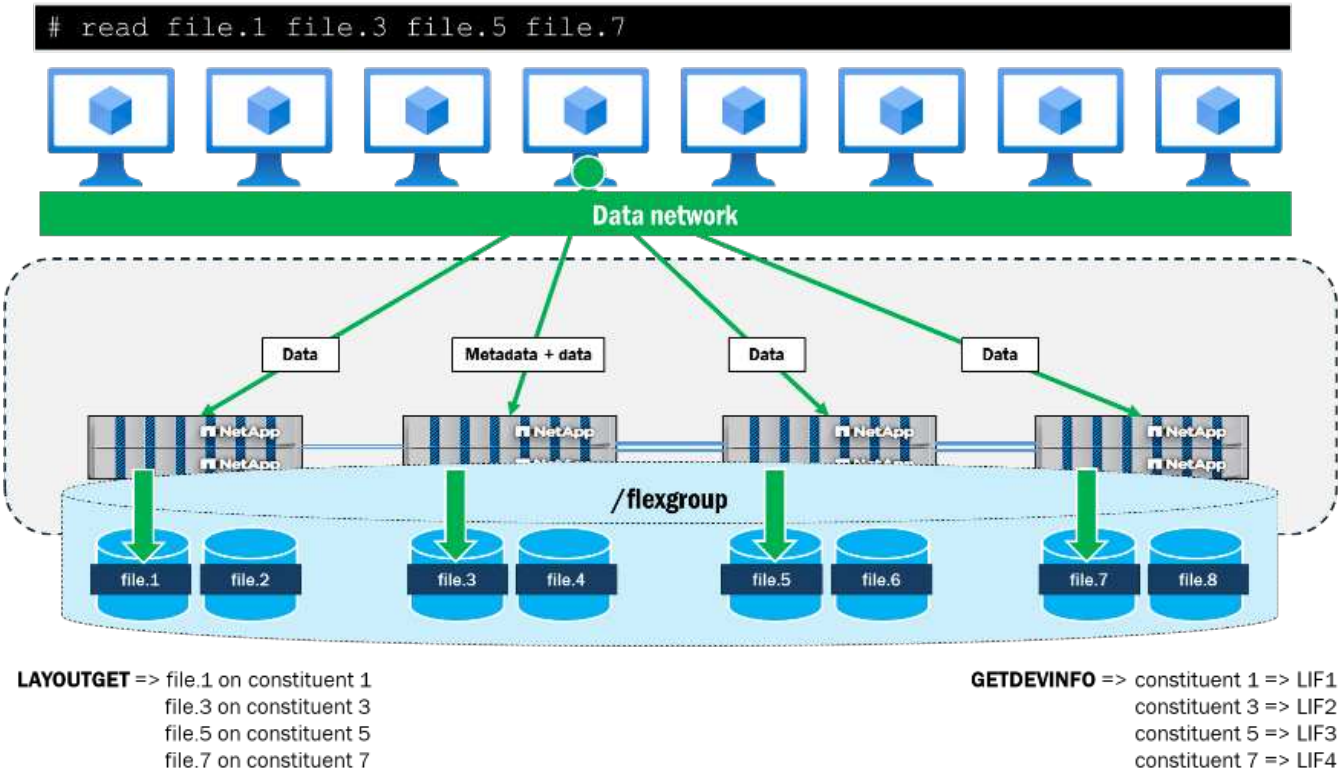


図 **8. pNFS**を使用した**FlexGroup**ボリューム内の複数の同時ファイルアクセス

以下は、単一のRHEL 9.5クライアント上で、2つのONTAPクラスタノードにまたがる異なるコンスティチュエントボリュームにそれぞれ存在する4つの10GBファイルをddを使用して並列に読み取るという簡単なテスト実行の結果です。各ファイルにおいて、pNFSを使用することで全体的なスループットと完了時間が向上しました。pNFSを使用せずにNFSv4.1を使用した場合、マウントポイントに対してローカルなファイルとリモートなファイル間のパフォーマンス差は、pNFSを使用した場合よりも大きくなっていました。

テスト	ファイルあたりのスループット（**MB**/秒）	ファイルあたりの完了時間
NFSv4.1：pNFSなし	• File.1–228（ローカル） • File.2–227（local） • File.3–192（リモート） • File.4–192（リモート）	• File.1–46（ローカル） • File.2–46.1（local） • File.3–54.5（リモート） • File.4–54.5（リモート）
NFSv4.1：pNFS を使用	• File.1–248（ローカル） • File.2–246（ローカル） • File.3–244（pNFS経由のローカル） • File.4–244（pNFS経由のローカル）	• File.1–42.3（local） • File.2–42.6（local） • File.3–43（pNFS経由のローカル） • File.4–43（pNFS経由のlocal）

関連情報

- "FlexGroupボリューム管理"
- "NetAppテクニカルレポート4571：FlexGroupベストプラクティス"

ONTAPにおけるpNFSのユースケース

pNFSをさまざまなONTAP機能と組み合わせて使用することで、パフォーマンスが向上し、NFSワークロードの柔軟性が向上します。

nconnect を使用した pNFS

NFSは、最近のクライアントとサーバーに新しいマウントオプションを導入しました。これにより、単一のIPアドレスをマウントしながら複数のTCP接続を提供することができます。これにより、処理の並列化が向上し、NFSサーバとクライアントの制限を回避できるようになり、特定のワークロードの全体的なパフォーマンスが向上する可能性があります。nconnectは、クライアントがnconnectをサポートしている場合、ONTAP 9.8以降でサポートされます。

pNFSでnconnectを使用する場合、NFSサーバによってアドバタイズされる各pNFSデバイスに対して、nconnectオプションを使用して接続が並列化されます。例えば、nconnectが4に設定され、pNFSに使用できるインターフェースが4つある場合、作成される接続の総数はマウントポイントごとに最大16（nconnect 4個 × IPアドレス4個）になります。

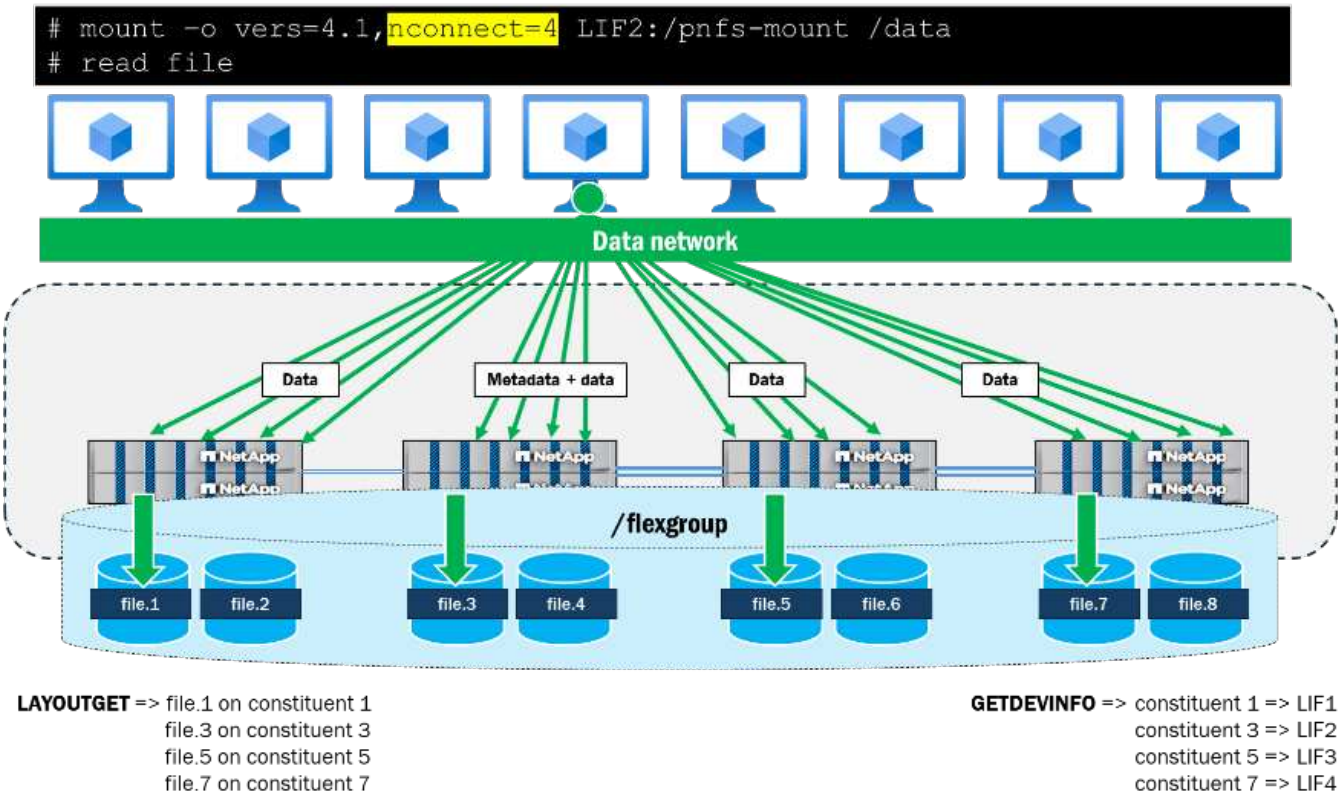


図 **9. nconnect** を **4** に設定した **pNFS**

"ONTAPのNFSv4.1サポートの詳細"

NFSv4.1セッショントランキングを使用したpNFS

NFSv4.1セッション トランキング("RFC 5661、セクション2.10.5"）は、クライアントとサーバ間で複数のTCP接続を使用することで、データ転送速度を向上させるものです。NFSv4.1セッション トランキングのサポートはONTAP 9.14.1で追加されており、セッション トランキングをサポートするクライアントと併用する必要があります。

ONTAPでは、クラスタ内の複数のノード間でセッション トランキングを使用することで、接続全体で追加のスループットと冗長性を実現できます。

セッション トランキングは、複数の方法で確立できます：

- *マウント オプションによる自動検出：*最近のほとんどのNFSクライアントでは、マウント オプション（OSベンダーのドキュメントを参照）を介してセッション トランキングを確立できます。マウント オプションはNFSサーバにセッション トランキングに関する情報をクライアントに返すよう指示します。この情報は、NFSパケットを介して `fs_location4`呼び出しとして表示されます。

  使用されるマウント オプションは、クライアントのOSバージョンによって異なります。たとえば、Ubuntu Linuxフレーバーでは通常、 `max_connect=n`を使用してセッション トランクの使用を通知します。RHEL Linuxディストリビューションでは、 `trunkdiscovery`マウント オプションが使用されます。

  Ubuntuの例

  ```
  mount -o vers=4.1,max_connect=8 10.10.10.10:/pNFS /mnt/pNFS
  ```

  RHELの例

  ```
  mount -o vers=4.1,trunkdiscovery 10.10.10.10:/pNFS /mnt/pNFS
  ```

  RHEL ディストリビューションで `max_connect`を使用しようとすると、代わりに nconnect として扱われ、セッション トランキングは期待どおりに機能しません。

- *手動で確立：*個々のIPアドレスを同じエクスポート パスとマウント ポイントにマウントすることで、セッション トランキングを手動で確立できます。例えば、エクスポート パスが `/pNFS`である同じノードに2つのIPアドレス（10.10.10.10と10.10.10.11）がある場合、マウント コマンドを2回実行します：

  ```
  mount -o vers=4.1 10.10.10.10:/pNFS /mnt/pNFS
  mount -o vers=4.1 10.10.10.11:/pNFS /mnt/pNFS
  ```

  トランクに参加させたいすべてのインターフェイスでこのプロセスを繰り返します。

各ノードには独自のセッション トランクがあります。トランクはノードをまたぎません。

pNFSを使用する場合は、セッション トランキング_または_nconnectのいずれかのみを使用してください。両方を使用すると、メタデータ サーバ接続のみがnconnectのメリットを受け、データ サーバは単一の接続を使用するなど、望ましくない動作が発生します。

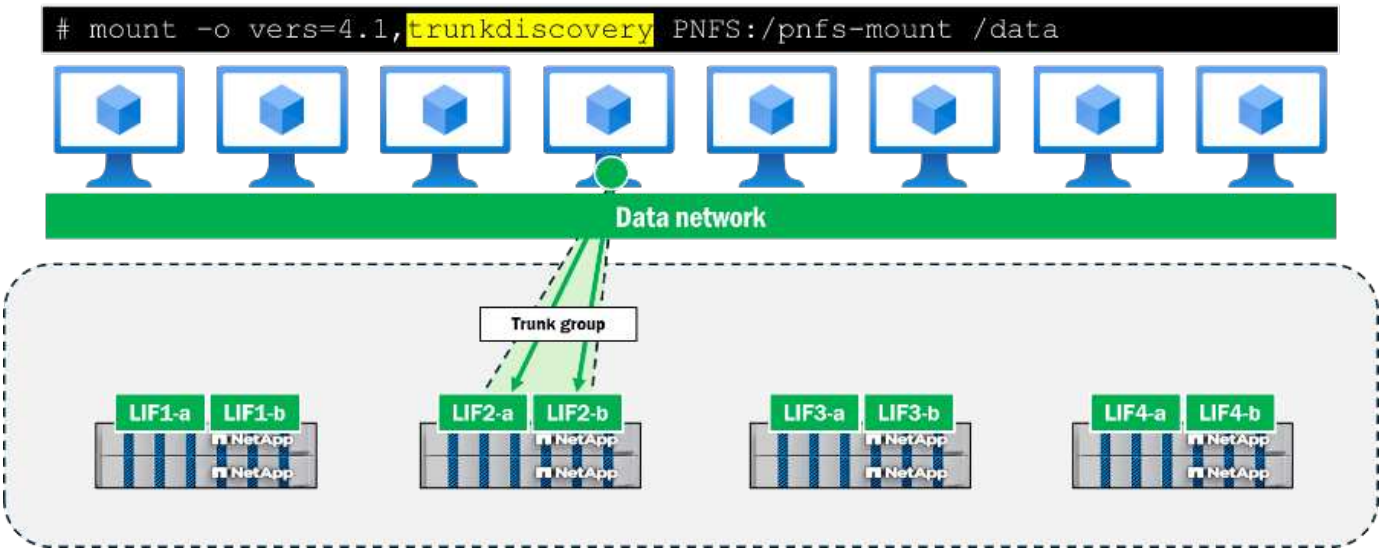
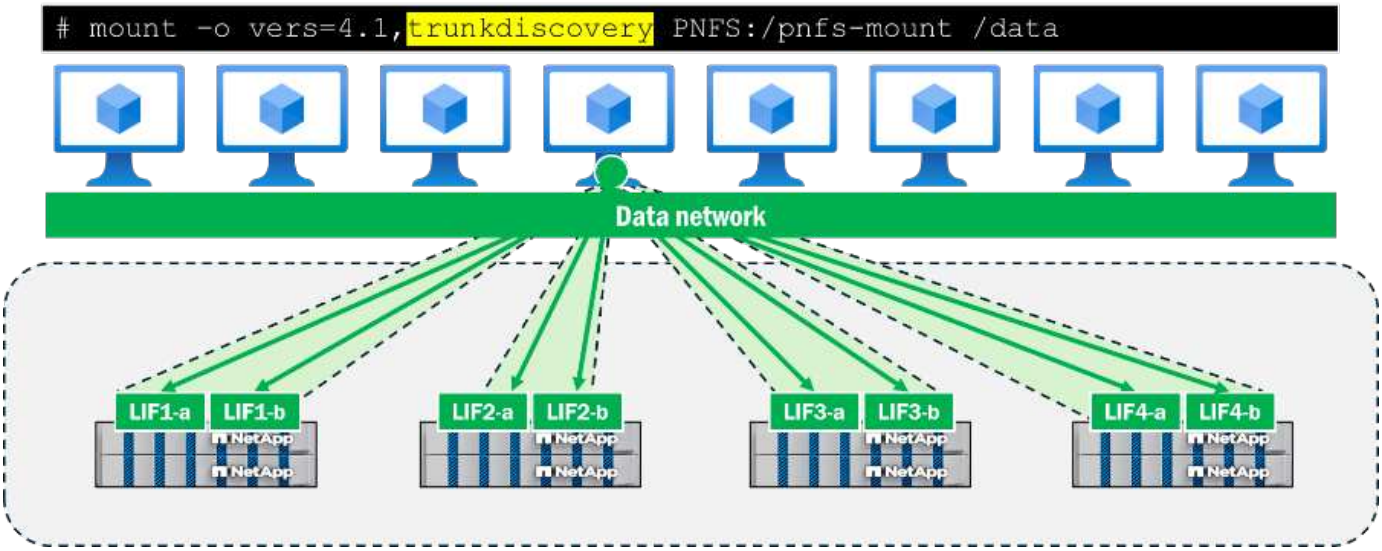


図 **10. ONTAP**での**NFSv4.1**セッション トランキング

pNFSは、クラスタ内の各参加ノードへのローカル パスを提供できます。また、セッション トランキングと併用すると、pNFSはノードごとにセッション トランクを活用して、クラスタ全体のスループットを最大化できます。



`trunkdiscovery`を使用すると、マウントインターフェイスが配置されているNFSサーバ ノード上のリストされたセッション トランク インターフェイスに対して、追加のGETATTR呼び出し（FS_Locations）が利用されます。これらのアドレスが返されると、以降のマウントは返されたアドレスに対して行われます。これは、マウント中のパケット キャプチャで確認できます。

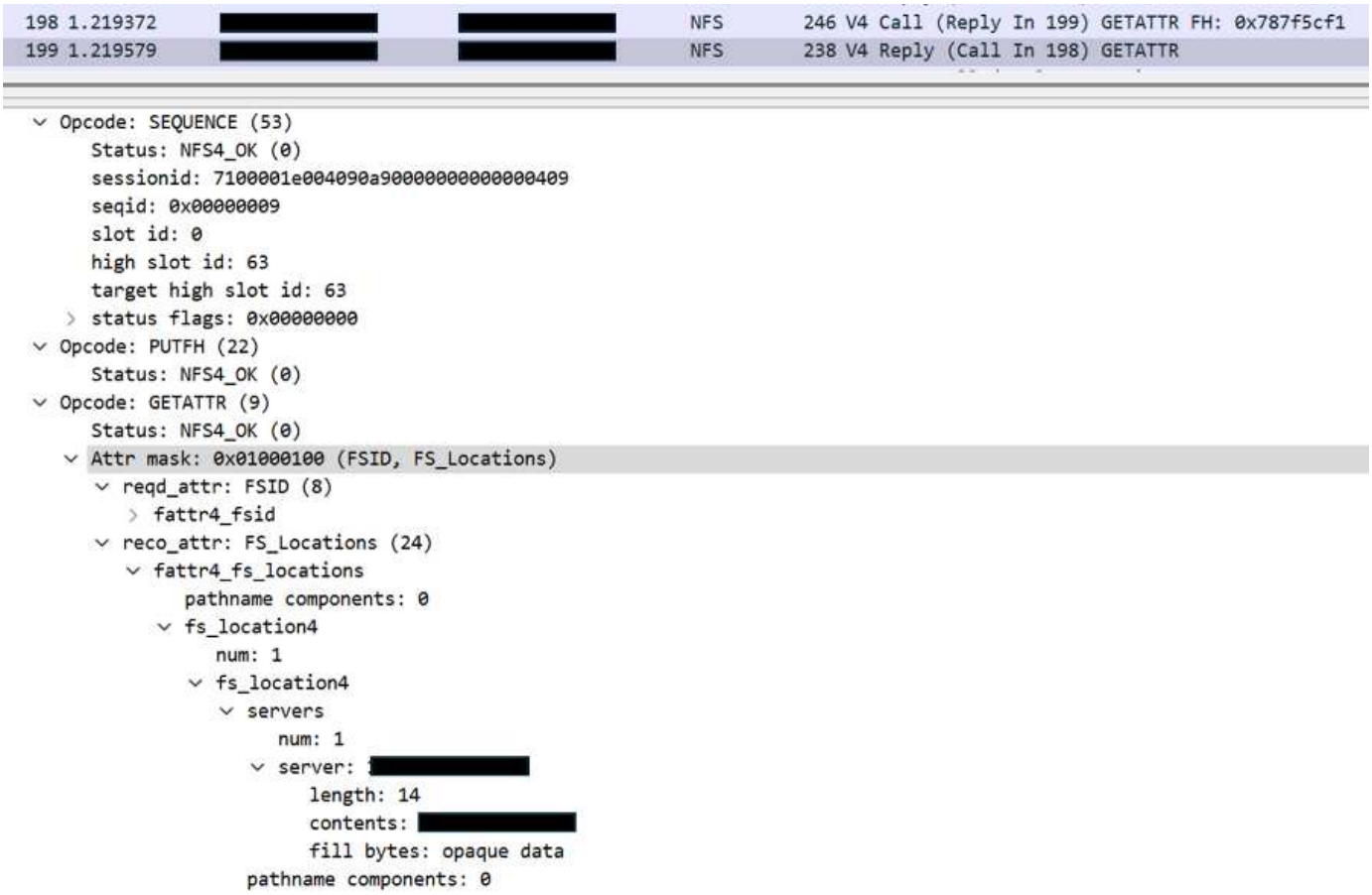


図 **11.** マウント中の**NFS**セッション トランキング検出：パケット キャプチャ

"NFSトランキングの詳細"

pNFS と **NFSv4.1** リファラル

NFSv4.1リファーラルは、マウント要求時にクライアントをボリュームの場所に誘導する初期マウント パス リダイレクト モードを提供します。NFSv4.1リファーラルは単一のSVM内で動作します。この機能は、NFS マウントをデータ ボリュームと同じノードにあるネットワーク インターフェースにローカライズしようとします。クライアントにマウントされている間にそのインターフェースまたはボリュームが別のノードに移動した場合、新しいマウントが確立されるまでデータ パスはローカライズされなくなります。

pNFSはマウント パスのローカライズを試みません。代わりに、マウント パスを使用してメタデータ サーバを確立し、必要に応じてデータ パスを動的にローカライズします。

NFSv4.1 リファラルは pNFS でも使用できますが、この機能は不要です。pNFS でリファラルを有効にしても、目立った効果は得られません。

"NFSv4リファーラルの有効化または無効化"

pNFSと高度な容量バランス調整の相互作用

"高度な容量バランシング"ONTAPでは、ファイルデータの一部をFlexGroupボリュームを構成する複数のボリュームに書き込みます（単一FlexVolボリュームではサポートされません）。ファイルのサイズが大きくなると、ONTAPは別の構成ボリューム（同じノードまたは異なるノード）上の新しいマルチパートinodeへのデータの書き込みを開始します。これらのマルチinodeファイルへの書き込み、読み取り、およびメタデータ操作は、クライアントに対して透過的で、中断を伴いません。高度な容量バランス調整により、FlexGroup構成ボ

リューム間のスペース管理が改善され、より安定したパフォーマンスが実現します。

pNFSは、NFSサーバに格納されているファイル レイアウト情報に応じて、データIOをローカライズされたネットワーク パスにリダイレクトできます。単一の大きなファイルが、クラスタ内の複数のノードにまたがる可能性のある複数の構成ボリュームに分割して作成される場合でも、ONTAPのpNFSは、すべてのファイル パートのファイル レイアウト情報も保持しているため、各ファイル パートにローカライズされたトラフィックを提供できます。ファイルが読み取られると、データ パスのローカル性は必要に応じて変化します。

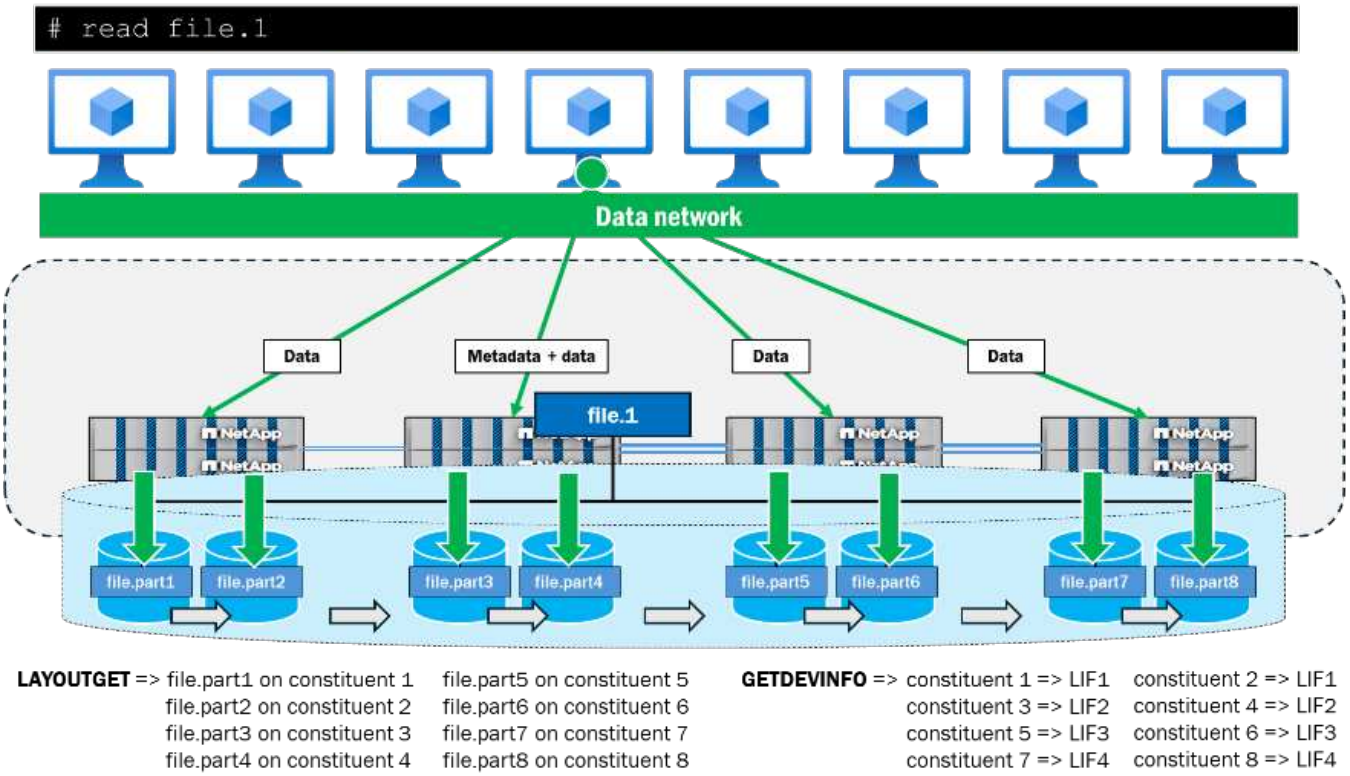


図 **12. pNFS**による高度な容量バランス調整

関連情報

- "FlexGroupボリューム構成"

ONTAPにおけるpNFS導入戦略

pNFSは、メタデータとデータ パスを分離し、データのローカライズを提供し、並列操作を可能にすることで、従来のNFSを改善するために導入されました。

従来のNFSの課題とpNFSの利点

次の表は、従来のNFSの課題を示し、ONTAPのpNFSがそれらの課題にどのように対処するかを説明しています。

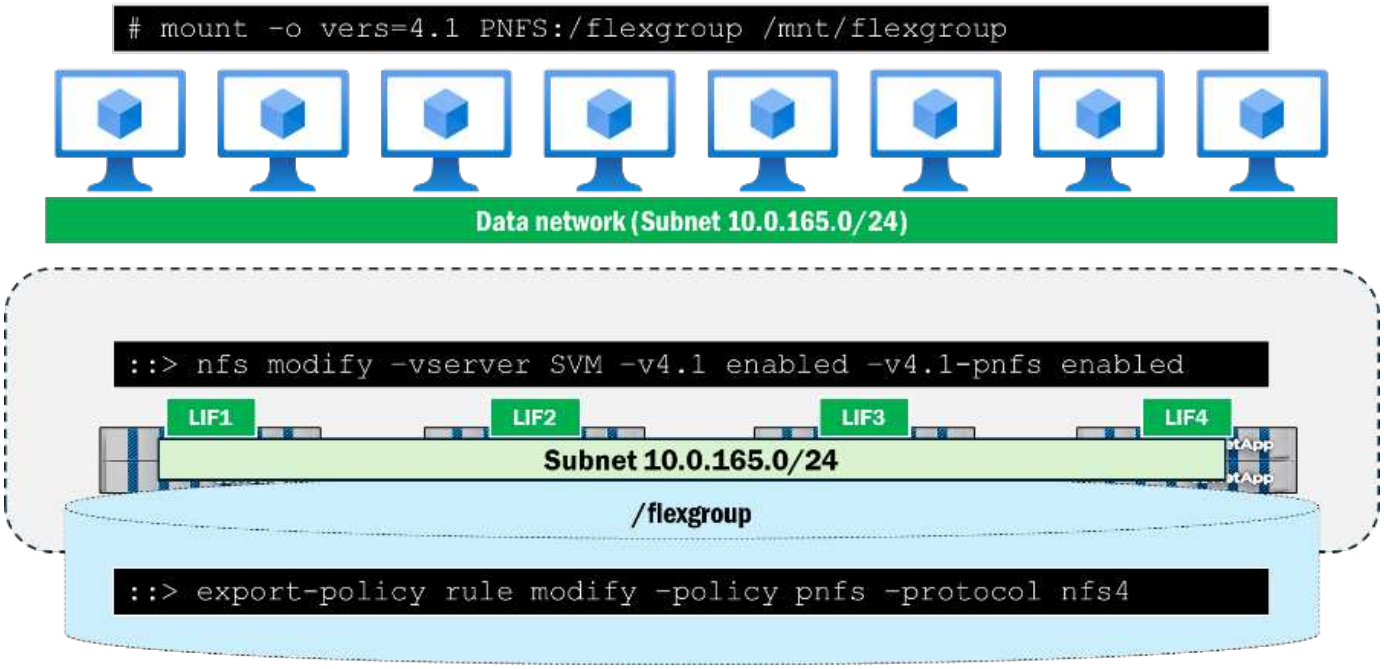
課題	**pNFS**の利点
メタデータとデータの同一パス 従来のNFSでは、メタデータとデータは同じパスを通過します。これにより、単一のパスがクラスタ内の単一のハードウェア ノードに接続されるため、ネットワークとCPUの両方が飽和状態になる可能性があります。多くのユーザーが同じNFSエクスポートにアクセスしようとすると、この問題はさらに悪化します。	メタデータ パスとデータ パスが分離され、データ パスが並列化されます NFS トラフィックのメタデータ パスとデータ パスを分離し、データ パスに複数のネットワーク パスを提供することで、ONTAPクラスタ内のCPUおよびネットワーク リソースが最大化され、ワークロードのスケールが向上します。
ワークロード分散の課題 ONTAP NASクラスタでは最大24ノードまで構成でき、各ノードは独自のデータボリュームとネットワークインターフェイスを持つことができます。各ボリュームは独自のワークロード、またはワークロードのサブセットをホストでき、FlexGroupボリュームでは、単純化のために単一のネームスペースにアクセスする複数のノードにまたがってワークロードを配置できます。クライアントがNFSエクスポートをマウントすると、ネットワーク トラフィックは単一のノードで確立されます。アクセス対象のデータがクラスタ内の別のノードに存在する場合、リモートトラフィックが発生し、ワークロードのレイテンシが増加し、管理が複雑になる可能性があります。	データ構造へのローカルな並列パス pNFSはメタデータからデータパスを分離し、クラスタ内のボリュームのローカル性に応じて複数の並列データパスを提供するため、クラスタ内のネットワーク トラフィックの距離を短縮し、クラスタ内の複数のハードウェアリソースを活用することでレイテンシを削減できます。また、ONTAPのpNFSはデータトラフィックを自動的にリダイレクトするため、管理者は複数のエクスポートパスと場所を管理する必要性が軽減されます。
NFSマウント ポイントの再配置 マウントポイントを確立した後、ボリュームのアンマウントと再マウントを行うと、システム停止が発生します。ONTAPは、ノード間でネットワーク インターフェイスを移行する機能を提供していますが、管理オーバーヘッドが増加し、NFSv4.xを使用したステートフルNFS接続ではシステム停止が発生します。マウント ポイントを再配置する理由の一部は、データの局所性に関する課題に関連しています。	自動パス再配置 pNFSでは、NFSサーバがネットワーク インターフェイスとボリュームの場所を示すテーブルを保持します。pNFSのメタデータ パスを介してクライアントからデータ構造が要求されると、サーバは最適化されたネットワーク パスをクライアントに提供し、クライアントはそのパスをデータ処理に使用します。これにより、ワークロードの管理オーバーヘッドが大幅に削減され、場合によってはパフォーマンスが向上する可能性があります。

構成要件

NetApp ONTAPでpNFSを設定するには、次のものが必要です：

- pNFSをサポートし、NFSv4.1以降でマウントされているNFSクライアント
- ONTAPのNFSサーバでNFSv4.1が有効になっている(`nfs modify -v4.1 enabled`（デフォルトではオフ）
- ONTAPのNFSサーバでpNFSが有効になっている(`nfs modify -v4.1-pnfs enabled`（デフォルトでは無効）
- ノードごとに少なくとも1つのネットワーク インターフェイスがあり、NFSクライアントにルーティング可能

- NFSv4を許可するエクスポート ポリシーとルールを持つSVM内のデータ ボリューム



上記の構成要件が満たされると、pNFS は単独で動作するようになります。

関連情報

- "NFSの設定"
- "ONTAPでのNFSv4.1のサポート"
- "pNFS のネットワーク インターフェイス接続"

Plan

pNFS 展開の計画

環境に pNFS を導入する前に、前提条件を満たしていること、相互運用性の要件と構成の制限を理解していることを確認してください。

前提条件

ONTAPでpNFSを有効にして使用する前に、次の要件が満たされていることを確認してください。

- NFSサーバでNFSv4.1以降が有効になっている
- NFSサーバをホストするSVM用のクラスタ内に少なくとも1つの"ノードごとにデータLIFが存在する"
- すべての"SVM内のデータLIFがルーティング可能である"NFSクライアントへ
- NFSクライアントはpNFSをサポートしています（2014年以降のほとんどの最新のLinuxディストリビューション）
- クライアントとSVM内のすべてのデータLIF間のネットワーク接続が機能している
- DNS解決（ホスト名を使用している場合）がすべてのデータLIFに対して適切に設定されている

- "FlexGroupボリューム"が設定されている（最良の結果を得るには推奨）
- "NFSv4.x IDドメインが一致"クライアントとONTAPの間
- "NFS Kerberos"（使用されている場合）がSVM内のすべてのデータLIFで有効になっている

ベストプラクティスの概要

環境に pNFS を実装する場合は、次のベスト プラクティスに従ってください：

- "FlexGroupボリューム"を使用して、最高のパフォーマンスと容量の拡張を実現
- すべての"SVM内のネットワークインターフェイスがルーティング可能である"をクライアントに確保する
- "NFSv4.0を無効にする"クライアントがNFSv4.1以降を使用するようにします
- マウント ポイントを複数のネットワーク インターフェイスとノードに分散する
- "load balancingメタデータサーバ"にラウンドロビンDNSを使用
- クライアントとサーバーで"NFSv4.x IDドメインが一致"を検証する
- メンテナンス期間中に"ネットワーク インターフェイスの移行"および"ストレージ フェイルオーバー"を実施
- Kerberos セキュリティを使用する場合は、すべてのデータ LIF で "NFS Kerberos" を有効にします。
- pNFSを使用する場合は"NFSv4.1リファーラル"を使用しないでください
- "nconnect設定"TCP接続制限の過大化を避けるため、慎重にテストしてください。
- "セッション トランキング"を"nconnect"の代替として検討してください（両方を併用しないでください）
- 展開前に"クライアントOSベンダーのサポート"pNFSを検証する

相互運用性

ONTAPのpNFSは、RFC準拠のNFSクライアントと連携するように設計されています。以下の点にご注意ください：

- 最新の"2014年以降のLinuxディストリビューション"では pNFS がサポートされています（RHEL 6.4 、Fedora 17 以降）
- クライアントOSベンダーにpNFSがサポートされていることを確認してください
- pNFSは、FlexVolと"FlexGroupボリューム"の両方で動作します。
- pNFSはNFSv4.1および"NFSv4.2"でサポートされています
- pNFSは"NFS Kerberos"（krb5、krb5i、krb5p）で使用できますが、パフォーマンスに影響が出る可能性があります
- pNFSは"nconnect"、または"セッション トランキング"（両方同時には使用できない）と併用できます。
- pNFSは"NFSv4.0"では動作しません

制限

ONTAPのpNFSには次の制限が適用されます。

- "TCP接続制限"ノードあたりの制限はプラットフォームによって異なります（具体的な制限について

は、NetApp Hardware Universeを確認してください）

- 最大ファイルサイズ：ボリュームタイプとONTAPバージョンによって異なります
- 最大ファイル数：最大2000億ファイル"FlexGroupボリューム"
- 最大容量：最大60PB（"FlexGroupボリューム"使用時）
- "ネットワークインターフェイス数"：ノードごとに少なくとも1つのデータLIFが必要です。ロード バランシングにはさらに多くのLIFが必要になる場合があります

"pNFS を使用した nconnect"を使用する場合、TCP接続数が急速に増加することに注意してください：

- nconnectを使用した各クライアントマウントは、データLIFごとに複数のTCP接続を作成します。
- 多くのクライアントが高いnconnect値を使用すると、"TCP接続制限"を超過する可能性があります
- TCP接続制限を超えると、既存の接続が解放されるまで新しい接続ができなくなります。

関連情報

- "pNFS のネットワーク インターフェイス接続"
- "NFSv4.1の有効化または無効化"
- "ONTAPでのNFSv4.1のサポート"
- "ONTAPでのNFSv4.2のサポート"
- "NetApp Hardware Universe"

pNFS のチューニングとパフォーマンスのベストプラクティス

ONTAPでpNFSを使用する場合は、最良の結果を得るために、次の考慮事項とベスト プラクティスに従ってください。

ボリューム タイプの推奨事項

ONTAPのpNFSはFlexVolボリュームとFlexGroupボリュームの両方で機能しますが、全体的に最良の結果を得るにはFlexGroupボリュームを使用します。

FlexGroupボリュームは以下を提供します：

- pNFSによるデータ トラフィックのローカライズを可能にしながら、クラスタ内の複数のハードウェア リソースにまたがる単一のマウント ポイント
- 大容量の可能性（最大60 PB）と高いファイル数（最大2,000億ファイル）
- 容量のバランシングと潜在的なパフォーマンス上のメリットのためのマルチパート ファイルのサポート
- 単一のワークロードをサポートするボリュームとハードウェアへの並列アクセス

"FlexGroupボリューム管理について学ぶ"

クライアントの推奨事項

すべてのNFSクライアントがpNFSをサポートしているわけではありませんが、最近のクライアントのほとんどはサポートしています。RHEL 6.4とFedora 17は、最初にpNFSをサポートしたクライアントです（2014年頃）。そのため、ここ数年でリリースされたクライアントバージョンは、この機能を完全にサポートしている

と想定しても問題ありません。ONTAPのNFSサポートに関するスタンスは、「クライアントが機能をサポートし、RFCに準拠しており、かつONTAPもその機能をサポートしている場合、その組み合わせはサポートされます」というものです。ただし、クライアントOSベンダーがpNFSをサポートしていることを確認することがベストプラクティスです。

ボリューム移動

ONTAPは、同一クラスタ内のノードまたはアグリゲート間でボリュームを無停止で移動できる機能を提供し、容量とパフォーマンスのバランスを柔軟に調整します。ONTAPでボリュームの移動が発生すると、pNFS デバイス マッピングが自動的に更新され、必要に応じてクライアントに新しいボリュームとインターフェイスの関係を使用するように通知されます。

"ボリュームの移動について"

ネットワーク インターフェイスの移行

ONTAPは、パフォーマンスのバランスとメンテナンスの柔軟性を実現するために、同一クラスタ内のノード間でネットワーク インターフェイスを移動する機能を提供します。ボリュームの移動と同様に、ONTAPでネットワーク インターフェイスの移行が行われると、pNFSデバイスのマッピングが自動的に更新され、必要に応じて新しいボリュームとインターフェイスの関係を使用するようにクライアントに通知されます。

ただし、NFSv4.1はステートフル プロトコルであるため、ネットワーク インターフェイスの移行は、NFSマウントをアクティブに使用しているクライアントに混乱をもたらす可能性があります。ネットワーク インターフェイスの移行はメンテナンス ウィンドウ内に実施し、ネットワークの混乱が発生する可能性があることをクライアントに通知することがベスト プラクティスです。

ストレージのフェイルオーバー/ギブバック

pNFSは、NFSv4.1と同じストレージフェイルオーバーの考慮事項に従います。これらの詳細については"NetAppテクニカル レポート4067：『NFS Best Practice and Implementation Guide』"を参照してください。一般に、pNFSに関連するストレージフェイルオーバー/ギブバックは、プロトコルのステートフル性によりストレージの中断が発生する可能性があることを考慮して、メンテナンスウィンドウ内で実行する必要があります。

メタデータのワークロード

メタデータ操作はサイズが小さいですが、ワークロード（大量のファイルを作成しているか、「find」コマンドを実行しているかなど）やファイル総数に応じて、操作回数が大きくなることがあります。そのため、メタデータ呼び出しが多いワークロードは、NFSサーバのCPUに負荷をかけ、単一の接続でボトルネックとなる可能性があります。pNFS（およびNFSv4.x全般）は、ステートフル性、ロックメカニズム、およびプロトコル バージョンのセキュリティ機能がCPU使用率とレイテンシに悪影響を与える可能性があるため、パフォーマンスに依存する高メタデータワークロードには適していません。これらのワークロードタイプ（GETATTRやSETATTRの呼び出しが多いなど）は、一般的にNFSv3の方が適しています。

メタデータサーバ

pNFSのメタデータ サーバは、NFSエクスポートの最初のマウント時に確立されます。マウント ポイントが確立されると、再マウントされるかデータ インターフェイスが移動されるまで、そのマウント ポイントはそのまま維持されます。そのため、同じボリュームにアクセスする複数のクライアントが、SVM全体の異なるノードとデータ インターフェイスにマウントされるようにすることがベスト プラクティスです。このアプローチにより、ノードとCPUリソース間でメタデータ サーバのロード バランシングが実現され、クラスタ内のネットワーク インターフェイスを最大限に活用できます。これを実現する方法の1つとして、ラウンドロビンDNS設定を確立することが挙げられます。これについては "NetAppテクニカルレポート4523：ONTAPにお

けるDNSロードバランシング"で説明します。

NFSv4.x IDドメイン

NFSv4.xは様々な方法でセキュリティ機能を提供します（詳細は "NetAppテクニカル レポート4067：『NFS Best Practice and Implementation Guide』"で説明されています）。NFSv4.x IDドメインはその一つで、NFSエクスポートでユーザーとグループを認証する際に、クライアントとサーバーはIDドメインについて合意する必要があります。IDドメインの不一致による副作用の一つとして、不要なアクセスを防ぐために、ユーザーまたはグループが匿名ユーザー（実質的には圧縮されたユーザー）として表示されることがあります。NFSv4.x（およびpNFS）では、クライアントとサーバーのNFSv4.x IDドメインが一致していることを確認することがベストプラクティスです。

nconnect

前述の通り、ONTAPのnconnectは一部のワークロードのパフォーマンス向上に役立ちます。pNFSでは、nconnectはストレージ システムへのTCP接続数を大幅に増加させることでパフォーマンスを向上させる一方で、多くのクライアントがマウント オプションを利用する場合、ストレージへのTCP接続が過負荷になり、問題が発生する可能性があることを理解しておくことが重要です。NetApp Hardware Universeでは、ノードあたりのTCP接続制限について説明しています。

ノードのTCP接続制限を超えると、既存の接続が解放されるまで新しいTCP接続は許可されません。これにより、マウント ストームが発生する可能性のある環境では、問題が発生する可能性があります。

次の表は、nconnect を使用した pNFS が TCP 接続制限を超える可能性があることを示しています：

クライアント数	**nconnect**値	マウントあたり、ノードあたりの潜在的な**TCP**接続の合計数
1	4	4
100	4	400
1000	8	8000
10000	8	80000
10000	16	160000 [1]

[1] ほとんどのONTAPシングルノードTCP接続制限を超えています

NFSv4.1セッション トランキング

ONTAPのセッション トランキングは、NFSv4.xマウントのスループットとパスの復元力を向上させるために使用できます。pNFSと併用すると、クラスタ内の各ノードでセッション トランクを確立できます。ただし、セッション トランクはノードごとに少なくとも2つのインターフェイスを必要とし、pNFSは意図したとおりに動作するためにノードごとに少なくとも1つのインターフェイスを必要とします。さらに、SVM内のすべてのインターフェイスがNFSクライアントにルーティング可能である必要があります。セッション トランキングとpNFSは、nconnectも併用すると正常に動作しません。nconnectとセッション トランキングは相互に排他的な機能であることを考慮してください。

"NFSトランキングについて"

ネットワーク インターフェイスの接続

pNFSが正常に機能するには、クラスタ内の各ノードにルーティング可能なネットワークインターフェースが必要です。pNFSをホストするNFSサーバと同じSVM内に、NFSクライアントにルーティングできないネット

ワークインターフェースが存在する場合でも、ONTAPはデバイスマッピングでそれらのインターフェースをクライアントにアドバタイズします。NFSクライアントが異なるサブネットのインターフェース経由でデータにアクセスしようとすると、接続できず、システム停止が発生します。pNFSを使用する場合は、SVM内でクライアントがアクセスできるネットワークインターフェースのみを許可するのがベストプラクティスです。

> デフォルトでは、pNFSデバイス リストにSVM内のすべてのデータLIFが入力されるため、pNFSではSVM内のすべてのデータLIFがNFSクライアントのインターフェイスにルーティング可能である必要があります。その結果、ルーティング不可能なデータLIFが選択され、停止シナリオが発生する可能性があります。ベスト プラクティスとして、pNFSを使用する場合は、ルーティング可能なデータLIFのみを設定してください。
>
> ONTAP 9.18.1 RC1以降では、サブネットごとにpNFSトラフィックの対象となるインターフェースを指定できるようになりました。これにより、ルーティング可能なインターフェースとルーティング不可能なインターフェースを混在させることができます。コマンドの詳細については、NetAppサポートにお問い合わせください。

NFSv4.0

NFSv4.0は、ONTAP NFSサーバでNFSv4.1と併用できるオプションです。ただし、pNFSはNFSv4.0上では動作しません。NFSサーバでNFSv4.0が有効になっている場合、クライアントが意図せずそのプロトコル バージョンをマウントし、pNFSを利用できなくなる可能性があります。そのため、pNFSを使用する場合は、NFSv4.0を明示的に無効にすることがベストプラクティスです。NFSv4.1は引き続き有効にする必要があり、NFSv4.0とは独立して動作します。

NFSv4.1リファーラル

NFSv4.1 参照は、クライアントからボリュームを所有するノード上のネットワーク インターフェイスへのマウント パスをローカライズします。pNFS はデータ パスをローカライズし、マウント パスはメタデータ サーバになります。

これら2つの機能は併用可能ですが、NFSv4.1のリファーラルをpNFSで使用すると、複数のメタデータ サーバを同一ノード上にスタックし、複数のクラスタ ノードにメタデータ サーバを分散させる能力が低下するという望ましくない影響が生じる可能性があります。pNFSを使用する場合、メタデータ サーバがクラスタ全体に均等に分散されていないと、単一ノードのCPUがメタデータ要求で過負荷になり、パフォーマンスのボトルネックが発生する可能性があります。

そのため、pNFSを使用する場合はNFSv4.1リファラルの使用を避けることがベストプラクティスです。代わりに、マウント ポイントをクラスタ内の複数のネットワーク インターフェイスとノードに分散させてください。

"NFSv4リファーラルの有効化または無効化について学習します"

NFS Kerberos

NFS Kerberos では、krb5 による認証の暗号化に加え、krb5i および krb5p によるデータパケットの暗号化が可能です。これは SVM 内のネットワークインターフェースごとに有効化され、詳細は "NetApp テクニカルレポート 4616：ONTAP における NFS Kerberos と Microsoft Active Directory"で説明されています。

pNFSはSVM内のノードおよびネットワーク インターフェイス間でデータ トラフィックをリダイレクトできるため、SVM内の各ネットワーク インターフェイスでNFS Kerberosが有効になっていて機能している必要があります。SVM内のいずれかのネットワーク インターフェイスでKerberosが有効になっていない場合、pNFSはそれらのインターフェイス上のデータ ボリュームにアクセスしようとしても正常に機能しません。

例えば、2つのネットワーク インターフェイス（Kerberosが有効になっているのは1つだけ）を備えたpNFS対応SVMで並列ddを用いた読み取りテストを実行したところ、Kerberos対応インターフェース上のファイルは正常に動作しましたが、Kerberosが有効になっていないインターフェースを持つノード上のファイルは読み取りを完了できませんでした。両方のインターフェースでKerberosを有効にすると、すべてのファイルが期待どおりに動作しました。

SVM内のすべてのネットワーク インターフェイスでNFS Kerberosが有効になっている場合、pNFSでNFS Kerberosを使用できます。NFS Kerberosはパケットの暗号化/復号化によってパフォーマンスが低下する可能性があることにご注意ください。そのため、パフォーマンスの低下がワークロードに過度の影響を及ぼさないことを確認するために、ワークロードでpNFSとNFS Kerberosを徹底的にテストすることをお勧めします。

以下は、RHEL 9.5 クライアント上の pNFS で krb5（認証）と krb5p（エンドツーエンド暗号化）を使用した場合の並列読み取りパフォーマンスの例です。このテストでは、krb5p のパフォーマンスが 70% 低下しました。

Kerberosフレーバー	**MB/s**	完了時間
krb5	• File1-243 • File2-243 • File3-238 • File4-238	• File1-43 • File2-43.1 • File3-44 • File4-44.1
krb5p	• File1-72.9 • File2-72.8 • File3-71.4 • File4-71.2	• File1-143.9 • File2-144.1 • File3-146.9 • File4-147.3

"強力なセキュリティを実現するNFSでのKerberosについて学ぶ"

NFSv4.2

NFSv4.2はONTAP 9.8に追加され、利用可能な最新のNFSv4.xバージョンです（RFC-7862）。NFSv4.2には、有効化/無効化を明示的に指定するオプションはありません。代わりに、NFSv4.1と同時に有効化/無効化されます(`-4.1 enabled`。クライアントがNFSv4.2をサポートしている場合、 `minorversion=2`マウント オプションで別途指定しない限り、マウント コマンドの実行中にサポートされているNFSの最新バージョンがネゴシエートされます。

ONTAPのNFSv4.2は次の機能をサポートしています：

- セキュリティ ラベル（MACラベル）
- 拡張属性
- スパース ファイル操作（FALLOCATE）

pNFS は NFSv4.1 で導入されましたが、NFSv4.2 でも、付随する機能とともにサポートされています。

"ONTAP の NFSv4.2 サポートについて学ぶ"

pNFSコマンド、統計、イベント ログ

これらのONTAP CLIコマンドはpNFSに特化しており、設定、トラブルシューティング、統計情報の収集に使用できます。

NFSv4.1の有効化

```
nfs modify -vserver SVM -v4.1 enabled
```

pNFSを有効にする

```
nfs modify -vserver SVM -v4.1-pnfs enabled
```

pNFSデバイスを表示する（**advanced**権限）

```
pnfs devices show -vserver SVM
```

```
Vserver Name     Mapping ID      Volume MSID     Mapping Status
Generation
---------------  --------------- --------------- ---------------
-------------
SVM              17              2157024470      notavailable    2
SVM              18              2157024463      notavailable    2
SVM              19              2157024469      available       3
SVM              20              2157024465      available       4
SVM              21              2157024467      available       3
SVM              22              2157024462      available       1
```

pNFSデバイス マッピングを表示する（**advanced**権限）

```
pnfs devices mappings show -vserver SVM
```

```
Vserver Name    Mapping ID      Dsid            LIF IP
--------------  --------------- --------------- --------------------
SVM             19              2449            10.x.x.x
SVM             20              2512            10.x.x.y
SVM             21              2447            10.x.x.x
SVM             22              2442            10.x.x.y
```

pNFS固有のパフォーマンス カウンターをキャプチャする（高度な権限）

```
statistics start -object nfsv4_1 -vserver SVM -sample-id [optional-name]
```

pNFS固有のパフォーマンス カウンタを表示する（**advanced**権限）

```
statistics show -object nfsv4_1 -vserver SVM
```

pNFS固有のカウンターのリストを表示する（高度な権限）

```
statistics catalog counter show -object nfsv4_1 -counter *layout*|*device*
```

```
Object: nfsv4_1

Counter                      Description
---------------------------- ----------------------------------------------
getdeviceinfo_avg_latency    Average latency of NFSv4.1 GETDEVICEINFO
operations.
getdeviceinfo_error          The number of failed NFSv4.1 GETDEVICEINFO
operations.
getdeviceinfo_percent        Percentage of NFSv4.1 GETDEVICEINFO
operations.
getdeviceinfo_success        The number of successful NFSv4.1 GETDEVICEINFO
operations.
getdeviceinfo_total          Total number of NFSv4.1 GETDEVICEINFO
operations.
getdevicelist_avg_latency    Average latency of NFSv4.1 GETDEVICELIST
operations.
getdevicelist_error          The number of failed NFSv4.1 GETDEVICELIST
operations.
getdevicelist_percent        Percentage of NFSv4.1 GETDEVICELIST
operations.
getdevicelist_success        The number of successful NFSv4.1 GETDEVICELIST
operations.
getdevicelist_total          Total number of NFSv4.1 GETDEVICELIST
operations.
layoutcommit_avg_latency     Average latency of NFSv4.1 LAYOUTCOMMIT
operations.
layoutcommit_error           The number of failed NFSv4.1 LAYOUTCOMMIT
operations.
layoutcommit_percent         Percentage of NFSv4.1 LAYOUTCOMMIT operations.
layoutcommit_success         The number of successful NFSv4.1 LAYOUTCOMMIT
```

```
operations.
layoutcommit_total          Total number of NFSv4.1 LAYOUTCOMMIT
operations.
layoutget_avg_latency       Average latency of NFSv4.1 LAYOUTGET
operations.
layoutget_error             The number of failed NFSv4.1 LAYOUTGET
operations.
layoutget_percent           Percentage of NFSv4.1 LAYOUTGET operations.
layoutget_success           The number of successful NFSv4.1 LAYOUTGET
operations.
layoutget_total             Total number of NFSv4.1 LAYOUTGET operations.
layoutreturn_avg_latency    Average latency of NFSv4.1 LAYOUTRETURN
operations.
layoutreturn_error          The number of failed NFSv4.1 LAYOUTRETURN
operations.
layoutreturn_percent        Percentage of NFSv4.1 LAYOUTRETURN operations.
layoutreturn_success        The number of successful NFSv4.1 LAYOUTRETURN
operations.
layoutreturn_total          Total number of NFSv4.1 LAYOUTRETURN
operations.
```

NFSのアクティブなネットワーク接続を表示する

`network connections active show`コマンドを使用して、SVMへの複数の TCP接続が確立されているかどうかを確認できます。

たとえば、NFSセッション トランクを表示する場合は、ノードごとに異なるインターフェイスを介して同じクライアントからの接続を探します：

```
cluster::*> network connections active show -node cluster-0* -vserver PNFS
              Vserver   Interface         Remote
      CID Ctx Name      Name:Local Port   Host:Port
Protocol/Service
--------- --- --------- ----------------- --------------------
----------------
Node: node-01
2304333128 14 PNFS      data1:2049        ubuntu22-224:740     TCP/nfs
2304333144 10 PNFS      data3:2049        ubuntu22-224:864     TCP/nfs
2304333151  5 PNFS      data1:2049        ubuntu22-226:848     TCP/nfs
2304333167 15 PNFS      data3:2049        ubuntu22-226:684     TCP/nfs
Node: node-02
2497668321 12 PNFS      data2:2049        ubuntu22-224:963     TCP/nfs
2497668337 18 PNFS      data4:2049        ubuntu22-224:859     TCP/nfs
2497668344 14 PNFS      data2:2049        ubuntu22-226:675     TCP/nfs
2497668360  7 PNFS      data4:2049        ubuntu22-226:903     TCP/nfs
```

接続されたクライアントの**NFS**バージョン情報を表示する

`nfs connected-clients show`コマンドで
NFS接続を表示することもできます。表示されるクライアント リストは、過去
48時間以内にアクティブなNFSトラフィックがあったクライアントであることに注意してください
。アイドル状態のNFSクライアント（マウントされている場合でも）は、マウントにアクセスするまで表示されない場合があります。 `-idle-
time`機能を指定することで、これらのクライアントをフィルタリングし、最近アクセスしたクライアントのみを表示できます。

たとえば、pNFS SVM の過去 10 分間にアクティビティがあったクライアントを表示するには：

```
cluster::*> nfs connected-clients show -vserver PNFS -idle-time <10m>

 Node: node-01

Vserver: PNFS Data-Ip: 10.x.x.x Local Remote Client-Ip Protocol Volume
Policy Idle-Time Reqs Reqs Trunking

10.x.x.a nfs4.2 PNFS_root default 9m 10s 0 149 false 10.x.x.a nfs4.2
FG_0001 default 9m 10s 135847 0 false 10.x.x.b nfs4.2 PNFS_root default 8m
12s 0 157 false 10.x.x.b nfs4.2 FG_0001 default 8m 12s 52111 0 false
```

関連情報

- "ONTAPでの並列NFS（pNFS）について"

著作権に関する情報

商標に関する情報