FlexCacheによるホットスポット修復

ONTAP 9

NetApp
March 13, 2025

目次

FlexCacheによるホットスポット修復

ONTAPによるハイパフォーマンスコンピューティングワークロードのホットスポットの修正

アニメーションレンダリングやEDAなど、多くのハイパフォーマンスコンピューティングワークロードで共通する問題はホットスポットです。ホットスポットは、クラスタまたはネットワークの特定の部分の負荷が他の領域と比較して著しく高い場合に発生する状況です。その結果、パフォーマンスのボトルネックが発生し、その場所に集中した過剰なデータトラフィックが原因で全体的な効率が低下します。たとえば、1つまたは複数のファイルが実行中のジョブに対して高い需要があるため、（ボリュームアフィニティ経由で）そのファイルへの要求を処理するために使用されるCPUでボトルネックが発生します。FlexCacheはこのボトルネックの解消に役立ちますが、適切に設定する必要があります。

このドキュメントでは、ホットスポットを修正するためのFlexCacheの設定方法について説明します。

> 2024年7月以降、これまでPDFとして公開されていたテクニカルレポートの内容がONTAPの製品ドキュメントに統合されました。このONTAPホットスポット修正テクニカルレポートの内容は、発行日時点でまったく新しいものであり、それ以前の形式は作成されていません。

主な概念

ホットスポットの修復を計画する際には、これらの重要な概念を理解することが重要です。

- 高密度**FlexCache**（**HDF**）：キャッシュ容量の要件で許容される数のノードにまたがるように凝縮されたFlexCache
- * HDFアレイ（HDFA）*：同じオリジンのキャッシュで構成され、クラスタ全体に分散されるHDFSのグループ
- * SVM間HDFA *：サーバ仮想マシン（SVM）ごとにHDFAから1つのHDF
- * SVM内HDFA *：1つのSVM内のHDFA内のすべてのHDFS
- * East-Westトラフィック*：間接データアクセスによって生成されるクラスタバックエンドトラフィック

次のステップ

- "高密度FlexCacheを使用してホットスポットの修復を支援する方法を理解する"
- "FlexCacheアレイ密度の決定"
- "HDFSの密度を確認し、SVM間HDFAとSVM内HDFAを使用してNFSを使用してHDFSにアクセスするかどうかを決定します。"
- "ONTAP構成でクラスタ内キャッシングを使用するメリットを実現するために、HDFAとデータLIFを設定する"
- "クライアント設定でONTAP NAS接続を分散するようにクライアントを設定する方法について説明します。"

ONTAP FlexCacheホットスポット修復ソリューションの設計

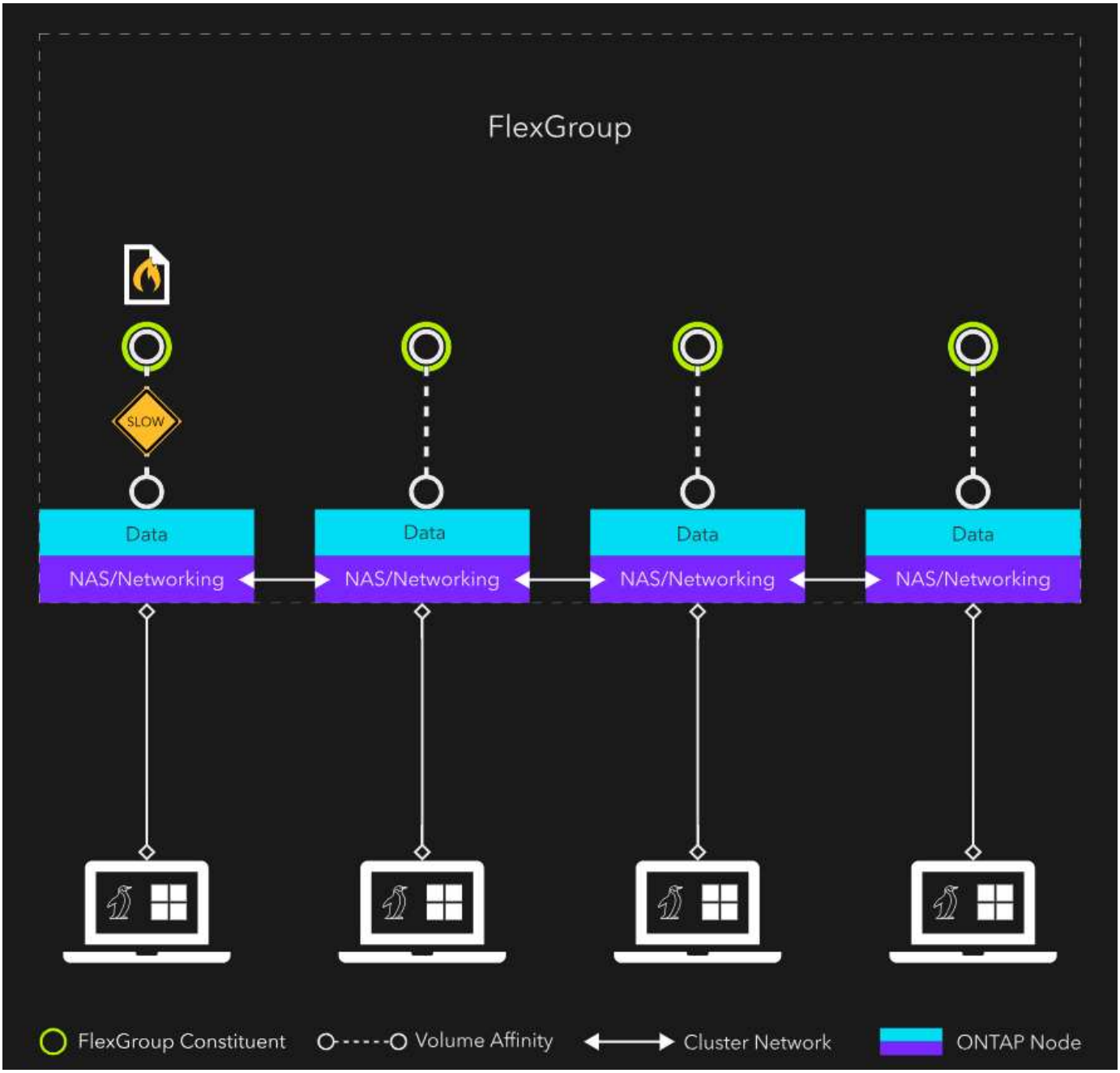
ホットスポットの問題を修正するには、ボトルネックの根本的な原因、自動プロビジョニングされたFlexCacheでは不十分な理由、FlexCacheソリューションを効果的に設計するために必要な技術的な詳細を確認します。高密度FlexCacheアレイ（HDFA）を理解して実装することで、パフォーマンスを最適化し、負荷の高いワークロードのボトルネックを解消できます。

ボトルネックの把握

次に、イメージ（ Image ）一般的なシングルファイルホットスポットのシナリオを示します。このボリュームは、各ノードにコンスティチュエントが1つだけ含まれるFlexGroupであり、ファイルはノード1に格納されます。

すべてのNASクライアントのネットワーク接続をクラスタ内の異なるノードに分散しても、ホットファイルが存在するボリュームアフィニティを提供するCPUのボトルネックは解消されません。また、ファイルが存在する以外のノードに接続されているクライアントからのコールに、クラスタネットワークトラフィック（イースト/ウェストトラフィック）を導入します。イースト/ウェストトラフィックのオーバーヘッドは通常は小さいですが、ハイパフォーマンスコンピューティングワークロードの場合は少しずつ考慮されます。

図1：FlexGroupの単一ファイルホットスポットのシナリオ

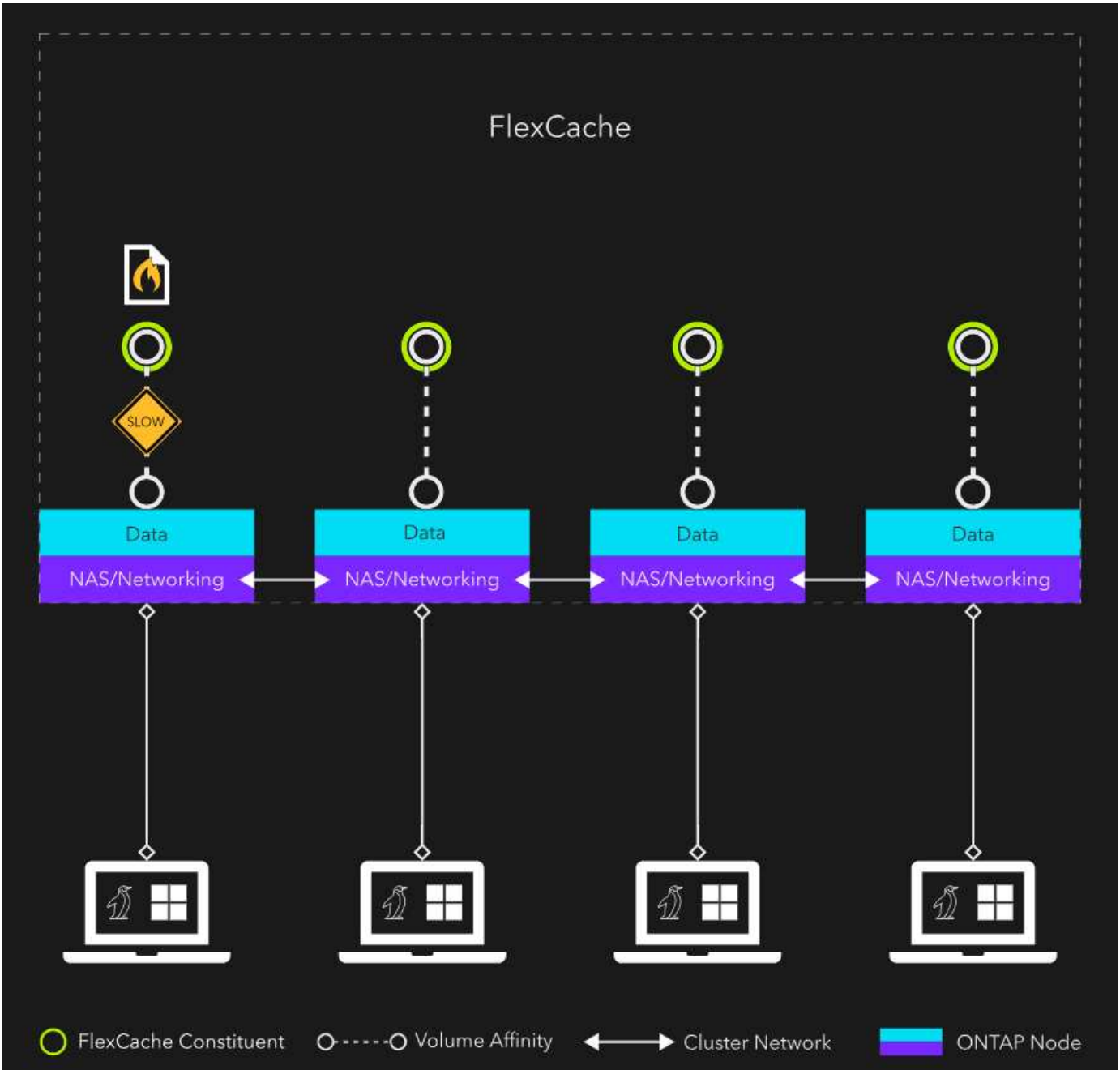


自動プロビジョニングされた**FlexCache**が答えにならない理由

ホットスポットの問題を解決するには、CPUのボトルネックを解消し、できれば東西のトラフィックも解消します。適切に設定されていれば、FlexCacheが役立ちます。

次の例では、FlexCacheがSystem Manager、BlueXP 、またはデフォルトのCLI引数を使用して自動プロビジョニングされます。図1図2最初は同じように見えます。どちらも4ノードの単一コンスティチュエントのNASコンテナです。唯一の違いは、図1のNASコンテナがFlexGroupで、図2のNASコンテナがFlexCacheであることです。各図は同じボトルネックを示しています。つまり、ホットファイルへのアクセスを提供するボリュームアフィニティ用のノード1のCPUと、レイテンシの原因となるイースト/ウェストトラフィックです。自動プロビジョニングされたFlexCacheでは、ボトルネックは解消されません。

図**2**：自動プロビジョニング**FlexCache**のシナリオ



FlexCacheの構造

ホットスポットの修復のためにFlexCacheを効果的に設計するには、FlexCacheに関する技術的な詳細を理解する必要があります。

FlexCacheは常にスパースFlexGroupです。FlexGroupは複数のFlexVolで構成されます。このFlexVolのことをFlexGroupコンスティチュエントと呼びます。デフォルトのFlexGroupレイアウトでは、クラスタ内のノードごとに1つ以上のコンスティチュエントがあります。コンスティチュエントは抽象化レイヤの下で「縫い合わせ」され、単一の大規模なNASコンテナとしてクライアントに提供されます。ファイルがFlexGroupに書き込まれると、取り込みのヒューリスティックによって、ファイルを格納するコンスティチュエントが決定されます。クライアントのNAS接続を含むコンスティチュエントでも別のノードでもかまいません。すべてが抽象化レイヤの下で動作し、クライアントからは見えないため、この場所は無関係です。

このFlexGroupの理解をFlexCacheに適用してみましょう。FlexCacheはFlexGroup上に構築されているため、

に示すように、デフォルトでは、クラスタ内のすべてのノードにコンスティチュエントを含むFlexCacheが1つだけ存在し図1ます。ほとんどの場合、これは素晴らしいことです。クラスタ内のすべてのリソースを利用している。

ただし、ホットファイルの修正には、単一ファイルのCPUとイースト/ウェストトラフィックの2つのボトルネックがあるため、これは理想的ではありません。ホットファイルの各ノードのコンスティチュエントで構成されるFlexCacheを作成した場合、そのファイルはそのうちの1つのコンスティチュエントにのみ配置されます。これは、ホットファイルへのすべてのアクセスを処理するCPUが1つあることを意味します。また、ホットファイルに到達するために必要なイースト/ウェストトラフィックの量も制限する必要があります。

このソリューションは、多数の高密度FlexCachesで構成されています。

高密度FlexCacheの構造

高密度FlexCache（HDF）では、キャッシュされたデータの容量要件が許容される数のノードにコンスティチュエントが配置されます。目的は、キャッシュを単一のノードに配置することです。容量の要件によってそれが不可能になった場合は、コンスティチュエントを少数のノードにしか配置できません。

たとえば、24ノードクラスタでは、次の3つの高密度FlexCachesを使用できます。

- ノード1~8にまたがるノード
- ノード9~16にまたがる秒
- 3分の1はノード17~24にまたがっています。

これら3つのHDFSは、1つの高密度FlexCacheアレイ（HDFA）を構成します。ファイルが各HDF内に均等に分散されている場合、クライアントから要求されたファイルがフロントエンドNAS接続に対してローカルに存在する可能性が8分の1になります。それぞれ2つのノードにまたがる12個のHDFSを使用する場合、ファイルがローカルである可能性は50%です。HDFを1つのノードに縮小して24個のノードを作成できれば、ファイルがローカルであることが保証されます。

この構成により、すべてのEast-Westトラフィックが排除され、最も重要なことは、ホットファイルにアクセスするために24個のCPU/ボリュームアフィニティが提供されることです。

次の手順

"FlexCacheアレイ密度の決定"

関連情報

"FlexGroupおよびTRに関するドキュメント"

ONTAP FlexCache密度の決定

最初のホットスポット修復設計の決定は、FlexCache密度を把握することです。次の例は4ノードクラスタです。ファイル数が各HDFのすべてのコンスティチュエントに均等に分散されているとします。また、すべてのノードにフロントエンドNAS接続が均等に分散されていると仮定します。

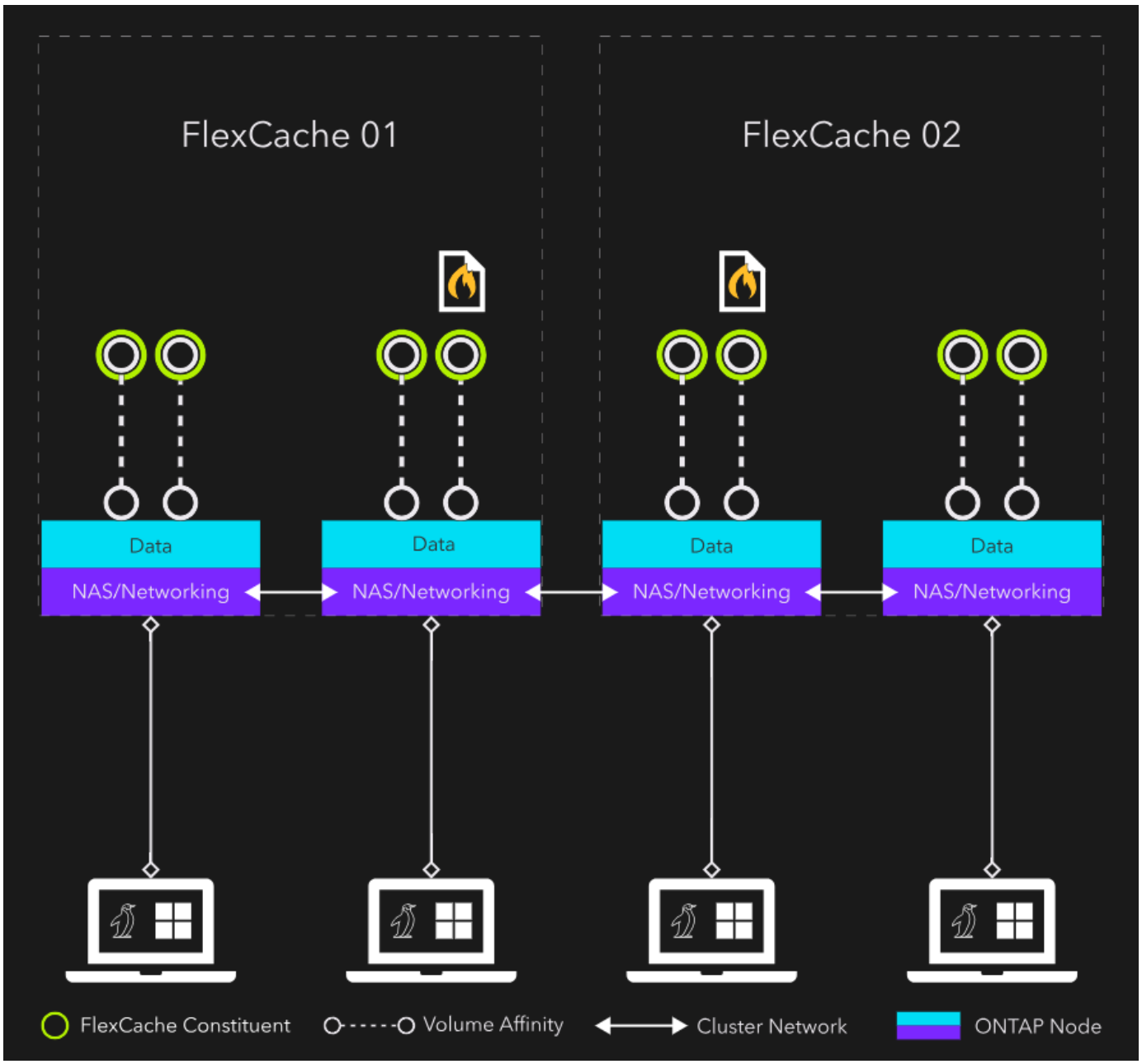
使用できる構成はこれらの例だけではありませんが、スペース要件と使用可能なリソースで許容される数のHDFSを作成するための設計原則を理解しておく必要があります。

HDFAは、次の構文で表されます。 `HDFs per HDFA x nodes per HDF x constituents per node per HDF`

2x2x2 HDFAノセツテイ

図1は、2x2x2のHDFA構成の例です。2つのHDFSがそれぞれ2つのノードにまたがり、各ノードに2つのコンスティチュエントボリュームが含まれています。この例では、各クライアントがホットファイルに直接アクセスできる可能性は50%です。4つのクライアントのうち2つがイーストウェストトラフィックを持っています。重要なことは、HDFSが2つになったことです。つまり、ホットファイルには2つの個別のキャッシュがあります。これで、ホットファイルへのアクセスを提供する2つのCPU /ボリュームアフィニティが確立されました。

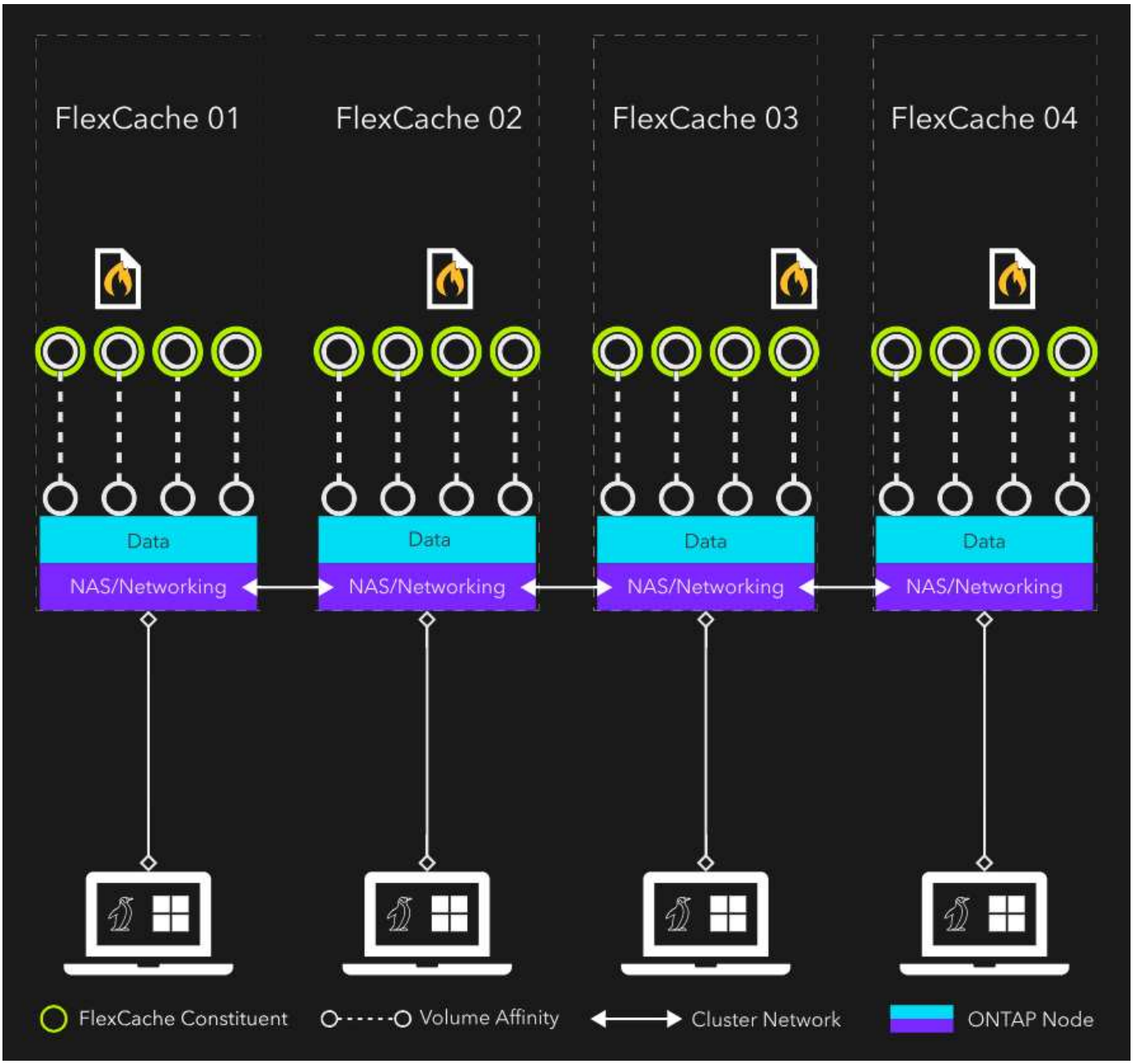
図1：2x2x2 HDFA構成



4x1x4 HDFAノセツテイ

図2は、最適な構成を表します。これは、4x1x4のHDFA構成の例です。4つのHDFSがそれぞれ1つのノードに含まれ、各ノードに4つのコンスティチュエントが含まれます。この例では、各クライアントがホットファイルのキャッシュに直接アクセスできることが保証されています。4つの異なるノードに4つのキャッシュファイルがあるため、4つの異なるCPU /ボリュームアフィニティがホットファイルへのサービスアクセスに役立ちます。また、イースト/ウェストトラフィックは生成されません。

図2：4x1x4 HDFA構成



次のステップ

HDFSの密度を決定したら、NFSを使用してHDFSにアクセスする場合は、設計上の別の決定を行う必要があります"SVM間HDFAとSVM内HDFA"。

ONTAPのSVM間またはSVM内のHDFAオプションを決定する

HDFSの密度を確認したら、NFSを使用してHDFSにアクセスするかどうかを決定し、SVM間HDFAとSVM内HDFAのオプションについて学習します。

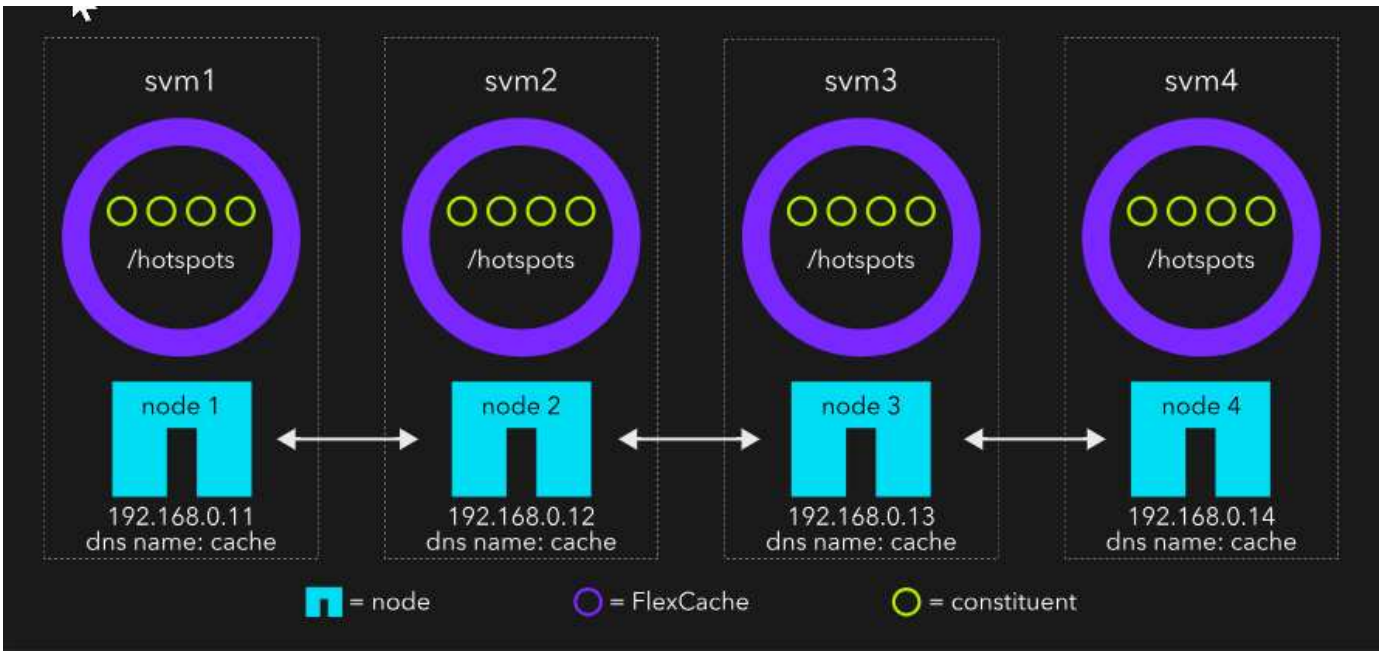
HDFSにアクセスするのがSMBクライアントだけの場合は、すべてのHDFSを1つのSVMに作成する必要があります。DFSターゲットを使用してロードバランシングを行う方法については、Windowsクライアントの設定を参照してください。

SVM間HDFAの導入

SVM間HDFAを使用するには、HDFA内のHDFごとにSVMを作成する必要があります。これにより、HDFA内のすべてのHDFSのジャンクションパスが同じになるため、クライアント側での設定が容易になります。

この例で図1は、各HDFが専用のSVMに配置されています。これはSVM間HDFA環境です。各HDFには/hotspotsのジャンクションパスがあります。また、すべてのIPにはホスト名キャッシュのDNS Aレコードがあります。この構成では、DNSラウンドロビンを使用して、異なるHDFS間でマウントの負荷を分散します。

図1：4x1x4のSVM間HDFA構成

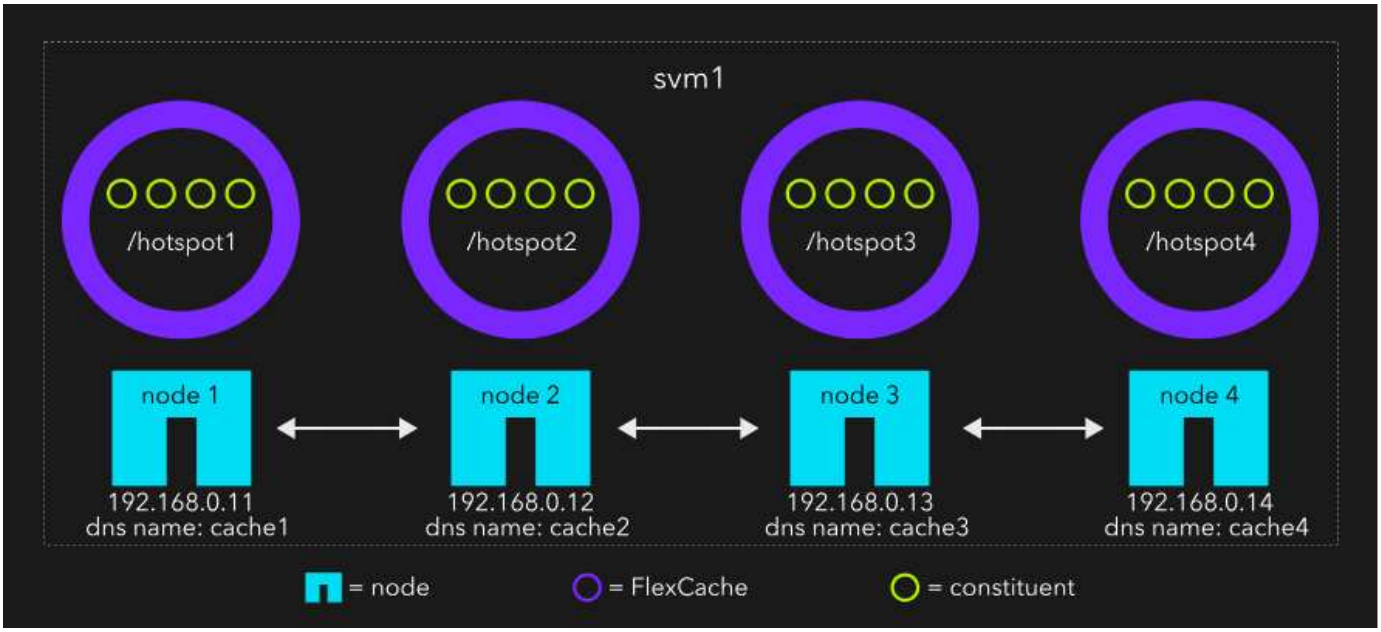


SVM内HDFAの導入

SVM内では、各HDFに一意のジャンクションパスが必要ですが、すべてのHDFSは1つのSVMに含まれます。このセットアップはONTAPでは1つのSVMしか必要ないため簡単ですが、ONTAPでおよびデータLIFを使用してLinux側でさらに高度な設定を行う必要があります `autofs`。

この例で図2は、すべてのHDFが同じSVM内にあります。これはSVM内HDFA環境であり、ジャンクションパスが一意である必要があります。ロードバランシングを適切に機能させるには、IPごとに一意のDNS名を作成し、ホスト名が解決されるデータLIFをHDFが存在するノードにのみ配置する必要があります。また、で説明されているように、複数のエントリを使用してを構成する必要があり `autofs`ます"Linuxクライアントノセッテイ"。

図2：4x1x4のSVM内HDFA構成



次のステップ

これで、HDFAをどのように展開するかがわかりました"HDFAを導入し、分散した方法でアクセスするようにクライアントを設定する"。

ONTAPでのHDFAとデータLIFの設定

このホットスポット修復ソリューションのメリットを実現するには、HDFAとデータLIFを適切に設定する必要があります。このソリューションでは、同じクラスタ内でオリジンとHDFAを使用したクラスタ内キャッシングが使用されます。

次に、2つのHDFAサンプル構成を示します。

- SVM間HDFA×2×2
- SVM内HDFA×4

タスクの内容

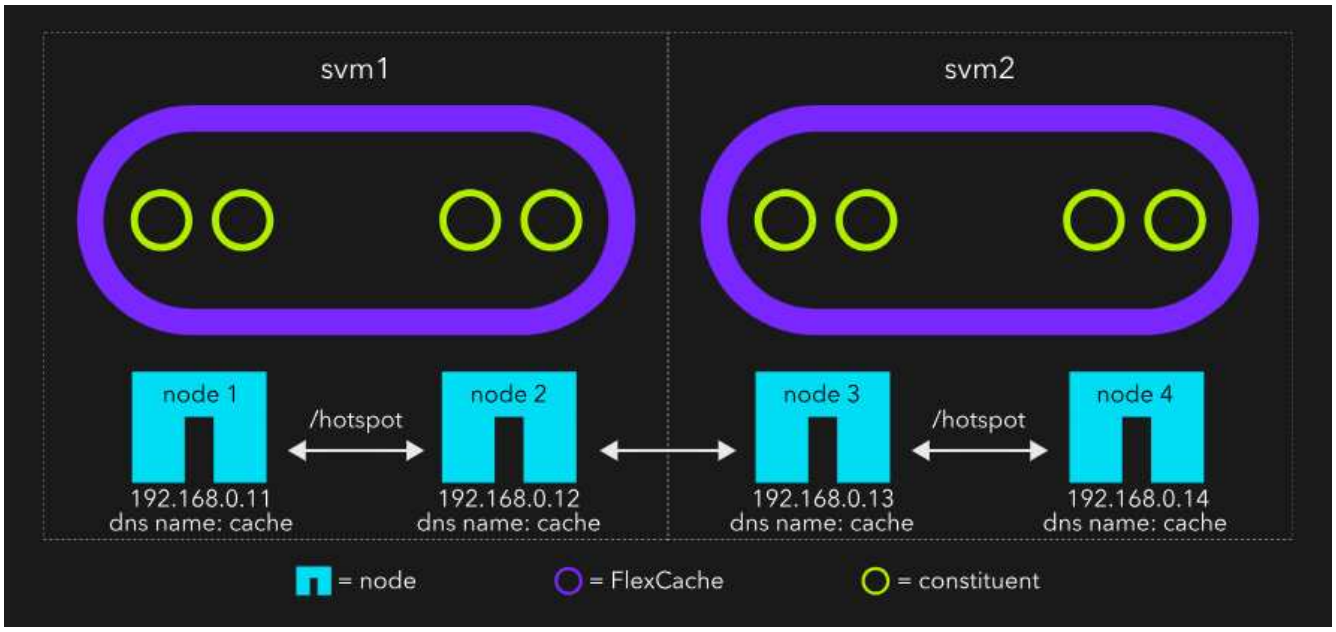
この高度な設定は、ONTAP CLIを使用して実行します。コマンドでは2つの設定を使用する必要があり`flexcache create`ます。また、が設定されていないことを確認する必要があります。

- `-aggr-list`：HDFを制限するノードまたはノードのサブセットにあるアグリゲート（アグリゲートのリスト）を指定します。
- `-aggr-list-multiplier`：オプションに表示されたアグリゲートごとに作成するコンスティチュエントの数を確認します `aggr-list`。アグリゲートが2つ表示されている場合にこの値をに設定すると、`2`コンスティチュエントが4つになります。NetAppでは、アグリゲートあたり最大8個のコンスティチュエントを推奨していますが、16個で十分です。
- `-auto-provision-as`：タブアウトすると、CLIは自動入力を試み、値をに設定します `flexgroup`。これが設定されていないことを確認してください。表示された場合は、削除します。

2x2x2のSVM間HDFA構成を作成

1. 図1に示すように、2x2x2のSVM間HDFAの構成を支援するために、準備シートを作成します。

   図**1**：**2x2x2**の**SVM**間**HDFA**レイアウト



SVM	HDFあたりのノード数	アグリゲート	ノードあたりのコンスティチュエン	ジャンクションパス	データLIFのIP
svm1	node1、node2	aggr1、aggr2	2	/ホットスポット	192.168.0.11、192.168.0.12
svm2	node3、node4	aggr3、aggr4	2	/ホットスポット	192.168.0.13、192.168.0.14

2. HDFSを作成します。次のコマンドを、準備シートの各行に対して1回ずつ、2回実行します。2回目のイテレーションの値と `aggr-list`値を調整して `vserver`ください。

```
cache::> flexcache create -vserver svm1 -volume hotspot -aggr-list
aggr1,aggr2 -aggr-list-multiplier 2 -origin-volume <origin_vol> -origin
-vserver <origin_svm> -size <size> -junction-path /hotspot
```

3. データLIFを作成します。このコマンドを4回実行し、準備シートに記載されているノードにSVMごとに2つのデータLIFを作成します。繰り返しごとに値を適切に調整してください。

```
cache::> net int create -vserver svm1 -home-port e0a -home-node node1
-address 192.168.0.11 -netmask-length 24
```

次のステップ

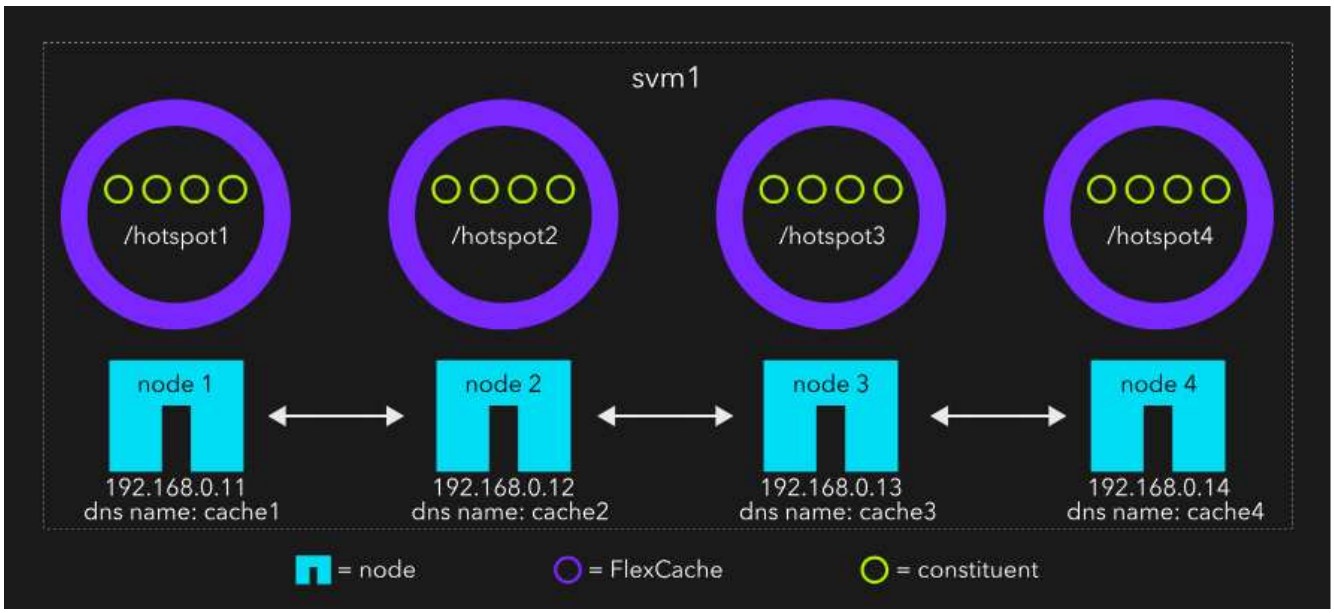
次に、HDFAを適切に利用するようにクライアントを設定する必要があります。を参照して "クライアント

セツテイ"

SVM内HDFA（4x1x4）を作成

1. 図2に示すように、4x1x4のSVM間HDFAの構成を支援するために、準備シートに必要事項を記入してください。

   図2：4x1x4のSVM内HDFAレイアウト



SVM	HDFあたりのノード数	アグリゲート	ノードあたりのコンスティチュエン	ジャンクションパス	データLIFのIP
svm1	node1	アグリゲート1	4	/hotspot1	192.168.0.11
svm1	ノード2	aggr2	4	/hotspot2	192.168.0.12
svm1	node3	aggr3	4	/hotspot3	192.168.0.13
svm1	node4	aggr4	4	/hotspot4	192.168.0.14

2. HDFSを作成します。次のコマンドを、準備シートの各行に1回ずつ、4回実行します。各イテレーションの値と `junction-path`値は必ず調整して `aggr-list`ください。

```
cache::> flexcache create -vserver svm1 -volume hotspot1 -aggr-list
aggr1 -aggr-list-multiplier 4 -origin-volume <origin_vol> -origin
-vserver <origin_svm> -size <size> -junction-path /hotspot1
```

3. データLIFを作成します。このコマンドを4回実行し、SVMに合計4つのデータLIFを作成します。ノードごとに1つのデータLIFが必要です。繰り返しごとに値を適切に調整してください。

```
cache::> net int create -vserver svm1 -home-port e0a -home-node node1
-address 192.168.0.11 -netmask-length 24
```

次のステップ

次に、HDFAを適切に利用するようにクライアントを設定する必要があります。を参照して "クライアントセツテイ"

ONTAP NAS接続を分散するためのクライアントの設定

ホットスポットの問題を解決するには、CPUのボトルネックを回避するためにクライアントを適切に設定します。

Linuxクライアントノセツテイ

SVM内とSVM間のどちらのHDFA環境を選択した場合でも、Linuxでを使用して、異なるHDFS間でクライアントのロードバランシングが行われるようにする必要があります `autofs`。 `autofs`設定はSVM間とSVM内で異なります。

開始する前に

必要であり、適切な依存関係がインストールされている必要があります `autofs`。詳細については、Linuxのマニュアルを参照してください。

タスクの内容

ここで説明する手順では、次のエントリを持つサンプルファイルを使用し `/etc/auto_master`ます。

```
/flexcache auto_hotspot
```

これにより、プロセスがディレクトリにアクセスしようとするときに、ディレクトリ `/flexcache`内で `/etc`呼び出されるファイルを検索するように `auto_hotspot`設定され `autofs`ます。ファイルの内容 `auto_hotspot`によって、ディレクトリ内でマウントするNFSサーバとジャンクションパスが決まり `/flexcache`ます。ここで説明する例は、ファイルのさまざまな構成 `auto_hotspot`です。

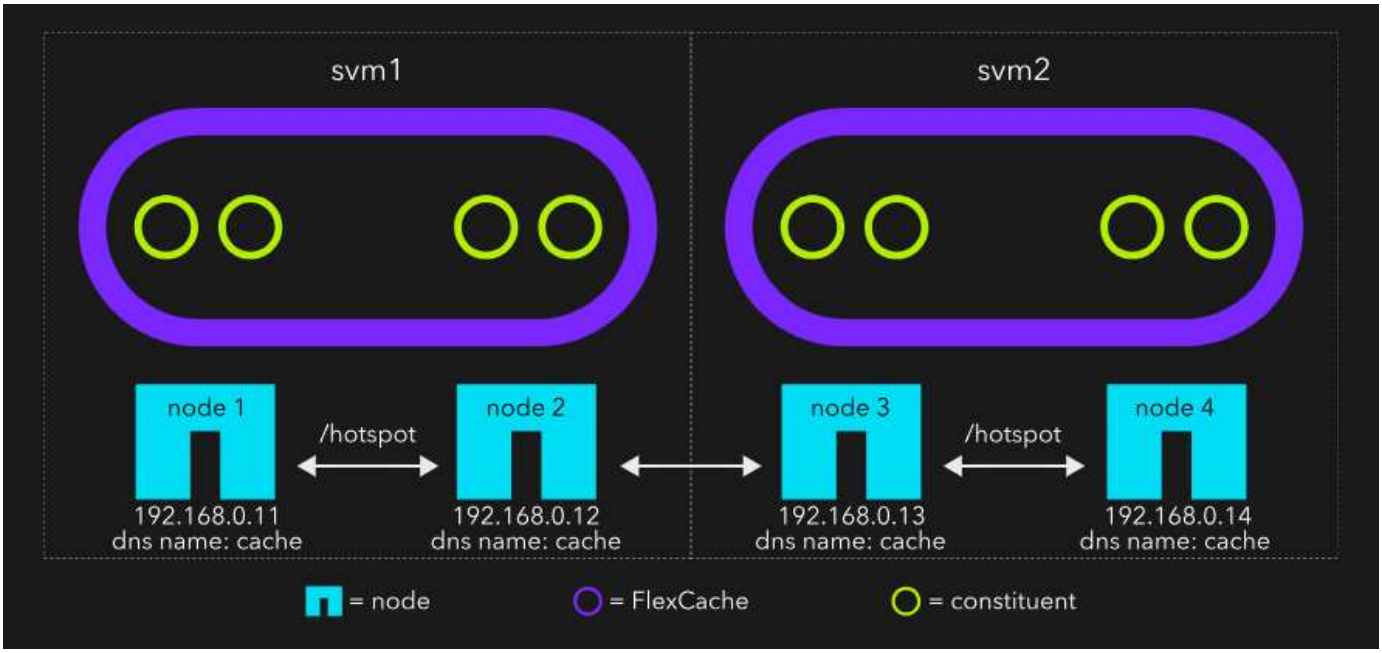
SVM内HDFA autofs設定

次の例では、の図のマップを作成し `autofs`図1ます。各キャッシュには同じジャンクションパスがあり、ホスト名には4つのDNS Aレコードがあるため、 `cache`必要なのは1行だけです。

```
hotspot cache:/hotspot
```

この1行だけで、NFSクライアントはホスト名のDNSルックアップを実行し `cache`ます。 `DNS`は、 `IP`をラウンドロビン方式で返すように設定されています。これにより、フロントエンド`NAS`接続が均等に分散されます。クライアントは`IP`を受信すると、ジャンクションパスをに `/flexcache/hotspot`マウントします `/hotspot`。SVM1、SVM2、SVM3、SVM4に接続することもできますが、特定のSVMは関係ありません。

図1：2x2x2のSVM間HDFA



SVM内HDFA autofs設定

次の例では、の図のマップを作成し `autofs`図2ます。HDFジャンクションパス環境に含まれるIPをNFSクライアントがマウントするようにする必要があります。つまり、IP 192.168.0.11以外でマウントしたくないということです `/hotspot1`。これを行うには、マップ内の1つのローカルマウントの場所について、4つのIP/ジャンクションパスのペアをすべてリストし `auto_hotspot`ます。

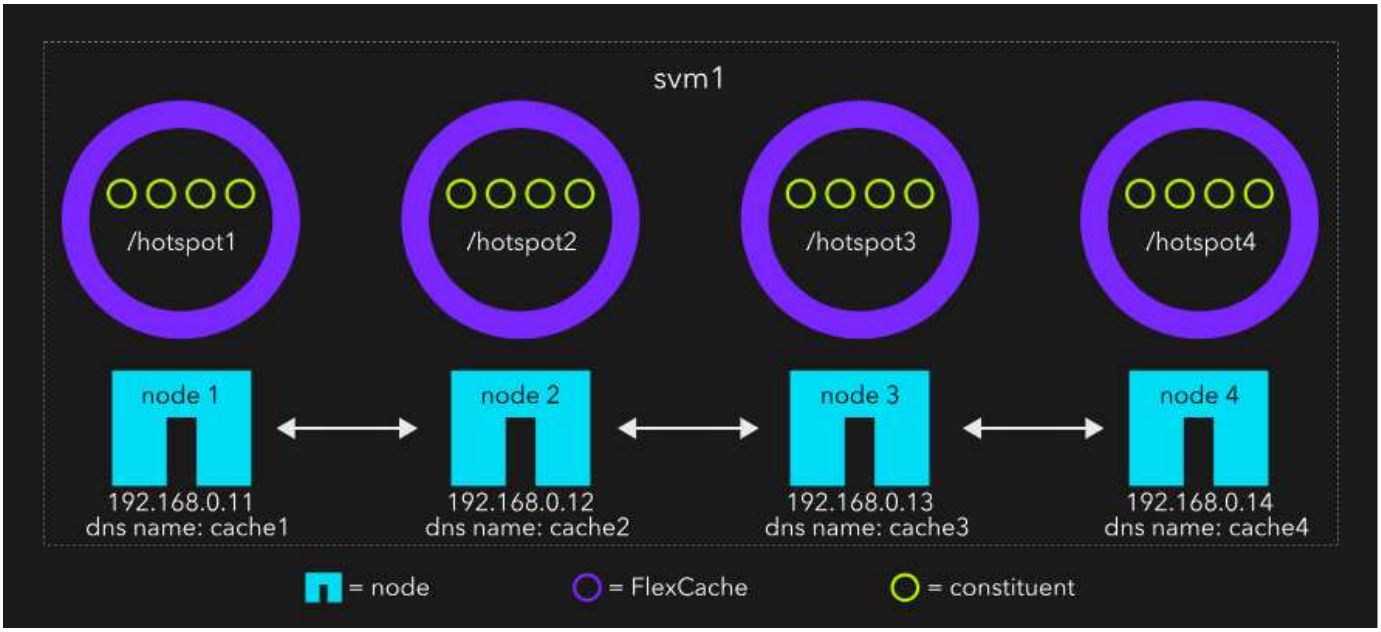
(`\`次の例のバックスラッシュ）は、エントリを次の行に続けて読みやすくします。

```
hotspot     cache1:/hostspot1 \
            cache2:/hostspot2 \
            cache3:/hostspot3 \
            cache4:/hostspot4
```

クライアントがにアクセスしようとすると `/flexcache/hotspot`、 `autofs`は4つのホスト名すべてに対して前方検索を実行します。4つのIPがすべてクライアントと同じサブネットにある場合、または別のサブネットにある場合、は `autofs`各IPにNFS NULL pingを発行します。

このNULL pingは、パケットをONTAPのNFSサービスで処理する必要がありますが、ディスクアクセスは必要ありません。最初に返されるpingはIPで、junction-pathが `autofs`mountを選択します。

図2：SVM内HDFA×4x1x4



Windowsクライアントセツテイ

Windowsクライアントでは、SVM内HDFAを使用する必要があります。SVM内の異なるHDFS間で負荷を分散するには、各HDFに一意の共有名を追加する必要があります。その後、の手順に従って、"Microsoftのドキュメント"同じフォルダに複数のDFSターゲットを実装します。

著作権に関する情報

商標に関する情報