ベストプラクティスと推奨事項
Trident

NetApp
March 05, 2026

This PDF was generated from https://docs.netapp.com/ja-jp/trident-2506/trident-reco/deploy-reco.html on March 05, 2026. Always check docs.netapp.com for the latest.

目次

ベストプラクティスと推奨事項

導入

Trident を展開するときは、ここに記載された推奨事項を使用してください。

専用の名前空間にデプロイする

"ネームスペース"異なるアプリケーション間の管理上の分離を提供し、リソース共有の障壁となります。たとえば、ある名前空間の PVC を別の名前空間で使用することはできません。 Trident は、 Kubernetes クラスター内のすべての名前空間に PV リソースを提供し、その結果、昇格された権限を持つサービス アカウントを活用します。

さらに、 Tridentポッドにアクセスすると、ユーザーはストレージ システムの資格情報やその他の機密情報にアクセスできる可能性があります。アプリケーション ユーザーと管理アプリケーションがTridentオブジェクト定義またはポッド自体にアクセスできないようにすることが重要です。

クォータと範囲制限を使用してストレージ消費を制御する

Kubernetes には 2 つの機能があり、これらを組み合わせることで、アプリケーションによるリソース消費を制限する強力なメカニズムが提供されます。その "ストレージクォータメカニズム"管理者は、グローバルおよびストレージ クラス固有の容量とオブジェクト数の消費制限を名前空間ごとに実装できます。さらに、 "範囲制限"要求がプロビジョナーに転送される前に、PVC 要求が最小値と最大値の範囲内であることを確認します。

これらの値は名前空間ごとに定義されます。つまり、各名前空間には、そのリソース要件に適合する値が定義されている必要があります。詳細については、こちらをご覧ください。 "割り当てを活用する方法" 。

ストレージ構成

NetAppポートフォリオの各ストレージ プラットフォームには、コンテナ化されているかどうかに関係なく、アプリケーションに役立つ独自の機能があります。

プラットフォームの概要

Trident はONTAPおよび Element と連携します。すべてのアプリケーションやシナリオに他のプラットフォームよりも適したプラットフォームは存在しませんが、プラットフォームを選択する際には、アプリケーションのニーズとデバイスを管理するチームを考慮する必要があります。

利用しているプロトコルに応じて、ホスト オペレーティング システムのベースライン ベスト プラクティスに従う必要があります。オプションとして、可能な場合は、バックエンド、ストレージ クラス、および PVC 設定にアプリケーションのベスト プラクティスを組み込んで、特定のアプリケーションのストレージを最適化することを検討することもできます。

ONTAPとCloud Volumes ONTAP のベストプラクティス

ONTAPおよびCloud Volumes ONTAP for Tridentを構成するためのベスト プラクティスについて説明します。

次の推奨事項は、 Tridentによって動的にプロビジョニングされるボリュームを消費するコンテナ化されたワークロード用にONTAPを構成するためのガイドラインです。それぞれの環境における適切性を考慮して評価する必要があります。

Trident専用の**SVM**を使用する

ストレージ仮想マシン (SVM) は、 ONTAPシステム上のテナント間の分離と管理の分離を実現します。SVMをアプリケーション専用にすることで、権限の委任が可能になり、リソース消費を制限するためのベスト プラクティスを適用できるようになります。

SVM の管理にはいくつかのオプションがあります。

- バックエンド構成でクラスタ管理インターフェイスと適切な資格情報を提供し、SVM 名を指定します。
- ONTAP System Manager または CLI を使用して、SVM 専用の管理インターフェイスを作成します。
- 管理ロールを NFS データ インターフェイスと共有します。

いずれの場合も、インターフェイスは DNS に存在する必要があり、 Trident を構成するときに DNS 名を使用する必要があります。これにより、ネットワーク ID 保持を使用しない SVM-DR など、一部の DR シナリオが容易になります。

SVM に専用の管理 LIF を使用するか、共有の管理 LIF を使用するかという優先順位はありませんが、ネットワーク セキュリティ ポリシーが選択したアプローチと一致していることを確認する必要があります。いずれにせよ、管理LIFはDNS経由でアクセスできるようにして、最大限の柔軟性を実現する必要があります。 "SVM-DR" Tridentと併用してください。

最大音量数を制限する

ONTAPストレージ システムには最大ボリューム数があり、これはソフトウェア バージョンとハードウェア プラットフォームによって異なります。参照 "NetApp Hardware Universe"正確な制限を確認するには、特定のプラットフォームとONTAPバージョンを参照してください。ボリューム数が使い果たされると、 Tridentだけでなく、すべてのストレージ要求のプロビジョニング操作が失敗します。

トライデントの `ontap-nas`そして `ontap-san`ドライバーは、作成された各 Kubernetes 永続ボリューム (PV) に対して FlexVolume をプロビジョニングします。その `ontap-nas-economy`ドライバーは、約 200 PV ごとに 1 つの FlexVolume を作成します (50 ～ 300 の間で構成可能)。その `ontap-san-economy`ドライバーは、100 PV ごとに約 1 つの FlexVolume を作成します (50 ～ 200 の間で構成可能)。Trident がストレージ システム上の使用可能なボリュームをすべて消費しないようにするには、SVM に制限を設定する必要があります。コマンドラインからこれを実行できます:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

の価値 `max-volumes`環境に固有のいくつかの基準によって異なります。

- ONTAPクラスタ内の既存ボリュームの数
- 他のアプリケーション用にTridentの外部にプロビジョニングする予定のボリュームの数
- Kubernetes アプリケーションによって消費されると予想される永続ボリュームの数

その `max-volumes`値は、個々のONTAPノードではなく、 ONTAPクラスタ内のすべてのノードにわたってプロビジョニングされたボリュームの合計です。その結果、 ONTAPクラスタ ノードに、他のノードよりもはる

かに多くの、または少ないTridentプロビジョニング ボリュームが存在する状況が発生する可能性があります。

たとえば、2 ノードのONTAPクラスタは、最大 2000 個のFlexVolボリュームをホストできます。最大ボリューム数を 1250 に設定するのは非常に合理的であると思われます。しかし、もし "アグリゲート"1 つのノードからのすべてのアグリゲートが SVM に割り当てられている場合、または 1 つのノードから割り当てられたアグリゲートがプロビジョニングできない場合 (たとえば、容量の問題など)、他のノードがすべてのTridentプロビジョニング ボリュームのターゲットになります。これは、そのノードのボリューム制限に達する前に、`max-volumes`値に達すると、 Tridentと、そのノードを使用するその他のボリューム操作の両方に影響します。 この状況を回避するには、クラスター内の各ノードからのアグリゲートが、**Trident**が使用する **SVM** に均等に割り当てられていることを確認します。

ボリュームをクローニングする

NetApp Tridentは、 `ontap-nas` 、 `ontap-san` 、 `solidfire-san` 、 そして `gcp-cvs`ストレージ ドライバー。使用する際は `ontap-nas-flexgroup`または `ontap-nas-economy`ドライバーの場合、クローン作成はサポートされません。既存のボリュームから新しいボリュームを作成すると、新しいスナップショットが作成されます。

> 異なる StorageClass に関連付けられている PVC の複製は避けてください。互換性を確保し、予期しない動作を防ぐために、同じ StorageClass 内でクローン操作を実行します。

Tridentによって作成されるボリュームの最大サイズを制限する

Tridentで作成できるボリュームの最大サイズを設定するには、 `limitVolumeSize`パラメータ `backend.json`意味。

ストレージ アレイでのボリューム サイズの制御に加えて、Kubernetes の機能も活用する必要があります。

Tridentによって作成される **FlexVol** の最大サイズを制限する

ontap-san-economyおよびontap-nas-economyドライバのプールとして使用されるFlexVolの最大サイズを設定するには、 `limitVolumePoolSize`パラメータ `backend.json`意味。

双方向**CHAP**を使用するように**Trident**を設定する

バックエンド定義で CHAP イニシエーターとターゲットのユーザー名とパスワードを指定し、 Trident でSVM 上で CHAP を有効にすることができます。使用して `useCHAP`バックエンド構成のパラメータを指定すると、 Trident はCHAP を使用してONTAPバックエンドの iSCSI 接続を認証します。

SVM QoSポリシーを作成して使用する

SVM に適用されたONTAP QoS ポリシーを活用すると、 Trident でプロビジョニングされたボリュームで消費可能な IOPS の数が制限されます。これは、 "いじめを防ぐ"または制御不能なコンテナがTrident SVM 外部のワークロードに影響を与えないようにします。

SVM の QoS ポリシーは数ステップで作成できます。最も正確な情報については、ご使用のONTAPバージョンのドキュメントを参照してください。以下の例では、SVM で使用可能な合計 IOPS を 5000 に制限する QoS ポリシーを作成します。

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

さらに、 ONTAPのバージョンがサポートしている場合は、コンテナ化されたワークロードへのスループットの量を保証するために QoS 最小値の使用を検討できます。アダプティブ QoS は SVM レベルのポリシーと互換性がありません。

コンテナ化されたワークロード専用の IOPS の数は、さまざまな側面によって異なります。とりわけ、次のようなものがあります:

- ストレージ アレイを使用するその他のワークロード。 Kubernetes デプロイメントに関連しない他のワークロードがストレージ リソースを利用している場合は、それらのワークロードが誤って悪影響を受けないように注意する必要があります。
- コンテナ内で実行されると予想されるワークロード。高い IOPS 要件を持つワークロードがコンテナ内で実行される場合、QoS ポリシーが低いとエクスペリエンスが悪くなります。

SVM レベルで割り当てられた QoS ポリシーにより、SVM にプロビジョニングされたすべてのボリュームが同じ IOPS プールを共有することになるということを覚えておくことが重要です。コンテナ化されたアプリケーションの 1 つまたは少数に高い IOPS 要件がある場合、他のコンテナ化されたワークロードに対して脅威となる可能性があります。このような場合は、外部自動化を使用してボリュームごとの QoS ポリシーを割り当てることを検討してください。

ONTAPバージョンが 9.8 より前の場合にのみ、QoS ポリシー グループを SVM に割り当てる必要があります。

Tridentの QoS ポリシー グループを作成する

サービス品質 (QoS) は、競合するワークロードによって重要なワークロードのパフォーマンスが低下しないことを保証します。 ONTAP QoS ポリシー グループはボリュームの QoS オプションを提供し、ユーザーが 1 つ以上のワークロードのスループット上限を定義できるようにします。 QoSの詳細については、以下を参照してください。 "QoSによるスループットの保証" 。バックエンドまたはストレージ プールで QoS ポリシー グループを指定すると、そのプールまたはバックエンドで作成された各ボリュームに適用されます。

ONTAP には、従来型とアダプティブ型の 2 種類の QoS ポリシー グループがあります。従来のポリシー グループは、IOPS で一定した最大 (または後のバージョンでは最小) スループットを提供します。アダプティブ QoS は、ワークロードのサイズの変化に応じて IOPS と TB|GB の比率を維持しながら、スループットをワークロードのサイズに合わせて自動的にスケーリングします。これは、大規模な展開で数百または数千のワークロードを管理する場合に大きな利点となります。

QoS ポリシー グループを作成するときは、次の点を考慮してください。

- 設定する必要があります `qosPolicy`キー入力 `defaults`バックエンド構成のブロック。次のバックエンド構成の例を参照してください。

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
  - labels:
      performance: extreme
    defaults:
      adaptiveQosPolicy: extremely-adaptive-pg
  - labels:
      performance: premium
    defaults:
      qosPolicy: premium-pg
```

- 各ボリュームがポリシー グループで指定されたスループット全体を取得できるように、ボリュームごとにポリシー グループを適用する必要があります。共有ポリシー グループはサポートされていません。

QoSポリシーグループの詳細については、以下を参照してください。 "ONTAPコマンド リファレンス" 。

Kubernetes クラスター メンバーへのストレージ リソース アクセスを制限する

Tridentによって作成された NFS ボリューム、iSCSI LUN、および FC LUN へのアクセスを制限することは、Kubernetes デプロイメントのセキュリティ体制の重要な要素です。そうすることで、Kubernetes クラスターの一部ではないホストがボリュームにアクセスして予期せずデータを変更する可能性を防ぐことができます。

名前空間は Kubernetes 内のリソースの論理的な境界であることを理解することが重要です。同じ名前空間内のリソースは共有できると想定されていますが、重要なのは、名前空間間の機能がないことです。つまり、PV はグローバル オブジェクトですが、PVC にバインドされている場合は、同じ名前空間にあるポッドからのみアクセスできます。 適切な場合には、名前空間を使用して分離を行うことが重要です。

Kubernetes コンテキストでのデータ セキュリティに関してほとんどの組織が主に懸念するのは、コンテナ内のプロセスが、ホストにマウントされているがコンテナ用ではないストレージにアクセスできることです。 "ネームスペース"この種の侵害を防ぐように設計されています。ただし、特権コンテナという例外が 1 つあります。

特権コンテナとは、通常よりも大幅に高いホストレベルの権限で実行されるコンテナです。これらはデフォルトでは拒否されないので、 "ポッドセキュリティポリシー" 。

Kubernetes と外部ホストの両方からのアクセスが必要なボリュームの場合、PV は管理者によって導入され、Tridentによって管理されない従来の方法でストレージを管理する必要があります。これにより、Kubernetes と外部ホストの両方が切断され、ボリュームを使用しなくなった場合にのみ、ストレージ ボリュームが破棄されるようになります。さらに、カスタム エクスポート ポリシーを適用して、Kubernetes クラスター ノー

ドおよび Kubernetes クラスター外部の対象サーバーからのアクセスを有効にすることもできます。

専用のインフラストラクチャ ノード (OpenShift など) またはユーザー アプリケーションをスケジュールできないその他のノードがあるデプロイメントでは、ストレージ リソースへのアクセスをさらに制限するために、別のエクスポート ポリシーを使用する必要があります。これには、インフラストラクチャ ノードにデプロイされるサービス (OpenShift Metrics サービスや Logging サービスなど) と、非インフラストラクチャ ノードにデプロイされる標準アプリケーションのエクスポート ポリシーの作成が含まれます。

専用のエクスポートポリシーを使用する

各バックエンドに対して、Kubernetes クラスター内に存在するノードへのアクセスのみを許可するエクスポート ポリシーが存在することを確認する必要があります。Trident はエクスポート ポリシーを自動的に作成および管理できます。このようにして、 Trident はKubernetes クラスター内のノードにプロビジョニングするボリュームへのアクセスを制限し、ノードの追加/削除を簡素化します。

あるいは、エクスポート ポリシーを手動で作成し、各ノード アクセス要求を処理する 1 つ以上のエクスポート ルールを設定することもできます。

- 使用 `vserver export-policy create`エクスポート ポリシーを作成するためのONTAP CLI コマンド。
- エクスポートポリシーにルールを追加するには、 `vserver export-policy rule create` ONTAP CLI コマンド。

これらのコマンドを実行すると、データにアクセスできる Kubernetes ノードを制限できます。

無効にする **`showmount`**アプリケーション**SVM**

その `showmount`この機能により、NFS クライアントは SVM に対して利用可能な NFS エクスポートのリストを照会できるようになります。 Kubernetesクラスタにデプロイされたポッドは、 `showmount -e`に対してコマンドを実行し、アクセスできないマウントも含め、使用可能なマウントのリストを受け取ります。これ自体はセキュリティ侵害にはなりませんが、権限のないユーザーが NFS エクスポートに接続する際に役立つ可能性のある不要な情報を提供します。

無効にする必要があります `showmount`SVM レベルのONTAP CLI コマンドを使用します。

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

SolidFireのベストプラクティス

Trident用にSolidFireストレージを構成するためのベスト プラクティスを学びます。

Solidfireアカウントを作成する

各SolidFireアカウントは一意のボリューム所有者を表し、独自のチャレンジ ハンドシェイク認証プロトコル (CHAP) 資格情報のセットを受け取ります。アカウントに割り当てられたボリュームには、アカウント名と相対 CHAP 資格情報を使用するか、ボリューム アクセス グループを通じてアクセスできます。 1 つのアカウントには最大 2,000 個のボリュームを割り当てることができますが、1 つのボリュームは 1 つのアカウントにのみ属することができます。

QoSポリシーを作成する

多数のボリュームに適用できる標準化されたサービス品質設定を作成して保存する場合は、 SolidFireのサービス品質 (QoS) ポリシーを使用します。

ボリュームごとに QoS パラメータを設定できます。 QoS を定義する 3 つの構成可能なパラメータ (最小 IOPS、最大 IOPS、バースト IOPS) を設定することで、各ボリュームのパフォーマンスを保証できます。

4Kb ブロック サイズで可能な最小、最大、およびバースト IOPS 値は次のとおりです。

IOPSパラメータ	定義	最小値	デフォルト値	最大値(4Kb)
最小 IOPS	ボリュームの保証されたパフォーマンスレベル。	50	50	15000
最大 IOPS	パフォーマンスはこの制限を超えることはありません。	50	15000	200,000
バースト IOPS	短いバーストシナリオで許可される最大 IOPS。	50	15000	200,000

最大 IOPS とバースト IOPS は最大 200,000 まで設定できますが、ボリュームの実際の最大パフォーマンスは、クラスターの使用状況とノードごとのパフォーマンスによって制限されます。

ブロック サイズと帯域幅は IOPS の数に直接影響します。ブロック サイズが大きくなるにつれて、システムはより大きなブロック サイズを処理するために必要なレベルまで帯域幅を増加させます。帯域幅が増加すると、システムが達成できる IOPS の数は減少します。参照 "SolidFireサービス品質"QoS とパフォーマンスの詳細については、こちらをご覧ください。

SolidFire認証

Element は、CHAP とボリューム アクセス グループ (VAG) の 2 つの認証方法をサポートしています。 CHAP は、CHAP プロトコルを使用して、バックエンドに対してホストを認証します。ボリューム アクセス グループは、プロビジョニングするボリュームへのアクセスを制御します。 NetApp、よりシンプルでスケーリングの制限がないため、認証には CHAP を使用することを推奨しています。

拡張 CSI プロビジョナーを備えたTrident は、 CHAP 認証の使用をサポートします。 VAG は、従来の非 CSI 動作モードでのみ使用する必要があります。

CHAP 認証 (イニシエーターが意図したボリューム ユーザーであることの検証) は、アカウント ベースのアクセス制御でのみサポートされます。認証に CHAP を使用する場合は、単方向 CHAP と双方向 CHAP の 2 つのオプションが利用できます。単方向 CHAP は、 SolidFireアカウント名とイニシエーター シークレットを使用してボリューム アクセスを認証します。双方向 CHAP オプションは、ボリュームがアカウント名とイニシエーター シークレットを使用してホストを認証し、次にホストがアカウント名とターゲット シークレットを使用してボリュームを認証するため、ボリュームを認証する最も安全な方法を提供します。

ただし、CHAP を有効にできず、VAG が必要な場合は、アクセス グループを作成し、ホスト イニシエーター

とボリュームをアクセス グループに追加します。アクセス グループに追加した各 IQN は、CHAP 認証の有無にかかわらず、グループ内の各ボリュームにアクセスできます。 iSCSI イニシエーターが CHAP 認証を使用するように構成されている場合は、アカウントベースのアクセス制御が使用されます。 iSCSI イニシエーターが CHAP 認証を使用するように構成されていない場合は、ボリューム アクセス グループのアクセス制御が使用されます。

さらに詳しい情報はどこで見つかりますか?

ベスト プラクティスのドキュメントの一部を以下に示します。検索 "NetAppライブラリ"最新バージョンについては。

ONTAP

- "NFS ベストプラクティスと実装ガイド"
- "SAN の管理"（iSCSIの場合）
- "RHEL の iSCSI エクスプレス構成"

エレメントソフトウェア

- "Linux用SolidFireの設定"

NetApp HCI

- "NetApp HCI導入の前提条件"
- "NetApp導入エンジンにアクセスする"

アプリケーションのベストプラクティス情報

- "ONTAP上の MySQL のベストプラクティス"
- "SolidFire上の MySQL のベストプラクティス"
- "NetApp SolidFireと Cassandra"
- "SolidFireにおけるOracleのベストプラクティス"
- "SolidFireにおけるPostgreSQLのベストプラクティス"

すべてのアプリケーションに特定のガイドラインがあるわけではないので、 NetAppチームと協力し、"NetAppライブラリ"最新のドキュメントを見つけます。

Tridentを統合

Trident を統合するには、ドライバーの選択とデプロイメント、ストレージ クラスの設計、仮想プールの設計、ストレージ プロビジョニングに対する Persistent Volume Claim (PVC) の影響、ボリューム操作、およびTrident を使用した OpenShift サービスのデプロイメントといった設計およびアーキテクチャ要素の統合が必要です。

ドライバーの選択と展開

ストレージ システムのバックエンド ドライバーを選択して展開します。

ONTAPバックエンド ドライバー

ONTAPバックエンド ドライバーは、使用されるプロトコルと、ストレージ システム上でボリュームがプロビジョニングされる方法によって区別されます。したがって、どのドライバーを展開するかを決定する際には慎重に検討してください。

より高いレベルでは、アプリケーションに共有ストレージを必要とするコンポーネント (同じ PVC にアクセスする複数のポッド) がある場合、NAS ベースのドライバーがデフォルトの選択肢になりますが、ブロックベースの iSCSI ドライバーは非共有ストレージのニーズを満たします。アプリケーションの要件と、ストレージおよびインフラストラクチャ チームの快適度に基づいてプロトコルを選択します。一般的に言えば、ほとんどのアプリケーションではそれらの間にほとんど違いがないため、共有ストレージ (複数のポッドが同時にアクセスする必要がある場合) が必要かどうかに基づいて決定することがよくあります。

利用可能なONTAPバックエンド ドライバーは次のとおりです。

- `ontap-nas`: プロビジョニングされた各 PV は完全なONTAP FlexVolume です。
- `ontap-nas-economy`: プロビジョニングされる各 PV は qtree であり、FlexVolume ごとに設定可能な数の qtree があります (デフォルトは 200)。
- `ontap-nas-flexgroup`: 各 PV は完全なONTAP FlexGroupとしてプロビジョニングされ、SVM に割り当てられたすべてのアグリゲートが使用されます。
- `ontap-san`: プロビジョニングされた各 PV は、独自の FlexVolume 内の LUN です。
- `ontap-san-economy`: プロビジョニングされる各 PV は LUN であり、FlexVolume ごとに構成可能な数の LUN (デフォルトは 100) があります。

3 つの NAS ドライバーのいずれかを選択すると、アプリケーションで利用できる機能にいくつかの影響が生じます。

以下の表では、すべての機能がTridentを通じて公開されているわけではないことに注意してください。一部の機能は、その機能が必要な場合、プロビジョニング後にストレージ管理者が適用する必要があります。上付き脚注は、機能およびドライバーごとに機能を区別します。

ONTAP NAS ドライバー	**Snapshot** 数	クローン	動的エクスポートポリシー	マルチアタッチ	**QoS**	サイズ変更	レプリケーション
`ontap-nas`	はい	はい	はい脚注:5[]	はい	はい脚注:1[]	はい	はい脚注:1[]
`ontap-nas-economy`	脚注:3[]	脚注:3[]	はい脚注:5[]	はい	脚注:3[]	はい	脚注:3[]
`ontap-nas-flexgroup`	はい脚注:1[]	いいえ	はい脚注:5[]	はい	はい脚注:1[]	はい	はい脚注:1[]

Trident はONTAP用の 2 つの SAN ドライバーを提供しており、その機能を以下に示します。

ONTAP SAN ドライバー	**Snapshot** 数	クローン	マルチアタッチ	双方向**CHAP**	**QoS**	サイズ変更	レプリケーション
`ontap-san`	はい	はい	はい脚注:4[]	はい	はい脚注:1[]	はい	はい脚注:1[]

ONTAP SAN ドライバー	**Snapshot** 数	クローン	マルチアタッチ	双方向**CHAP**	**QoS**	サイズ変更	レプリケーション
`ontap-san-economy`	はい	はい	はい脚注:4[]	はい	脚注:3[]	はい	脚注:3[]

> 上記の表の脚注: はい脚注:1[]: Tridentによって管理されません はい脚注:2[]: Tridentによって管理されますが、PV 細分化されません いいえ脚注:3[]: Tridentによって管理されず、PV 細分化されません はい脚注:4[]: 生のブロック ボリュームでサポートされます はい脚注:5[]: Tridentによってサポートされます

PV 粒度ではない機能は FlexVolume 全体に適用され、すべての PV (つまり、共有 FlexVol 内の qtree または LUN) が共通のスケジュールを共有します。

上の表からわかるように、 `ontap-nas`そして `ontap-nas-economy`同じです。しかし、 `ontap-nas-economy`ドライバーは `PV` 単位の粒度でスケジュールを制御する機能を制限します。これは特に災害復旧とバックアップ計画に影響を与える可能性があります。 `ONTAP`ストレージで`PVC`クローン機能を活用したい開発チームにとって、これは `ontap-nas`、 `ontap-san`または `ontap-san-economy`ドライバー。

> その `solidfire-san`ドライバーは PVC の複製も可能です。

Cloud Volumes ONTAPバックエンド ドライバー

Cloud Volumes ONTAP は、ファイル共有や、NAS および SAN プロトコル (NFS、SMB/CIFS、iSCSI) を提供するブロック レベルのストレージなど、さまざまなユース ケースに対応するエンタープライズ クラスのストレージ機能とともにデータ制御を提供します。 Cloud Volume ONTAPと互換性のあるドライバーは `ontap-nas`、 `ontap-nas-economy` 、 `ontap-san`そして `ontap-san-economy`。これらは、Cloud Volume ONTAP for Azure、Cloud Volume ONTAP for GCP に適用されます。

Amazon FSx for ONTAPバックエンドドライバー

Amazon FSx for NetApp ONTAPすると、使い慣れたNetApp の機能、パフォーマンス、管理機能を活用しながら、AWS にデータを保存するシンプルさ、俊敏性、セキュリティ、スケーラビリティを活用できます。 FSx for ONTAP は、多くのONTAPファイル システム機能と管理 API をサポートしています。 Cloud Volume ONTAPと互換性のあるドライバーは `ontap-nas`、 `ontap-nas-economy` 、 `ontap-nas-flexgroup` 、 `ontap-san`そして `ontap-san-economy`。

NetApp HCI/ SolidFireバックエンド ドライバー

その `solidfire-san`NetApp HCI/ SolidFireプラットフォームで使用されるドライバーは、管理者が QoS 制限に基づいてTridentの Element バックエンドを構成するのに役立ちます。 Tridentによってプロビジョニングされたボリュームに特定の QoS 制限を設定するようにバックエンドを設計する場合は、 `type`バックエンド ファイルのパラメータ。管理者は、ストレージ上に作成できるボリュームサイズを制限することもできます。 `limitVolumeSize`パラメータ。現在、ボリュームのサイズ変更やボリュームの複製などのElementストレージ機能は、 `solidfire-san`ドライバ。これらの操作は、Element Software Web UI を通じて手動で実行する必要があります。

SolidFireドライバー	Snapshot数	クローン	マルチアタッチ	チャップ	QoS	サイズ変更	レプリケーション
`solidfire-san`	はい	はい	はい脚注:2[]	はい	はい	はい	はい脚注:1[]

> 脚注: はい脚注:1[]: Tridentでは管理されません はい脚注:2[]: raw ブロックボリュームでサポートされます

Azure NetApp Filesバックエンド ドライバー

Tridentは `azure-netapp-files`ドライバーを管理する"Azure NetApp Files"サービス。

このドライバの詳細と設定方法については、"Azure NetApp FilesのTridentバックエンド構成" 。

Azure NetApp Filesドライバー	Snapshot数	クローン	マルチアタッチ	QoS	拡張	レプリケーション
`azure-netapp-files`	はい	はい	はい	はい	はい	はい脚注:1[]

> 脚注: はい脚注:1[]:Tridentによって管理されていない

Google Cloud バックエンド ドライバー上のCloud Volumes Service

Tridentは `gcp-cvs`Google Cloud 上のCloud Volumes Serviceにリンクするためのドライバー。

その `gcp-cvs`ドライバーは仮想プールを使用してバックエンドを抽象化し、 Trident がボリュームの配置を決定できるようにします。管理者は仮想プールを `backend.json`ファイル。ストレージ クラスはセレクターを使用して、ラベルによって仮想プールを識別します。

- バックエンドで仮想プールが定義されている場合、 Trident はそれらの仮想プールが制限されている Google Cloud ストレージ プールにボリュームを作成しようとします。
- バックエンドで仮想プールが定義されていない場合、 Trident はリージョン内の利用可能なストレージ プールから Google Cloud ストレージ プールを選択します。

TridentでGoogle Cloudバックエンドを構成するには、以下を指定する必要があります。 `projectNumber` 、 `apiRegion` 、 そして `apiKey`バックエンドファイル内。プロジェクト番号は Google Cloud コンソールで確認できます。 API キーは、Google Cloud でCloud Volumes Serviceの API アクセスを設定するときに作成したサービス アカウントの秘密キー ファイルから取得されます。

Google CloudのCloud Volumes Serviceのサービスタイプとサービスレベルの詳細については、以下を参照してください。"GCP 向け CVS のTridentサポートについて学ぶ" 。

Google Cloud ドライバー用の**Cloud Volumes Service**	**Snapshot** 数	クローン	マルチアタッチ	**QoS**	拡張	レプリケーション
`gcp-cvs`	はい	はい	はい	はい	はい	CVS-Performance サービスタイプでのみ利用可能です。

レプリケーションノート

- レプリケーションはTridentによって管理されません。
- クローン は、ソース ボリュームと同じストレージ プールに作成されます。

ストレージクラスの設計

Kubernetes ストレージ クラス オブジェクトを作成するには、個々のストレージ クラスを構成して適用する必要があります。このセクションでは、アプリケーションのストレージ クラスを設計する方法について説明します。

特定のバックエンドの利用

特定のストレージ クラス オブジェクト内でフィルタリングを使用すると、その特定のストレージ クラスで使用されるストレージ プールまたはプール セットを決定できます。ストレージ クラスでは、次の 3 セットのフィルターを設定できます。 `storagePools` 、 `additionalStoragePools` 、および/または `excludeStoragePools`。

その `storagePools`パラメーターは、指定された属性に一致するプールのセットにストレージを制限するのに役立ちます。その `additionalStoragePools`パラメータは、属性によって選択されたプールのセットとともに、 Tridentがプロビジョニングに使用するプールのセットを拡張するために使用されます。 `storagePools`パラメータ。いずれかのパラメータを単独で、または両方を一緒に使用して、適切なストレージ プールのセットが選択されていることを確認できます。

その `excludeStoragePools`パラメータは、属性に一致するリストされたプールのセットを具体的に除外するために使用されます。

QoSポリシーをエミュレートする

ストレージクラスを設計してサービス品質ポリシーをエミュレートしたい場合は、 `media`属性として `hdd`または `ssd`。に基づいて `media`ストレージクラスで指定された属性に基づいて、 Tridentは適切なバックエンドを選択します。 `hdd`または `ssd`アグリゲートをメディア属性と一致させてから、特定のアグリゲートにボリュームのプロビジョニングを指示します。したがって、PREMIUMというストレージクラスを作成すると、 `media`属性設定 `ssd`これは PREMIUM QoS ポリシーとして分類できます。メディア属性を「hdd」に設定し、STANDARD QoS ポリシーとして分類できる別のストレージ クラス STANDARD を作成できます。また、ストレージ クラスの「IOPS」属性を使用して、QoS ポリシーとして定義できる Element アプライアンスにプロビジョニングをリダイレクトすることもできます。

特定の機能に基づいてバックエンドを活用する

ストレージ クラスは、シン プロビジョニング、シック プロビジョニング、スナップショット、クローン、暗

号化などの機能が有効になっている特定のバックエンドでボリュームのプロビジョニングを直接行うように設計できます。使用するストレージを指定するには、必要な機能が有効になっている適切なバックエンドを指定するストレージ クラスを作成します。

仮想プール

仮想プールはすべてのTridentバックエンドで利用できます。 Trident が提供する任意のドライバーを使用して、任意のバックエンドの仮想プールを定義できます。

仮想プールを使用すると、管理者はストレージ クラスを通じて参照できるバックエンドの抽象化レベルを作成できるため、バックエンドでのボリュームの配置の柔軟性と効率性が向上します。同じサービス クラスで異なるバックエンドを定義できます。さらに、同じバックエンド上に、異なる特性を持つ複数のストレージ プールを作成することもできます。ストレージ クラスが特定のラベルを持つセレクタで構成されている場合、 Trident はすべてのセレクタ ラベルに一致するバックエンドを選択してボリュームを配置します。ストレージ クラス セレクターのラベルが複数のストレージ プールと一致する場合、 Trident はそのうちの 1 つを選択してボリュームをプロビジョニングします。

仮想プールの設計

バックエンドを作成する際には、一般的に一連のパラメータを指定できます。管理者が同じストレージ認証情報と異なるパラメータセットを持つ別のバックエンドを作成することは不可能でした。仮想プールの導入により、この問題は軽減されました。仮想プールは、バックエンドとKubernetesストレージクラスの間に導入されたレベル抽象化であり、管理者は、バックエンドに依存しない方法で、Kubernetesストレージクラスを介してセレクタとして参照できるラベルとともにパラメータを定義できます。仮想プールは、 Tridentを使用してサポートされているすべてのNetAppバックエンドに対して定義できます。そのリストには、 SolidFire/NetApp HCI、 ONTAP、GCPのCloud Volumes Service 、 Azure NetApp Filesが含まれます。

> ⓘ 仮想プールを定義するときは、バックエンド定義内の既存の仮想プールの順序を変更しないことをお勧めします。既存の仮想プールの属性を編集/変更せず、代わりに新しい仮想プールを定義することもお勧めします。

異なるサービスレベル/**QoS**のエミュレーション

サービス クラスをエミュレートするための仮想プールを設計することが可能です。 Cloud Volume Service for Azure NetApp Filesの仮想プール実装を使用して、さまざまなサービス クラスを設定する方法を調べてみましょう。異なるパフォーマンス レベルを表す複数のラベルを使用して、 Azure NetApp Filesバックエンドを構成します。セット `servicelevel`側面を適切なパフォーマンス レベルに設定し、各ラベルの下にその他の必要な側面を追加します。次に、異なる仮想プールにマップする異なる Kubernetes ストレージ クラスを作成します。使用して `parameters.selector`フィールドでは、各 StorageClass はボリュームをホストするために使用できる仮想プールを呼び出します。

特定の側面のセットを割り当てる

単一のストレージ バックエンドから、特定の側面を持つ複数の仮想プールを設計できます。そのためには、バックエンドを複数のラベルで構成し、各ラベルの下に必要な側面を設定します。次に、Kubernetesストレージクラスを作成します。 `parameters.selector`異なる仮想プールにマップされるフィールド。バックエンドでプロビジョニングされるボリュームには、選択した仮想プールで定義された側面が設定されます。

ストレージのプロビジョニングに影響する**PVC**の特性

要求されたストレージ クラスを超える一部のパラメーターは、PVC の作成時にTridentプロビジョニングの決定プロセスに影響を与える可能性があります。

アクセス モード

PVC 経由でストレージを要求する場合、必須フィールドの 1 つはアクセス モードです。希望するモードは、ストレージ要求をホストするために選択されたバックエンドに影響する可能性があります。

Trident は、次のマトリックスに従って、使用されるストレージ プロトコルと指定されたアクセス メソッドを一致させようとします。これは、基盤となるストレージ プラットフォームとは独立しています。

	一度だけ読み書き可能	読み取り専用	読み取り書き込み多数
iSCSI	はい	はい	はい（生のブロック）
NFS	はい	はい	はい

NFS バックエンドが構成されていないTridentデプロイメントに ReadWriteMany PVC の要求を送信すると、ボリュームはプロビジョニングされません。このため、リクエスト側はアプリケーションに適したアクセス モードを使用する必要があります。

ボリューム操作

永続ボリュームを変更する

永続ボリュームは、2 つの例外を除き、Kubernetes 内の不変オブジェクトです。作成したら、再利用ポリシーとサイズを変更できます。ただし、これによってボリュームの一部の側面が Kubernetes の外部で変更されるのを防ぐことはできません。これは、特定のアプリケーション用にボリュームをカスタマイズしたり、容量が誤って消費されないようにしたり、あるいは何らかの理由でボリュームを別のストレージ コントローラに移動したりする場合に望ましいことがあります。

> Kubernetes のツリー内プロビジョナーは、現時点では NFS、iSCSI、または FC PV のボリューム サイズ変更操作をサポートしていません。 Trident は、 NFS、iSCSI、FC ボリュームの拡張をサポートします。

PV の接続詳細は作成後に変更できません。

オンデマンドのボリュームスナップショットを作成する

Trident は、 CSI フレームワークを使用したオンデマンドのボリューム スナップショットの作成とスナップショットからの PVC の作成をサポートします。スナップショットは、データの特定時点のコピーを維持する便利な方法を提供し、Kubernetes のソース PV から独立したライフサイクルを持ちます。これらのスナップショットを使用して PVC を複製できます。

スナップショットからボリュームを作成する

Trident は、ボリューム スナップショットからの PersistentVolume の作成もサポートしています。これを実現するには、PersistentVolumeClaimを作成し、 `datasource`ボリュームを作成するために必要なスナップショットとして。 Trident は、スナップショットに存在するデータを含むボリュームを作成することで、この PVC を処理します。この機能を使用すると、リージョン間でデータを複製したり、テスト環境を作成したり、破損または壊れた本番ボリューム全体を交換したり、特定のファイルやディレクトリを取得して別の接続されたボリュームに転送したりすることが可能になります。

クラスター内のボリュームを移動する

ストレージ管理者は、ストレージ コンシューマーに支障をきたすことなく、 ONTAPクラスタ内のアグリゲートとコントローラ間でボリュームを移動できます。宛先アグリゲートがTridentが使用している SVM がアクセスできるものである限り、この操作はTridentまたは Kubernetes クラスターに影響を与えません。重要なのは、集計が SVM に新しく追加された場合、それをTridentに再度追加してバックエンドを更新する必要があることです。これにより、 Trident はSVM の再インベントリを実行し、新しいアグリゲートが認識されるようになります。

ただし、バックエンド間でのボリュームの移動は、 Tridentでは自動的にはサポートされません。これには、同じクラスター内の SVM 間、クラスター間、または異なるストレージ プラットフォーム (そのストレージ システムがTridentに接続されている場合も含む) が含まれます。

ボリュームを別の場所にコピーする場合は、ボリューム インポート機能を使用して現在のボリュームをTridentにインポートできます。

ボリュームを拡張する

Trident は、 NFS、iSCSI、および FC PV のサイズ変更をサポートしています。これにより、ユーザーは Kubernetes レイヤーを通じてボリュームのサイズを直接変更できるようになります。ボリューム拡張は、ONTAP、 SolidFire/ NetApp HCI 、 Cloud Volumes Serviceバックエンドを含むすべての主要なNetAppストレージ プラットフォームで可能です。後で拡張できるように設定するには `allowVolumeExpansion`に `true`ボリュームに関連付けられた StorageClass 内。永続ボリュームのサイズを変更する必要がある場合は、`spec.resources.requests.storage`永続ボリューム要求の注釈を必要なボリューム サイズに設定します。Trident は、ストレージ クラスター上のボリュームのサイズ変更を自動的に処理します。

既存のボリュームを**Kubernetes**にインポートする

ボリューム インポートでは、既存のストレージ ボリュームを Kubernetes 環境にインポートできます。これは現在、 `ontap-nas` 、 `ontap-nas-flexgroup` 、 `solidfire-san` 、 `azure-netapp-files` 、 そして`gcp-cvs`ドライバー。この機能は、既存のアプリケーションを Kubernetes に移植する場合や、災害復旧シナリオ時に役立ちます。

ONTAPを使用する場合 `solidfire-san`ドライバーの場合は、コマンド `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml`既存のボリュームを Kubernetes にインポートして、 Tridentで管理できるようにします。インポート ボリューム コマンドで使用される PVC YAML または JSON ファイルは、Trident をプロビジョナーとして識別するストレージ クラスを指します。 NetApp HCI/ SolidFireバックエンドを使用する場合は、ボリューム名が一意であることを確認してください。ボリューム名が重複している場合は、ボリュームのインポート機能で区別できるように、ボリュームを一意の名前で複製します。

もし `azure-netapp-files`または `gcp-cvs`ドライバーを使用する場合は、コマンド `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml`Tridentで管理できるようにボリュームを Kubernetes にインポートします。これにより、一意のボリューム参照が保証されます。

上記のコマンドを実行すると、 Trident はバックエンド上のボリュームを見つけてそのサイズを読み取ります。設定された PVC のボリューム サイズが自動的に追加され (必要に応じて上書きされます)。次に、Trident は新しい PV を作成し、Kubernetes は PVC を PV にバインドします。

特定のインポートされた PVC を必要とするコンテナーがデプロイされた場合、PVC/PV ペアがボリューム インポート プロセスによってバインドされるまで、コンテナーは保留状態のままになります。 PVC/PV ペアがバインドされた後、他の問題がなければコンテナーが起動するはずです。

レジストリサービス

レジストリのストレージの展開と管理については、"ネットアップ"の中で"ブログ"。

ログサービス

他の OpenShift サービスと同様に、ログ サービスは、プレイブックに提供されるインベントリ ファイル (ホスト) によって提供される構成パラメーターを使用して Ansible を使用してデプロイされます。説明するインストール方法は 2 つあります。OpenShift の初期インストール時にログをデプロイする方法と、OpenShift のインストール後にログをデプロイする方法です。

> Red Hat OpenShift バージョン 3.9 の時点では、データ破損の懸念があるため、公式ドキュメントではログ サービスに NFS を使用しないことを推奨しています。これは Red Hat による自社製品のテストに基づいています。 ONTAP NFS サーバーにはこれらの問題はなく、ログの展開を簡単にバックアップできます。最終的に、ログ サービスのプロトコルの選択はユーザー次第ですが、 NetAppプラットフォームを使用する場合はどちらも問題なく機能し、NFS が好みであれば NFS を避ける理由はありません。

ログサービスでNFSを使用する場合は、Ansible変数を設定する必要があります。 `openshift_enable_unsupported_configurations`に `true`インストーラーが失敗するのを防ぐためです。

始めましょう

オプションで、ログ サービスは、アプリケーションだけでなく、OpenShift クラスター自体のコア操作にもデプロイできます。操作ログを展開することを選択した場合は、変数を指定して `openshift_logging_use_ops`として `true`、サービスのインスタンスが 2 つ作成されます。操作のログインスタンスを制御する変数には「ops」が含まれますが、アプリケーションのインスタンスには含まれません。

基盤となるサービスによって正しいストレージが確実に利用されるようにするには、デプロイメント方法に応じて Ansible 変数を構成することが重要です。それぞれの展開方法のオプションを見てみましょう。

> 以下の表には、ログ サービスに関連するストレージ構成に関係する変数のみが含まれています。その他のオプションについては、"Red Hat OpenShift のログ記録ドキュメント"これらは、展開に応じて確認、構成、および使用する必要があります。

以下の表の変数により、Ansible プレイブックは提供された詳細を使用して、ログ サービス用の PV と PVC を作成します。この方法は、OpenShift のインストール後にコンポーネント インストール プレイブックを使用するよりも柔軟性が大幅に低くなりますが、既存のボリュームが利用できる場合はオプションとして使用できます。

変数	詳細
`openshift_logging_storage_kind`	設定 `nfs`インストーラーでログ サービス用の NFS PV を作成します。
`openshift_logging_storage_host`	NFS ホストのホスト名または IP アドレス。これは仮想マシンの dataLIF に設定する必要があります。
`openshift_logging_storage_nfs_directory`	NFS エクスポートのマウント パス。たとえば、ボリュームが次のようにジャンクションされている場合 `/openshift_logging`、この変数にはそのパスを使用します。

変数	詳細
`openshift_logging_storage_volume_name`	名前、例 `pv_ose_logs`、作成するPVの。
`openshift_logging_storage_volume_size`	NFSエクスポートのサイズ、例 `100Gi`。

OpenShift クラスターがすでに実行されており、 Trident がデプロイおよび構成されている場合は、インストーラーは動的プロビジョニングを使用してボリュームを作成できます。次の変数を設定する必要があります。

変数	詳細
`openshift_logging_es_pvc_dynamic`	動的にプロビジョニングされたボリュームを使用するには、true に設定します。
`openshift_logging_es_pvc_storage_class_name`	PVC で使用されるストレージ クラスの名前。
`openshift_logging_es_pvc_size`	PVC で要求されたボリュームのサイズ。
`openshift_logging_es_pvc_prefix`	ログ サービスによって使用される PVC のプレフィックス。
`openshift_logging_es_ops_pvc_dynamic`	設定 `true`オペレーションログインスタンスに動的にプロビジョニングされたボリュームを使用します。
`openshift_logging_es_ops_pvc_storage_class_name`	オペレーションログインスタンスのストレージクラスの名前。
`openshift_logging_es_ops_pvc_size`	オペレーションインスタンスのボリューム要求のサイズ。
`openshift_logging_es_ops_pvc_prefix`	ops インスタンス PVC のプレフィックス。

ログスタックをデプロイする

ログ記録を OpenShift の初期インストール プロセスの一部としてデプロイする場合は、標準のデプロイ プロセスに従うだけで済みます。 Ansible は必要なサービスと OpenShift オブジェクトを構成およびデプロイし、Ansible が完了するとすぐにサービスが利用できるようになります。

ただし、初期インストール後にデプロイする場合は、Ansible でコンポーネント プレイブックを使用する必要があります。このプロセスはOpenShiftのバージョンによって多少異なる可能性がありますので、必ず読んで従ってください。"Red Hat OpenShift Container Platform 3.11 ドキュメント"あなたのバージョン用。

メトリクスサービス

メトリクス サービスは、OpenShift クラスターのステータス、リソース使用率、可用性に関する貴重な情報を管理者に提供します。これはポッドの自動スケール機能にも必要であり、多くの組織はチャージバックやショーバック アプリケーションにメトリック サービスのデータを使用しています。

ログ サービスや OpenShift 全体と同様に、メトリック サービスのデプロイには Ansible が使用されます。また、ログ サービスと同様に、メトリック サービスは、クラスターの初期セットアップ時、またはコンポーネント インストール方法を使用して運用開始後にデプロイできます。次の表には、メトリック サービスの永続ストレージを構成するときに重要な変数が含まれています。

以下の表には、メトリック サービスに関連するストレージ構成に関係する変数のみが含まれています。ドキュメントには他にも多くのオプションが記載されており、展開に応じて確認、構成、使用する必要があります。

変数	詳細
`openshift_metrics_storage_kind`	設定 `nfs`インストーラーでログ サービス用の NFS PV を作成します。
`openshift_metrics_storage_host`	NFS ホストのホスト名または IP アドレス。これは、SVM の dataLIF に設定する必要があります。
`openshift_metrics_storage_nfs_directory`	NFS エクスポートのマウント パス。たとえば、ボリュームが次のようにジャンクションされている場合 `/openshift_metrics`、この変数にはそのパスを使用します。
`openshift_metrics_storage_volume_name`	名前、例 `pv_ose_metrics`、作成するPVの。
`openshift_metrics_storage_volume_size`	NFSエクスポートのサイズ、例 `100Gi`。

OpenShift クラスターがすでに実行されており、 Trident がデプロイおよび構成されている場合は、インストーラーは動的プロビジョニングを使用してボリュームを作成できます。次の変数を設定する必要があります。

変数	詳細
`openshift_metrics_cassandra_pvc_prefix`	メトリック PVC に使用するプレフィックス。
`openshift_metrics_cassandra_pvc_size`	要求するボリュームのサイズ。
`openshift_metrics_cassandra_storage_type`	メトリックに使用するストレージのタイプ。Ansible が適切なストレージ クラスを持つ PVC を作成するには、これを dynamic に設定する必要があります。
`openshift_metrics_cassanda_pvc_storage_class_name`	使用するストレージ クラスの名前。

メトリクスサービスをデプロイする

hosts/inventory ファイルに適切な Ansible 変数を定義して、Ansible を使用してサービスをデプロイします。OpenShift のインストール時にデプロイする場合は、PV が自動的に作成され、使用されます。コンポーネント プレイブックを使用してデプロイする場合は、OpenShift のインストール後に Ansible が必要な PVC を作成し、 Trident がそれらのストレージをプロビジョニングした後、サービスをデプロイします。

上記の変数とデプロイのプロセスは、OpenShift のバージョンごとに変更される可能性があります。必ず確認して従ってください"Red HatのOpenShiftデプロイメントガイド"お使いの環境に合わせて構成されるように、バージョンに応じて変更してください。

データ保護とディザスタ リカバリ

TridentおよびTrident を使用して作成されたボリュームの保護および回復オプションについて説明します。永続性を必要とするアプリケーションごとに、データ保護および回復戦略を用意する必要があります。

Tridentの複製と回復

災害発生時にTrident を復元するためのバックアップを作成できます。

Trident複製

Trident は、 Kubernetes CRD を使用して自身の状態を保存および管理し、Kubernetes クラスター etcd を使用してメタデータを保存します。

手順

1. Kubernetesクラスタetcdをバックアップするには"Kubernetes: etcd クラスターのバックアップ"。
2. バックアップアーティファクトをFlexVol volumeに配置する

> NetApp、 FlexVol が存在する SVM を別の SVM とのSnapMirror関係で保護することを推奨しています。

Tridentの回復

Kubernetes CRD と Kubernetes クラスター etcd スナップショットを使用して、 Trident を回復できます。

手順

1. 宛先 SVM から、Kubernetes etcd データ ファイルと証明書を含むボリュームを、マスター ノードとしてセットアップされるホストにマウントします。
2. Kubernetesクラスタに関連する必要な証明書をすべてコピーします。 `/etc/kubernetes/pki`そして、以下のetcdメンバーファイル `/var/lib/etcd`。
3. etcdバックアップからKubernetesクラスターを復元するには、"Kubernetes: etcd クラスターの復元" 。
4. 走る `kubectl get crd`すべてのTridentカスタム リソースが起動していることを確認し、 Tridentオブジェクトを取得してすべてのデータが利用可能であることを確認します。

SVMのレプリケーションとリカバリ

Tridentはレプリケーション関係を設定することはできませんが、ストレージ管理者は"ONTAPSnapMirror"SVM を複製します。

災害が発生した場合は、 SnapMirror の宛先 SVM をアクティブ化してデータの提供を開始できます。システムが復元されたら、プライマリに切り替えることができます。

タスク概要

SnapMirror SVM レプリケーション機能を使用する場合は、次の点を考慮してください。

- SVM-DR が有効になっている各 SVM ごとに個別のバックエンドを作成する必要があります。
- レプリケーションを必要としないボリュームが SVM-DR をサポートするバックエンドにプロビジョニングされることを回避するために、必要な場合にのみレプリケートされたバックエンドを選択するようにストレージ クラスを構成します。
- アプリケーション管理者は、レプリケーションに関連する追加コストと複雑さを理解し、このプロセスを開始する前にリカバリ計画を慎重に検討する必要があります。

SVMレプリケーション

使用できます"ONTAP: SnapMirror SVM レプリケーション"SVM レプリケーション関係を作成します。

SnapMirror を使用すると、複製する内容を制御するオプションを設定できます。実行時に選択したオプションを知っておく必要がありますTridentを使用したSVM回復。

- "-アイデンティティ保存真"SVM 構成全体を複製します。
- "-discard-configs ネットワーク"LIF および関連するネットワーク設定は除外されます。
- "-identity-preserve 偽"ボリュームとセキュリティ構成のみを複製します。

Tridentを使用した**SVM**回復

Trident はSVM の障害を自動的に検出しません。災害が発生した場合、管理者は新しい SVM へのTridentフェイルオーバーを手動で開始できます。

手順

1. スケジュールされた進行中のSnapMirror転送をキャンセルし、レプリケーション関係を解除し、ソース SVM を停止してから、 SnapMirror宛先 SVM をアクティブ化します。
2. 指定した場合 `-identity-preserve false`または `-discard-config network`SVMレプリケーションを設定するときは、 `managementLIF`そして `dataLIF`Tridentバックエンド定義ファイル内。
3. 確認する `storagePrefix`Tridentバックエンド定義ファイルに存在します。このパラメータは変更できません。省略 `storagePrefix`バックエンドの更新が失敗します。
4. 次のコマンドを使用して、新しい宛先 SVM 名を反映するように必要なすべてのバックエンドを更新します。

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. 指定した場合 `-identity-preserve false`または `discard-config network`、すべてのアプリケーション ポッドをバウンスする必要があります。

> 指定した場合 `-identity-preserve true`宛先 SVM がアクティブ化されると、 Tridentによってプロビジョニングされたすべてのボリュームがデータの提供を開始します。

ボリュームの複製と回復

TridentはSnapMirrorのレプリケーション関係を設定することはできませんが、ストレージ管理者は"ONTAP SnapMirrorのレプリケーションとリカバリ"Tridentによって作成されたボリュームを複製します。

その後、回復したボリュームをTridentにインポートすることができます。"tridentctl ボリュームインポート"。

> インポートはサポートされていません `ontap-nas-economy`、 `ontap-san-economy` 、 または `ontap-flexgroup-economy`ドライバー。

スナップショットデータ保護

以下を使用してデータを保護および復元できます。

- 永続ボリューム (PV) の Kubernetes ボリューム スナップショットを作成するための外部スナップショット コントローラーと CRD。

  "ボリュームスナップショット"

- ONTAPスナップショットを使用して、ボリュームの内容全体を復元したり、個々のファイルまたは LUN を回復したりします。

  "ONTAPスナップショット"

セキュリティ

セキュリティ

ここで挙げた推奨事項に従って、 Trident のインストールが安全であることを確認してください。

Trident を独自の名前空間で実行する

信頼性の高いストレージを確保し、潜在的な悪意のあるアクティビティをブロックするには、アプリケーション、アプリケーション管理者、ユーザー、および管理アプリケーションがTridentオブジェクト定義またはポッドにアクセスできないようにすることが重要です。

他のアプリケーションやユーザーをTridentから分離するには、常にTrident を独自の Kubernetes 名前空間にインストールします。(`trident`）。 Trident を独自の名前空間に配置すると、Kubernetes 管理担当者だけがTridentポッドと、名前空間の CRD オブジェクトに保存されている成果物 (該当する場合はバックエンドや CHAP シークレットなど) にアクセスできるようになります。管理者のみがTrident名前空間にアクセスできるようにし、 `tridentctl`応用。

ONTAP SANバックエンドで**CHAP**認証を使用する

TridentはONTAP SANワークロードのCHAPベースの認証をサポートします（ `ontap-san`そして `ontap-san-economy`ドライバー)。 NetApp、ホストとストレージ バックエンド間の認証に、 Tridentを使用した双方向 CHAP を使用することを推奨しています。

SANストレージドライバを使用するONTAPバックエンドの場合、 Tridentは双方向CHAPを設定し、CHAPのユーザー名とシークレットを管理できます。 `tridentctl` 。参照"ONTAP SAN ドライバーを使用してバックエンドを構成する準備をする"Trident がONTAPバックエンドで CHAP を設定する方法を理解します。

NetApp HCIおよび**SolidFire**バックエンドで**CHAP**認証を使用する

NetApp、ホストとNetApp HCIおよびSolidFireバックエンド間の認証を確実にするために、双方向 CHAP を導入することを推奨しています。 Trident は、テナントごとに 2 つの CHAP パスワードを含む秘密オブジェクトを使用します。 Tridentがインストールされると、CHAPシークレットを管理し、それを `tridentvolume`それぞれの PV の CR オブジェクト。 PV を作成すると、 Trident はCHAP シークレットを使用して iSCSI セッションを開始し、CHAP 経由でNetApp HCIおよびSolidFireシステムと通信します。

Tridentによって作成されたボリュームは、どのボリューム アクセス グループにも関連付けられていません。

Trident を**NVE** および **NAE** と併用する

NetApp ONTAP は、ディスクが盗難、返却、または再利用された場合に機密データを保護するために、保存データの暗号化を提供します。詳細については、"NetApp Volume Encryptionの設定 - 概要" 。

- バックエンドで NAE が有効になっている場合、 Tridentでプロビジョニングされたすべてのボリュームは NAE が有効になります。
  - NVE暗号化フラグを設定するには `""`NAE 対応ボリュームを作成します。
- NAEがバックエンドで有効になっていない場合、NVE暗号化フラグが設定されていない限り、 TridentでプロビジョニングされたボリュームはすべてNVE対応になります。 `false` (デフォルト値) をバックエンド構成で指定します。

NAE 対応バックエンド上のTridentで作成されたボリュームは、NVE または NAE で暗号化されている必要があります。

- NVE暗号化フラグを設定するには `true`Tridentバックエンド構成で NAE 暗号化をオーバーライドし、ボリュームごとに特定の暗号化キーを使用します。
- NVE暗号化フラグを設定する `false`NAE 対応のバックエンドでは、`NAE` 対応のボリュームが作成されます。 `NVE`暗号化フラグを次のように設定して`NAE`暗号化を無効にすることはできません。 `false` 。

- TridentでNVEボリュームを手動で作成するには、NVE暗号化フラグを明示的に設定します。 `true` 。

バックエンド構成オプションの詳細については、以下を参照してください。

- "ONTAP SAN構成オプション"
- "ONTAP NAS 構成オプション"

Linux 統合キー設定 (LUKS)

Linux Unified Key Setup (LUKS) を有効にして、 Trident上のONTAP SAN およびONTAP SAN ECONOMY ボリュームを暗号化できます。 Trident は、 LUKS で暗号化されたボリュームのパスフレーズローテーションとボリューム拡張をサポートします。

Tridentでは、LUKSで暗号化されたボリュームは、aes-xts-plain64暗号とモードを使用します。これは、"NIST" 。

LUKS 暗号化はASA r2 システムではサポートされていません。 ASA r2システムの詳細については、以下を参照してください。"ASA r2 ストレージシステムについて学ぶ" 。

開始する前に

- ワーカー ノードには、cryptsetup 2.1 以上 (3.0 未満) がインストールされている必要があります。詳細については、"Gitlab: 暗号化セットアップ" 。
- パフォーマンス上の理由から、 NetAppワーカー ノードで Advanced Encryption Standard New

Instructions (AES-NI) をサポートすることを推奨しています。 AES-NI のサポートを確認するには、次のコマンドを実行します。

```
grep "aes" /proc/cpuinfo
```

何も返されない場合、プロセッサは AES-NI をサポートしていません。 AES-NI の詳細については、以下をご覧ください。"インテル: 高度暗号化標準命令 (AES-NI)" 。

LUKS暗号化を有効にする

ONTAP SAN およびONTAP SAN ECONOMY ボリュームでは、Linux Unified Key Setup (LUKS) を使用して、ボリュームごとのホスト側暗号化を有効にすることができます。

手順

1. バックエンド構成で LUKS 暗号化属性を定義します。 ONTAP SANのバックエンド構成オプションの詳細については、以下を参照してください。"ONTAP SAN構成オプション" 。

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. 使用 `parameters.selector`LUKS 暗号化を使用してストレージ プールを定義します。例えば：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. LUKS パスフレーズを含むシークレットを作成します。例えば：

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

制限事項

LUKS で暗号化されたボリュームは、 ONTAPの重複排除と圧縮を利用できません。

LUKSボリュームをインポートするためのバックエンド構成

LUKSボリュームをインポートするには、 `luksEncryption`に(`true`バックエンドで。その`luksEncryption`オプションはボリュームがLUKS準拠かどうかをTridentに伝える(`true`）またはLUKSに準拠していない(false) を次の例のように入力します。

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

LUKSボリュームをインポートするためのPVC構成

LUKSボリュームを動的にインポートするには、アノテーションを設定します `trident.netapp.io/luksEncryption`に `true`この例に示すように、PVC に LUKS 対応ストレージ クラスを含めます。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

LUKSパスフレーズをローテーションする

LUKS パスフレーズをローテーションし、ローテーションを確認することができます。

> パスフレーズがボリューム、スナップショット、またはシークレットによって参照されなくなったことを確認するまで、パスフレーズを忘れないようにしてください。参照されたパスフレーズが失われた場合、ボリュームをマウントできなくなり、データは暗号化されたままアクセスできなくなる可能性があります。

タスク概要

新しい LUKS パスフレーズが指定された後にボリュームをマウントするポッドが作成されると、LUKS パスフレーズのローテーションが発生します。新しいポッドが作成されると、 Trident はボリューム上の LUKS パスフレーズをシークレット内のアクティブなパスフレーズと比較します。

- ボリューム上のパスフレーズがシークレット内のアクティブなパスフレーズと一致しない場合は、ローテーションが発生します。
- ボリューム上のパスフレーズがシークレット内のアクティブなパスフレーズと一致する場合、 `previous-luks-passphrase`パラメータは無視されます。

手順

1. 追加する `node-publish-secret-name`そして `node-publish-secret-namespace`StorageClass パラメータ。例えば：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. ボリュームまたはスナップショット上の既存のパスフレーズを識別します。

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["A"]
```

3. ボリュームの LUKS シークレットを更新して、新しいパスフレーズと以前のパスフレーズを指定します。確保する `previous-luke-passphrase-name`そして `previous-luks-passphrase`以前のパスフレーズと一致します。

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. ボリュームをマウントする新しいポッドを作成します。これは回転を開始するために必要です。

5. パスフレーズがローテーションされたことを確認します。

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

結果

ボリュームとスナップショットで新しいパスフレーズのみが返されたときに、パスフレーズがローテーションされました。

> たとえば、2つのパスフレーズが返された場合 `luksPassphraseNames: ["B", "A"]`、回転は不完全です。新しいポッドをトリガーして、回転を完了させることができます。

ボリューム拡張を有効にする

LUKS で暗号化されたボリュームでボリューム拡張を有効にすることができます。

手順

1. 有効にする `CSINodeExpandSecret`機能ゲート (ベータ 1.25 以上)。参照 "Kubernetes 1.25: CSIボリュームのノード駆動型拡張にSecretを使用する" 詳細については。
2. 追加する `node-expand-secret-name`そして `node-expand-secret-namespace`StorageClass パラメータ。例えば：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

結果

オンライン ストレージ拡張を開始すると、kubelet は適切な資格情報をドライバーに渡します。

Kerberos のインフライト暗号化

Kerberos インフライト暗号化を使用すると、マネージド クラスターとストレージ バックエンド間のトラフィックの暗号化を有効にして、データ アクセスのセキュリティを向上させることができます。

Trident は、ストレージ バックエンドとしてのONTAPの Kerberos 暗号化をサポートしています。

- オンプレミス**ONTAP** - Trident は、 Red Hat OpenShift およびアップストリーム Kubernetes クラスターからオンプレミスONTAPボリュームへの NFSv3 および NFSv4 接続を介した Kerberos 暗号化をサポートします。

NFS 暗号化を使用するボリュームを作成、削除、サイズ変更、スナップショット、クローン、読み取り専用クローン、およびインポートできます。

オンプレミスの**ONTAP**ボリュームでインフライト **Kerberos** 暗号化を構成する

管理対象クラスターとオンプレミスのONTAPストレージ バックエンド間のストレージ トラフィックで Kerberos 暗号化を有効にできます。

> オンプレミスのONTAPストレージバックエンドを使用したNFSトラフィックのKerberos暗号化は、 `ontap-nas`ストレージ ドライバー。

開始する前に

- アクセスできることを確認してください `tridentctl`ユーティリティ。
- ONTAPストレージ バックエンドへの管理者アクセス権があることを確認します。
- ONTAPストレージ バックエンドから共有するボリュームの名前を必ず確認してください。
- NFS ボリュームの Kerberos 暗号化をサポートするようにONTAPストレージ VM を準備していることを確認します。参照 "データLIFでKerberosを有効にする"手順についてはこちらをご覧ください。

- Kerberos 暗号化で使用する NFSv4 ボリュームが正しく構成されていることを確認します。 NetApp NFSv4ドメイン構成セクション（13ページ）を参照してください。 "NetApp NFSv4 の機能強化とベストプラクティスガイド" 。

ONTAPエクスポートポリシーを追加または変更する

既存のONTAPエクスポート ポリシーにルールを追加するか、 ONTAPストレージ VM ルート ボリュームと、アップストリーム Kubernetes クラスターと共有されているすべてのONTAPボリュームに対して Kerberos 暗号化をサポートする新しいエクスポート ポリシーを作成する必要があります。追加するエクスポート ポリシー ルール、または作成する新しいエクスポート ポリシーは、次のアクセス プロトコルとアクセス許可をサポートする必要があります。

アクセス プロトコル

NFS、NFSv3、および NFSv4 アクセス プロトコルを使用してエクスポート ポリシーを構成します。

アクセス詳細

ボリュームのニーズに応じて、3 つの異なるバージョンの Kerberos 暗号化のいずれかを構成できます。

- **Kerberos 5** - (認証と暗号化)
- **Kerberos 5i** - (ID保護を備えた認証と暗号化)
- **Kerberos 5p** - (アイデンティティとプライバシー保護を備えた認証と暗号化)

適切なアクセス権限を使用してONTAPエクスポート ポリシー ルールを設定します。たとえば、クラスターが Kerberos 5i と Kerberos 5p 暗号化を組み合わせて NFS ボリュームをマウントする場合は、次のアクセス設定を使用します。

タイプ	読み取り専用アクセス	読み取り/書き込みアクセス	スーパーユーザーアクセス
UNIX	有効	有効	有効
Kerberos 5i	有効	有効	有効
Kerberos 5p	有効	有効	有効

ONTAPエクスポート ポリシーとエクスポート ポリシー ルールを作成する方法については、次のドキュメントを参照してください。

- "エクスポート ポリシーの作成"
- "エクスポート ポリシーへのルールの追加"

ストレージバックエンドを作成する

Kerberos 暗号化機能を含むTridentストレージ バックエンド構成を作成できます。

タスク概要

Kerberos暗号化を設定するストレージバックエンド構成ファイルを作成するときに、次のオプションを使用して3つの異なるバージョンのKerberos暗号化のいずれかを指定できます。 `spec.nfsMountOptions`パラメータ:

- `spec.nfsMountOptions: sec=krb5`（認証と暗号化）
- `spec.nfsMountOptions: sec=krb5i`（ID保護を備えた認証と暗号化）

- `spec.nfsMountOptions: sec=krb5p`（アイデンティティとプライバシー保護を備えた認証と暗号化）

Kerberos レベルを 1 つだけ指定します。パラメータ リストで複数の Kerberos 暗号化レベルを指定した場合、最初のオプションのみが使用されます。

手順

1. マネージド クラスターで、次の例を使用してストレージ バックエンド構成ファイルを作成します。括弧<>内の値は、環境の情報で置き換えます。

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. 前の手順で作成した構成ファイルを使用して、バックエンドを作成します。

```
tridentctl create backend -f <backend-configuration-file>
```

バックエンドの作成に失敗した場合、バックエンドの構成に問題があります。次のコマンドを実行すると、ログを表示して原因を特定できます。

```
tridentctl logs
```

構成ファイルの問題を特定して修正したら、create コマンドを再度実行できます。

ストレージクラスを作成する

Kerberos 暗号化を使用してボリュームをプロビジョニングするためのストレージ クラスを作成できます。

タスク概要

ストレージクラスオブジェクトを作成するときに、Kerberos暗号化の3つの異なるバージョンのいずれかを指定できます。 `mountOptions`パラメータ:

- `mountOptions: sec=krb5`（認証と暗号化）
- `mountOptions: sec=krb5i`（ID保護を備えた認証と暗号化）
- `mountOptions: sec=krb5p`（アイデンティティとプライバシー保護を備えた認証と暗号化）

Kerberos レベルを 1 つだけ指定します。パラメータ リストで複数の Kerberos 暗号化レベルを指定した場合、最初のオプションのみが使用されます。ストレージ バックエンド構成で指定した暗号化レベルがストレージ クラス オブジェクトで指定したレベルと異なる場合は、ストレージ クラス オブジェクトが優先されます。

手順

1. 次の例を使用して、StorageClass Kubernetes オブジェクトを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions:
  - sec=krb5i #can be krb5, krb5i, or krb5p
parameters:
  backendType: ontap-nas
  storagePools: ontapnas_pool
  trident.netapp.io/nasType: nfs
allowVolumeExpansion: true
```

2. ストレージ クラスを作成します。

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. ストレージ クラスが作成されたことを確認します。

```
kubectl get sc ontap-nas-sc
```

次のような出力が表示されます。

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

プロビジョニングボリューム

ストレージ バックエンドとストレージ クラスを作成したら、ボリュームをプロビジョニングできるようになります。手順については、 "ボリュームをプロビジョニングする" 。

Azure NetApp Filesボリュームでインフライト Kerberos 暗号化を構成する

マネージド クラスターと単一のAzure NetApp Filesストレージ バックエンドまたはAzure NetApp Filesストレージ バックエンドの仮想プール間のストレージ トラフィックに対して Kerberos 暗号化を有効にできます。

開始する前に

- 管理対象 Red Hat OpenShift クラスターでTridentが有効になっていることを確認します。
- アクセスできることを確認してください `tridentctl`ユーティリティ。
- 要件に注意し、以下の手順に従って、Kerberos暗号化用のAzure NetApp Filesストレージバックエンドを準備したことを確認します。 "Azure NetApp Files のドキュメント" 。
- Kerberos 暗号化で使用する NFSv4 ボリュームが正しく構成されていることを確認します。 NetApp NFSv4ドメイン構成セクション（13ページ）を参照してください。 "NetApp NFSv4 の機能強化とベストプラクティスガイド" 。

ストレージバックエンドを作成する

Kerberos 暗号化機能を含むAzure NetApp Filesストレージ バックエンド構成を作成できます。

タスク概要

Kerberos 暗号化を構成するストレージ バックエンド構成ファイルを作成するときに、次の 2 つのレベルのいずれかで適用されるように定義できます。

- *ストレージバックエンドレベル*は、 `spec.kerberos`分野
- *仮想プールレベル*を使用して `spec.storage.kerberos`分野

仮想プール レベルで構成を定義する場合、ストレージ クラスのラベルを使用してプールが選択されます。

どちらのレベルでも、Kerberos 暗号化の 3 つの異なるバージョンのいずれかを指定できます。

- `kerberos: sec=krb5`（認証と暗号化）
- `kerberos: sec=krb5i`（ID保護を備えた認証と暗号化）
- `kerberos: sec=krb5p`（アイデンティティとプライバシー保護を備えた認証と暗号化）

手順

1. 管理対象クラスターで、ストレージ バックエンドを定義する必要がある場所 (ストレージ バックエンド レベルまたは仮想プール レベル) に応じて、次のいずれかの例を使用してストレージ バックエンド構成ファイルを作成します。括弧<>内の値は、環境の情報で置き換えます。

ストレージバックエンドレベルの例

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

仮想プールレベルの例

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

2. 前の手順で作成した構成ファイルを使用して、バックエンドを作成します。

```
tridentctl create backend -f <backend-configuration-file>
```

バックエンドの作成に失敗した場合、バックエンドの構成に問題があります。次のコマンドを実行すると、ログを表示して原因を特定できます。

```
tridentctl logs
```

構成ファイルの問題を特定して修正したら、create コマンドを再度実行できます。

ストレージクラスを作成する

Kerberos 暗号化を使用してボリュームをプロビジョニングするためのストレージ クラスを作成できます。

手順

1. 次の例を使用して、StorageClass Kubernetes オブジェクトを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: azure-netapp-files
  trident.netapp.io/nasType: nfs
  selector: type=encryption
```

2. ストレージ クラスを作成します。

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. ストレージ クラスが作成されたことを確認します。

```
kubectl get sc -sc-nfs
```

次のような出力が表示されます。

```
NAME          PROVISIONER             AGE
sc-nfs        csi.trident.netapp.io   15h
```

プロビジョニングボリューム

ストレージ バックエンドとストレージ クラスを作成したら、ボリュームをプロビジョニングできるようになります。手順については、 "ボリュームをプロビジョニングする" 。

著作権に関する情報

Copyright © 2026 NetApp, Inc. All Rights Reserved. Printed in the U.S.このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為（過失またはそうでない場合を含む）にかかわらず、一切の責任を負いません。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップの特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1つ以上の米国特許、その他の国の特許、および出願中の特許によって保護されている場合があります。

権利の制限について：政府による使用、複製、開示は、DFARS 252.227-7013（2014年2月）およびFAR 5252.227-19（2007年12月）のRights in Technical Data -Noncommercial Items（技術データ - 非商用品目に関する諸権利）条項の(b)(3)項、に規定された制限が適用されます。

本書に含まれるデータは商用製品および / または商用サービス（FAR 2.101の定義に基づく）に関係し、データの所有権はNetApp, Inc.にあります。本契約に基づき提供されるすべてのネットアップの技術データおよびコンピュータ ソフトウェアは、商用目的であり、私費のみで開発されたものです。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用権を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用権については、DFARS 252.227-7015(b)項（2014年2月）で定められた権利のみが認められます。

商標に関する情報

NetApp、NetAppのロゴ、http://www.netapp.com/TMに記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。