



Avec Astra Trident

Astra Trident

NetApp
November 14, 2025

Sommaire

Avec Astra Trident	1
Préparez le nœud de travail	1
Choisir les bons outils	1
Détection des services de nœud	1
Volumes NFS	1
Volumes iSCSI	2
Configuration des systèmes back-end	5
Azure NetApp Files	6
Configurer un système Cloud Volumes Service pour Google Cloud backend	18
Configurer un système NetApp HCI ou SolidFire backend	34
Configurer un système back-end avec les pilotes SAN ONTAP	41
Configurer un système NAS backend ONTAP	63
Amazon FSX pour NetApp ONTAP	89
Création de systèmes back-end avec kubectl	101
TridentBackendConfig	101
Présentation des étapes	102
Étape 1 : créez un code secret Kubernetes	103
Étape 2 : créez le TridentBackendConfig CR	104
Étape 3 : vérifier l'état du TridentBackendConfig CR	105
(Facultatif) étape 4 : pour plus de détails	105
Effectuer la gestion back-end avec kubectl	107
Supprimer un back-end	107
Affichez les systèmes back-end existants	107
Mettre à jour un back-end	107
Gestion back-end avec tridentctl	108
Créer un back-end	108
Supprimer un back-end	108
Affichez les systèmes back-end existants	109
Mettre à jour un back-end	109
Identifier les classes de stockage qui utilisent un système back-end	109
Passez d'une option de gestion back-end à une autre	110
Gérez tridentctl utilisation de systèmes back-end TridentBackendConfig	110
Gérez TridentBackendConfig utilisation de systèmes back-end tridentctl	114
Gérer les classes de stockage	116
Concevez une classe de stockage	116
Créer une classe de stockage	116
Supprimer une classe de stockage	116
Afficher les classes de stockage existantes	117
Définir une classe de stockage par défaut	117
Identifier le système back-end pour une classe de stockage	118
Réaliser des opérations de volume	118
Utiliser la topologie CSI	118

Travailler avec des instantanés	126
Développement des volumes	130
Importer des volumes	137
Partager un volume NFS entre les espaces de noms	143
Caractéristiques	143
Démarrage rapide	144
Configurer les espaces de noms source et de destination	145
Supprimer un volume partagé	146
Utiliser <code>tridentctl get</code> pour interroger des volumes subordonnés	146
Limites	147
Pour en savoir plus	147
Contrôle d'Astra Trident	147
Étape 1 : définir une cible Prometheus	148
Étape 2 : créer un ServiceMonitor Prometheus	148
Étape 3 : interroger les mesures Trident avec PromQL	149
Découvrez la télémétrie AutoSupport d'Astra Trident	150
Désactivation des metrics d'Astra Trident	151

Avec Astra Trident

Préparez le nœud de travail

Tous les nœuds workers du cluster Kubernetes doivent pouvoir monter les volumes provisionnés pour vos pods. Pour préparer les nœuds workers, vous devez installer des outils NFS ou iSCSI en fonction de votre sélection de pilotes.

Choisir les bons outils

Si vous utilisez une combinaison de pilotes, vous devez installer les outils NFS et iSCSI.

Outils NFS

Installez les outils NFS si vous utilisez : `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`

Outils iSCSI

Installez les outils iSCSI si vous utilisez : `ontap-san`, `ontap-san-economy`, `solidfire-san`



Les versions récentes de RedHat CoreOS ont installé NFS et iSCSI par défaut.

Détection des services de nœud

Astra Trident tente de détecter automatiquement si le nœud peut exécuter des services iSCSI ou NFS.



La découverte de services de nœuds identifie les services détectés, mais ne garantit pas que les services sont correctement configurés. Inversement, l'absence d'un service découvert ne garantit pas l'échec du montage du volume.

Révision des événements

Astra Trident crée des événements pour le nœud afin d'identifier les services découverts. Pour passer en revue ces événements, exécutez :

```
kubectl get event -A --field-selector involvedObject.name=<Kubernetes node name>
```

Examiner les services découverts

Astra Trident identifie les services activés pour chaque nœud du nœud Trident CR. Pour afficher les services découverts, exécutez :

```
tridentctl get node -o wide -n <Trident namespace>
```

Volumes NFS

Installez les outils NFS à l'aide des commandes de votre système d'exploitation. Assurez-vous que le service NFS est démarré pendant le démarrage.

RHEL 8+

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```



Redémarrez les nœuds workers après l'installation des outils NFS afin d'éviter toute défaillance lors de la connexion des volumes aux conteneurs.

Volumes iSCSI

Astra Trident peut établir automatiquement une session iSCSI, analyser les LUN et détecter les périphériques à chemins d'accès multiples, les formater et les monter sur un pod.

Fonctionnalités d'auto-rétablissement de l'iSCSI

Pour les systèmes ONTAP, Astra Trident exécute l'auto-rétablissement iSCSI toutes les cinq minutes pour :

1. **Identifier** l'état de session iSCSI souhaité et l'état de session iSCSI en cours.
2. **Comparer** l'état souhaité à l'état actuel pour identifier les réparations nécessaires. Astra Trident détermine les priorités de réparation et le moment auquel les réparations doivent être effectuées.
3. **Effectuez les réparations** requises pour rétablir l'état de session iSCSI actuel à l'état de session iSCSI souhaité.



Les journaux d'activité d'auto-rétablissement se trouvent dans le `trident-main` Conteneur sur le pod `DemOnset` respectif. Pour afficher les journaux, vous devez avoir défini `debug « Vrai »` lors de l'installation d'Astra Trident.

Avec les fonctionnalités d'auto-rétablissement iSCSI d'Astra Trident,

- Sessions iSCSI obsolètes ou non saines pouvant survenir après un problème de connectivité réseau. Dans le cas d'une session obsolète, Astra Trident attend sept minutes avant de se déconnecter pour rétablir la connexion avec un portail.



Par exemple, si les secrets CHAP ont tourné sur le contrôleur de stockage et que le réseau perd la connectivité, les anciens secrets CHAP (*obsolète*) pourraient persister. L'auto-rétablissement peut reconnaître ceci et rétablir automatiquement la session pour appliquer les secrets CHAP mis à jour.

- Sessions iSCSI manquantes
- LUN manquantes

Installez les outils iSCSI

Installez les outils iSCSI à l'aide des commandes de votre système d'exploitation.

Avant de commencer

- Chaque nœud du cluster Kubernetes doit avoir un IQN unique. **C'est une condition préalable nécessaire.**
- En cas d'utilisation de RHCOS version 4.5 ou ultérieure ou d'une autre distribution Linux compatible RHEL, avec le `solidfire-san` Pilote et Element OS 12.5 ou version antérieure, assurez-vous que l'algorithme d'authentification CHAP est défini sur MD5 dans `/etc/iscsi/iscsid.conf`. Les algorithmes CHAP sécurisés conformes à la norme FIPS SHA1, SHA-256 et SHA3-256 sont disponibles avec Element 12.7.

```
sudo sed -i 's/^\(node.session.auth.chap_algs\) .*/\1 = MD5/'  
/etc/iscsi/iscsid.conf
```

- Lorsque vous utilisez des nœuds workers exécutant RHEL/RedHat CoreOS avec iSCSI PVS, spécifiez le `discard` MounOption dans la classe de stockage pour effectuer la réclamation d'espace en ligne. Voir "[La documentation de Red Hat](#)".

RHEL 8+

1. Installez les packages système suivants :

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_utils device-  
mapper-multipath
```

2. Vérifiez que la version iscsi-initiator-utils est 6.2.0.874-2.el7 ou ultérieure :

```
rpm -q iscsi-initiator-utils
```

3. Définir la numérisation sur manuelle :

```
sudo sed -i 's/^\(node.session.scan\) .*/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. Activer les chemins d'accès multiples :

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```



Bien sûr `etc/multipath.conf` contient `find_multipaths no` sous `defaults`.

5. S'assurer que `iscsid` et `multipathd` sont en cours d'exécution :

```
sudo systemctl enable --now iscsid multipathd
```

6. Activer et démarrer `iscsi`:

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. Installez les packages système suivants :

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools  
scsitools
```

2. Vérifiez que la version Open-iscsi est 2.0.874-5ubuntu2.10 ou ultérieure (pour bionique) ou 2.0.874-7.1ubuntu6.1 ou ultérieure (pour focaux) :

```
dpkg -l open-iscsi
```

3. Définir la numérisation sur manuelle :

```
sudo sed -i 's/^\(node.session.scan\).*\/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. Activer les chemins d'accès multiples :

```
sudo tee /etc/multipath.conf <<-'EOF'  
defaults {  
    user_friendly_names yes  
    find_multipaths no  
}  
EOF  
sudo systemctl enable --now multipath-tools.service  
sudo service multipath-tools restart
```



Bien sûr `etc/multipath.conf` contient `find_multipaths no` sous `defaults`.

5. S'assurer que `open-iscsi` et `multipath-tools` sont activées et en cours d'exécution :

```
sudo systemctl status multipath-tools  
sudo systemctl enable --now open-iscsi.service  
sudo systemctl status open-iscsi
```



Pour Ubuntu 18.04, vous devez découvrir les ports cibles avec `iscsiadm` avant de commencer `open-iscsi` Pour que le démon iSCSI démarre. Vous pouvez également modifier le `iscsi` service à démarrer `iscsid` automatiquement.



Redémarrez les nœuds workers après l'installation des outils iSCSI pour éviter toute défaillance lors de la connexion des volumes aux conteneurs.

Configuration des systèmes back-end

Un système back-end définit la relation entre Astra Trident et un système de stockage. Il explique à Astra Trident comment communiquer avec ce système de stockage et comment Astra Trident doit provisionner des volumes à partir de celui-ci.

Astra Trident propose automatiquement des pools de stockage back-end correspondant aux exigences définies par une classe de stockage. Découvrez comment configurer le système back-end pour votre système

de stockage.

- ["Configurer un back-end Azure NetApp Files"](#)
- ["Configurer un système back-end Cloud Volumes Service pour Google Cloud Platform"](#)
- ["Configurer un système NetApp HCI ou SolidFire backend"](#)
- ["Configurer un système back-end avec des pilotes NAS ONTAP ou Cloud Volumes ONTAP"](#)
- ["Configurer un système back-end avec des pilotes ONTAP ou Cloud Volumes ONTAP SAN"](#)
- ["Utilisez Astra Trident avec Amazon FSX pour NetApp ONTAP"](#)

Azure NetApp Files

Configurer un back-end Azure NetApp Files

Vous pouvez configurer Azure NetApp Files (ANF) comme back-end pour Astra Trident. Vous pouvez relier des volumes NFS et SMB à l'aide d'un back-end ANF.

- ["Préparation"](#)
- ["Exemples et options de configuration"](#)

Considérations

- Le service Azure NetApp Files ne prend pas en charge des volumes de moins de 100 Go. Astra Trident crée automatiquement des volumes de 100 Go en cas de demande d'un volume plus petit.
- Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.
- Astra Trident ne prend pas en charge l'architecture Windows ARM.

Préparez la configuration d'un back-end Azure NetApp Files

Avant de pouvoir configurer le système back-end Azure NetApp Files, vous devez vous assurer que les exigences suivantes sont respectées.



Si vous utilisez Azure NetApp Files pour la première fois ou dans un nouvel emplacement, une configuration initiale est requise pour configurer Azure NetApp Files et créer un volume NFS. Reportez-vous à la section ["Azure : configurez Azure NetApp Files et créez un volume NFS"](#).

Prérequis pour les volumes NFS et SMB

Pour configurer et utiliser un ["Azure NetApp Files"](#) back-end, vous avez besoin des éléments suivants :

- Un pool de capacité. Reportez-vous à la section ["Microsoft : créez un pool de capacité pour Azure NetApp Files"](#).
- Sous-réseau délégué à Azure NetApp Files. Reportez-vous à la section ["Microsoft : déléguer un sous-réseau à Azure NetApp Files"](#).
- `subscriptionID` Depuis un abonnement Azure avec Azure NetApp Files activé.
- `tenantID`, `clientID`, et `clientSecret` à partir d'un ["Enregistrement d'applications"](#) Dans Azure Active Directory avec les autorisations suffisantes pour le service Azure NetApp Files. L'enregistrement de l'application doit utiliser l'une des options suivantes :

- Rôle propriétaire ou contributeur ["Prédéfinie par Azure"](#).
- A ["Rôle de contributeur personnalisé"](#) au niveau de l'abonnement (assignableScopes) Avec les autorisations suivantes qui sont limitées à ce qu'exige Astra Trident. Après avoir créé le rôle personnalisé, ["Attribuez le rôle à l'aide du portail Azure"](#).

```
{
  "id": "/subscriptions/<subscription-id>/providers/Microsoft.Authorization/roleDefinitions/<role-definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [
          "Microsoft.NetApp/netAppAccounts/capacityPools/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/write",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/write",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/delete",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/write",
          "Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/delete",

```

```

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/Get
Metadata/action",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTargets/r
ead",
        "Microsoft.Network/virtualNetworks/read",
        "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/delete",
        "Microsoft.Features/features/read",
        "Microsoft.Features/operations/read",
        "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
    ],
    "notActions": [],
    "dataActions": [],
    "notDataActions": []
  }
]
}
}

```

- Azure location qui contient au moins un "sous-réseau délégué". À partir de Trident 22.01, le location le paramètre est un champ obligatoire au niveau supérieur du fichier de configuration back-end. Les valeurs d'emplacement spécifiées dans les pools virtuels sont ignorées.

Exigences supplémentaires pour les volumes SMB

Pour créer un volume SMB, vous devez disposer des éléments suivants :

- Active Directory configuré et connecté à Azure NetApp Files. Reportez-vous à la section "[Microsoft : création et gestion des connexions Active Directory pour Azure NetApp Files](#)".
- Cluster Kubernetes avec un nœud de contrôleur Linux et au moins un nœud worker Windows exécutant Windows Server 2019. Astra Trident prend en charge les volumes SMB montés sur des pods qui

s'exécutent uniquement sur des nœuds Windows.

- Au moins un secret Astra Trident contenant vos informations d'identification Active Directory pour que Azure NetApp Files puisse s'authentifier auprès d'Active Directory. Pour générer un secret `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user  
--from-literal password='password'
```

- Un proxy CSI configuré en tant que service Windows. Pour configurer un `csi-proxy`, voir "[GitHub : proxy CSI](#)" ou "[GitHub : proxy CSI pour Windows](#)" Pour les nœuds Kubernetes s'exécutant sur Windows.

Exemples et options de configuration du back-end Azure NetApp Files

Découvrez les options de configuration du back-end NFS et SMB pour ANF et examinez les exemples de configuration.

Astra Trident utilise votre configuration back-end (sous-réseau, réseau virtuel, niveau de service et emplacement), pour créer des volumes ANF dans des pools de capacité disponibles à l'emplacement demandé et correspondre au niveau de service et au sous-réseau requis.



Astra Trident ne prend pas en charge les pools de capacité manuels de QoS.

Options de configuration du back-end

Les systèmes back-end ANF proposent ces options de configuration.

Paramètre	Description	Valeur par défaut
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	« azure-netapp-files »
<code>backendName</code>	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + caractères aléatoires
<code>subscriptionID</code>	L'ID d'abonnement de votre abonnement Azure	
<code>tenantID</code>	ID locataire d'un enregistrement d'application	
<code>clientID</code>	L'ID client d'un enregistrement d'application	
<code>clientSecret</code>	Secret client d'un enregistrement d'application	
<code>serviceLevel</code>	Un de Standard, Premium, ou Ultra	« » (aléatoire)
<code>location</code>	Nom de l'emplacement Azure dans lequel les nouveaux volumes seront créés	

Paramètre	Description	Valeur par défaut
resourceGroups	Liste des groupes de ressources pour le filtrage des ressources découvertes	« [] » (sans filtre)
netappAccounts	Liste des comptes NetApp permettant de filtrer les ressources découvertes	« [] » (sans filtre)
capacityPools	Liste des pools de capacité pour le filtrage des ressources découvertes	« [] » (sans filtre, aléatoire)
virtualNetwork	Nom d'un réseau virtuel avec un sous-réseau délégué	« »
subnet	Nom d'un sous-réseau délégué à Microsoft.Netapp/volumes	« »
networkFeatures	L'ensemble des fonctions de vnet pour un volume peut être Basic ou Standard. Les fonctions réseau ne sont pas disponibles dans toutes les régions et peuvent être activées dans un abonnement. Spécification networkFeatures lorsque la fonctionnalité n'est pas activée, le provisionnement du volume échoue.	« »
nfsMountOptions	Contrôle précis des options de montage NFS. Ignoré pour les volumes SMB. Pour monter des volumes à l'aide de NFS version 4.1, incluez nfsvers=4 Dans la liste des options de montage délimitées par des virgules, choisissez NFS v4.1. Les options de montage définies dans une définition de classe de stockage remplacent les options de montage définies dans la configuration backend.	« nfsvers=3 »
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur	« » (non appliqué par défaut)
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Exemple <code>\{"api": false, "method": true, "discovery": true\}</code> . Ne l'utilisez pas à moins que vous ne soyez en mesure de résoudre les problèmes et que vous ayez besoin d'un vidage détaillé des journaux.	nul

Paramètre	Description	Valeur par défaut
nasType	Configurez la création de volumes NFS ou SMB. Les options sont <code>nfs</code> , <code>smb</code> ou <code>nul</code> . La valeur null par défaut sur les volumes NFS.	<code>nfs</code>



Pour plus d'informations sur les fonctionnalités réseau, reportez-vous à la section "[Configurer les fonctions réseau d'un volume Azure NetApp Files](#)".

Autorisations et ressources requises

Si vous recevez une erreur "aucun pool de capacité détecté" lors de la création d'une demande de volume persistant, il est probable que votre enregistrement d'application ne dispose pas des autorisations et ressources requises (sous-réseau, réseau virtuel, pool de capacité) associées. Si le débogage est activé, Astra Trident consigne les ressources Azure découvertes lors de la création du back-end. Vérifiez que vous utilisez un rôle approprié.

Les valeurs de `resourceGroups`, `netappAccounts`, `capacityPools`, `virtualNetwork`, et `subnet` peut être spécifié à l'aide de noms courts ou complets. Les noms complets sont recommandés dans la plupart des cas, car les noms abrégés peuvent faire correspondre plusieurs ressources avec le même nom.

Le `resourceGroups`, `netappAccounts`, et `capacityPools` les valeurs sont des filtres qui limitent l'ensemble des ressources découvertes aux ressources disponibles pour ce stockage back-end et peuvent être spécifiés dans n'importe quelle combinaison. Les noms complets suivent le format suivant :

Type	Format
Groupe de ressources	<groupe de ressources>
Compte NetApp	<groupe de ressources>/<compte netapp>
Pool de capacité	<groupe de ressources>/<compte netapp>/<pool de capacité>
Réseau virtuel	<groupe de ressources>/<réseau virtuel>
Sous-réseau	<groupe de ressources>/<réseau virtuel>/<sous-réseau>

Provisionnement de volume

Vous pouvez contrôler le provisionnement de volume par défaut en spécifiant les options suivantes dans une section spéciale du fichier de configuration. Reportez-vous à la section [Exemples de configurations](#) pour plus d'informations.

Paramètre	Description	Valeur par défaut
exportRule	Règles d'exportation pour les nouveaux volumes. exportRule Doit être une liste séparée par des virgules d'une combinaison d'adresses IPv4 ou de sous-réseaux IPv4 en notation CIDR. Ignoré pour les volumes SMB.	« 0.0.0.0/0 »
snapshotDir	Contrôle la visibilité du répertoire .snapshot	« faux »
size	Taille par défaut des nouveaux volumes	« 100 G »
unixPermissions	Les autorisations unix des nouveaux volumes (4 chiffres octaux). Ignoré pour les volumes SMB.	« » (fonction d'aperçu, liste blanche requise dans l'abonnement)

Exemples de configurations

Exemple 1 : configuration minimale

Il s'agit de la configuration back-end minimale absolue. Avec cette configuration, Astra Trident détecte tous vos comptes, pools de capacité et sous-réseaux NetApp délégués à ANF à l'emplacement configuré et place les nouveaux volumes sur l'un de ces pools et sous-réseaux de manière aléatoire. Parce que `nasType` est omis, le `nfs` La valeur par défaut s'applique et le système back-end provisionne les volumes NFS.

Cette configuration est idéale pour commencer avec ANF et essayer certaines choses. Toutefois, dans la pratique, vous voulez fournir des fonctionnalités supplémentaires pour déterminer les volumes que vous provisionnez.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
```

Exemple 2 : configuration de niveau de service spécifique avec des filtres de pool de capacité

Cette configuration back-end place les volumes dans des Azure `eastus` emplacement dans un `Ultra` pool de capacité. Astra Trident détecte automatiquement tous les sous-réseaux délégués à ANF dans cet emplacement et place un nouveau volume de façon aléatoire sur l'un d'entre eux.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
- application-group-1/account-1/ultra-1
- application-group-1/account-1/ultra-2
```

Exemple 3 : configuration avancée

Cette configuration back-end réduit davantage l'étendue du placement des volumes sur un seul sous-réseau et modifie également certains paramètres par défaut du provisionnement des volumes.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
- application-group-1/account-1/ultra-1
- application-group-1/account-1/ultra-2
virtualNetwork: my-virtual-network
subnet: my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: 'true'
  size: 200Gi
  unixPermissions: '0777'
```

Exemple 4 : configuration de pool virtuel

Cette configuration back-end définit plusieurs pools de stockage dans un seul fichier. Cette fonction est utile lorsque plusieurs pools de capacité prennent en charge différents niveaux de service, et que vous souhaitez créer des classes de stockage dans Kubernetes qui les représentent. Des étiquettes de pools virtuels ont été utilisées pour différencier les pools en fonction de performance.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
- application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
- labels:
  performance: gold
  serviceLevel: Ultra
  capacityPools:
  - ultra-1
  - ultra-2
  networkFeatures: Standard
- labels:
  performance: silver
  serviceLevel: Premium
  capacityPools:
  - premium-1
- labels:
  performance: bronze
  serviceLevel: Standard
  capacityPools:
  - standard-1
  - standard-2
```

Définitions des classes de stockage

Les éléments suivants `StorageClass` les définitions font référence aux pools de stockage ci-dessus.

Exemples de définitions utilisant `parameter.selector` légale

À l'aide de `parameter.selector` vous pouvez spécifier pour chaque `StorageClass` pool virtuel utilisé pour héberger un volume. Les aspects définis dans le pool sélectionné seront définis pour le volume.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze"
allowVolumeExpansion: true
```

Exemples de définitions pour les volumes SMB

À l'aide de `nasType`, `node-stage-secret-name`, et `node-stage-secret-namespace`, Vous pouvez spécifier un volume SMB et fournir les informations d'identification Active Directory requises.

Exemple 1 : configuration de base sur l'espace de noms par défaut

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Exemple 2 : utilisation de secrets différents par espace de noms

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Exemple 3 : utilisation de différents secrets par volume

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```



`nasType: `smb` Filtres pour les pools qui prennent en charge les volumes SMB. `nasType: `nfs` ou `nasType: `null` Filtres pour pools NFS.

Créer le backend

Après avoir créé le fichier de configuration backend, exécutez la commande suivante :

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

Si la création du back-end échoue, la configuration du back-end est erronée. Vous pouvez afficher les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs
```

Après avoir identifié et corrigé le problème avec le fichier de configuration, vous pouvez exécuter de nouveau la commande `create`.

Configurer un système Cloud Volumes Service pour Google Cloud backend

Découvrez comment configurer NetApp Cloud Volumes Service pour Google Cloud en tant que backend pour votre installation d'Astra Trident à l'aide des exemples de configuration fournis.

En savoir plus sur la prise en charge d'Astra Trident pour Cloud Volumes Service pour Google Cloud

Astra Trident peut créer des volumes Cloud Volumes Service dans un des deux "[types de service](#)" :

- **CVS-Performance** : le type de service Astra Trident par défaut. Ce type de service aux performances optimisées est parfaitement adapté aux charges de travail de production qui exigent des performances élevées. Le type de service CVS-Performance est une option matérielle prenant en charge les volumes d'une taille minimale de 100 Gio. Vous pouvez choisir l'une des options "[trois niveaux de service](#)" :
 - `standard`
 - `premium`
 - `extreme`
- **CVS** : Le type de service CVS fournit une haute disponibilité zonale avec des niveaux de performance limités à modérés. Le type de service CVS est une option logicielle utilisant des pools de stockage pour prendre en charge des volumes de 1 Gio. Le pool de stockage peut contenir jusqu'à 50 volumes dans lesquels tous les volumes partagent la capacité et les performances du pool. Vous pouvez choisir l'une des options "[deux niveaux de service](#)" :
 - `standardsw`
 - `zoneredundantstandardsw`

Ce dont vous avez besoin

Pour configurer et utiliser le "[Cloud Volumes Service pour Google Cloud](#)" back-end, vous avez besoin des éléments suivants :

- Un compte Google Cloud configuré avec NetApp Cloud Volumes Service
- Numéro de projet de votre compte Google Cloud
- Compte de service Google Cloud avec le `netappcloudvolumes.admin` rôle
- Fichier de clé API pour votre compte Cloud Volumes Service

Options de configuration du back-end

Chaque back-end provisionne les volumes dans une seule région Google Cloud. Pour créer des volumes dans d'autres régions, vous pouvez définir des systèmes back-end supplémentaires.

Paramètre	Description	Valeur par défaut
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	« <code>gcp-cvs</code> »
<code>backendName</code>	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + partie de la clé API
<code>storageClass</code>	Paramètre facultatif utilisé pour spécifier le type de service CVS. Utiliser <code>software</code> Pour sélectionner le type de service CVS. Sinon, Astra Trident suppose un type de service CVS-Performance (<code>hardware</code>).	
<code>storagePools</code>	Type de service CVS uniquement. Paramètre facultatif utilisé pour spécifier les pools de stockage pour la création du volume.	
<code>projectNumber</code>	Numéro de projet de compte Google Cloud. La valeur est disponible sur la page d'accueil du portail Google Cloud.	
<code>hostProjectNumber</code>	Requis si l'utilisation d'un réseau VPC partagé. Dans ce scénario, <code>projectNumber</code> est le projet de service, et <code>hostProjectNumber</code> est le projet hôte.	
<code>apiRegion</code>	Région Google Cloud dans laquelle Astra Trident crée des volumes Cloud Volumes Service. Lors de la création de clusters Kubernetes inter-région, de volumes créés dans un <code>apiRegion</code> Peut être utilisé pour des charges de travail planifiées sur des nœuds sur plusieurs régions Google Cloud. Le trafic entre les régions coûte plus cher.	

Paramètre	Description	Valeur par défaut
apiKey	Clé API pour le compte de service Google Cloud avec le <code>netappcloudvolumes.admin</code> rôle. Il inclut le contenu au format JSON du fichier de clé privée d'un compte de service Google Cloud (copié en compte dans le fichier de configuration back-end).	
proxyURL	URL proxy si le serveur proxy doit se connecter au compte CVS. Le serveur proxy peut être un proxy HTTP ou HTTPS. Pour un proxy HTTPS, la validation du certificat est ignorée pour permettre l'utilisation de certificats auto-signés dans le serveur proxy. Les serveurs proxy avec authentification activée ne sont pas pris en charge.	
nfsMountOptions	Contrôle précis des options de montage NFS.	« nfsvers=3 »
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur.	« » (non appliqué par défaut)
serviceLevel	Niveau de service CVS-Performance ou CVS pour les nouveaux volumes. Les valeurs CVS-Performance sont <code>standard</code> , <code>premium</code> , ou <code>extreme</code> . Les valeurs CVS sont <code>standardsw</code> ou <code>zoneredundantstandardsw</code> .	CVS-Performance par défaut est « standard ». CVS default est "standardsw".
network	Réseau Google Cloud utilisé pour les volumes Cloud Volumes Service.	« par défaut »
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Exemple <code>\{"api":false, "method":true}</code> . Ne l'utilisez pas à moins que vous ne soyez en mesure de résoudre les problèmes et que vous ayez besoin d'un vidage détaillé des journaux.	nul

Paramètre	Description	Valeur par défaut
<code>allowedTopologies</code>	Pour activer l'accès inter-région, votre définition de classe de stockage pour <code>allowedTopologies</code> doit inclure toutes les régions. Par exemple : <ul style="list-style-type: none"> - key: <code>topology.kubernetes.io/region</code> values: <ul style="list-style-type: none"> - <code>us-east1</code> - <code>europa-west1</code> 	

Options de provisionnement de volumes

Vous pouvez contrôler le provisionnement de volume par défaut dans le `defaults` section du fichier de configuration.

Paramètre	Description	Valeur par défaut
<code>exportRule</code>	Règles d'exportation pour les nouveaux volumes. Doit être une liste séparée par des virgules d'une combinaison d'adresses IPv4 ou de sous-réseaux IPv4 en notation CIDR.	« 0.0.0.0/0 »
<code>snapshotDir</code>	Accès au <code>.snapshot</code> répertoire	« faux »
<code>snapshotReserve</code>	Pourcentage de volume réservé pour les snapshots	« » (Accepter CVS par défaut de 0)
<code>size</code>	La taille des nouveaux volumes. CVS-Performance minimum est de 100 Gio. CVS est au minimum de 1 Gio.	Le type de service CVS-Performance utilise par défaut « 100 Gio ». Le type de service CVS n'est pas défini par défaut mais nécessite au moins 1 Gio.

Exemples de type de service CVS-Performance

Les exemples suivants fournissent des exemples de configuration pour le type de service CVS-Performance.


```
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```



```
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```



```

znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
XsYg6gyxy4zq7OlwWgLwGa==
-----END PRIVATE KEY-----
client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
client_id: '123456789012345678901'
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
  region: us-west2
storage:
- labels:
  performance: extreme
  protection: extra
  serviceLevel: extreme
  defaults:
  snapshotDir: 'true'
  snapshotReserve: '10'
  exportRule: 10.0.0.0/24
- labels:
  performance: extreme
  protection: standard
  serviceLevel: extreme
- labels:
  performance: premium
  protection: extra
  serviceLevel: premium
  defaults:
  snapshotDir: 'true'
  snapshotReserve: '10'
- labels:
  performance: premium
  protection: standard
  serviceLevel: premium
- labels:
  performance: standard

```

```
serviceLevel: standard
```

Définitions des classes de stockage

Les définitions de classe de stockage suivantes s'appliquent à l'exemple de configuration de pool virtuel. À l'aide de `parameters.selector`, Vous pouvez spécifier pour chaque classe de stockage le pool virtuel utilisé pour héberger un volume. Les aspects définis dans le pool sélectionné seront définis pour le volume.

Exemple de classe de stockage

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: netapp.io/trident
parameters:
  selector: "performance=standard"
allowVolumeExpansion: true
```

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "protection=extra"
allowVolumeExpansion: true
```

- La première classe de stockage (`cvs-extreme-extra-protection`) correspond au premier pool virtuel. Il s'agit du seul pool offrant des performances extrêmes avec une réserve Snapshot de 10 %.
- La dernière classe de stockage (`cvs-extra-protection`) appelle tout pool de stockage qui fournit une réserve d'instantanés de 10%. Astra Trident décide du pool virtuel sélectionné et s'assure que les exigences de la réserve de snapshots sont respectées.

Exemples de type de service CVS

Les exemples suivants fournissent des exemples de configuration pour le type de service CVS.


```
client_id: '123456789012345678901'  
auth_uri: https://accounts.google.com/o/oauth2/auth  
token_uri: https://oauth2.googleapis.com/token  
auth_provider_x509_cert_url:  
https://www.googleapis.com/oauth2/v1/certs  
client_x509_cert_url:  
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-  
sa%40my-gcp-project.iam.gserviceaccount.com  
serviceLevel: standardsw
```

Exemple 2 : configuration du pool de stockage

Cet exemple de configuration back-end utilise `storagePools` pour configurer un pool de stockage.

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    MIIIEvAIBADANBgkqhkiG9w0BAQEFAASCbKwggSiAgEAAoIBAQDaT+Oui9FBAw19
    L1AGEkrYU5xd9K5NlO5jMkIFND5wCD+Nv+jd1GvtFRLaLK5RvXyF5wzvztmODNS+
    qtScpQ+5cFpQkuGtv9U9+N6qtuVYYO3b504Kp5CtqVPJCgMJaK2j8pZTIqUiMum/
    5/Y9oTbZrjAHSMsgJm2nHzFq2X0rqVmaHghI6ATm4DOuWx8XGWKGTGIPlc0qPqJlqS
    LLaWOH4VIZQZCAyW5IUp9CAmwqHgdG0uhFNfCgMmED6PBUvVLsLvcq86X+QSWR9k
    ETqElj/sGCenPF7ti1DhGBFafd9hPnxg9PZY29ArEZwY9G/ZjZQX7WPgs0VvxiNR
    DxZRC3GXAgMBAAECggEACn5c59bG/qnVEVI1CwMAalM5M2z09JFh1L1ljKwntNPj
    Vilw2eTW2+UE7HbJru/S7KQgA5Dnn9kvCraEahPRuddUMrD0vG4kTl/IODV6uFuk
    Y0sZfbqd4jMUQ21smvGsqFzwloYWS5qzO1W83ivXH/HW/iqkmY2eW+EPRS/hwSSu
    SscR+Soji7PB0BWSJhlV4yqYf3vcd/D95el2CVHfRCkL85DKumeZ+yHEnpiXGZAE
    t8xSs4a500Pm6NHhevCw2a/UQ95/foXNUR450HtbjieJo5o+FF6EYZQGfU2ZHZO8
    37FBKuaJkdGW5xqaI9TL7aqkGkFMF4F2qvOZM+vy8QKBgQD4oVuOkJDlhkTHP86W
    esFlw1kpWyJR9ZA7LI0g/rVpslnX+XdDq0WQf4umdLNau5hYEH9LU6ZSGs1Xk3/B
    NHwR6OXFuqEKNiu83d0zSlHhTy7PZpOZdj5a/vVvQfPDMz7OvsqLRd7YCAbdzuQ0
    +Ahq0Ztwvg0HQ64hdW0ukpYRRwKBgQDgyHj98oqsw0YuIa+pP1yS0pPwLmjwKyNm
    /HayzCp+Qjiiyy7Tzg8AUqlH1Ou83XbV428jvg7kDh07PCCKFq+mMmfqHmTpb0Maq
    KpKnZg4ipsqPlyHNNEOrmcailXbwIhCLewMqMrggUiLOmCw4PscL5nK+4GKu2XE1
    jLqjWAZFMQKBgFHkQ9XXRAJ1kR3XpGHoGN890pZ0kCVSrqju6aUef/5KY1FCt8ew
    F/+aIxM2iQsvmWQYOvVCnhuY/F2GfAQ7d0om3decuwI0CX/xy7PjHMkLXa2uaZs4
    WR17sLduj62RqXRLX0c0QkwBiNFyHbRcpdkZJQujbyMhBa+7j7SxT4BtAoGAWMWT
    UucocRXZm/pdvz9wteNH3YDwnJLMxm1KC06qMXbBoYrliY4sm3ywJWMC+iCd/H8A
    Gecxd/xVu5mA2L2N3KMq18Zhz8Th0G5DwKyDRJgOQ0Q46yuNXOoYEjlo4Wjyk8Me
    +tlQ8iK98E0UmZnhTgfSpSNElbz2AqnzQ3MN9uECgYAqdvdpVnKGFvdtZ2DjyMoJ
    E89UIC41WjjJGmHsd8W65+3X0RwMzKMT6aZc5tK9J5dHvmWIETnbM+1TImdbBFga
    NWOC6f3r2xbGXHhaWS1+nobpTuvlo56ZRJVvVk7lFMsiddzMuHH8pxfgNJemwA4P
    ThDHcejv035NNV6Kyo00tA==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
  data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
```

```
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw
```

Et la suite ?

Après avoir créé le fichier de configuration backend, exécutez la commande suivante :

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

Si la création du back-end échoue, la configuration du back-end est erronée. Vous pouvez afficher les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs
```

Après avoir identifié et corrigé le problème avec le fichier de configuration, vous pouvez exécuter de nouveau la commande create.

Configurer un système NetApp HCI ou SolidFire backend

Découvrez comment créer et utiliser un système Element backend avec votre installation d'Astra Trident.

Ce dont vous avez besoin

- Système de stockage pris en charge exécutant le logiciel Element.
- Identifiants de locataire ou administrateur de cluster NetApp HCI/SolidFire pouvant gérer les volumes
- Tous vos nœuds workers Kubernetes doivent avoir installé les outils iSCSI appropriés. Voir "[informations de préparation du nœud de travail](#)".

Ce que vous devez savoir

Le `solidfire-san` le pilote de stockage prend en charge les deux modes de volume : fichier et bloc. Pour le `Filesystem` En mode volume, Astra Trident crée un volume et crée un système de fichiers. Le type de système de fichiers est spécifié par la classe de stockage.

Conducteur	Protocole	Mode Volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
solidfire-san	ISCSI	Bloc	RWO,ROX,RWX	Aucun système de fichiers. Périphérique de bloc brut.
solidfire-san	ISCSI	Bloc	RWO,ROX,RWX	Aucun système de fichiers. Périphérique de bloc brut.
solidfire-san	ISCSI	Système de fichiers	RWO,ROX	xf _s , ext3, ext4
solidfire-san	ISCSI	Système de fichiers	RWO,ROX	xf _s , ext3, ext4



Astra Trident utilise le protocole CHAP lorsqu'il fonctionne comme un mécanisme de provisionnement CSI amélioré. Si vous utilisez CHAP (qui est la valeur par défaut pour CSI), aucune autre préparation n'est requise. Il est recommandé de définir explicitement le `UseCHAP` Possibilité d'utiliser CHAP avec Trident non CSI. Sinon, voir "[ici](#)".



Les groupes d'accès aux volumes sont uniquement pris en charge par le framework classique non CSI pour Astra Trident. Lorsqu'il est configuré pour fonctionner en mode CSI, Astra Trident utilise le protocole CHAP.

Si aucun de ces deux cas `AccessGroups` ou `UseCHAP` sont définies, l'une des règles suivantes s'applique :

- Si la valeur par défaut `trident` groupe d'accès détecté, groupes d'accès utilisés.
- Si aucun groupe d'accès n'est détecté et que la version de Kubernetes est 1.7 ou ultérieure, CHAP est utilisé.

Options de configuration du back-end

Voir le tableau suivant pour les options de configuration du back-end :

Paramètre	Description	Valeur par défaut
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	Toujours « solidfire-san ».
<code>backendName</code>	Nom personnalisé ou système back-end de stockage	"SolidFire_" + adresse IP de stockage (iSCSI)
<code>Endpoint</code>	MVIP pour le cluster SolidFire avec les identifiants de locataire	
<code>SVIP</code>	Port et adresse IP de stockage (iSCSI)	

Paramètre	Description	Valeur par défaut
labels	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes.	« »
TenantName	Nom du locataire à utiliser (créé si introuvable)	
InitiatorIFace	Limitez le trafic iSCSI à une interface hôte spécifique	« par défaut »
UseCHAP	Utilisez CHAP pour authentifier iSCSI	vrai
AccessGroups	Liste des ID de groupes d'accès à utiliser	Recherche l'ID d'un groupe d'accès nommé « trident »
Types	Spécifications de QoS	
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur	« » (non appliqué par défaut)
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Exemple, {"api":false, "méthode":true}	nul



Ne pas utiliser `debugTraceFlags` à moins que vous ne soyez en mesure de dépanner et que vous ayez besoin d'un vidage détaillé des journaux.

Exemple 1 : configuration back-end pour `solidfire-san` avec trois types de volume

Cet exemple montre un fichier back-end utilisant l'authentification CHAP et la modélisation de trois types de volumes avec des garanties de QoS spécifiques. Il est fort probable que vous définiriez ensuite des classes de stockage pour consommer chacune de ces catégories à l'aide de l' `IOPS` paramètre de classe de stockage.

```

---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: "<svip>:3260"
TenantName: "<tenant>"
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000

```

Exemple 2 : configuration du back-end et de la classe de stockage pour `solidfire-san` pilote avec pools virtuels

Cet exemple représente le fichier de définition du back-end configuré avec des pools virtuels ainsi que des classes de stockage qui les renvoient.

Astra Trident copie les étiquettes présentes sur un pool de stockage vers le LUN de stockage back-end lors du provisionnement. Pour plus de commodité, les administrateurs du stockage peuvent définir des étiquettes par pool virtuel et les volumes de groupe par étiquette.

Dans l'exemple de fichier de définition de back-end illustré ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, qui définissent le `type` Du niveau Silver. Les pools virtuels sont définis dans le `storage` section. Dans cet exemple, certains pools de stockage définissent leur propre type et certains pools remplacent les valeurs par défaut définies ci-dessus.

```

---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0

```

```

SVIP: "<svip>:3260"
TenantName: "<tenant>"
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
- labels:
  performance: gold
  cost: '4'
  zone: us-east-1a
  type: Gold
- labels:
  performance: silver
  cost: '3'
  zone: us-east-1b
  type: Silver
- labels:
  performance: bronze
  cost: '2'
  zone: us-east-1c
  type: Bronze
- labels:
  performance: silver
  cost: '1'
  zone: us-east-1d

```

Les définitions de classe de stockage suivantes font référence aux pools virtuels ci-dessus. À l'aide du

`parameters.selector` Chaque classe de stockage indique quel(s) pool(s) virtuel(s) peut(s) être utilisé(s) pour héberger un volume. Les aspects définis dans le pool virtuel sélectionné seront définis pour le volume.

La première classe de stockage (`solidfire-gold-four`) sera mappé sur le premier pool virtuel. Il s'agit du seul pool offrant des performances Gold avec un `Volume Type QoS` De l'or. La dernière classe de stockage (`solidfire-silver`) appelle n'importe quel pool de stockage qui offre une performance silver. Astra Trident va décider du pool virtuel sélectionné et s'assurer que les besoins en stockage sont satisfaits.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold; cost=4"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=3"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze; cost=2"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=1"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
  fsType: "ext4"
```

Trouvez plus d'informations

- ["Groupes d'accès de volume"](#)

Configurer un système back-end avec les pilotes SAN ONTAP

Découvrez comment configurer un back-end ONTAP avec les pilotes ONTAP et Cloud Volumes ONTAP SAN.

- ["Préparation"](#)
- ["Configuration et exemples"](#)

Astra Control assure une protection, une reprise d'activité et une mobilité transparentes (en déplaçant des volumes entre les clusters Kubernetes) pour les volumes créés avec le système `ontap-nas`, `ontap-nas-flexgroup`, et `ontap-san` pilotes. Voir ["Conditions préalables à la réplication d'Astra Control"](#) pour plus d'informations.



- Vous devez utiliser `ontap-nas` adapté aux charges de travail de production qui nécessitent une protection des données, une reprise d'activité et la mobilité.
- Utiliser `ontap-san-economy` Lorsque vous prévoyez une utilisation de volume, celle-ci devrait être bien supérieure à celle prise en charge par ONTAP.
- Utiliser `ontap-nas-economy` Ce n'est que lorsque l'utilisation prévue des volumes sera beaucoup plus élevée que ce que prend en charge ONTAP, et le `ontap-san-economy` le pilote ne peut pas être utilisé.
- Ne pas utiliser `ontap-nas-economy` si vous prévoyez d'avoir besoin en termes de protection des données, de reprise sur incident ou de mobilité.

Autorisations utilisateur

Astra Trident devrait être exécuté en tant qu'administrateur de ONTAP ou du SVM, généralement à l'aide du `admin` utilisateur du cluster ou un `vsadmin` Utilisateur d'un SVM ou un utilisateur avec un autre nom qui a le même rôle. Pour les déploiements Amazon FSX pour NetApp ONTAP, Astra Trident devrait être exécuté en tant qu'administrateur ONTAP ou SVM, à l'aide du cluster `fsxadmin` utilisateur ou un `vsadmin` Utilisateur d'un SVM ou un utilisateur avec un autre nom qui a le même rôle. Le `fsxadmin` l'utilisateur remplace limitée l'utilisateur administrateur du cluster.



Si vous utilisez le `limitAggregateUsage` paramètre, des autorisations d'administration du cluster sont requises. Avec Amazon FSX pour NetApp ONTAP avec Astra Trident, le `limitAggregateUsage` le paramètre ne fonctionne pas avec le `vsadmin` et `fsxadmin` comptes d'utilisateur. L'opération de configuration échoue si vous spécifiez ce paramètre.

S'il est possible de créer un rôle plus restrictif au sein de ONTAP qu'un pilote Trident peut utiliser, nous ne le recommandons pas. La plupart des nouvelles versions de Trident appellent des API supplémentaires qui devront être prises en compte, ce qui complique les mises à niveau et risque d'erreurs.

Préparez la configuration du système back-end avec les pilotes SAN ONTAP

Découvrez comment vous préparer à configurer un système ONTAP backend avec les pilotes SAN ONTAP. Pour tous les systèmes back-end ONTAP, Astra Trident requiert au moins un agrégat affecté à la SVM.

N'oubliez pas que vous pouvez également exécuter plusieurs pilotes et créer des classes de stockage qui pointent vers l'un ou l'autre. Par exemple, vous pouvez configurer un `san-dev` classe qui utilise le `ontap-`

san conducteur et a `san-default` classe qui utilise le `ontap-san-economy` une seule.

Tous vos nœuds workers Kubernetes doivent avoir installé les outils iSCSI appropriés. Voir "[ici](#)" pour en savoir plus.

Authentification

Astra Trident propose deux modes d'authentification d'un système back-end ONTAP.

- Basé sur les informations d'identification : nom d'utilisateur et mot de passe pour un utilisateur ONTAP disposant des autorisations requises. Il est recommandé d'utiliser un rôle de connexion de sécurité prédéfini, par exemple `admin` ou `vsadmin`. Pour garantir une compatibilité maximale avec les versions ONTAP.
- Basé sur des certificats : Astra Trident peut également communiquer avec un cluster ONTAP à l'aide d'un certificat installé sur le système back-end. Dans ce cas, la définition backend doit contenir des valeurs encodées Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance, le cas échéant (recommandé).

Vous pouvez mettre à jour les systèmes back-end existants pour passer d'une méthode basée sur les identifiants à une méthode basée sur les certificats. Toutefois, une seule méthode d'authentification est prise en charge à la fois. Pour passer à une méthode d'authentification différente, vous devez supprimer la méthode existante de la configuration backend.



Si vous tentez de fournir **les deux identifiants et les certificats**, la création du back-end échoue avec une erreur indiquant que plus d'une méthode d'authentification a été fournie dans le fichier de configuration.

Activer l'authentification basée sur les informations d'identification

Astra Trident nécessite les identifiants d'un administrateur SVM-scoped/cluster-scoped pour communiquer avec le ONTAP backend. Il est recommandé d'utiliser des rôles standard prédéfinis tels que `admin` ou `vsadmin`. Il est ainsi possible d'assurer une compatibilité avec les futures versions d'ONTAP et d'exposer les API de fonctionnalités à utiliser avec les futures versions d'Astra Trident. Un rôle de connexion de sécurité personnalisé peut être créé et utilisé avec Astra Trident, mais il n'est pas recommandé.

Voici un exemple de définition du back-end :

YAML

```
Version: 1 backendName: ExempleBackend storageDriverName: ontap-san managementLIF: 10.0.0.1
svm: svm_nfs username: Vsadmin password: Password
```

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Gardez à l'esprit que la définition du back-end est le seul endroit où les informations d'identification sont stockées en texte brut. Une fois le système backend créé, les noms d'utilisateur/mots de passe sont codés avec Base64 et stockés sous forme de secrets Kubernetes. La création ou la mise à jour d'un back-end est la seule étape qui nécessite la connaissance des informations d'identification. Il s'agit donc d'une opération uniquement administrative, qui doit être effectuée par l'administrateur Kubernetes/du stockage.

Activez l'authentification basée sur les certificats

Les systèmes back-end, nouveaux et existants, peuvent utiliser un certificat et communiquer avec le système back-end ONTAP. Trois paramètres sont requis dans la définition du back-end.

- `ClientCertificate` : valeur encodée en Base64 du certificat client.
- `ClientPrivateKey` : valeur encodée en Base64 de la clé privée associée.
- `TrustedCACertificate` : valeur encodée Base64 du certificat CA de confiance. Si vous utilisez une autorité de certification approuvée, ce paramètre doit être fourni. Ceci peut être ignoré si aucune autorité de certification approuvée n'est utilisée.

Un flux de travail type comprend les étapes suivantes.

Étapes

1. Générez un certificat client et une clé. Lors de la génération, définissez le nom commun (CN) sur l'utilisateur ONTAP pour qu'il s'authentifie.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. Ajoutez un certificat d'autorité de certification de confiance au cluster ONTAP. Il se peut déjà que l'administrateur de stockage gère cet espace. Ignorer si aucune autorité de certification approuvée n'est utilisée.

```
security certificate install -type server -cert-name <trusted-ca-cert-name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca <cert-authority>
```

3. Installez le certificat client et la clé (à partir de l'étape 1) sur le cluster ONTAP.

```
security certificate install -type client-ca -cert-name <certificate-name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Vérifiez que le rôle de connexion de sécurité ONTAP est pris en charge cert methode d'authentification.

```
security login create -user-or-group-name admin -application ontapi -authentication-method cert
security login create -user-or-group-name admin -application http -authentication-method cert
```

5. Testez l'authentification à l'aide d'un certificat généré. Remplacer <ONTAP Management LIF> et <vserver name> par Management LIF IP et SVM name.

```
curl -X POST -Lk https://<ONTAP-Management-LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key --cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp xmlns="http://www.netapp.com/filer/admin" version="1.21" vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Encodez le certificat, la clé et le certificat CA de confiance avec Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Créez le back-end à l'aide des valeurs obtenues à partir de l'étape précédente.

```

cat cert-backend.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "SanBackend",
  "managementLIF": "1.2.3.4",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuueeeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "trustedCACertificate": "QNFinfO...SiqOyN",
  "storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |           UUID           |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |         0 |
+-----+-----+-----+-----+
+-----+-----+

```

Mettre à jour les méthodes d'authentification ou faire pivoter les informations d'identification

Vous pouvez mettre à jour un back-end existant pour utiliser une méthode d'authentification différente ou pour faire pivoter leurs informations d'identification. Cela fonctionne de deux manières : les systèmes back-end qui utilisent le nom d'utilisateur/mot de passe peuvent être mis à jour pour utiliser des certificats ; les systèmes back-end qui utilisent des certificats peuvent être mis à jour en fonction du nom d'utilisateur/mot de passe. Pour ce faire, vous devez supprimer la méthode d'authentification existante et ajouter la nouvelle méthode d'authentification. Utilisez ensuite le fichier backend.json mis à jour contenant les paramètres requis à exécuter `tridentctl backend update`.

```

cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |          UUID          |
STATE | VOLUMES |
+-----+-----+-----+
+-----+-----+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |      9 |
+-----+-----+-----+
+-----+-----+

```



Lors de la rotation des mots de passe, l'administrateur du stockage doit d'abord mettre à jour le mot de passe de l'utilisateur sur ONTAP. Cette opération est suivie d'une mise à jour du back-end. Lors de la rotation de certificats, plusieurs certificats peuvent être ajoutés à l'utilisateur. Le back-end est ensuite mis à jour pour utiliser le nouveau certificat, en suivant lequel l'ancien certificat peut être supprimé du cluster ONTAP.

La mise à jour d'un back-end n'interrompt pas l'accès aux volumes qui ont déjà été créés, et n'a aucun impact sur les connexions de volume effectuées après. Une mise à jour réussie indique qu'Astra Trident peut communiquer avec le système back-end ONTAP et gérer les opérations de volumes à venir.

Spécifiez les igroups

Astra Trident utilise des igroups pour contrôler l'accès aux volumes (LUN) qu'il provisionne. Dans le cas de la spécification des igroups pour un système back-end, les administrateurs ont deux options :

- Astra Trident peut créer et gérer automatiquement un groupe initiateur par système back-end. Si `igroupName` n'est pas inclus dans la définition du système back-end, Astra Trident crée un groupe initiateur nommé `trident-<backend-UUID>` sur le SVM. Cela permet de s'assurer que chaque système back-end dispose d'un groupe initiateur dédié et de gérer l'ajout/la suppression automatiques d'IQN de nœud Kubernetes.
- Alternativement, les igroups pré-crées peuvent être fournis dans une définition de back-end. Pour ce faire,

utilisez le `igroupName` paramètre config. Astra Trident ajoute/supprime des IQN de nœud Kubernetes au groupe initiateur préexistant.

Pour les systèmes back-end dont ils ont besoin `igroupName` défini, le `igroupName` peut être supprimé avec un `tridentctl backend update` Pour bénéficier des igrups à manipulation automatique avec Astra Trident. L'accès aux volumes déjà rattachés aux charges de travail ne sera pas perturbé. Les futures connexions seront gérées à l'aide du groupe initiateur Astra Trident.



Dédier un groupe initiateur à chaque instance unique d'Astra Trident est une bonne pratique bénéfique pour l'administrateur Kubernetes et l'administrateur du stockage. CSI Trident automatise l'ajout et la suppression des IQN du nœud du cluster au groupe initiateur, ce qui simplifie considérablement sa gestion. Lorsque vous utilisez le même SVM sur tous les environnements Kubernetes (et avec des installations Trident d'Astra), un groupe initiateur dédié permet de s'assurer que les modifications apportées à un cluster Kubernetes n'influencent pas les groupes initiateurs associés à un autre. En outre, il est important de s'assurer que chaque nœud du cluster Kubernetes dispose d'un IQN unique. Comme mentionné ci-dessus, Astra Trident s'occupe automatiquement de l'ajout et de la suppression des IQN. La réutilisation d'IQN sur des hôtes peut entraîner des scénarios indésirables où les hôtes se confondent les uns avec les autres et où l'accès aux LUN est refusé.

Si Astra Trident est configuré pour fonctionner comme un provisionnement CSI, les IQN du nœud Kubernetes sont automatiquement ajoutés ou supprimés du groupe initiateur. Lorsque des nœuds sont ajoutés à un cluster Kubernetes, `trident-csi` DemonSet déploie un pod (`trident-csi-xxxxx` dans les versions antérieures à 23.01 ou `trident-node<operating system>-xxxx` dans 23.01 et versions ultérieures) sur les nouveaux nœuds ajoutés et enregistre les nouveaux nœuds sur lesquels il peut attacher des volumes. Les IQN du nœud sont également ajoutés au groupe initiateur du back-end. Un ensemble d'étapes similaire gère la suppression des IQN lorsque le(s) nœud(s) est cordeleted, drainé et supprimé de Kubernetes.

Si Astra Trident ne s'exécute pas comme un provisionnement CSI, le groupe initiateur doit être mis à jour manuellement pour contenir les IQN iSCSI de chaque nœud worker du cluster Kubernetes. Les IQN des nœuds qui rejoignent le cluster Kubernetes devront être ajoutés au groupe initiateur. De même, les IQN des nœuds qui sont supprimés du cluster Kubernetes doivent être supprimés du groupe initiateur.

Authentifier les connexions avec le protocole CHAP bidirectionnel

Astra Trident peut authentifier les sessions iSCSI avec le protocole CHAP bidirectionnel pour le `ontap-san` et `ontap-san-economy` pilotes. Pour cela, il faut activer `useCHAP` dans votre définition backend. Lorsqu'il est réglé sur `true`, Astra Trident configure la sécurité de l'initiateur par défaut du SVM en CHAP bidirectionnel et définit le nom d'utilisateur et les secrets du fichier backend. NetApp recommande d'utiliser le protocole CHAP bidirectionnel pour l'authentification des connexions. Voir l'exemple de configuration suivant :

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
igroupName: trident
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
```



Le `useCHAP` Paramètre est une option booléenne qui ne peut être configurée qu'une seule fois. Elle est définie sur `FALSE` par défaut. Une fois la valeur `true` définie, vous ne pouvez pas la définir sur `false`.

En plus de `useCHAP=true`, le `chapInitiatorSecret`, `chapTargetInitiatorSecret`, `chapTargetUsername`, et `chapUsername` les champs doivent être inclus dans la définition back-end. Les secrets peuvent être modifiés après la création d'un back-end en cours d'exécution `tridentctl update`.

Comment cela fonctionne

Par réglage `useCHAP` À vrai dire, l'administrateur du stockage demande à Astra Trident de configurer le protocole CHAP sur le système back-end. Ceci inclut les éléments suivants :

- Configuration du protocole CHAP sur le SVM :
 - Si le type de sécurité de l'initiateur par défaut du SVM n'est pas défini (défini par défaut) **et** il n'y a pas de LUN préexistantes dans le volume, Astra Trident définit le type de sécurité par défaut sur `CHAP` Et procédez à la configuration de l'initiateur CHAP et du nom d'utilisateur cible et des secrets.
 - Si le SVM contient des LUN, Astra Trident n'active pas le protocole CHAP sur le SVM. Cela permet de garantir que l'accès aux LUN déjà présentes sur le SVM n'est pas restreint.
- Configuration de l'initiateur CHAP et du nom d'utilisateur cible et des secrets ; ces options doivent être spécifiées dans la configuration backend (comme indiqué ci-dessus).
- Gestion de l'ajout d'initiateurs au système `igroupName` donné en arrière-plan. Si ce n'est pas spécifié, la valeur par défaut est `trident`.

Une fois le système back-end créé, Astra Trident crée un correspondant `tridentbackend` CRD et stocke les secrets et noms d'utilisateur CHAP sous forme de secrets Kubernetes. Tous les volumes persistants créés par Astra Trident sur ce back-end seront montés et rattachés au protocole CHAP.

Rotation des identifiants et mise à jour des systèmes back-end

Vous pouvez mettre à jour les informations d'identification CHAP en mettant à jour les paramètres CHAP dans le `backend.json` fichier. Cela nécessitera la mise à jour des secrets CHAP et l'utilisation de `tridentctl`

update pour refléter ces modifications.



Lors de la mise à jour des secrets CHAP pour un back-end, vous devez utiliser `tridentctl` pour mettre à jour le backend. Ne mettez pas à jour les identifiants du cluster de stockage via l'interface de ligne de commande/ONTAP car Astra Trident ne pourra pas détecter ces modifications.

```
cat backend-san.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "ontap_san_chap",
  "managementLIF": "192.168.0.135",
  "svm": "ontap_iscsi_svm",
  "useCHAP": true,
  "username": "vsadmin",
  "password": "password",
  "igroupName": "trident",
  "chapInitiatorSecret": "cl9qxUpDaTeD",
  "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
  "chapTargetUsername": "iJF4heBRT0TCwxyz",
  "chapUsername": "uh2aNCLSD6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|  NAME          | STORAGE DRIVER |          UUID          |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffb5c |
online |       7 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Les connexions existantes ne seront pas affectées. Elles restent actives si les identifiants sont mis à jour par Astra Trident sur le SVM. Les nouvelles connexions utiliseront les informations d'identification mises à jour et les connexions existantes continuent de rester actives. La déconnexion et la reconnexion des anciens volumes persistants se traduiront par l'utilisation des identifiants mis à jour.

Options et exemples de configuration des SAN ONTAP

Découvrez comment créer et utiliser des pilotes SAN ONTAP avec votre installation d'Astra Trident. Cette section présente des exemples de configuration du back-end et des détails sur le mappage des systèmes back-end aux classes de stockage.

Options de configuration du back-end

Voir le tableau suivant pour les options de configuration du back-end :

Paramètre	Description	Valeur par défaut
version		Toujours 1
storageDriverName	Nom du pilote de stockage	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san », « ontap-san », « ontap-économie san »
backendName	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + dataLIF
managementLIF	Adresse IP d'un cluster ou d'une LIF de gestion SVM pour un basculement MetroCluster transparent, vous devez spécifier une LIF de gestion SVM. Un nom de domaine complet (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide du <code>--use-ipv6</code> drapeau. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	« 10.0.0.1 », « [2001:1234:abcd::fefe] »
dataLIF	Adresse IP de la LIF de protocole. Ne pas spécifier pour iSCSI. utilisations d'Astra Trident "Mappage de LUN sélectif ONTAP" Pour découvrir les LIFs iSCSI nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si dataLIF est explicitement défini.	Dérivé par la SVM
useCHAP	Utilisez CHAP pour authentifier iSCSI pour les pilotes SAN ONTAP [Boolean]. Réglez sur <code>true</code> Pour qu'Astra Trident configure et utilise le protocole CHAP bidirectionnel comme authentification par défaut pour la SVM donnée en back-end. Reportez-vous à la section "Préparez la configuration du système back-end avec les pilotes SAN ONTAP" pour plus d'informations.	faux
chapInitiatorSecret	Secret de l'initiateur CHAP. Requis si <code>useCHAP=true</code>	« »

Paramètre	Description	Valeur par défaut
labels	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »
chapTargetInitiatorSecret	Secret de l'initiateur cible CHAP. Requis si useCHAP=true	« »
chapUsername	Nom d'utilisateur entrant. Requis si useCHAP=true	« »
chapTargetUsername	Nom d'utilisateur cible. Requis si useCHAP=true	« »
clientCertificate	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
clientPrivateKey	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »
trustedCACertificate	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification basée sur des certificats.	« »
username	Le nom d'utilisateur doit communiquer avec le cluster ONTAP. Utilisé pour l'authentification basée sur les identifiants.	« »
password	Mot de passe requis pour communiquer avec le cluster ONTAP. Utilisé pour l'authentification basée sur les identifiants.	« »
svm	Serveur virtuel de stockage à utiliser	Dérivé d'un SVM managementLIF est spécifié
igroupName	Nom du groupe initiateur à utiliser pour les volumes SAN. Reportez-vous à la section pour en savoir plus.	Trident-<backend-UUID>
storagePrefix	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être modifié ultérieurement. Pour mettre à jour ce paramètre, vous devez créer un nouveau backend.	trident

Paramètre	Description	Valeur par défaut
limitAggregateUsage	Echec du provisionnement si l'utilisation est supérieure à ce pourcentage. Si vous utilisez un système Amazon FSX pour le système back-end NetApp ONTAP, ne spécifiez pas limitAggregateUsage. Le fourni fsxadmin et vsadmin Ne contiennent pas les autorisations requises pour récupérer l'utilisation d'agrégats et le limiter à l'aide d'Astra Trident.	« » (non appliqué par défaut)
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN.	« » (non appliqué par défaut)
lunsPerFlexvol	Nombre maximal de LUN par FlexVol, doit être compris dans la plage [50, 200]	"100"
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Par exemple, {"api":false, "méthode":true} ne pas utiliser sauf si vous effectuez un dépannage et que vous avez besoin d'un vidage détaillé des journaux.	nul

Paramètre	Description	Valeur par défaut
useREST	<p>Paramètre booléen pour utiliser les API REST de ONTAP. Aperçu technique</p> <p>useREST est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur <code>true</code>, Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au <code>ontap client</code> supplémentaire. Ceci est satisfait par le pré-défini <code>vsadmin</code> et <code>cluster-admin</code> rôles.</p> <p>useREST N'est pas pris en charge par MetroCluster.</p>	faux

Détails sur `igroupName`

`igroupName` Peut être défini sur un groupe initiateur déjà créé sur le cluster ONTAP. Si non spécifié, Astra Trident crée automatiquement un groupe initiateur nommé `trident-<backend-UUID>`.

Si vous disposez d'un nom de groupe prédéfini, nous vous recommandons d'utiliser un groupe initiateur par cluster Kubernetes si le SVM doit être partagé entre les environnements. Cela est nécessaire pour qu'Astra Trident conserve automatiquement les ajouts et suppressions d'IQN.

- `igroupName` Peut être mis à jour afin de désigner un nouveau groupe initiateur créé et géré sur la SVM en dehors d'Astra Trident.
- `igroupName` peut être omis. Dans ce cas, Astra Trident crée et gère un groupe initiateur nommé `trident-<backend-UUID>` automatiquement.

Dans les deux cas, les pièces jointes de volume continueront d'être accessibles. Les pièces jointes futures utilisent le groupe initiateur mis à jour. Cette mise à jour n'interrompt pas l'accès aux volumes présents sur le back-end.

Options de configuration back-end pour les volumes de provisionnement

Vous pouvez contrôler le provisionnement par défaut à l'aide de ces options dans `defaults` section de la configuration. Pour un exemple, voir les exemples de configuration ci-dessous.

Paramètre	Description	Valeur par défaut
<code>spaceAllocation</code>	Allocation d'espace pour les LUN	« vrai »

Paramètre	Description	Valeur par défaut
spaceReserve	Mode de réservation d'espace ; "none" (fin) ou "volume" (épais)	« aucun »
snapshotPolicy	Règle Snapshot à utiliser	« aucun »
qosPolicy	QoS policy group à affecter pour les volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end. Avec Astra Trident, les groupes de règles de QoS doivent être utilisés avec ONTAP 9.8 ou version ultérieure. Nous recommandons l'utilisation d'un groupe de règles de qualité de service non partagé et nous assurer que le groupe de règles est appliqué à chaque composant individuellement. Un groupe de règles de QoS partagé appliquera le plafond du débit total de toutes les charges de travail.	« »
adaptiveQosPolicy	Groupe de règles de QoS adaptative à attribuer aux volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end	« »
snapshotReserve	Pourcentage du volume réservé pour les instantanés "0"	Si snapshotPolicy est « aucun », sinon « »
splitOnClone	Séparer un clone de son parent lors de sa création	« faux »
encryption	Activez NetApp Volume Encryption (NVE) sur le nouveau volume. La valeur par défaut est <code>false</code> . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE" .	« faux »
luksEncryption	Activez le cryptage LUKS. Reportez-vous à la section "Utiliser la configuration de clé unifiée Linux (LUKS)" .	« »
securityStyle	Style de sécurité pour les nouveaux volumes	unix

Paramètre	Description	Valeur par défaut
tieringPolicy	La stratégie de hiérarchisation à utiliser « none »	Snapshot uniquement pour une configuration SVM-DR pré-ONTAP 9.5

Exemples de provisionnement de volumes

Voici un exemple avec des valeurs par défaut définies :

```

---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: password
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
igroupName: custom
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'

```



Pour tous les volumes créés à l'aide de `ontap-san` Avec d'autres pilotes, Astra Trident ajoute une capacité supplémentaire de 10 % au système FlexVol pour prendre en charge les métadonnées de LUN. La LUN sera provisionnée avec la taille exacte que l'utilisateur demande dans la demande de volume persistant. Astra Trident ajoute 10 % au système FlexVol (dont la taille disponible dans ONTAP). Les utilisateurs obtiennent à présent la capacité utilisable requise. Cette modification empêche également que les LUN ne soient en lecture seule, à moins que l'espace disponible soit pleinement utilisé. Cela ne s'applique pas à l'économie d'`ontap-san`.

Pour les systèmes back-end définis `snapshotReserve`, Astra Trident calcule la taille des volumes comme suit :

```

Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1

```

Le modèle 1.1 est le modèle 10 % d'Astra Trident supplémentaire qui s'ajoute à la baie FlexVol pour prendre en charge les métadonnées de la LUN. Pour `snapshotReserve = 5 %` et demande de volume persistant = 5 Gio, la taille totale du volume est de 5,7 Gio et la taille disponible est de 5,5 Gio. Le `volume show` la commande doit afficher des résultats similaires à cet exemple :

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
		_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4	online	RW	10GB	5.00GB	0%
		_pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d	online	RW	5.79GB	5.50GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

3 entries were displayed.

Actuellement, le redimensionnement est le seul moyen d'utiliser le nouveau calcul pour un volume existant.

Exemples de configuration minimaux

Les exemples suivants montrent des configurations de base qui laissent la plupart des paramètres par défaut. C'est la façon la plus simple de définir un back-end.



Si vous utilisez Amazon FSX sur NetApp ONTAP avec Astra Trident, il est recommandé de spécifier des noms DNS pour les LIF au lieu d'adresses IP.

ontap-san pilote avec authentification par certificat

Il s'agit d'un exemple de configuration back-end minimal. `clientCertificate`, `clientPrivateKey`, et `trustedCACertificate` (Facultatif, si vous utilisez une autorité de certification approuvée) est renseigné `backend.json` Et prendre les valeurs codées en base64 du certificat client, de la clé privée et du certificat CA de confiance, respectivement.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
igroupName: trident
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

ontap-san **Pilote avec CHAP bidirectionnel**

Il s'agit d'un exemple de configuration back-end minimal. Cette configuration de base crée un `ontap-san` backend avec `useCHAP` réglé sur `true`.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
igroupName: trident
username: vsadmin
password: password
```

ontap-san-economy **conducteur**

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
igroupName: trident
username: vsadmin
password: password
```

Exemples de systèmes back-end avec pools virtuels

Dans l'exemple de fichier de définition backend ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, par exemple `spaceReserve` aucune, `spaceAllocation` lors de la fausse idée, et `encryption` faux. Les pools virtuels sont définis dans la section `stockage`.

Astra Trident définit les étiquettes de provisionnement dans le champ « Commentaires ». Les commentaires sont définis sur le FlexVol. Astra Trident copie toutes les étiquettes présentes sur un pool virtuel vers le volume

de stockage lors du provisionnement. Pour plus de commodité, les administrateurs du stockage peuvent définir des étiquettes par pool virtuel et les volumes de groupe par étiquette.

Dans cet exemple, certains pools de stockage sont propriétaires de leur propre pool `spaceReserve`, `spaceAllocation`, et `encryption` les valeurs et certains pools remplacent les valeurs par défaut définies ci-dessus.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
igroupName: trident
username: vsadmin
password: password
defaults:
  spaceAllocation: 'false'
  encryption: 'false'
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  protection: gold
  creditpoints: '40000'
  zone: us_east_1a
  defaults:
    spaceAllocation: 'true'
    encryption: 'true'
    adaptiveQosPolicy: adaptive-extreme
- labels:
  protection: silver
  creditpoints: '20000'
  zone: us_east_1b
  defaults:
    spaceAllocation: 'false'
    encryption: 'true'
    qosPolicy: premium
- labels:
  protection: bronze
  creditpoints: '5000'
  zone: us_east_1c
  defaults:
    spaceAllocation: 'true'
    encryption: 'false'
```

Voici un exemple iSCSI pour le ontap-san-economy pilote :

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
igroupName: trident
username: vsadmin
password: password
defaults:
  spaceAllocation: 'false'
  encryption: 'false'
labels:
  store: san_economy_store
region: us_east_1
storage:
- labels:
  app: oracledb
  cost: '30'
  zone: us_east_1a
  defaults:
    spaceAllocation: 'true'
    encryption: 'true'
- labels:
  app: postgresdb
  cost: '20'
  zone: us_east_1b
  defaults:
    spaceAllocation: 'false'
    encryption: 'true'
- labels:
  app: mysqldb
  cost: '10'
  zone: us_east_1c
  defaults:
    spaceAllocation: 'true'
    encryption: 'false'
```

Mappage des systèmes back-end aux classes de stockage

Les définitions de classe de stockage suivantes font référence aux pools virtuels ci-dessus. À l'aide du `parameters.selector` Chaque classe de stockage indique quel(s) pool(s) virtuel(s) peut(s) être utilisé(s) pour héberger un volume. Les aspects définis dans le pool virtuel sélectionné seront définis pour le volume.

- La première classe de stockage (`protection-gold`) sera mappé sur le premier, deuxième pool virtuel dans le `ontap-nas-flexgroup` système back-end et le premier pool virtuel dans le `ontap-san` back-end. Il s'agit du seul pool offrant une protection de niveau Gold.
- La deuxième classe de stockage (`protection-not-gold`) sera mappé sur le troisième, quatrième pool virtuel dans `ontap-nas-flexgroup` back-end et le deuxième, troisième pool virtuel dans `ontap-san` back-end. Ce sont les seuls pools offrant un niveau de protection autre que l'or.
- La troisième classe de stockage (`app-mysqldb`) sera mappé sur le quatrième pool virtuel dans `ontap-nas` back-end et le troisième pool virtuel dans `ontap-san-economy` back-end. Ce sont les seuls pools offrant une configuration de pool de stockage pour l'application de type `mysqldb`.
- La quatrième classe de stockage (`protection-silver-creditpoints-20k`) sera mappé sur le troisième pool virtuel dans `ontap-nas-flexgroup` back-end et le second pool virtuel dans `ontap-san` back-end. Ce sont les seules piscines offrant une protection de niveau or à 20000 points de solvabilité.
- La cinquième classe de stockage (`creditpoints-5k`) sera mappé sur le second pool virtuel dans `ontap-nas-economy` back-end et le troisième pool virtuel dans `ontap-san` back-end. Ce sont les seules offres de piscine à 5000 points de solvabilité.

Astra Trident va décider du pool virtuel sélectionné et s'assurer que les besoins en stockage sont satisfaits.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: netapp.io/trident
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"

```

Configurer un système NAS backend ONTAP

Découvrez comment configurer un back-end ONTAP avec les pilotes ONTAP et NAS Cloud Volumes ONTAP.

- ["Préparation"](#)
- ["Configuration et exemples"](#)

Astra Control assure une protection, une reprise d'activité et une mobilité transparentes (en déplaçant des volumes entre les clusters Kubernetes) pour les volumes créés avec le système `ontap-nas`, `ontap-nas-flexgroup`, et `ontap-san` pilotes. Voir ["Conditions préalables à la réplication d'Astra Control"](#) pour plus d'informations.



- Vous devez utiliser `ontap-nas` adapté aux charges de travail de production qui nécessitent une protection des données, une reprise d'activité et la mobilité.
- Utiliser `ontap-san-economy` Lorsque vous prévoyez une utilisation de volume, celle-ci devrait être bien supérieure à celle prise en charge par ONTAP.
- Utiliser `ontap-nas-economy` Ce n'est que lorsque l'utilisation prévue des volumes sera beaucoup plus élevée que ce que prend en charge ONTAP, et le `ontap-san-economy` le pilote ne peut pas être utilisé.
- Ne pas utiliser `ontap-nas-economy` si vous prévoyez d'avoir besoin en termes de protection des données, de reprise sur incident ou de mobilité.

Autorisations utilisateur

Astra Trident devrait être exécuté en tant qu'administrateur de ONTAP ou du SVM, généralement à l'aide du `admin` utilisateur du cluster ou un `vsadmin` Utilisateur d'un SVM ou un utilisateur avec un autre nom qui a le même rôle. Pour les déploiements Amazon FSX pour NetApp ONTAP, Astra Trident devrait être exécuté en tant qu'administrateur ONTAP ou SVM, à l'aide du cluster `fsxadmin` utilisateur ou un `vsadmin` Utilisateur d'un SVM ou un utilisateur avec un autre nom qui a le même rôle. Le `fsxadmin` l'utilisateur remplace limitée l'utilisateur administrateur du cluster.



Si vous utilisez le `limitAggregateUsage` paramètre, des autorisations d'administration du cluster sont requises. Avec Amazon FSX pour NetApp ONTAP avec Astra Trident, le `limitAggregateUsage` le paramètre ne fonctionne pas avec le `vsadmin` et `fsxadmin` comptes d'utilisateur. L'opération de configuration échoue si vous spécifiez ce paramètre.

S'il est possible de créer un rôle plus restrictif au sein de ONTAP qu'un pilote Trident peut utiliser, nous ne le recommandons pas. La plupart des nouvelles versions de Trident appellent des API supplémentaires qui devront être prises en compte, ce qui complique les mises à niveau et risque d'erreurs.

Préparez la configuration d'un système back-end avec les pilotes NAS ONTAP

Découvrez comment vous préparer à configurer un back-end ONTAP avec les pilotes NAS ONTAP. Pour tous les systèmes back-end ONTAP, Astra Trident requiert au moins un agrégat affecté à la SVM.

Pour tous les systèmes back-end ONTAP, Astra Trident requiert au moins un agrégat affecté à la SVM.

N'oubliez pas que vous pouvez également exécuter plusieurs pilotes et créer des classes de stockage qui pointent vers l'un ou l'autre. Par exemple, vous pouvez configurer une classe Gold qui utilise le `ontap-nas` Pilote et une classe Bronze qui utilise le `ontap-nas-economy` une seule.

Tous vos nœuds workers Kubernetes doivent avoir installé les outils NFS appropriés. Voir "[ici](#)" pour en savoir plus.

Authentification

Astra Trident propose deux modes d'authentification d'un système back-end ONTAP.

- Basé sur les informations d'identification : nom d'utilisateur et mot de passe pour un utilisateur ONTAP disposant des autorisations requises. Il est recommandé d'utiliser un rôle de connexion de sécurité prédéfini, par exemple `admin` ou `vsadmin`. Pour garantir une compatibilité maximale avec les versions ONTAP.
- Basé sur des certificats : Astra Trident peut également communiquer avec un cluster ONTAP à l'aide d'un certificat installé sur le système back-end. Dans ce cas, la définition backend doit contenir des valeurs encodées Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance, le cas échéant (recommandé).

Vous pouvez mettre à jour les systèmes back-end existants pour passer d'une méthode basée sur les identifiants à une méthode basée sur les certificats. Toutefois, une seule méthode d'authentification est prise en charge à la fois. Pour passer à une méthode d'authentification différente, vous devez supprimer la méthode existante de la configuration backend.



Si vous tentez de fournir **les deux identifiants et les certificats**, la création du back-end échoue avec une erreur indiquant que plus d'une méthode d'authentification a été fournie dans le fichier de configuration.

Activer l'authentification basée sur les informations d'identification

Astra Trident nécessite les identifiants d'un administrateur SVM-scoped/cluster-scoped pour communiquer avec le ONTAP backend. Il est recommandé d'utiliser des rôles standard prédéfinis tels que `admin` ou `vsadmin`. Il est ainsi possible d'assurer une compatibilité avec les futures versions d'ONTAP et d'exposer les API de fonctionnalités à utiliser avec les futures versions d'Astra Trident. Un rôle de connexion de sécurité personnalisé peut être créé et utilisé avec Astra Trident, mais il n'est pas recommandé.

Voici un exemple de définition du back-end :

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Gardez à l'esprit que la définition du back-end est le seul endroit où les informations d'identification sont stockées en texte brut. Une fois le système backend créé, les noms d'utilisateur/mots de passe sont codés avec Base64 et stockés sous forme de secrets Kubernetes. La création/la conversion d'un back-end est la seule étape qui nécessite la connaissance des informations d'identification. Il s'agit donc d'une opération uniquement administrative, qui doit être effectuée par l'administrateur Kubernetes/du stockage.

Activez l'authentification basée sur les certificats

Les systèmes back-end, nouveaux et existants, peuvent utiliser un certificat et communiquer avec le système back-end ONTAP. Trois paramètres sont requis dans la définition du back-end.

- ClientCertificate : valeur encodée en Base64 du certificat client.
- ClientPrivateKey : valeur encodée en Base64 de la clé privée associée.
- TrustedCACertificate : valeur encodée Base64 du certificat CA de confiance. Si vous utilisez une autorité de certification approuvée, ce paramètre doit être fourni. Ceci peut être ignoré si aucune autorité de certification approuvée n'est utilisée.

Un flux de travail type comprend les étapes suivantes.

Étapes

1. Générez un certificat client et une clé. Lors de la génération, définissez le nom commun (CN) sur

l'utilisateur ONTAP pour qu'il s'authentifie.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key  
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Ajoutez un certificat d'autorité de certification de confiance au cluster ONTAP. Il se peut déjà que l'administrateur de stockage gère cet espace. Ignorer si aucune autorité de certification approuvée n'est utilisée.

```
security certificate install -type server -cert-name <trusted-ca-cert-  
name> -vserver <vserver-name>  
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled  
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca  
<cert-authority>
```

3. Installez le certificat client et la clé (à partir de l'étape 1) sur le cluster ONTAP.

```
security certificate install -type client-ca -cert-name <certificate-  
name> -vserver <vserver-name>  
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Vérifiez que le rôle de connexion de sécurité ONTAP est pris en charge cert methode d'authentification.

```
security login create -user-or-group-name vsadmin -application ontapi  
-authentication-method cert -vserver <vserver-name>  
security login create -user-or-group-name vsadmin -application http  
-authentication-method cert -vserver <vserver-name>
```

5. Testez l'authentification à l'aide d'un certificat généré. Remplacer <ONTAP Management LIF> et <vserver name> par Management LIF IP et SVM name. Vous devez vous assurer que le LIF a sa politique de service définie sur default-data-management.

```
curl -X POST -Lk https://<ONTAP-Management-  
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key  
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp  
xmlns="http://www.netapp.com/filer/admin" version="1.21"  
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Encodez le certificat, la clé et le certificat CA de confiance avec Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Créez le back-end à l'aide des valeurs obtenues à partir de l'étape précédente.

```
cat cert-backend-updated.json
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "NasBackend",
  "managementLIF": "1.2.3.4",
  "dataLIF": "1.2.3.8",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuueeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |          UUID          |
STATE | VOLUMES |
+-----+-----+-----+
+-----+-----+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |      9 |
+-----+-----+-----+
+-----+-----+

```

Mettre à jour les méthodes d'authentification ou faire pivoter les informations d'identification

Vous pouvez mettre à jour un back-end existant pour utiliser une méthode d'authentification différente ou pour faire pivoter leurs informations d'identification. Cela fonctionne de deux manières : les systèmes back-end qui utilisent le nom d'utilisateur/mot de passe peuvent être mis à jour pour utiliser des certificats ; les systèmes back-end qui utilisent des certificats peuvent être mis à jour en fonction du nom d'utilisateur/mot de passe. Pour ce faire, vous devez supprimer la méthode d'authentification existante et ajouter la nouvelle méthode d'authentification. Utilisez ensuite le fichier backend.json mis à jour contenant les paramètres requis à exécuter `tridentctl update backend`.

```

cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+-----+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |           UUID           |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |          9 |
+-----+-----+-----+-----+
+-----+-----+

```



Lors de la rotation des mots de passe, l'administrateur du stockage doit d'abord mettre à jour le mot de passe de l'utilisateur sur ONTAP. Cette opération est suivie d'une mise à jour du back-end. Lors de la rotation de certificats, plusieurs certificats peuvent être ajoutés à l'utilisateur. Le back-end est ensuite mis à jour pour utiliser le nouveau certificat, en suivant lequel l'ancien certificat peut être supprimé du cluster ONTAP.

La mise à jour d'un back-end n'interrompt pas l'accès aux volumes qui ont déjà été créés, et n'a aucun impact sur les connexions de volume effectuées après. Une mise à jour réussie indique qu'Astra Trident peut communiquer avec le système back-end ONTAP et gérer les opérations de volumes à venir.

Gestion des règles d'exportation NFS

Astra Trident utilise les règles d'exportation NFS pour contrôler l'accès aux volumes qu'il provisionne.

Astra Trident propose deux options pour l'utilisation des règles d'exportation :

- Astra Trident peut gérer la règle d'exportation de manière dynamique. Dans ce mode de fonctionnement, l'administrateur du stockage spécifie une liste de blocs CIDR qui représentent les adresses IP admissibles. Astra Trident ajoute automatiquement des adresses IP de nœud qui font partie de ces plages à la règle d'exportation. En outre, lorsqu'aucun CIDRS n'est spécifié, toute adresse IP unicast globale trouvée sur les nœuds est ajoutée à la règle d'exportation.

- Les administrateurs du stockage peuvent créer une export-policy et ajouter des règles manuellement. Astra Trident utilise la export policy par défaut, sauf si un nom différent de export policy est spécifié dans la configuration.

Gérez les règles d'exportation de manière dynamique

La version 20.04 de CSI Trident permet de gérer de manière dynamique les règles d'exportation pour les systèmes back-end ONTAP. Cela permet à l'administrateur du stockage de spécifier un espace d'adresse autorisé pour les adresses IP du nœud de travail, au lieu de définir manuellement des règles explicites. Il simplifie considérablement la gestion des export policy ; les modifications apportées à l'export policy ne nécessitent plus d'intervention manuelle sur le cluster de stockage. De plus, cela permet de limiter l'accès au cluster de stockage uniquement aux nœuds workers dont les adresses IP sont comprises dans la plage spécifiée, ce qui prend en charge une gestion automatisée et précise.



La gestion dynamique des règles d'exportation n'est disponible que pour CSI Trident. Il est important de s'assurer que les nœuds de travail ne sont pas NATed.

Exemple

Deux options de configuration doivent être utilisées. Voici un exemple de définition du backend :

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
- 192.168.0.0/24
autoExportPolicy: true
```



Lorsque vous utilisez cette fonctionnalité, vous devez vous assurer que la jonction root dans votre SVM possède une export policy précédemment créée avec une règle d'exportation qui autorise le bloc CIDR (comme la export policy par défaut) du nœud. Suivez toujours la meilleure pratique recommandée par NetApp pour dédier un SVM à Astra Trident.

Voici une explication du fonctionnement de cette fonction à l'aide de l'exemple ci-dessus :

- `autoExportPolicy` est défini sur `true`. Cela signifie qu'Astra Trident va créer une export policy pour le `svm1` SVM et gère l'ajout et la suppression de règles à l'aide de `autoExportCIDRs` blocs d'adresse. Par exemple, un backend avec UUID `403b5326-8482-40db-96d0-d83fb3f4daec` et `autoExportPolicy` réglé sur `true` crée une export-policy nommée `trident-403b5326-8482-40db-96d0-d83fb3f4daec` Sur le SVM.
- `autoExportCIDRs` contient une liste de blocs d'adresses. Ce champ est facultatif et il prend par défaut la valeur `["0.0.0.0/0", "*/0"]`. S'il n'est pas défini, Astra Trident ajoute toutes les adresses de diffusion individuelle à périmètre global présentes sur les nœuds du worker.

Dans cet exemple, le 192.168.0.0/24 l'espace d'adressage est fourni. Cela indique que les adresses IP des nœuds Kubernetes qui appartiennent à cette plage d'adresse seront ajoutées à la règle d'exportation créée par Astra Trident. Lorsque Astra Trident enregistre un nœud sur lequel il s'exécute, il récupère les adresses IP du nœud et les vérifie par rapport aux blocs d'adresse fournis dans `autoExportCIDRs`. Après avoir filtré les adresses IP, Astra Trident crée des règles de politique d'exportation pour les adresses IP clientes qu'il détecte, avec une règle pour chaque nœud qu'il identifie.

Vous pouvez mettre à jour `autoExportPolicy` et `autoExportCIDRs` pour les systèmes back-end après leur création. Vous pouvez ajouter de nouveaux rapports CIDR pour un back-end qui est géré automatiquement ou supprimé des rapports CIDR existants. Faites preuve de prudence lors de la suppression des CIDR pour vous assurer que les connexions existantes ne sont pas tombées. Vous pouvez également choisir de désactiver `autoExportPolicy` pour un back-end et revenir à une export policy créée manuellement. Pour ce faire, vous devrez définir le `exportPolicy` dans votre configuration backend.

Après la création ou la mise à jour d'Astra Trident, vous pouvez vérifier le système back-end à l'aide de `tridentctl` ou le correspondant `tridentbackend` CRD :

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileType: ext4
```

Lorsque des nœuds sont ajoutés à un cluster Kubernetes et enregistrés avec le contrôleur Trident Astra, les règles d'exportation des systèmes back-end existants sont mises à jour (à condition qu'elles tombent dans la plage d'adresse spécifiée dans la `autoExportCIDRs` pour le back-end).

Lorsqu'un nœud est retiré, Astra Trident vérifie tous les systèmes back-end en ligne afin de supprimer la règle d'accès du nœud. En supprimant cette IP de nœud des règles d'exportation des systèmes back-end gérés, Astra Trident empêche les montages erratiques, à moins que cette adresse IP soit réutilisée par un nouveau nœud du cluster.

Pour les systèmes back-end existants, mise à jour du système back-end avec `tridentctl update backend` S'assure qu'Astra Trident gère automatiquement les règles d'exportation. Cela créera une nouvelle

export policy nommée après l'UUID et les volumes du backend qui sont présents sur le back-end, utilisera la export policy nouvellement créée lorsqu'ils sont de nouveau montés.



La suppression d'un back-end avec des règles d'exportation gérées automatiquement supprimera l'export policy créée de manière dynamique. Si le back-end est recréés, il est traité comme un nouveau backend et entraîne la création d'une nouvelle export policy.

Si l'adresse IP d'un nœud actif est mise à jour, vous devez redémarrer le pod Astra Trident sur le nœud. Astra Trident va ensuite mettre à jour la règle d'exportation pour les systèmes back-end qu'il gère pour tenir compte de ce changement d'IP.

Options et exemples de configuration du NAS ONTAP

Découvrez comment créer et utiliser des pilotes NAS ONTAP avec votre installation d'Astra Trident. Cette section présente des exemples de configuration du back-end et des détails sur le mappage des systèmes back-end aux classes de stockage.

Options de configuration du back-end

Voir le tableau suivant pour les options de configuration du back-end :

Paramètre	Description	Valeur par défaut
version		Toujours 1
storageDriverName	Nom du pilote de stockage	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san », « ontap-san », « ontap-économie san »
backendName	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + dataLIF
managementLIF	Adresse IP d'un cluster ou d'une LIF de gestion SVM pour un basculement MetroCluster transparent, vous devez spécifier une LIF de gestion SVM. Un nom de domaine complet (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide du <code>--use-ipv6</code> drapeau. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	« 10.0.0.1 », « [2001:1234:abcd::fefe] »

Paramètre	Description	Valeur par défaut
<code>dataLIF</code>	Adresse IP de la LIF de protocole. Nous vous recommandons de spécifier <code>dataLIF</code> . Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. Peut être modifié après le réglage initial. Reportez-vous à la section . Peut être configuré pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide du <code>--use-ipv6</code> drapeau. Les adresses IPv6 doivent être définies entre crochets, telles que <code>[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]</code> .	Adresse spécifiée ou dérivée d'un SVM, si non spécifiée (non recommandé)
<code>autoExportPolicy</code>	Activer la création et la mise à jour automatiques des règles d'exportation [booléennes]. À l'aide du <code>autoExportPolicy</code> et <code>autoExportCIDRs</code> Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	faux
<code>autoExportCIDRs</code>	Liste des CIDR pour filtrer les adresses IP du nœud Kubernetes par rapport à quand <code>autoExportPolicy</code> est activé. À l'aide du <code>autoExportPolicy</code> et <code>autoExportCIDRs</code> Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	<code>["0.0.0.0/0", "::/0"]</code>
<code>labels</code>	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »
<code>clientCertificate</code>	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
<code>clientPrivateKey</code>	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »

Paramètre	Description	Valeur par défaut
trustedCACertificate	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification par certificat	« »
username	Nom d'utilisateur pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants	
password	Mot de passe pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants	
svm	Serveur virtuel de stockage à utiliser	Dérivé d'un SVM managementLIF est spécifié
storagePrefix	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être mis à jour une fois que vous l'avez défini	trident
limitAggregateUsage	Echec du provisionnement si l'utilisation est supérieure à ce pourcentage. Ne s'applique pas à Amazon FSX pour ONTAP	« » (non appliqué par défaut)
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur.	« » (non appliqué par défaut)
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN, et la qtreesPerFlexvol L'option permet de personnaliser le nombre maximal de qtree par FlexVol.	« » (non appliqué par défaut)
lunsPerFlexvol	Nombre maximal de LUN par FlexVol, doit être compris dans la plage [50, 200]	"100"
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Par exemple, {"api":false, "méthode":true} ne pas utiliser debugTraceFlags à moins que vous ne soyez en mesure de dépanner et que vous ayez besoin d'un vidage détaillé des journaux.	nul

Paramètre	Description	Valeur par défaut
<code>nfsMountOptions</code>	Liste des options de montage NFS séparée par des virgules. Les options de montage des volumes Kubernetes persistants sont généralement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Astra Trident utilisera les options de montage spécifiées dans le fichier de configuration du système back-end. Si aucune option de montage n'est spécifiée dans la classe de stockage ou le fichier de configuration, Astra Trident ne définit aucune option de montage sur un volume persistant associé.	« »
<code>qtreesPerFlexvol</code>	Nombre maximal de qtrees par FlexVol, qui doit être compris dans la plage [50, 300]	"200"
<code>useREST</code>	Paramètre booléen pour utiliser les API REST de ONTAP. Aperçu technique <code>useREST</code> est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur <code>true</code> , Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au <code>ontap client</code> supplémentaire. Ceci est satisfait par le pré-défini <code>vsadmin</code> et <code>cluster-admin</code> rôles. <code>useREST</code> N'est pas pris en charge par MetroCluster.	faux

Options de configuration back-end pour les volumes de provisionnement

Vous pouvez contrôler le provisionnement par défaut à l'aide de ces options dans `defaults` section de la configuration. Pour un exemple, voir les exemples de configuration ci-dessous.

Paramètre	Description	Valeur par défaut
spaceAllocation	Allocation d'espace pour les LUN	« vrai »
spaceReserve	Mode de réservation d'espace ; "none" (fin) ou "volume" (épais)	« aucun »
snapshotPolicy	Règle Snapshot à utiliser	« aucun »
qosPolicy	QoS policy group à affecter pour les volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end	« »
adaptiveQosPolicy	Groupe de règles de QoS adaptative à attribuer aux volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end. Non pris en charge par l'économie ontap-nas.	« »
snapshotReserve	Pourcentage du volume réservé pour les instantanés "0"	Si snapshotPolicy est « aucun », sinon « »
splitOnClone	Séparer un clone de son parent lors de sa création	« faux »
encryption	Activez NetApp Volume Encryption (NVE) sur le nouveau volume. La valeur par défaut est <code>false</code> . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : " Fonctionnement d'Astra Trident avec NVE et NAE ".	« faux »
tieringPolicy	La stratégie de hiérarchisation à utiliser « none »	Snapshot uniquement pour une configuration SVM-DR pré-ONTAP 9.5
unixPermissions	Mode pour les nouveaux volumes	"777" pour volumes NFS ; vide (non applicable) pour volumes SMB
snapshotDir	Contrôle la visibilité du <code>.snapshot</code> répertoire	« faux »
exportPolicy	Export policy à utiliser	« par défaut »
securityStyle	Style de sécurité pour les nouveaux volumes. Prise en charge de NFS <code>mixed</code> et <code>unix</code> styles de sécurité. SMB prend en charge <code>mixed</code> et <code>ntfs</code> styles de sécurité.	NFS par défaut est <code>unix</code> . SMB par défaut est <code>ntfs</code> .



Avec Astra Trident, les groupes de règles de QoS doivent être utilisés avec ONTAP 9.8 ou version ultérieure. Il est recommandé d'utiliser un groupe de règles de qualité de service non partagé et de s'assurer que le groupe de règles est appliqué à chaque composant individuellement. Un groupe de règles de QoS partagé appliquera le plafond du débit total de toutes les charges de travail.

Exemples de provisionnement de volumes

Voici un exemple avec des valeurs par défaut définies :

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: password
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: '10'
```

Pour `ontap-nas` et `ontap-nas-flexgroups`, Astra Trident utilise maintenant un nouveau calcul pour s'assurer que la FlexVol est correctement dimensionnée avec le pourcentage de snapshots et la demande de volume persistant. Lorsque l'utilisateur demande de volume persistant, Astra Trident crée le FlexVol d'origine avec plus d'espace en utilisant le nouveau calcul. Ce calcul garantit que l'utilisateur reçoit l'espace inscriptible demandé dans la demande de volume persistant et qu'il ne dispose pas d'un espace minimal par rapport à ce qu'il a demandé. Avant le 21.07, lorsque l'utilisateur demande une demande de volume persistant (par exemple, 5 Gio), et le `snapshotReserve` à 50 %, ils ne bénéficient que d'un espace inscriptible de 2,5 Gio. En effet, le nom d'utilisateur requis correspond à l'intégralité du volume et `snapshotReserve` représente un pourcentage de cela. Avec Trident 21.07, il s'agit de l'espace inscriptible demandé par l'utilisateur et d'Astra Trident définit le `snapshotReserve` nombre comme pourcentage de l'intégralité du volume. Cela ne s'applique pas à `ontap-nas-economy`. Voir l'exemple suivant pour voir comment cela fonctionne :

Le calcul est le suivant :

```
Total volume size = (PVC requested size) / (1 - (snapshotReserve
percentage) / 100)
```

Pour les snapshots Reserve = 50 %, et demande en volume PVC = 5 Gio, la taille totale du volume est 2/0,5 = 10 Gio et la taille disponible est de 5 Gio, ce que l'utilisateur a demandé dans la demande de demande de volume persistant. Le `volume show` la commande doit afficher des résultats similaires à cet exemple :

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
	_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4		online	RW	10GB	5.00GB	0%
	_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba		online	RW	1GB	511.8MB	0%

2 entries were displayed.

Les systèmes back-end des installations précédentes provisionnent les volumes comme expliqué ci-dessus lors de la mise à niveau d'Astra Trident. Pour les volumes que vous avez créés avant la mise à niveau, vous devez redimensionner leurs volumes afin que la modification puisse être observée. Par exemple, un PVC de 2 Gio avec `snapshotReserve=50` Auparavant, un volume doté d'un espace inscriptible de 1 Gio. Le redimensionnement du volume à 3 Gio, par exemple, fournit l'application avec 3 Gio d'espace inscriptible sur un volume de 6 Gio.

Exemples

Exemples de configuration minimaux

Les exemples suivants montrent des configurations de base qui laissent la plupart des paramètres par défaut. C'est la façon la plus simple de définir un back-end.



Si vous utilisez Amazon FSX sur NetApp ONTAP avec Trident, nous vous recommandons de spécifier des noms DNS pour les LIF au lieu d'adresses IP.

Options par défaut activées `ontap-nas-economy`

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Authentification basée sur des certificats

Il s'agit d'un exemple de configuration back-end minimal. `clientCertificate`, `clientPrivateKey`, et `trustedCACertificate` (Facultatif, si vous utilisez une autorité de certification approuvée) est renseigné `backend.json` Et prendre les valeurs codées en base64 du certificat client, de la clé privée et du certificat CA de confiance, respectivement.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Export policy auto

Ces exemples vous montrent comment vous pouvez demander à Astra Trident d'utiliser des règles d'exportation dynamiques pour créer et gérer automatiquement les règles d'exportation. Cela fonctionne de la même manière pour le `ontap-nas-economy` et `ontap-nas-flexgroup` pilotes.

pilote `ontap-nas`

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

`ontap-nas-flexgroup` conducteur

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: test-cluster-east-1b
  backend: test1-ontap-cluster
svm: svm_nfs
username: vsadmin
password: password
```

Utilisation des adresses IPv6

Cet exemple montre managementLIF Utilisation d'une adresse IPv6.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

ontap-nas-economy **conducteur**

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

ontap-nas **Pilote pour Amazon FSX pour ONTAP utilisant des volumes SMB**

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Exemples de systèmes back-end avec pools virtuels

Dans l'exemple de fichier de définition backend ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, par exemple `spaceReserve` aucune, `spaceAllocation` lors de la fausse idée, et `encryption` faux. Les pools virtuels sont définis dans la section stockage.

Astra Trident définit les étiquettes de provisionnement dans le champ « Commentaires ». Les commentaires sont définis sur FlexVol pour `ontap-nas` Ou FlexGroup pour `ontap-nas-flexgroup`. Astra Trident copie toutes les étiquettes présentes sur un pool virtuel vers le volume de stockage lors du provisionnement. Pour plus de commodité, les administrateurs du stockage peuvent définir des étiquettes par pool virtuel et les volumes de groupe par étiquette.

Dans cet exemple, certains pools de stockage sont propriétaires de leur propre pool `spaceReserve`, `spaceAllocation`, et `encryption` les valeurs et certains pools remplacent les valeurs par défaut définies ci-dessus.

<code>ontap-nas</code> conducteur

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: admin
password: password
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: 'false'
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  app: msoffice
  cost: '100'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
    adaptiveQosPolicy: adaptive-premium
- labels:
  app: slack
  cost: '75'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  app: wordpress
  cost: '50'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
```

```
app: mysqldb
cost: '25'
zone: us_east_1d
defaults:
  spaceReserve: volume
  encryption: 'false'
  unixPermissions: '0775'
```

<code>ontap-nas-flexgroup</code> conducteur

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
defaults:
  spaceReserve: none
  encryption: 'false'
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  protection: gold
  creditpoints: '50000'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  protection: gold
  creditpoints: '30000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  protection: silver
  creditpoints: '20000'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
  protection: bronze
  creditpoints: '10000'
  zone: us_east_1d
```

```
defaults:  
  spaceReserve: volume  
  encryption: 'false'  
  unixPermissions: '0775'
```

<code>ontap-nas-economy</code> conducteur

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
defaults:
  spaceReserve: none
  encryption: 'false'
labels:
  store: nas_economy_store
region: us_east_1
storage:
- labels:
  department: finance
  creditpoints: '6000'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  department: legal
  creditpoints: '5000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  department: engineering
  creditpoints: '3000'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
  department: humanresource
  creditpoints: '2000'
  zone: us_east_1d
  defaults:
```

```
spaceReserve: volume
encryption: 'false'
unixPermissions: '0775'
```

Mise à jour dataLIF après la configuration initiale

Vous pouvez modifier la LIF de données après la configuration initiale en exécutant la commande suivante pour fournir le nouveau fichier JSON back-end avec la LIF de données mise à jour.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```



Si des demandes de volume persistant sont associées à un ou plusieurs pods, tous les pods correspondants doivent être arrêtés, puis réintégrés dans le but de permettre la nouvelle LIF de données d'être effective.

Mappage des systèmes back-end aux classes de stockage

Les définitions de classe de stockage suivantes font référence aux pools virtuels ci-dessus. À l'aide du `parameters.selector` Chaque classe de stockage indique quel(s) pool(s) virtuel(s) peut(s) être utilisé(s) pour héberger un volume. Les aspects définis dans le pool virtuel sélectionné seront définis pour le volume.

- La première classe de stockage (`protection-gold`) sera mappé sur le premier, deuxième pool virtuel dans le `ontap-nas-flexgroup` système back-end et le premier pool virtuel dans le `ontap-san` back-end. Il s'agit du seul pool offrant une protection de niveau Gold.
- La deuxième classe de stockage (`protection-not-gold`) sera mappé sur le troisième, quatrième pool virtuel dans `ontap-nas-flexgroup` back-end et le deuxième, troisième pool virtuel dans `ontap-san` back-end. Ce sont les seuls pools offrant un niveau de protection autre que l'or.
- La troisième classe de stockage (`app-mysqldb`) sera mappé sur le quatrième pool virtuel dans `ontap-nas` back-end et le troisième pool virtuel dans `ontap-san-economy` back-end. Ce sont les seuls pools offrant une configuration de pool de stockage pour l'application de type `mysqldb`.
- La quatrième classe de stockage (`protection-silver-creditpoints-20k`) sera mappé sur le troisième pool virtuel dans `ontap-nas-flexgroup` back-end et le second pool virtuel dans `ontap-san` back-end. Ce sont les seules piscines offrant une protection de niveau or à 20000 points de solvabilité.
- La cinquième classe de stockage (`creditpoints-5k`) sera mappé sur le second pool virtuel dans `ontap-nas-economy` back-end et le troisième pool virtuel dans `ontap-san` back-end. Ce sont les seules offres de piscine à 5000 points de solvabilité.

Astra Trident va décider du pool virtuel sélectionné et s'assurer que les besoins en stockage sont satisfaits.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: netapp.io/trident
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Amazon FSX pour NetApp ONTAP

Utilisez Astra Trident avec Amazon FSX pour NetApp ONTAP

"Amazon FSX pour NetApp ONTAP" Est un service AWS entièrement géré qui permet aux clients de lancer et d'exécuter des systèmes de fichiers optimisés par le système d'exploitation du stockage NetApp ONTAP. La solution FSX pour ONTAP vous permet d'exploiter les fonctionnalités, les performances et les capacités d'administration de NetApp que vous connaissez bien, tout en profitant de la simplicité, de l'agilité, de la sécurité et de l'évolutivité du stockage de données sur AWS. FSX pour ONTAP prend en charge les fonctionnalités du système de fichiers ONTAP et les API d'administration.

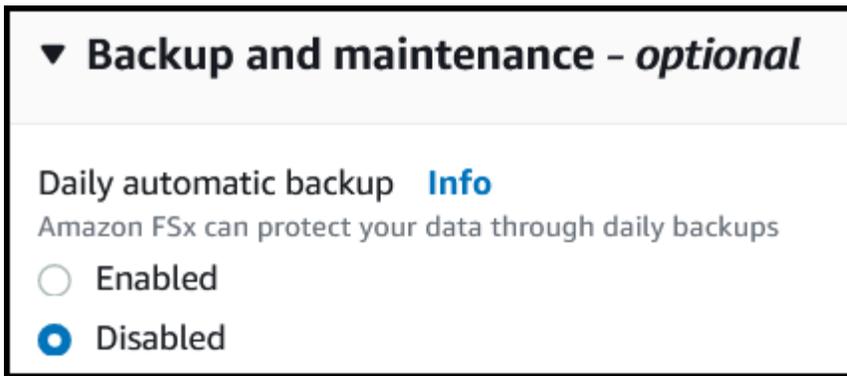
Un système de fichiers est la ressource principale d'Amazon FSX, similaire à un cluster ONTAP sur site. Au sein de chaque SVM, vous pouvez créer un ou plusieurs volumes, qui sont des conteneurs de données qui stockent les fichiers et les dossiers dans votre système de fichiers. Avec Amazon FSX pour NetApp ONTAP, Data ONTAP sera fourni en tant que système de fichiers géré dans le cloud. Le nouveau type de système de fichiers est appelé **NetApp ONTAP**.

Avec Astra Trident avec Amazon FSX pour NetApp ONTAP, vous pouvez vous assurer que les clusters Kubernetes exécutés dans Amazon Elastic Kubernetes Service (EKS) peuvent provisionner des volumes persistants de bloc et de fichier sauvegardés par ONTAP.

Amazon FSX pour NetApp ONTAP utilise "[FabricPool](#)" pour gérer les tiers de stockage. Elle vous permet de stocker les données au niveau le plus important, selon que celles-ci sont fréquemment utilisées.

Considérations

- Volumes SMB :
 - Les volumes SMB sont pris en charge à l'aide de `ontap-nas` conducteur uniquement.
 - Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.
 - Astra Trident ne prend pas en charge l'architecture Windows ARM.
- Les volumes créés sur des systèmes de fichiers Amazon FSX dont les sauvegardes automatiques sont activées ne peuvent pas être supprimés par Trident. Pour supprimer des demandes de volume persistant, vous devez supprimer manuellement le volume PV et le volume FSX pour ONTAP. Pour éviter ce problème :
 - N'utilisez pas **création rapide** pour créer le système de fichiers FSX pour ONTAP. Le flux de création rapide active les sauvegardes automatiques et ne propose pas d'option de désinscription.
 - Lorsque vous utilisez **création standard**, désactivez la sauvegarde automatique. La désactivation des sauvegardes automatiques permet à Trident de supprimer un volume sans intervention manuelle supplémentaire.



Pilotes

Vous pouvez intégrer Astra Trident avec Amazon FSX pour NetApp ONTAP à l'aide des pilotes suivants :

- `ontap-san`: Chaque volume persistant provisionné est un LUN au sein de son propre volume Amazon FSX pour NetApp ONTAP.
- `ontap-san-economy`: Chaque volume persistant provisionné est un LUN avec un nombre configurable de LUN par Amazon FSX pour le volume NetApp ONTAP.
- `ontap-nas`: Chaque volume persistant provisionné est un volume Amazon FSX complet pour NetApp ONTAP.
- `ontap-nas-economy`: Chaque volume persistant provisionné est un qtree, avec un nombre configurable de qtrees par Amazon FSX pour le volume NetApp ONTAP.
- `ontap-nas-flexgroup`: Chaque volume persistant provisionné est un volume Amazon FSX complet pour NetApp ONTAP FlexGroup.

Pour plus d'informations sur le pilote, reportez-vous à la section "[Pilotes ONTAP](#)".

Authentification

Astra Trident propose deux modes d'authentification.

- Basé sur des certificats : Astra Trident communiquera avec le SVM sur votre système de fichiers FSX à l'aide d'un certificat installé sur votre SVM.
- Basé sur les identifiants : vous pouvez utiliser le `fsxadmin` utilisateur pour votre système de fichiers ou `vsadmin` Configuré pour votre SVM.



Astra Trident devrait être exécuté en tant que `A. vsadmin` Utilisateur SVM ou en tant qu'utilisateur avec un nom différent qui a le même rôle. Amazon FSX pour NetApp ONTAP en a un `fsxadmin` Utilisateur qui remplace le ONTAP de manière limitée `admin` utilisateur du cluster. Nous vous recommandons vivement d'utiliser `vsadmin` Avec Astra Trident.

Vous pouvez mettre à jour les systèmes back-end pour passer d'une méthode basée sur les identifiants à une méthode basée sur les certificats. Toutefois, si vous tentez de fournir des identifiants et des certificats *, la création du back-end échouera. Pour passer à une méthode d'authentification différente, vous devez supprimer la méthode existante de la configuration backend.

Pour plus d'informations sur l'activation de l'authentification, reportez-vous à la section authentification de votre type de pilote :

- ["Authentification NAS ONTAP"](#)
- ["Authentification SAN de ONTAP"](#)

Trouvez plus d'informations

- ["Documentation Amazon FSX pour NetApp ONTAP"](#)
- ["Billet de blog sur Amazon FSX pour NetApp ONTAP"](#)

Intégration d'Amazon FSX pour NetApp ONTAP

Vous pouvez intégrer votre système de fichiers Amazon FSX pour NetApp ONTAP avec Astra Trident pour vous assurer que les clusters Kubernetes exécutés dans Amazon Elastic Kubernetes Service (EKS) peuvent provisionner des volumes persistants de bloc et de fichier sauvegardés par ONTAP.

Avant de commencer

En plus de ["Exigences d'Astra Trident"](#), Pour intégrer FSX pour ONTAP avec Astra Trident, vous avez besoin de :

- Un cluster Amazon EKS existant ou un cluster Kubernetes autogéré avec `kubectl` installé.
- Un système de fichiers Amazon FSX pour NetApp ONTAP existant et une machine virtuelle de stockage (SVM) accessible depuis les nœuds workers de votre cluster.
- Nœuds worker prêts pour ["NFS ou iSCSI"](#).



Assurez-vous de suivre les étapes de préparation des nœuds requises pour Amazon Linux et Ubuntu ["Images de machine Amazon"](#) (AMIS) en fonction de votre type ami EKS.

Exigences supplémentaires pour les volumes SMB

- Cluster Kubernetes avec un nœud de contrôleur Linux et au moins un nœud worker Windows exécutant Windows Server 2019. Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.
- Au moins un secret Astra Trident contenant vos identifiants Active Directory. Pour générer un secret `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user  
--from-literal password='password'
```

- Un proxy CSI configuré en tant que service Windows. Pour configurer un `csi-proxy`, voir ["GitHub : proxy CSI"](#) ou ["GitHub : proxy CSI pour Windows"](#) Pour les nœuds Kubernetes s'exécutant sur Windows.

Intégration des pilotes SAN et NAS de ONTAP



Si vous configurez la configuration pour les volumes SMB, vous devez lire [Préparez-vous au provisionnement des volumes SMB](#) avant de créer le backend.

Étapes

1. Déployez Astra Trident avec l'un des ["méthodes de déploiement"](#).

2. Collectez votre nom DNS de la LIF de gestion du SVM. Par exemple, recherchez le sur l'interface de ligne de commandes AWS `DNSName` entrée sous `Endpoints` → `Management` après avoir exécuté la commande suivante :

```
aws fsx describe-storage-virtual-machines --region <file system region>
```

3. Créer et installer des certificats pour "[Authentification NAS backend](#)" ou "[Authentification SAN backend](#)".



Vous pouvez vous connecter à votre système de fichiers (par exemple pour installer des certificats) à l'aide de SSH à partir de n'importe quel endroit qui peut atteindre votre système de fichiers. Utilisez le `fsxadmin` User, le mot de passe que vous avez configuré lors de la création de votre système de fichiers et le nom DNS de gestion à partir de `aws fsx describe-file-systems`.

4. Créer un fichier backend en utilisant vos certificats et le nom DNS de votre LIF de gestion, comme indiqué dans l'exemple ci-dessous :

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: customBackendName
managementLIF: svm-XXXXXXXXXXXXXXXXXXXX.fs-XXXXXXXXXXXXXXXXXXXX.fsx.us-
east-2.aws.internal
svm: svm01
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "customBackendName",
  "managementLIF": "svm-XXXXXXXXXXXXXXXXXXXX.fs-
XXXXXXXXXXXXXXXXXXXX.fsx.us-east-2.aws.internal",
  "svm": "svm01",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz"
}
```

Pour plus d'informations sur la création des systèmes back-end, voir les liens suivants :

- ["Configurer un système back-end avec les pilotes NAS ONTAP"](#)
- ["Configurer un système back-end avec les pilotes SAN ONTAP"](#)

Résultats

Après déploiement, vous pouvez créer un ["classe de stockage, provisionnez un volume et montez le volume dans un pod"](#).

Préparez-vous au provisionnement des volumes SMB

Vous pouvez provisionner des volumes SMB à l'aide de `ontap-nas` conducteur. Avant de terminer [Intégration des pilotes SAN et NAS de ONTAP](#) procédez comme suit.

Étapes

1. Création de partages SMB. Vous pouvez créer les partages d'administration SMB de deux manières à l'aide de l' ["Console de gestion Microsoft"](#) Dossier partagé snap-in ou à l'aide de l'interface de ligne de commande ONTAP. Pour créer les partages SMB à l'aide de l'interface de ligne de commandes ONTAP :
 - a. Si nécessaire, créez la structure du chemin d'accès au répertoire pour le partage.

Le `vserver cifs share create` commande vérifie le chemin spécifié dans l'option `-path` lors de la création du partage. Si le chemin spécifié n'existe pas, la commande échoue.

- b. Créer un partage SMB associé au SVM spécifié :

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

- c. Vérifiez que le partage a été créé :

```
vserver cifs share show -share-name share_name
```



Reportez-vous à la section ["Créez un partage SMB"](#) pour en savoir plus.

2. Lors de la création du back-end, vous devez configurer le suivant pour spécifier les volumes SMB. Pour toutes les options de configuration back-end FSX pour ONTAP, voir ["Exemples et options de configuration de FSX pour ONTAP"](#).

Paramètre	Description	Exemple
<code>smbShare</code>	Nom du partage SMB créé à l'aide du dossier partagé Microsoft Management Console. Par exemple « <code>smb-share</code> ». Requis pour les volumes SMB.	<code>smb-share</code>
<code>nasType</code>	Doit être défini sur <code>smb</code> , si elle est nulle, la valeur par défaut est <code>nfs</code> .	<code>smb</code>

Paramètre	Description	Exemple
securityStyle	Style de sécurité pour les nouveaux volumes. Doit être défini sur ntfs ou mixed Pour les volumes SMB.	ntfs ou mixed Pour les volumes SMB
unixPermissions	Mode pour les nouveaux volumes. Doit rester vide pour les volumes SMB.	« »

Exemples et options de configuration de FSX pour ONTAP

Découvrez les options de configuration back-end pour Amazon FSX pour ONTAP. Cette section fournit des exemples de configuration back-end.

Options de configuration du back-end

Voir le tableau suivant pour les options de configuration du back-end :

Paramètre	Description	Exemple
version		Toujours 1
storageDriverName	Nom du pilote de stockage	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san », « ontap-san », « ontap-économie san »
backendName	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + dataLIF
managementLIF	Adresse IP d'un cluster ou d'une LIF de gestion SVM pour un basculement MetroCluster transparent, vous devez spécifier une LIF de gestion SVM. Un nom de domaine complet (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide du --use-ipv6 drapeau. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	« 10.0.0.1 », « [2001:1234:abcd::fefe] »

Paramètre	Description	Exemple
dataLIF	Adresse IP de la LIF de protocole. Pilotes NAS ONTAP: Nous vous recommandons de spécifier dataLIF. Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. Peut être modifié après le réglage initial. Reportez-vous à la section . Pilotes SAN ONTAP : ne pas spécifier pour iSCSI. Astra Trident utilise le mappage de LUN sélectif de ONTAP pour découvrir les LIFs iSCSI nécessaires pour établir une session multi-chemins. Un avertissement est généré si dataLIF est explicitement défini. Peut être configuré pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide du <code>--use-ipv6</code> drapeau. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
autoExportPolicy	Activer la création et la mise à jour automatiques des règles d'exportation [booléennes]. À l'aide du <code>autoExportPolicy</code> et <code>autoExportCIDRs</code> Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	« faux »
autoExportCIDRs	Liste des CIDR pour filtrer les adresses IP du nœud Kubernetes par rapport à quand <code>autoExportPolicy</code> est activé. À l'aide du <code>autoExportPolicy</code> et <code>autoExportCIDRs</code> Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	« [« 0.0.0.0/0 », « :/0 »] »
labels	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »

Paramètre	Description	Exemple
clientCertificate	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
clientPrivateKey	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »
trustedCACertificate	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification basée sur des certificats.	« »
username	Nom d'utilisateur pour la connexion au cluster ou au SVM. Utilisé pour l'authentification basée sur les identifiants. Par exemple, vsadmin.	
password	Mot de passe pour se connecter au cluster ou au SVM. Utilisé pour l'authentification basée sur les identifiants.	
svm	Serveur virtuel de stockage à utiliser	Dérivé si une LIF de gestion SVM est spécifiée.
igroupName	Nom du groupe initiateur à utiliser pour les volumes SAN. Reportez-vous à la section .	Trident-<backend-UUID>
storagePrefix	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être modifié après sa création. Pour mettre à jour ce paramètre, vous devez créer un nouveau backend.	trident
limitAggregateUsage	Ne pas spécifier pour Amazon FSX pour NetApp ONTAP fsxadmin et vsadmin Ne contiennent pas les autorisations requises pour récupérer l'utilisation d'agrégats et le limiter à l'aide d'Astra Trident.	Ne pas utiliser.
limitVolumeSize	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN, et la qtreesPerFlexvol L'option permet de personnaliser le nombre maximal de qtree par FlexVol.	« » (non appliqué par défaut)

Paramètre	Description	Exemple
lunsPerFlexvol	Le nombre maximal de LUN par FlexVol doit être compris dans la plage [50, 200]. SAN uniquement.	« 100 »
debugTraceFlags	Indicateurs de débogage à utiliser lors du dépannage. Par exemple, {"api":false, "méthode":true} ne pas utiliser debugTraceFlags à moins que vous ne soyez en mesure de dépanner et que vous ayez besoin d'un vidage détaillé des journaux.	nul
nfsMountOptions	Liste des options de montage NFS séparée par des virgules. Les options de montage des volumes Kubernetes persistants sont généralement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Astra Trident utilisera les options de montage spécifiées dans le fichier de configuration du système back-end. Si aucune option de montage n'est spécifiée dans la classe de stockage ou le fichier de configuration, Astra Trident ne définit aucune option de montage sur un volume persistant associé.	« »
nasType	Configurez la création de volumes NFS ou SMB. Les options sont <code>nfs</code> , <code>smb</code> , ou <code>nul</code> . Doit être défini sur <code>smb</code> Pour les volumes SMB. la valeur NULL est définie par défaut sur les volumes NFS.	nfs
qtreesPerFlexvol	Nombre maximal de qtrees par FlexVol, qui doit être compris dans la plage [50, 300]	« 200 »
smbShare	Nom du partage SMB créé à l'aide du dossier partagé Microsoft Management Console. Requis pour les volumes SMB.	« partage smb »

Paramètre	Description	Exemple
useREST	<p>Paramètre booléen pour utiliser les API REST de ONTAP. Aperçu technique</p> <p>useREST est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur <code>true</code>, Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au <code>ontap client</code> supplémentaire. Ceci est satisfait par le pré-défini <code>vsadmin</code> et <code>cluster-admin</code> rôles.</p>	« faux »

Détails sur `igroupName`

`igroupName` Peut être défini sur un groupe initiateur déjà créé sur le cluster ONTAP. Si non spécifié, Astra Trident crée automatiquement un groupe initiateur nommé `trident-<backend-UUID>`.

Si vous disposez d'un nom de groupe prédéfini, nous vous recommandons d'utiliser un groupe initiateur par cluster Kubernetes si le SVM doit être partagé entre les environnements. Cela est nécessaire pour qu'Astra Trident conserve automatiquement les ajouts et suppressions d'IQN.

- `igroupName` Peut être mis à jour afin de désigner un nouveau groupe initiateur créé et géré sur la SVM en dehors d'Astra Trident.
- `igroupName` peut être omis. Dans ce cas, Astra Trident crée et gère un groupe initiateur nommé `trident-<backend-UUID>` automatiquement.

Dans les deux cas, les pièces jointes de volume continueront d'être accessibles. Les pièces jointes futures utilisent le groupe initiateur mis à jour. Cette mise à jour n'interrompt pas l'accès aux volumes présents sur le back-end.

Mise à jour `dataLIF` après la configuration initiale

Vous pouvez modifier la LIF de données après la configuration initiale en exécutant la commande suivante pour fournir le nouveau fichier JSON back-end avec la LIF de données mise à jour.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-with-updated-dataLIF>
```



Si des demandes de volume persistant sont associées à un ou plusieurs pods, tous les pods correspondants doivent être arrêtés, puis réintégrés dans le but de permettre la nouvelle LIF de données d'être effective.

Options de configuration back-end pour les volumes de provisionnement

Vous pouvez contrôler le provisionnement par défaut à l'aide de ces options dans `defaults` section de la configuration. Pour un exemple, voir les exemples de configuration ci-dessous.

Paramètre	Description	Valeur par défaut
<code>spaceAllocation</code>	Allocation d'espace pour les LUN	« vrai »
<code>spaceReserve</code>	Mode de réservation d'espace ; "none" (fin) ou "volume" (épais)	« aucun »
<code>snapshotPolicy</code>	Règle Snapshot à utiliser	« aucun »
<code>qosPolicy</code>	QoS policy group à affecter pour les volumes créés. Choisissez une de <code>qosPolicy</code> ou <code>adaptiveQosPolicy</code> par pool de stockage ou back-end. Avec Astra Trident, les groupes de règles de QoS doivent être utilisés avec ONTAP 9.8 ou version ultérieure. Nous recommandons l'utilisation d'un groupe de règles de qualité de service non partagé et nous assurer que le groupe de règles est appliqué à chaque composant individuellement. Un groupe de règles de QoS partagé appliquera le plafond du débit total de toutes les charges de travail.	« »
<code>adaptiveQosPolicy</code>	Groupe de règles de QoS adaptative à attribuer aux volumes créés. Choisissez une de <code>qosPolicy</code> ou <code>adaptiveQosPolicy</code> par pool de stockage ou back-end. Non pris en charge par l'économie <code>ontap-nas</code> .	« »
<code>snapshotReserve</code>	Pourcentage du volume réservé pour les instantanés "0"	Si <code>snapshotPolicy</code> est « aucun », sinon « »
<code>splitOnClone</code>	Séparer un clone de son parent lors de sa création	« faux »

Paramètre	Description	Valeur par défaut
encryption	Activez NetApp Volume Encryption (NVE) sur le nouveau volume. La valeur par défaut est <code>false</code> . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE" .	« faux »
luksEncryption	Activez le cryptage LUKS. Reportez-vous à la section "Utiliser la configuration de clé unifiée Linux (LUKS)" . SAN uniquement.	« »
tieringPolicy	La stratégie de hiérarchisation à utiliser « none »	Snapshot uniquement pour une configuration SVM-DR pré-ONTAP 9.5
unixPermissions	Mode pour les nouveaux volumes. Laisser vide pour les volumes SMB.	« »
securityStyle	Style de sécurité pour les nouveaux volumes. Prise en charge de NFS <code>mixed</code> et <code>unix</code> styles de sécurité. SMB prend en charge <code>mixed</code> et <code>ntfs</code> styles de sécurité.	NFS par défaut est <code>unix</code> . SMB par défaut est <code>ntfs</code> .

Exemple

À l'aide de `nasType`, `node-stage-secret-name`, et `node-stage-secret-namespace`, Vous pouvez spécifier un volume SMB et fournir les informations d'identification Active Directory requises. Les volumes SMB sont pris en charge à l'aide de `ontap-nas` conducteur uniquement.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nas-smb-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"

```

Création de systèmes back-end avec kubectl

Un système back-end définit la relation entre Astra Trident et un système de stockage. Il explique à Astra Trident comment communiquer avec ce système de stockage et comment Astra Trident doit provisionner des volumes à partir de celui-ci. Après l'installation d'Astra Trident, l'étape suivante consiste à créer un système back-end. Le `TridentBackendConfig` La définition de ressource personnalisée (CRD) vous permet de créer et de gérer des systèmes back-end Trident directement via l'interface Kubernetes. Vous pouvez le faire en utilisant `kubectl` Ou l'outil CLI équivalent pour votre distribution Kubernetes.

TridentBackendConfig

`TridentBackendConfig` (`tbc`, `tbconfig`, `tbackendconfig`) Est un CRD au rythme de noms qui vous permet de gérer les systèmes back-end Astra Trident à l'aide de `kubectl`. Avec Kubernetes et les administrateurs de stockage, il est désormais possible de créer et de gérer des systèmes back-end directement via la CLI Kubernetes sans avoir besoin d'un utilitaire de ligne de commande dédié (`tridentctl`).

Lors de la création d'un `TridentBackendConfig` objet :

- Un système back-end est créé automatiquement par Astra Trident en fonction de la configuration que vous fournissez. Il est représenté en interne en tant que `TridentBackend` (`tbe`, `tridentbackend`) CR.
- Le `TridentBackendConfig` est lié de manière unique à un `TridentBackend` Créé par Astra Trident.

Chacun `TridentBackendConfig` gère un mappage un-à-un avec un `TridentBackend`. La première est l'interface fournie à l'utilisateur pour concevoir et configurer les systèmes back-end, tandis que `Trident` représente l'objet back-end réel.



`TridentBackend` ASTRA Trident crée automatiquement des CRS. Vous ne devez pas les modifier. Si vous voulez effectuer des mises à jour vers les systèmes back-end, modifiez le `TridentBackendConfig` objet.

Reportez-vous à l'exemple suivant pour connaître le format du `TridentBackendConfig` CR :

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Vous pouvez également consulter les exemples de la "[programme d'installation trident](#)" répertoire des

exemples de configuration pour la plate-forme/le service de stockage souhaité.

Le `spec` prend des paramètres de configuration spécifiques au back-end. Dans cet exemple, le back-end utilise le `ontap-san` pilote de stockage et utilise les paramètres de configuration qui sont présentés ici. Pour obtenir la liste des options de configuration pour le pilote de stockage souhaité, reportez-vous à la section "[informations de configuration backend pour votre pilote de stockage](#)".

Le `spec` la section inclut également `credentials` et `deletionPolicy` les champs qui viennent d'être introduits dans le `TridentBackendConfig` CR :

- `credentials`: Ce paramètre est un champ obligatoire et contient les informations d'identification utilisées pour s'authentifier auprès du système/service de stockage. Cette configuration est définie sur un code secret Kubernetes créé par l'utilisateur. Les informations d'identification ne peuvent pas être transmises en texte brut et entraînent une erreur.
- `deletionPolicy`: Ce champ définit ce qui doit se produire lorsque `TridentBackendConfig` est supprimé. Il peut prendre l'une des deux valeurs possibles :
 - `delete`: Cela entraîne la suppression des deux `TridentBackendConfig` CR et le back-end associé. Il s'agit de la valeur par défaut.
 - `retain`: Lorsqu'un `TridentBackendConfig` La demande de modification est supprimée, la définition de l'arrière-plan est toujours présente et peut être gérée avec `tridentctl`. Définition de la stratégie de suppression sur `retain` permet aux utilisateurs de revenir à une version antérieure (avant la version 21.04) et de conserver les systèmes back-end créés. La valeur de ce champ peut être mise à jour après un `TridentBackendConfig` est créé.



Le nom d'un backend est défini à l'aide de `spec.backendName`. S'il n'est pas spécifié, le nom du back-end est défini sur le nom du `TridentBackendConfig` objet (`metadata.name`). Il est recommandé de définir explicitement les noms backend à l'aide de `spec.backendName`.



Systèmes back-end créés avec `tridentctl` n'avez pas de lien associé `TridentBackendConfig` objet. Vous avez la possibilité de choisir de gérer de tels systèmes back-end avec `kubectl` en créant un `TridentBackendConfig` CR. Vous devez veiller à spécifier des paramètres de configuration identiques (par exemple `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, etc.). Astra Trident lie automatiquement le nouveau produit `TridentBackendConfig` avec le back-end existant.

Présentation des étapes

Pour créer un nouveau back-end à l'aide de `kubectl`, vous devez effectuer les opérations suivantes :

1. Créer un "[Le secret de Kubernetes](#)". Le secret est qu'Astra Trident doit communiquer avec le cluster/service de stockage.
2. Créer un `TridentBackendConfig` objet. Elle contient des informations spécifiques sur le cluster/service de stockage et fait référence au secret créé à l'étape précédente.

Après avoir créé un back-end, vous pouvez observer son état en utilisant `kubectl get tbc <tbc-name> -n <trident-namespace>` et recueillez des détails supplémentaires.

Étape 1 : créez un code secret Kubernetes

Créez un secret qui contient les informations d'identification d'accès pour le back-end. Ce point est unique à chaque service/plateforme de stockage. Voici un exemple :

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

Ce tableau récapitule les champs à inclure dans le Secret pour chaque plate-forme de stockage :

Description des champs secrets de la plate-forme de stockage	Secret	Description des champs
Azure NetApp Files	ID client	ID client d'un enregistrement d'application
Cloud Volumes Service pour GCP	id_clé_privée	ID de la clé privée. Partie de la clé API pour le compte de service GCP avec le rôle d'administrateur CVS
Cloud Volumes Service pour GCP	clé_privée	Clé privée. Partie de la clé API pour le compte de service GCP avec le rôle d'administrateur CVS
Element (NetApp HCI/SolidFire)	Point final	MVIP pour le cluster SolidFire avec les identifiants de locataire
ONTAP	nom d'utilisateur	Nom d'utilisateur pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants
ONTAP	mot de passe	Mot de passe pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants
ONTAP	ClientPrivateKey	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification basée sur des certificats

Description des champs secrets de la plate-forme de stockage	Secret	Description des champs
ONTAP	ChapUsername	Nom d'utilisateur entrant. Requis si useCHAP=vrai. Pour ontap-san et ontap-san-economy
ONTAP	Chapeau InitiatorSecret	Secret de l'initiateur CHAP. Requis si useCHAP=vrai. Pour ontap-san et ontap-san-economy
ONTAP	ChapTargetUsername	Nom d'utilisateur cible. Requis si useCHAP=vrai. Pour ontap-san et ontap-san-economy
ONTAP	ChapTargetInitiatorSecret	Secret de l'initiateur cible CHAP. Requis si useCHAP=vrai. Pour ontap-san et ontap-san-economy

Le secret créé dans cette étape sera référencé dans le `spec.credentials` champ du `TridentBackendConfig` objet créé à l'étape suivante.

Étape 2 : créez le `TridentBackendConfig` CR

Vous êtes maintenant prêt à créer votre `TridentBackendConfig` CR. Dans cet exemple, un back-end qui utilise le `ontap-san` le pilote est créé à l'aide du `TridentBackendConfig` objet illustré ci-dessous :

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Étape 3 : vérifier l'état du TridentBackendConfig CR

Maintenant que vous avez créé le TridentBackendConfig CR, vous pouvez vérifier l'état. Voir l'exemple suivant :

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                                BACKEND NAME                                BACKEND UUID
PHASE    STATUS
backend-tbc-ontap-san    ontap-san-backend    8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8    Bound    Success
```

Un back-end a été créé avec succès et lié au TridentBackendConfig CR.

La phase peut prendre l'une des valeurs suivantes :

- **Bound:** Le TridentBackendConfig La demande de modification est associée à un back-end, et ce backend contient configRef réglez sur TridentBackendConfig L'uid de CR.
- **Unbound:** Représenté en utilisant "". Le TridentBackendConfig l'objet n'est pas lié à un back-end. Tout nouveau TridentBackendConfig Les CRS sont dans cette phase par défaut. Une fois la phase modifiée, elle ne peut plus revenir à Unbound.
- **Deleting:** Le TridentBackendConfig CR deletionPolicy a été configuré pour supprimer. Lorsque le TridentBackendConfig La demande de modification est supprimée, elle passe à l'état Suppression.
 - Si aucune demande de volume persistant n'existe sur le back-end, supprimez le TridentBackendConfig Il en résultera la suppression du système back-end et du système Astra Trident TridentBackendConfig CR.
 - Si un ou plusieurs ESV sont présents sur le back-end, il passe à l'état de suppression. Le TridentBackendConfig La CR entre ensuite la phase de suppression. Le back-end et TridentBackendConfig Sont supprimés uniquement après la suppression de tous les ESV.
- **Lost:** Le back-end associé à l' TridentBackendConfig Le CR a été accidentellement ou délibérément supprimé et le TridentBackendConfig La CR a toujours une référence au back-end supprimé. Le TridentBackendConfig La CR peut toujours être supprimée, quel que soit le deletionPolicy valeur.
- **Unknown:** Astra Trident n'est pas en mesure de déterminer l'état ou l'existence du back-end associé au TridentBackendConfig CR. Par exemple, si le serveur d'API ne répond pas ou si tridentbackends.trident.netapp.io CRD manquant. Cela peut nécessiter l'intervention de l'utilisateur.

À ce stade, un système back-end est créé avec succès ! Plusieurs opérations peuvent également être traitées, par exemple "[mises à jour du système back-end et suppressions](#)".

(Facultatif) étape 4 : pour plus de détails

Vous pouvez exécuter la commande suivante pour obtenir plus d'informations sur votre système back-end :

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

NAME	BACKEND NAME	BACKEND UUID	
PHASE	STATUS	STORAGE DRIVER	DELETION POLICY
backend-tbc-ontap-san	ontap-san-backend	8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8	Bound
	Success	ontap-san	delete

En outre, vous pouvez également obtenir un vidage YAML/JSON de `TridentBackendConfig`.

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: "2021-04-21T20:45:11Z"
  finalizers:
  - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo` contient le `backendName` et le `backendUUID` du back-end créé en réponse à `TridentBackendConfig` CR. Le `lastOperationStatus` champ représente l'état de la dernière opération du `TridentBackendConfig` CR, qui peut être déclenché par l'utilisateur (par exemple, l'utilisateur a modifié quelque chose dans `spec`) Ou déclenché par Astra Trident (par exemple lors du redémarrage d'Astra Trident). Il peut être réussi ou échoué. `phase` représente l'état de la relation entre `TridentBackendConfig` CR et le

backend. Dans l'exemple ci-dessus, `phase A` a la valeur limitée, ce qui signifie que le `TridentBackendConfig` CR est associé au back-end.

Vous pouvez exécuter le `kubectl -n trident describe tbc <tbc-cr-name>` commande pour obtenir des détails sur les journaux d'événements.



Vous ne pouvez pas mettre à jour ou supprimer un backend qui contient un associé `TridentBackendConfig` objet utilisant `tridentctl`. Pour comprendre les étapes de passage d'un à l'autre `tridentctl` et `TridentBackendConfig`, ["voir ici"](#).

Effectuer la gestion back-end avec kubectl

Découvrez comment effectuer des opérations de gestion back-end à l'aide de `kubectl`.

Supprimer un back-end

En supprimant un `TridentBackendConfig`, Vous demandez à Astra Trident de supprimer/conservé les systèmes back-end (sur la base `deletionPolicy`). Pour supprimer un back-end, assurez-vous que `deletionPolicy` est configuré pour supprimer. Pour supprimer uniquement le `TridentBackendConfig`, assurez-vous que `deletionPolicy` est défini sur `conservé`. Cela permet de s'assurer que le système backend est toujours présent et qu'il peut être géré à l'aide de `tridentctl`.

Exécutez la commande suivante :

```
kubectl delete tbc <tbc-name> -n trident
```

Astra Trident ne supprime pas les secrets Kubernetes qui étaient utilisés par `TridentBackendConfig`. L'utilisateur Kubernetes est chargé de nettoyer les secrets. Il faut faire attention lors de la suppression des secrets. Vous devez supprimer les secrets uniquement s'ils ne sont pas utilisés par les systèmes back-end.

Affichez les systèmes back-end existants

Exécutez la commande suivante :

```
kubectl get tbc -n trident
```

Vous pouvez également exécuter `tridentctl get backend -n trident` ou `tridentctl get backend -o yaml -n trident` pour obtenir une liste de tous les systèmes back-end existants, Cette liste comprend également les systèmes back-end créés avec `tridentctl`.

Mettre à jour un back-end

Il peut y avoir plusieurs raisons de mettre à jour un backend :

- Les informations d'identification du système de stockage ont été modifiées. Pour mettre à jour les identifiants, le code secret Kubernetes utilisé dans le `TridentBackendConfig` l'objet doit être mis à jour. Avec Astra Trident, le système back-end est automatiquement mis à jour avec les dernières informations d'identification fournies. Exécutez la commande suivante pour mettre à jour le code secret Kubernetes :

```
kubectl apply -f <updated-secret-file.yaml> -n trident
```

- Les paramètres (tels que le nom du SVM ONTAP utilisé) doivent être mis à jour. Dans ce cas, `TridentBackendConfig` Les objets peuvent être mis à jour directement via Kubernetes.

```
kubectl apply -f <updated-backend-file.yaml>
```

Vous pouvez également apporter des modifications à l'existant `TridentBackendConfig` CR en exécutant la commande suivante :

```
kubectl edit tbc <tbc-name> -n trident
```

En cas d'échec d'une mise à jour du back-end, le système back-end continue de rester dans sa dernière configuration connue. Vous pouvez afficher les journaux pour déterminer la cause en cours d'exécution `kubectl get tbc <tbc-name> -o yaml -n trident` ou `kubectl describe tbc <tbc-name> -n trident`.

Après avoir identifié et corrigé le problème avec le fichier de configuration, vous pouvez relancer la commande `update`.

Gestion back-end avec `tridentctl`

Découvrez comment effectuer des opérations de gestion back-end à l'aide de `tridentctl`.

Créer un back-end

Après avoir créé un "[fichier de configuration back-end](#)", exécutez la commande suivante :

```
tridentctl create backend -f <backend-file> -n trident
```

Si la création du système back-end échoue, la configuration du système back-end était erronée. Vous pouvez afficher les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Une fois que vous avez identifié et corrigé le problème avec le fichier de configuration, vous pouvez simplement exécuter le `create` commande de nouveau.

Supprimer un back-end

Pour supprimer un back-end d'Astra Trident, procédez comme suit :

1. Récupérer le nom du système back-end :

```
tridentctl get backend -n trident
```

2. Supprimer le backend :

```
tridentctl delete backend <backend-name> -n trident
```



Si Astra Trident a provisionné des volumes et des snapshots à partir de ce backend qui existe toujours, la suppression du back-end empêche les nouveaux volumes d'être provisionnés. Le système back-end continuera à exister dans un état « Suppression » et Trident continuera à gérer ces volumes et ces snapshots jusqu'à leur suppression.

Affichez les systèmes back-end existants

Pour afficher les systèmes back-end dont Trident a conscience, procédez comme suit :

- Pour obtenir un récapitulatif, exécutez la commande suivante :

```
tridentctl get backend -n trident
```

- Pour obtenir tous les détails, exécutez la commande suivante :

```
tridentctl get backend -o json -n trident
```

Mettre à jour un back-end

Après avoir créé un nouveau fichier de configuration back-end, exécutez la commande suivante :

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

En cas d'échec de la mise à jour back-end, quelque chose était incorrect avec la configuration back-end ou vous avez tenté une mise à jour non valide. Vous pouvez afficher les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Une fois que vous avez identifié et corrigé le problème avec le fichier de configuration, vous pouvez simplement exécuter le `update` commande de nouveau.

Identifier les classes de stockage qui utilisent un système back-end

Voici un exemple de questions que vous pouvez répondre avec le fichier JSON `tridentctl` sorties des

objets back-end. Ceci utilise le `jq` utilitaire que vous devez installer.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name, storageClasses: [.storage[].storageClasses]|unique}]'
```

Cela s'applique également aux systèmes back-end créés par l'utilisation `TridentBackendConfig`.

Passez d'une option de gestion back-end à une autre

Découvrez les différentes façons de gérer les systèmes back-end avec Astra Trident. Avec l'introduction de `TridentBackendConfig`, les administrateurs ont désormais deux méthodes uniques de gestion des systèmes back-end. Ceci pose les questions suivantes :

- Les systèmes back-end peuvent être créés avec `tridentctl` être géré avec `TridentBackendConfig`?
- Les systèmes back-end peuvent être créés avec `TridentBackendConfig` gestion via `tridentctl`?

Gérez `tridentctl` utilisation de systèmes back-end `TridentBackendConfig`

Cette section aborde les étapes requises pour gérer les systèmes back-end créés à l'aide de `tridentctl` Directement via l'interface Kubernetes en créant la `TridentBackendConfig` objets.

Cela s'applique aux scénarios suivants :

- Les systèmes back-end préexistants, qui n'ont pas de système `TridentBackendConfig` parce qu'ils ont été créés avec `tridentctl`.
- Nouveaux systèmes back-end créés avec `tridentctl`, tandis que d'autres `TridentBackendConfig` les objets existent.

Dans les deux cas, le système back-end restera présent. Avec Astra Trident, qui planifie les volumes et les exécute. Les administrateurs peuvent choisir l'une des deux options suivantes :

- Continuer à utiliser `tridentctl` pour gérer les systèmes back-end créés en utilisant ces systèmes.
- Lier les systèmes back-end créés à l'aide de `tridentctl` à un nouveau `TridentBackendConfig` objet. Ainsi, le système back-end sera géré à l'aide de `kubectl` et non `tridentctl`.

Pour gérer un système back-end existant à l'aide de `kubectl`, vous devez créer un `TridentBackendConfig` cela se lie au back-end existant. Voici un aperçu du fonctionnement de ces éléments :

1. Créez un code secret Kubernetes. Le secret est qu'Astra Trident doit communiquer avec le cluster/service de stockage.
2. Créer un `TridentBackendConfig` objet. Elle contient des informations spécifiques sur le cluster/service de stockage et fait référence au secret créé à l'étape précédente. Vous devez veiller à spécifier des paramètres de configuration identiques (par exemple `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, etc.). `spec.backendName` doit être défini sur le nom du back-end existant.

Étape 0 : identifier le back-end

Pour créer un `TridentBackendConfig` qui se lie à un back-end existant, vous devez obtenir la configuration back-end. Dans cet exemple, supposons qu'un back-end a été créé à l'aide de la définition JSON suivante :

```
tridentctl get backend ontap-nas-backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE  | VOLUMES |                   |                   |
+-----+-----+-----+-----+
| ontap-nas-backend    | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |          25 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

cat ontap-nas-backend.json

{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",

  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels":{"store":"nas_store"},
  "region": "us_east_1",
  "storage": [
    {
      "labels":{"app":"msoffice", "cost":"100"},
      "zone":"us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
```

```

    "labels":{"app":"mysqldb", "cost":"25"},
    "zone":"us_east_1d",
    "defaults": {
      "spaceReserve": "volume",
      "encryption": "false",
      "unixPermissions": "0775"
    }
  ]
}

```

Étape 1 : créez un code secret Kubernetes

Créez un secret qui contient les informations d'identification du back-end, comme indiqué dans cet exemple :

```

cat tbc-ontap-nas-backend-secret.yaml

apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password

kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created

```

Étape 2 : créer un TridentBackendConfig CR

L'étape suivante consiste à créer un `TridentBackendConfig` CR qui se lie automatiquement au pré-existant `ontap-nas-backend` (comme dans cet exemple). Assurez-vous que les exigences suivantes sont respectées :

- Le même nom de back-end est défini dans `spec.backendName`.
- Les paramètres de configuration sont identiques au back-end d'origine.
- Les pools virtuels (le cas échéant) doivent conserver le même ordre que dans le back-end d'origine.
- Les identifiants sont fournis via un code secret Kubernetes et non en texte brut.

Dans ce cas, le `TridentBackendConfig` se présente comme suit :

```

cat backend-tbc-ontap-nas.yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
    app: msoffice
    cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
    app: mysqldb
    cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'

kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created

```

Étape 3 : vérifier l'état du TridentBackendConfig CR

Après le TridentBackendConfig a été créée, sa phase doit être Bound. Il devrait également refléter le même nom de back-end et UUID que celui du back-end existant.

```
kubectl -n trident get tbc tbc-ontap-nas-backend -n trident
NAME                                BACKEND NAME                BACKEND UUID
PHASE    STATUS
tbc-ontap-nas-backend  ontap-nas-backend          52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7    Bound    Success
```

#confirm that no new backends were created (i.e., TridentBackendConfig did not end up creating a new backend)

```
tridentctl get backend -n trident
```

```
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE  | VOLUMES |          |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| ontap-nas-backend     | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |          25 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Le système back-end sera désormais entièrement géré à l'aide du système tbc-ontap-nas-backend TridentBackendConfig objet.

Gérez TridentBackendConfig utilisation de systèmes back-end tridentctl

`tridentctl` possibilité d'afficher la liste des systèmes back-end créés à l'aide de `TridentBackendConfig`. En outre, les administrateurs ont la possibilité de choisir entre la gestion complète de ces systèmes back-end `tridentctl` en supprimant `TridentBackendConfig` et en fait bien sûr `spec.deletionPolicy` est défini sur `retain`.

Étape 0 : identifier le back-end

Par exemple, supposons que le back-end suivant a été créé à l'aide de TridentBackendConfig:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME                BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend  81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san  delete

tridentctl get backend ontap-san-backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID
| STATE  | VOLUMES |
+-----+-----+
+-----+-----+-----+-----+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-----+-----+
+-----+-----+-----+-----+
```

À partir de la sortie, on voit cela TridentBackendConfig A été créé avec succès et est lié à un back-end [observez l'UUID du backend].

Étape 1 : confirmer deletionPolicy est défini sur retain

Passons en revue les avantages de deletionPolicy. Il doit être défini sur retain. Cela permet de s'assurer que lorsqu'un TridentBackendConfig La demande de modification est supprimée, la définition de l'arrière-plan est toujours présente et peut être gérée avec tridentctl.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME                BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend  81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san  delete

# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME                BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend  81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san  retain
```



Ne pas passer à l'étape suivante sauf si `deletionPolicy` est défini sur `retain`.

Étape 2 : supprimez le `TridentBackendConfig` CR

La dernière étape consiste à supprimer le `TridentBackendConfig` CR. Après avoir confirmé le `deletionPolicy` est défini sur `retain`, vous pouvez poursuivre la suppression :

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

tridentctl get backend ontap-san-backend -n trident
+-----+-----+-----+
+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE | VOLUMES |          |
+-----+-----+-----+
+-----+-----+-----+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |          33 |
+-----+-----+-----+
+-----+-----+-----+
```

Lors de la suppression du `TridentBackendConfig` Objet : Astra Trident la supprime simplement sans le système back-end.

Gérer les classes de stockage

Recherchez des informations sur la création d'une classe de stockage, la suppression d'une classe de stockage et l'affichage des classes de stockage existantes.

Concevez une classe de stockage

Voir ["ici"](#) pour plus d'informations sur les classes de stockage et leur configuration.

Créer une classe de stockage

Après avoir un fichier de classe de stockage, exécutez la commande suivante :

```
kubectl create -f <storage-class-file>
```

`<storage-class-file>` doit être remplacé par votre nom de fichier de classe de stockage.

Supprimer une classe de stockage

Pour supprimer une classe de stockage de Kubernetes, exécutez la commande suivante :

```
kubectl delete storageclass <storage-class>
```

<storage-class> doit être remplacé par votre classe de stockage.

Tout volume persistant créé dans le cadre de cette classe de stockage n'est pas affecté. Astra Trident va continuer à les gérer.



L'ASTRA Trident applique un blanc `fsType` pour les volumes qu'elle crée. Pour les systèmes back-end iSCSI, il est recommandé d'appliquer la configuration `parameters.fsType` Dans la classe de stockage. Vous devez supprimer des classes de stockage existantes et les recréer à l'aide de `parameters.fsType` spécifié.

Afficher les classes de stockage existantes

- Pour afficher les classes de stockage Kubernetes existantes, exécutez la commande suivante :

```
kubectl get storageclass
```

- Pour afficher les détails de la classe de stockage Kubernetes, exécutez la commande suivante :

```
kubectl get storageclass <storage-class> -o json
```

- Pour afficher les classes de stockage synchronisées d'Astra Trident, exécutez la commande suivante :

```
tridentctl get storageclass
```

- Pour afficher les détails de la classe de stockage synchronisée d'Astra Trident, exécutez la commande suivante :

```
tridentctl get storageclass <storage-class> -o json
```

Définir une classe de stockage par défaut

Kubernetes 1.6 a ajouté la possibilité de définir une classe de stockage par défaut. Cette classe de stockage sera utilisée pour provisionner un volume persistant si un utilisateur ne en spécifie pas une dans une demande de volume persistant.

- Définissez une classe de stockage par défaut en définissant l'annotation `storageclass.kubernetes.io/is-default-class` vrai dans la définition de classe de stockage. Selon la spécification, toute autre valeur ou absence de l'annotation est interprétée comme fausse.
- Vous pouvez configurer une classe de stockage existante comme classe de stockage par défaut à l'aide de la commande suivante :

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- De même, vous pouvez supprimer l'annotation de classe de stockage par défaut à l'aide de la commande suivante :

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Il existe également des exemples dans le bundle du programme d'installation de Trident qui incluent cette annotation.



Vous ne devez avoir qu'une seule classe de stockage par défaut dans votre cluster à un moment donné. Kubernetes n'empêche pas techniquement d'en avoir plusieurs, mais il se comporte comme s'il n'existait aucune classe de stockage par défaut.

Identifier le système back-end pour une classe de stockage

Voici un exemple de questions que vous pouvez répondre avec le fichier JSON `tridentctl` Sorties pour les objets back-end Astra Trident. Ceci utilise le `jq` utilitaire, que vous devrez peut-être installer en premier.

```
tridentctl get storageclass -o json | jq '[.items[] | {storageClass:  
.Config.name, backends: [.storage]|unique}]'
```

Réaliser des opérations de volume

Découvrez les fonctionnalités d'Astra Trident pour la gestion de vos volumes.

- ["Utiliser la topologie CSI"](#)
- ["Travailler avec des instantanés"](#)
- ["Développement des volumes"](#)
- ["Importer des volumes"](#)

Utiliser la topologie CSI

Astra Trident peut créer et relier de façon sélective des volumes aux nœuds présents dans un cluster Kubernetes en utilisant le ["Fonction de topologie CSI"](#). Grâce à la fonction de topologie CSI, l'accès aux volumes peut être limité à un sous-ensemble de nœuds, en fonction des régions et des zones de disponibilité. Les fournisseurs cloud permettent aujourd'hui aux administrateurs Kubernetes de frayer des nœuds basés sur une zone. Les nœuds peuvent se trouver dans différentes zones de disponibilité au sein d'une région ou entre différentes régions. Astra Trident utilise la topologie CSI pour faciliter le provisionnement des volumes pour les charges de travail dans une architecture multi-zones.



En savoir plus sur la fonction de topologie CSI ["ici"](#).

Kubernetes propose deux modes de liaison de volumes :

- Avec `VolumeBindingMode` réglé sur `Immediate`, Astra Trident crée le volume sans la reconnaissance de la topologie. La liaison de volumes et le provisionnement dynamique sont gérés au moment de la création de la demande de volume persistant. Il s'agit de la valeur par défaut `VolumeBindingMode` et convient aux clusters qui n'appliquent pas les contraintes de topologie. Les volumes persistants sont créés sans dépendance vis-à-vis des exigences de planification du pod qui en fait la demande.
- Avec `VolumeBindingMode` réglé sur `WaitForFirstConsumer`, La création et la liaison d'un volume persistant pour une demande de volume persistant sont retardées jusqu'à ce qu'un pod qui utilise la demande de volume persistant soit planifié et créé. De cette façon, les volumes sont créés pour répondre aux contraintes de planification appliquées en fonction des besoins de topologie.



Le `WaitForFirstConsumer` le mode de liaison ne nécessite pas d'étiquettes de topologie. Il peut être utilisé indépendamment de la fonction de topologie CSI.

Ce dont vous avez besoin

Pour utiliser la topologie CSI, vous devez disposer des éléments suivants :

- Un cluster Kubernetes exécutant un "[Version Kubernetes prise en charge](#)"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedaafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedaafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Les nœuds du cluster doivent avoir des étiquettes qui permettent la prise en charge de la topologie (`topology.kubernetes.io/region` et `topology.kubernetes.io/zone`). Ces étiquettes **doivent être présentes sur les nœuds du cluster** avant d'installer Astra Trident pour qu'Astra Trident soit compatible avec la topologie.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[.metadata.name],
{.metadata.labels}}{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Étape 1 : création d'un back-end conscient de la topologie

Les systèmes back-end de stockage Astra Trident peuvent être conçus pour provisionner des volumes de manière sélective selon les zones de disponibilité. Chaque système back-end peut être équipé d'une option `supportedTopologies` bloc qui représente une liste de zones et de régions qui doivent être prises en charge. Pour les classes de stockage qui utilisent un tel backend, un volume ne sera créé que si une application est planifiée dans une région/zone prise en charge.

Voici un exemple de définition de back-end :

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-a
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-b
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-a"},
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-b"}
  ]
}
```



`supportedTopologies` sert à fournir une liste de régions et de zones par backend. Ces régions et ces zones représentent la liste des valeurs admissibles qui peuvent être fournies dans une classe de stockage. Pour les classes de stockage qui contiennent un sous-ensemble de régions et de zones qu'il fournit en back-end, Astra Trident crée un volume en interne.

Vous pouvez définir `supportedTopologies` par pool de stockage également. Voir l'exemple suivant :

```

---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-centrall
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-centrall
  topology.kubernetes.io/zone: us-centrall-a
- topology.kubernetes.io/region: us-centrall
  topology.kubernetes.io/zone: us-centrall-b
storage:
- labels:
    workload: production
    region: Iowa-DC
    zone: Iowa-DC-A
    supportedTopologies:
    - topology.kubernetes.io/region: us-centrall
      topology.kubernetes.io/zone: us-centrall-a
- labels:
    workload: dev
    region: Iowa-DC
    zone: Iowa-DC-B
    supportedTopologies:
    - topology.kubernetes.io/region: us-centrall
      topology.kubernetes.io/zone: us-centrall-b

```

Dans cet exemple, le `region` et `zone` les étiquettes correspondent à l'emplacement du pool de stockage. `topology.kubernetes.io/region` et `topology.kubernetes.io/zone` déterminer à partir de où les pools de stockage peuvent être consommés.

Étape 2 : définissez des classes de stockage qui prennent en compte la topologie

Les classes de stockage peuvent être définies en fonction des labels de topologie fournis aux nœuds du cluster, et contenir des informations de topologie. Cela déterminera les pools de stockage qui servent de candidats aux demandes de volume persistant faites et le sous-ensemble de nœuds qui peuvent utiliser les volumes provisionnés par Trident.

Voir l'exemple suivant :

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
- key: topology.kubernetes.io/zone
  values:
  - us-east1-a
  - us-east1-b
- key: topology.kubernetes.io/region
  values:
  - us-east1
parameters:
  fsType: "ext4"

```

Dans la définition de classe de stockage décrite ci-dessus, `volumeBindingMode` est défini sur `WaitForFirstConsumer`. Les demandes de volume persistant demandées pour cette classe de stockage ne seront pas traitées tant qu'elles ne seront pas référencées dans un pod. Et, `allowedTopologies` fournit les zones et la région à utiliser. Le `netapp-san-us-east1` `StorageClass` crée des ESV sur le `san-backend-us-east1` système back-end défini ci-dessus.

Étape 3 : création et utilisation d'une demande de volume persistant

Une fois la classe de stockage créée et mappée à un back-end, vous pouvez désormais créer des demandes de volume persistant.

Voir l'exemple `spec` ci-dessous :

```

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1

```

La création d'une demande de volume persistant à l'aide de ce manifeste se traduit par les éléments suivants :

```

kubect1 create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubect1 get pvc
NAME          STATUS      VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san      Pending
2s
kubect1 describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type      Reason              Age   From
  ----      -
  Normal    WaitForFirstConsumer 6s    persistentvolume-controller
waiting
for first consumer to be created before binding

```

Pour que Trident puisse créer un volume et le lier à la demande de volume persistant, utilisez la demande de volume persistant dans un pod. Voir l'exemple suivant :

```

apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          - matchExpressions:
              - key: topology.kubernetes.io/region
                operator: In
                values:
                  - us-east1
            preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 1
                preference:
                  matchExpressions:
                    - key: topology.kubernetes.io/zone
                      operator: In
                      values:
                        - us-east1-a
                        - us-east1-b
      securityContext:
        runAsUser: 1000
        runAsGroup: 3000
        fsGroup: 2000
    volumes:
      - name: voll
        persistentVolumeClaim:
          claimName: pvc-san
    containers:
      - name: sec-ctx-demo
        image: busybox
        command: [ "sh", "-c", "sleep 1h" ]
        volumeMounts:
          - name: voll
            mountPath: /data/demo
        securityContext:
          allowPrivilegeEscalation: false

```

Ce podSpec demande à Kubernetes de planifier le pod sur les nœuds présents dans le us-east1 et choisissez parmi les nœuds présents dans le us-east1-a ou us-east1-b zones.

Voir le résultat suivant :

```
kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP              NODE
NOMINATED NODE READINESS GATES
app-pod-1    1/1     Running   0          19s   192.168.25.131  node2
<none>      <none>
kubectl get pvc -o wide
NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS          AGE   VOLUMEMODE
pvc-san      Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b  300Mi
RWO          netapp-san-us-east1  48s   Filesystem
```

Mise à jour des systèmes back-end pour inclure `supportedTopologies`

Les systèmes back-end pré-existants peuvent être mis à jour pour inclure une liste de `supportedTopologies` à l'aide de `tridentctl backend update`. Cela n'affecte pas les volumes qui ont déjà été provisionnés et ne sera utilisé que pour les demandes de volume virtuel suivantes.

Trouvez plus d'informations

- ["Gestion des ressources pour les conteneurs"](#)
- ["Outil de sélection de nœud"](#)
- ["Affinité et anti-affinité"](#)
- ["Teintes et tolérances"](#)

Travailler avec des instantanés

Vous pouvez créer des copies Snapshot de volume Kubernetes VolumeSnapshot (copie de volume) de volumes persistants pour conserver des copies instantanées de volumes Trident. Vous pouvez également créer un nouveau volume, également appelé *clone*, à partir d'un snapshot de volume existant. Le snapshot de volume est pris en charge par `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, et `azure-netapp-files` pilotes.

Avant de commencer

Vous devez disposer d'un contrôleur de snapshot externe et de définitions de ressources personnalisées (CRD). Cela relève de la responsabilité de l'orchestrateur Kubernetes (par exemple : Kubeadm, GKE, OpenShift).

Si votre distribution Kubernetes n'inclut pas le contrôleur de snapshot et les CRD, reportez-vous à la [Déploiement d'un contrôleur de snapshot de volume](#).



Ne créez pas de contrôleur de snapshot si vous créez des snapshots de volume à la demande dans un environnement GKE. GKE utilise un contrôleur de snapshot caché intégré.

Étape 1 : créer un `VolumeSnapshotClass`

Cet exemple crée une classe de snapshot de volume.

```
cat snap-sc.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

Le driver Indique le conducteur CSI d'Astra Trident. `deletionPolicy` peut être `Delete` ou `Retain`. Lorsqu'il est réglé sur `Retain`, le snapshot physique sous-jacent sur le cluster de stockage est conservé même lorsque `VolumeSnapshot` l'objet a été supprimé.

Pour plus d'informations, consultez le lien [../trident-Reference/objects.html#kubernetes-volumesnapshotclass-objects](https://docs.netapp.com/us-en/trident-Reference/objects.html#kubernetes-volumesnapshotclass-objects)[`VolumeSnapshotClass`].

Étape 2 : création d'un snapshot d'un volume persistant existant

Dans cet exemple, nous allons créer un snapshot d'un volume persistant existant.

```
cat snap.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: pvcl-snap
spec:
  volumeSnapshotClassName: csi-snapclass
  source:
    persistentVolumeClaimName: pvcl
```

Dans cet exemple, le snapshot est créé pour une demande de volume persistant nommée `pvcl` et le nom du snapshot est défini sur `pvcl-snap`.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvcl-snap created

kubectl get volumesnapshots
NAME                AGE
pvcl-snap           50s
```

Cela a créé un `VolumeSnapshot` objet. Un instantané `VolumeSnapshot` est similaire à une demande de volume persistant et est associé à une `VolumeSnapshotContent` objet qui représente le snapshot réel.

Il est possible d'identifier le `VolumeSnapshotContent` objet pour le `pvcl-snap` `VolumeSnapshot` en le décrivant.

```
kubectl describe volumesnapshots pvcl-snap
Name:          pvcl-snap
Namespace:    default
.
.
.
Spec:
  Snapshot Class Name:  pvcl-snap
  Snapshot Content Name: snapcontent-e8d8a0ca-9826-11e9-9807-525400f3f660
  Source:
    API Group:
    Kind:      PersistentVolumeClaim
    Name:      pvcl
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:  true
  Restore Size:  3Gi
.
.
```

Le `Snapshot Content Name` identifie l'objet `VolumeSnapshotContent` qui sert ce snapshot. Le `Ready To Use` paramètre indique que l'instantané peut être utilisé pour créer une nouvelle demande de volume persistant.

Étape 3 : création de demandes de volume persistant à partir de copies Snapshot VolumeCas

Dans cet exemple, vous créez une demande de volume persistant à l'aide d'un snapshot :

```
cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvcl-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

`dataSource` La montre que la demande de volume persistant doit être créée à l'aide d'un Snapshot `VolumeSnapshot` nommé `pvcl-snap` comme source des données. Cela demande à Astra Trident de créer un volume persistant à partir du snapshot. Une fois la demande de volume persistant créée, elle peut être connectée à un pod et utilisée comme n'importe quel autre PVC.



Lors de la suppression d'un volume persistant avec les snapshots associés, le volume Trident correspondant est mis à jour et passe à un état « Suppression ». Pour supprimer le volume Astra Trident, il est nécessaire de supprimer les snapshots du volume.

Déploiement d'un contrôleur de snapshot de volume

Si votre distribution Kubernetes n'inclut pas le contrôleur de snapshot et les CRD, vous pouvez les déployer comme suit.

Étapes

1. Création de CRD de snapshot de volume.

```
cat snapshot-setup.sh
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Créer le contrôleur snapshot dans l'espace de noms souhaité. Modifiez les manifestes YAML ci-dessous pour modifier l'espace de noms.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```

Liens connexes

- ["Snapshots de volume"](#)
- ["VolumeSnapshotClass"](#)

Développement des volumes

Astra Trident permet aux utilisateurs de Kubernetes d'étendre leurs volumes après leur création. Trouvez des informations sur les configurations requises pour développer les volumes iSCSI et NFS.

Développez un volume iSCSI

Vous pouvez développer un volume persistant iSCSI à l'aide du mécanisme de provisionnement CSI.



L'extension de volume iSCSI est prise en charge par `ontap-san`, `ontap-san-economy`, `solidfire-san` Pilotes et requiert Kubernetes 1.16 et version ultérieure.

Présentation

L'extension d'un volume persistant iSCSI comprend les étapes suivantes :

- Modification de la définition de classe de stockage pour définir le `allowVolumeExpansion` champ à

true.

- Modification de la définition de PVC et mise à jour de `spec.resources.requests.storage` pour refléter la nouvelle taille souhaitée, qui doit être supérieure à la taille d'origine.
- Pour redimensionner le volume persistant, vous devez le connecter à un pod. Lors du redimensionnement d'un volume persistant iSCSI, deux scénarios sont possibles :
 - Si le volume persistant est connecté à un pod, Astra Trident étend le volume en back-end, reanalyse le système et redimensionne le système de fichiers.
 - Pour redimensionner un volume persistant non connecté, Astra Trident étend le volume sur le back-end. Une fois le volume de volume persistant lié à un pod, Trident analyse de nouveau le périphérique et redimensionne le système de fichiers. Kubernetes met ensuite à jour la taille de la demande de volume persistant une fois l'opération d'extension terminée.

L'exemple ci-dessous montre le fonctionnement de l'extension de PV iSCSI.

Étape 1 : configurer la classe de stockage pour prendre en charge l'extension de volume

```
cat storageclass-ontapsan.yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Pour une classe de stockage déjà existante, modifiez-la pour l'inclure `allowVolumeExpansion` paramètre.

Étape 2 : créez une demande de volume persistant avec la classe de stockage que vous avez créée

```
cat pvc-ontapsan.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

Astra Trident crée un volume persistant qui l'associe à cette demande de volume persistant.

```
kubectl get pvc
NAME          STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS  AGE
san-pvc      Bound     pvc-8a814d62-bd58-4253-b0d1-82f2885db671  1Gi
RWO          ontap-san    8s

kubectl get pv
NAME          CAPACITY  ACCESS MODES
RECLAIM POLICY  STATUS    CLAIM                STORAGECLASS  REASON  AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671  1Gi      RWO
Delete          Bound     default/san-pvc     ontap-san    10s
```

Étape 3 : définissez un pod qui fixe la demande de volume persistant

Dans cet exemple, un pod est créé et utilise le `san-pvc`.

```
kubectl get pod
NAME          READY    STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0           65s

kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
               csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Étape 4 : développez le volume persistant

Pour redimensionner la PV créée de 1Gi à 2Gi, modifiez la définition de la demande de volume persistant et mettez à jour la `spec.resources.requests.storage` à 2Gi.

```
kubectl edit pvc san-pvc
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  ...
```

Étape 5 : valider l'extension

Vous pouvez valider le bon fonctionnement de l'extension en contrôlant la taille de la demande de volume persistant, du volume persistant et du volume Astra Trident :

```

kubect1 get pvc san-pvc
NAME          STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS  AGE
san-pvc      Bound      pvc-8a814d62-bd58-4253-b0d1-82f2885db671  2Gi
RWO          ontap-san    11m
kubect1 get pv
NAME          CAPACITY  ACCESS MODES
RECLAIM POLICY  STATUS    CLAIM          STORAGECLASS  REASON  AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671  2Gi      RWO
Delete        Bound      default/san-pvc  ontap-san    12m
tridentctl get volumes -n trident
+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          |  STATE  |  MANAGED  |
+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san    |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----+-----+-----+
+-----+-----+-----+-----+

```

Développez un volume NFS

Astra Trident prend en charge l'extension de volume pour les volumes persistants NFS provisionnés sur ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, gcp-cvs, et azure-netapp-files systèmes back-end.

Étape 1 : configurer la classe de stockage pour prendre en charge l'extension de volume

Pour redimensionner un volume persistant NFS, l'administrateur doit d'abord configurer la classe de stockage afin de permettre l'extension du volume en paramétrant le `allowVolumeExpansion` champ à `true`:

```

cat storageclass-ontapnas.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true

```

Si vous avez déjà créé une classe de stockage sans cette option, vous pouvez simplement modifier la classe de stockage existante en utilisant `kubect1 edit storageclass` pour permettre l'extension de volume.

Étape 2 : créez une demande de volume persistant avec la classe de stockage que vous avez créée

```
cat pvc-ontapnas.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Astra Trident doit créer un volume persistant NFS 20MiB pour cette demande de volume persistant :

```
kubectl get pvc
NAME                STATUS    VOLUME
CAPACITY            ACCESS MODES  STORAGECLASS  AGE
ontapnas20mb       Bound     pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  20Mi
RWO                 ontapnas          9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                CAPACITY  ACCESS MODES
RECLAIM POLICY     STATUS    CLAIM                STORAGECLASS  REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  20Mi     RWO
Delete            Bound     default/ontapnas20mb  ontapnas
2m42s
```

Étape 3 : développez le volume persistant

Pour redimensionner le volume persistant 20MiB nouvellement créé à 1 Gio, modifiez la demande de volume persistant et définissez-la `spec.resources.requests.storage` à 1Go :

```
kubectl edit pvc ontapnas20mb
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  ...
```

Étape 4 : valider l'extension

Vous pouvez valider le redimensionnement correctement en contrôlant la taille de la demande de volume persistant, de la volume persistant et du volume Astra Trident :

```

kubect1 get pvc ontapnas20mb
NAME                STATUS      VOLUME
CAPACITY            ACCESS MODES  STORAGECLASS  AGE
ontapnas20mb       Bound       pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  1Gi
RWO                 ontapnas     4m44s

kubect1 get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                CAPACITY  ACCESS MODES
RECLAIM POLICY     STATUS    CLAIM          STORAGECLASS  REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  1Gi      RWO
Delete            Bound     default/ontapnas20mb  ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          |  STATE  |  MANAGED  |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file      | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true      |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

```

Importer des volumes

Vous pouvez importer des volumes de stockage existants sous forme de volume persistant Kubernetes à l'aide de `tridentctl import`.

Pilotes prenant en charge l'importation de volumes

Ce tableau décrit les pilotes qui prennent en charge l'importation de volumes et la version dans laquelle ils ont été introduits.

Conducteur	Relâchez
ontap-nas	19.04
ontap-nas-flexgroup	19.04
solidfire-san	19.04
azure-netapp-files	19.04

Conducteur	Relâchez
gcp-cvs	19.04
ontap-san	19.04

Pourquoi importer des volumes ?

L'importation d'un volume dans Trident est possible dans plusieurs champs d'application :

- Conaerisation d'une application et réutilisation de son jeu de données existant
- L'utilisation d'un clone du jeu de données pour une application éphémère
- Reconstruction d'un cluster Kubernetes défaillant
- Migration des données applicatives pendant la reprise sur incident

Comment fonctionne l'importation ?

Le fichier de demande de volume persistant est utilisé par le processus d'importation des volumes pour créer la demande de volume persistant. Au minimum, le fichier PVC doit inclure les champs Nom, espace de noms, Access modes et storageClassName, comme indiqué dans l'exemple suivant.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

Le `tridentctl` le client est utilisé pour importer un volume de stockage existant. Trident importe le volume en persistant des métadonnées de volume et en créant la demande de volume persistant et le volume persistant.

```
tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-file>
```

Pour importer un volume de stockage, spécifiez le nom du back-end Astra Trident contenant le volume, ainsi que le nom qui identifie de manière unique le volume sur le stockage (par exemple : ONTAP FlexVol, Element Volume, chemin du volume CVS). Le volume de stockage doit autoriser l'accès en lecture/écriture et être accessible par le back-end Trident spécifié de l'Astra. Le `-f` L'argument de chaîne est requis et spécifie le chemin d'accès au fichier PVC YAML ou JSON.

Lorsque Astra Trident reçoit la demande de volume d'importation, la taille du volume existant est déterminée et définie dans le volume persistant. Une fois le volume importé par le pilote de stockage, le volume persistant est créé avec un `SécurRef` dans la demande de volume persistant. La règle de récupération est initialement définie sur `retain` Dans la PV. Une fois que Kubernetes a réussi à relier la demande de volume persistant et

le volume persistant, la règle de récupération est mise à jour pour correspondre à la règle de récupération de la classe de stockage. Si la règle de récupération de la classe de stockage est `delete`, Le volume de stockage sera supprimé lorsque le volume persistant est supprimé.

Lorsqu'un volume est importé avec le `--no-manage` Argument, Trident n'effectue aucune opération supplémentaire sur la demande de volume persistant ou la volume persistant pour le cycle de vie des objets. Trident ignore les événements PV et PVC pour `--no-manage` Objets, le volume de stockage n'est pas supprimé lors de la suppression du volume persistant. D'autres opérations, telles que le clone de volume et le redimensionnement des volumes, sont également ignorées. Cette option est utile si vous souhaitez utiliser Kubernetes pour des workloads conteneurisés, mais que vous souhaitez gérer le cycle de vie du volume de stockage en dehors de Kubernetes.

Une annotation est ajoutée pour la demande de volume persistant et la volume persistant, qui servent un double objectif : indiquer l'importation du volume et gérer la demande de volume persistant. Cette annotation ne doit pas être modifiée ni supprimée.

Trident 19.07 et les versions ultérieures traitent la pièce jointe des volumes persistants et monte le volume dans le cadre de son importation. Pour les importations utilisant les versions antérieures d'Astra Trident, il n'y aura aucune opération dans le chemin d'accès aux données et l'importation de volume ne vérifiera pas si le volume peut être monté. En cas d'erreur lors de l'importation de volumes (par exemple, la classe de stockage n'est pas correcte), vous pouvez effectuer une restauration en changeant la règle de récupération du volume persistant à `retain`, Suppression de la demande de volume persistant et nouvelle tentative de la commande d'importation du volume.

`ontap-nas` et `ontap-nas-flexgroup` importations

Chaque volume créé avec le `ontap-nas` Le pilote est un FlexVol sur le cluster ONTAP. Importation de volumes FlexVol avec `ontap-nas` le pilote fonctionne de la même manière. Une FlexVol qui existe déjà sur un cluster ONTAP peut être importée en tant que `ontap-nas` PVC. De même, les volumes FlexGroup peuvent être importés en tant que `ontap-nas-flexgroup` ESV.



Pour être importé par Trident, un volume ONTAP doit être de type `rw`. Si un volume est de type `dp`, il s'agit d'un volume de destination `SnapMirror` ; vous devez rompre la relation du miroir avant d'importer le volume dans Trident.



Le `ontap-nas` le pilote ne peut pas importer et gérer les `qtrees`. Le `ontap-nas` et `ontap-nas-flexgroup` les pilotes n'autorisent pas les noms de volumes dupliqués.

Par exemple, pour importer un volume nommé `managed_volume` sur un système back-end nommé `ontap_nas`, utilisez la commande suivante :

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          | SIZE | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          | STATE | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file      | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true      |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Pour importer un volume nommé `unmanaged_volume` (sur le `ontap_nas` backend), que Trident ne gère pas, utilisez la commande suivante :

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-file>
--no-manage
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          | SIZE | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          | STATE | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file      | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false     |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Lorsque vous utilisez le `--no-manage` Argument, Trident ne renomme pas le volume ni ne valide si le volume a été monté. L'opération d'importation du volume échoue si le volume n'a pas été monté manuellement.



Un bogue existant précédemment avec l'importation de volumes avec Unixpermissions personnalisées a été corrigé. Vous pouvez spécifier `unixpermissions` dans votre définition de PVC ou configuration back-end et demander à Astra Trident d'importer le volume en conséquence.

ontap-san importer

Astra Trident peut également importer des volumes FlexVol SAN de ONTAP contenant un seul LUN. Ceci est cohérent avec le `ontap-san` Pilote, qui crée un FlexVol pour chaque demande de volume persistant et une LUN au sein de la FlexVol. Vous pouvez utiliser le `tridentctl import` commande de la même manière que dans les autres cas :

- Inclure le nom du `ontap-san` back-end.


```
tridentctl import volume element_default element-managed -f pvc-basic-import.yaml -n trident -d
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          | SIZE | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          | STATE | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block   | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true   |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```



Le pilote d'élément prend en charge les noms de volume dupliqués. S'il existe des noms de volume dupliqués, le processus d'importation de volume de Trident renvoie une erreur. Pour contourner ce problème, clonez le volume et fournissez un nom de volume unique. Importez ensuite le volume cloné.

gcp-cvs importer



Pour importer un volume sauvegardé par NetApp Cloud Volumes Service dans GCP, identifiez le volume par son chemin d'accès au volume et non son nom.

Pour importer un `gcp-cvs` volume sur le back-end appelé `gcpcvs_YEppr` avec le chemin de volume de `adroit-jolly-swift`, utilisez la commande suivante :

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-file> -n trident
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          | SIZE | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          | STATE | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | gcp-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true         |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```



Le chemin du volume correspond à la partie du chemin d'exportation du volume après `:/`. Par exemple, si le chemin d'exportation est `10.0.0.1:/adroit-jolly-swift`, le chemin du volume est `adroit-jolly-swift`.

azure-netapp-files importer

Pour importer un azure-netapp-files volume sur le back-end appelé `azurenetaappfiles_40517` avec le chemin de volume `importvoll1`, exécutez la commande suivante :

```
tridentctl import volume azurenetaappfiles_40517 importvoll1 -f <path-to-pvc-file> -n trident
```

```
+-----+-----+-----+
+-----+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS |
PROTOCOL |          BACKEND UUID          |  STATE  |  MANAGED  |
+-----+-----+-----+
+-----+-----+-----+-----+-----+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage |
file      | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true      |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```



Le chemin de volume du volume ANF est présent dans le chemin de montage après `:/`. Par exemple, si le chemin de montage est `10.0.0.2:/importvoll1`, le chemin du volume est `importvoll1`.

Partager un volume NFS entre les espaces de noms

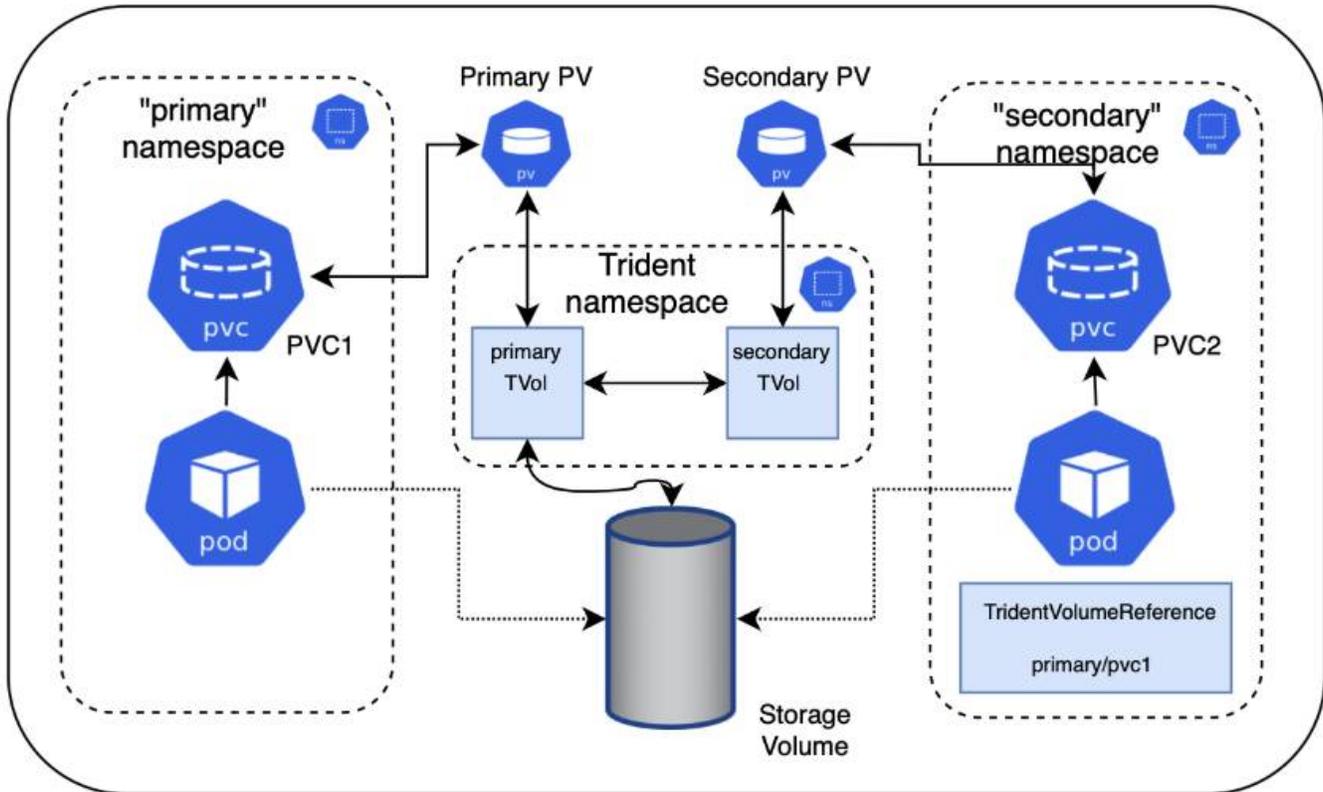
Avec Astra Trident, vous pouvez créer un volume dans un espace de noms principal et le partager dans un ou plusieurs espaces de noms secondaires.

Caractéristiques

Le système Astra TridentVolumeReference CR vous permet de partager en toute sécurité des volumes NFS ReadWriteMany (RWX) sur un ou plusieurs espaces de noms Kubernetes. Cette solution Kubernetes-native présente plusieurs avantages :

- Plusieurs niveaux de contrôle d'accès pour assurer la sécurité
- Fonctionne avec tous les pilotes de volume NFS Trident
- Pas de dépendance à `tridentctl` ou à toute autre fonctionnalité Kubernetes non native

Ce schéma illustre le partage de volumes NFS entre deux espaces de noms Kubernetes.



Démarrage rapide

Vous pouvez configurer le partage de volumes NFS en quelques étapes seulement.

1

Configurez la demande de volume persistant source pour partager le volume

Le propriétaire de l'espace de noms source autorise l'accès aux données dans la demande de volume persistant source.

2

Accorder l'autorisation de créer une demande de modification dans l'espace de noms de destination

L'administrateur de cluster accorde l'autorisation au propriétaire de l'espace de noms de destination pour créer le CR TridentVolumeReference.

3

Créer TridentVolumeReference dans l'espace de noms de destination

Le propriétaire de l'espace de noms de destination crée le CR TridentVolumeReference pour faire référence au PVC source.

4

Créer le PVC subalterne dans l'espace de noms de destination

Le propriétaire de l'espace de noms de destination crée le PVC subalterne pour utiliser la source de données à partir du PVC source.

Configurer les espaces de noms source et de destination

Pour garantir la sécurité, le partage de l'espace de noms croisé nécessite une collaboration et une action du propriétaire de l'espace de noms source, de l'administrateur de cluster et du propriétaire de l'espace de noms de destination. Le rôle utilisateur est désigné dans chaque étape.

Étapes

1. **Propriétaire de l'espace de noms source:** Créez le PVC (`pvc1`) dans l'espace de noms source qui accorde l'autorisation de partager avec l'espace de noms de destination (`namespace2`) à l'aide de l'`shareToNamespace` annotation.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Astra Trident crée le volume persistant et son volume de stockage NFS back-end.



- Vous pouvez partager le PVC sur plusieurs espaces de noms à l'aide d'une liste délimitée par des virgules. Par exemple :
`trident.netapp.io/shareToNamespace: namespace2, namespace3, namespace4.`
- Vous pouvez partager avec tous les espaces de noms à l'aide de `*`. Par exemple :
`trident.netapp.io/shareToNamespace: *`
- Vous pouvez mettre à jour le PVC pour inclure le `shareToNamespace` annotation à tout moment.

2. **Cluster admin:** Créez le rôle personnalisé et kubeconfig pour accorder l'autorisation au propriétaire de l'espace de noms de destination de créer le CR `TridentVolumeReference` dans l'espace de noms de destination.
3. **Propriétaire de l'espace de noms de destination :** Créez un CR `TridentVolumeReference` dans l'espace de noms de destination qui fait référence à l'espace de noms source `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propriétaire de l'espace de noms de destination:** Créer un PVC (`pvc2`) dans l'espace de noms de destination (`namespace2`) à l'aide de l' `shareFromPVC` Annotation pour désigner la PVC source.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```



La taille du PVC de destination doit être inférieure ou égale à la PVC source.

Résultats

Découvrez Astra Trident `shareFromPVC` Annotation sur la demande de volume persistant de destination et création du volume persistant de destination en tant que volume subalterne, sans ressource de stockage correspondant au volume persistant source et partage la ressource de stockage PV source. La demande de volume persistant et la demande de volume persistant de destination apparaissent comme normales.

Supprimer un volume partagé

Vous pouvez supprimer un volume partagé entre plusieurs namespaces. Astra Trident supprimera l'accès au volume de l'espace de noms source et maintiendra l'accès aux autres espaces de noms qui partagent le volume. Lorsque tous les namespaces qui référencent le volume sont supprimés, Astra Trident supprime le volume.

Utiliser `tridentctl get` pour interroger des volumes subordonnés

À l'aide du `tridentctl` vous pouvez exécuter l' `get` commande pour obtenir des volumes subordonnés.

Pour plus d'informations, consultez le lien [../trident-Reference/tridentctl.html](https://trident-reference.netapp.com/tridentctl.html) [tridentctl commandes et options].

Usage:

```
tridentctl get [option]
```

Alarmes :

- `-h, --help`: Aide pour les volumes.
- `--parentOfSubordinate string`: Limiter la requête au volume source subordonné.
- `--subordinateOf string`: Limiter la requête aux subordonnés du volume.

Limites

- Astra Trident ne peut pas empêcher les espaces de noms de destination d'écrire sur le volume partagé. Nous vous recommandons d'utiliser un verrouillage de fichiers ou d'autres processus pour éviter d'écraser les données du volume partagé.
- Vous ne pouvez pas révoquer l'accès au PVC source en retirant le `shareToNamespace` ou `shareFromNamespace` annotations ou suppression du `TridentVolumeReference` CR. Pour annuler l'accès, vous devez supprimer le PVC subalterne.
- Les snapshots, clones et la mise en miroir ne sont pas possibles sur les volumes subordonnés.

Pour en savoir plus

Pour en savoir plus sur l'accès aux volumes multi-espaces de noms :

- Visitez "[Partage de volumes entre les espaces de noms : dites bonjour à l'accès aux volumes situés à l'échelle d'un espace de noms](#)".
- Regardez la démo sur "[NetAppTV](#)".

Contrôle d'Astra Trident

Astra Trident fournit un ensemble de terminaux de metrics de Prometheus que vous pouvez utiliser pour surveiller les performances d'Astra Trident.

Avec les metrics d'Astra Trident, vous pouvez :

- Surveillez l'état et la configuration d'Astra Trident. Vous avez la possibilité d'examiner la réussite des opérations et de savoir si elles peuvent communiquer avec les systèmes back-end comme prévu.
- Examiner les informations d'utilisation du système back-end et comprendre le nombre de volumes provisionnés sur un système back-end, ainsi que la quantité d'espace consommé, etc.
- Conservez un mappage de la quantité de volumes provisionnés sur les systèmes back-end disponibles.
- Suivi des performances. Découvrez le temps nécessaire pour communiquer avec Astra Trident aux systèmes back-end et effectuer les opérations.



Par défaut, les metrics de Trident sont visibles sur le port cible 8001 au `/metrics` point final. Ces mesures sont **activées par défaut** lors de l'installation de Trident.

Ce dont vous avez besoin

- Cluster Kubernetes avec Astra Trident installé.
- Instance Prometheus. Il peut s'agir d'un ["Déploiement conteneurisé par Prometheus"](#) Vous pouvez également utiliser Prometheus en tant que ["application native"](#).

Étape 1 : définir une cible Prometheus

Vous devez définir une cible Prometheus pour collecter les metrics et obtenir des informations sur les systèmes back-end gérés par Trident, les volumes qu'elle crée, etc. C'est ça ["Blog"](#) Vous apprendrez comment utiliser Prometheus et Grafana avec Astra Trident pour récupérer des metrics. Ce blog explique comment exécuter Prometheus en tant qu'opérateur dans votre cluster Kubernetes et créer un ServiceMonitor pour obtenir les metrics d'Astra Trident.

Étape 2 : créer un ServiceMonitor Prometheus

Pour consommer les metrics Trident, vous devez créer un ServiceMonitor Prometheus qui surveille la `trident-csi` service et écoute sur le `metrics` port. Un exemple de ServiceMonitor se présente comme suit :

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
    release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
      - trident
  endpoints:
    - port: metrics
      interval: 15s
```

Cette définition de ServiceMonitor récupère les mesures renvoyées par le `trident-csi` et recherche spécifiquement le `metrics` point d'extrémité du service. Par conséquent, Prometheus est désormais configuré pour être en mesure de comprendre les metrics d'Astra Trident.

Outre les metrics directement disponibles par Astra Trident, kubelet expose beaucoup `kubelet_volume_*` metrics via son propre terminal de metrics. Kubelet peut fournir des informations sur les volumes reliés, ainsi que sur les pods et autres opérations internes qu'elle gère. Voir ["ici"](#).

Étape 3 : interroger les mesures Trident avec PromQL

PromQL est bon pour la création d'expressions qui renvoient des séries chronologiques ou des données tabulaires.

Voici quelques questions PromQL que vous pouvez utiliser :

Accédez aux informations sur l'état de santé de Trident

- **Pourcentage de réponses HTTP 2XX d'Astra Trident**

```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()  
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- **Pourcentage de réponses REST d'Astra Trident par le code d'état**

```
(sum (trident_rest_ops_seconds_total_count) by (status_code) / scalar  
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- **Durée moyenne en ms des opérations effectuées par Astra Trident**

```
sum by (operation)  
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by  
(operation)  
(trident_operation_duration_milliseconds_count{success="true"})
```

Découvrez les informations d'utilisation d'Astra Trident

- **Taille moyenne du volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Espace volume total provisionné par chaque back-end**

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

Utiliser individuellement le volume



Cette activation est uniquement possible si les indicateurs kubelet sont également collectés.

- **Pourcentage d'espace utilisé pour chaque volume**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Découvrez la télémétrie AutoSupport d'Astra Trident

Par défaut, Astra Trident envoie des metrics de Prometheus et des informations de base back-end à NetApp dans un rythme quotidien.

- Pour empêcher Astra Trident d'envoyer des metrics de Prometheus et des informations de base back-end à NetApp, passez le `--silence-autosupport` Indicateur lors de l'installation d'Astra Trident.
- Astra Trident peut également envoyer des journaux de conteneur à NetApp support à la demande via `tridentctl send autosupport`. Vous devrez déclencher Astra Trident pour télécharger ses journaux. Avant de communiquer les journaux, vous devez accepter celui de NetApp <https://www.netapp.com/company/legal/privacy-policy/>["politique de confidentialité"].
- Sauf mention contraire, Astra Trident extrait les journaux des 24 dernières heures.
- Vous pouvez spécifier la période de rétention du journal avec le `--since` drapeau. Par exemple : `tridentctl send autosupport --since=1h`. Ces informations sont collectées et envoyées via un `trident-autosupport` Conteneur installé avec Astra Trident. Vous pouvez obtenir l'image conteneur à ["AutoSupport Trident"](#).
- Le AutoSupport Trident ne collecte pas et ne transmet pas d'informations à caractère personnel (PII) ou de données personnelles. Il est fourni avec un ["CLUF"](#) qui ne s'applique pas à l'image du conteneur Trident. Pour en savoir plus sur l'engagement de NetApp en faveur de la sécurité et de la confiance des données ["ici"](#).

Voici un exemple de charge utile envoyée par Astra Trident :

```
---
items:
- backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
  protocol: file
  config:
    version: 1
    storageDriverName: ontap-nas
    debug: false
    debugTraceFlags:
    disableDelete: false
    serialNumbers:
    - nwkvzfanek_SN
    limitVolumeSize: ''
  state: online
  online: true
```

- Les messages AutoSupport sont envoyés au terminal AutoSupport de NetApp. Si vous utilisez un registre privé pour stocker des images de conteneur, vous pouvez utiliser le `--image-registry` drapeau.
- Vous pouvez également configurer des URL proxy en générant les fichiers YAML d'installation. Pour ce

faire, utilisez `tridentctl install --generate-custom-yaml` Pour créer les fichiers YAML et ajouter le `--proxy-url` argument pour le `trident-autosupport` conteneur `trident-deployment.yaml`.

Désactivation des metrics d'Astra Trident

Pour désactiver** les mesures signalées, vous devez générer des YAML personnalisées (à l'aide de l'`--generate-custom-yaml` marquer) et modifiez-les pour supprimer le `--metrics` indicateur d'être appelé pour le `trident-main` conteneur.

Informations sur le copyright

Copyright © 2025 NetApp, Inc. Tous droits réservés. Imprimé aux États-Unis. Aucune partie de ce document protégé par copyright ne peut être reproduite sous quelque forme que ce soit ou selon quelque méthode que ce soit (graphique, électronique ou mécanique, notamment par photocopie, enregistrement ou stockage dans un système de récupération électronique) sans l'autorisation écrite préalable du détenteur du droit de copyright.

Les logiciels dérivés des éléments NetApp protégés par copyright sont soumis à la licence et à l'avis de non-responsabilité suivants :

CE LOGICIEL EST FOURNI PAR NETAPP « EN L'ÉTAT » ET SANS GARANTIES EXPRESSES OU TACITES, Y COMPRIS LES GARANTIES TACITES DE QUALITÉ MARCHANDE ET D'ADÉQUATION À UN USAGE PARTICULIER, QUI SONT EXCLUES PAR LES PRÉSENTES. EN AUCUN CAS NETAPP NE SERA TENU POUR RESPONSABLE DE DOMMAGES DIRECTS, INDIRECTS, ACCESSOIRES, PARTICULIERS OU EXEMPLAIRES (Y COMPRIS L'ACHAT DE BIENS ET DE SERVICES DE SUBSTITUTION, LA PERTE DE JOUISSANCE, DE DONNÉES OU DE PROFITS, OU L'INTERRUPTION D'ACTIVITÉ), QUELLES QU'EN SOIENT LA CAUSE ET LA DOCTRINE DE RESPONSABILITÉ, QU'IL S'AGISSE DE RESPONSABILITÉ CONTRACTUELLE, STRICTE OU DÉLICTELLE (Y COMPRIS LA NÉGLIGENCE OU AUTRE) DÉCOULANT DE L'UTILISATION DE CE LOGICIEL, MÊME SI LA SOCIÉTÉ A ÉTÉ INFORMÉE DE LA POSSIBILITÉ DE TELS DOMMAGES.

NetApp se réserve le droit de modifier les produits décrits dans le présent document à tout moment et sans préavis. NetApp décline toute responsabilité découlant de l'utilisation des produits décrits dans le présent document, sauf accord explicite écrit de NetApp. L'utilisation ou l'achat de ce produit ne concède pas de licence dans le cadre de droits de brevet, de droits de marque commerciale ou de tout autre droit de propriété intellectuelle de NetApp.

Le produit décrit dans ce manuel peut être protégé par un ou plusieurs brevets américains, étrangers ou par une demande en attente.

LÉGENDE DE RESTRICTION DES DROITS : L'utilisation, la duplication ou la divulgation par le gouvernement sont sujettes aux restrictions énoncées dans le sous-paragraphe (b)(3) de la clause Rights in Technical Data-Noncommercial Items du DFARS 252.227-7013 (février 2014) et du FAR 52.227-19 (décembre 2007).

Les données contenues dans les présentes se rapportent à un produit et/ou service commercial (tel que défini par la clause FAR 2.101). Il s'agit de données propriétaires de NetApp, Inc. Toutes les données techniques et tous les logiciels fournis par NetApp en vertu du présent Accord sont à caractère commercial et ont été exclusivement développés à l'aide de fonds privés. Le gouvernement des États-Unis dispose d'une licence limitée irrévocable, non exclusive, non cessible, non transférable et mondiale. Cette licence lui permet d'utiliser uniquement les données relatives au contrat du gouvernement des États-Unis d'après lequel les données lui ont été fournies ou celles qui sont nécessaires à son exécution. Sauf dispositions contraires énoncées dans les présentes, l'utilisation, la divulgation, la reproduction, la modification, l'exécution, l'affichage des données sont interdits sans avoir obtenu le consentement écrit préalable de NetApp, Inc. Les droits de licences du Département de la Défense du gouvernement des États-Unis se limitent aux droits identifiés par la clause 252.227-7015(b) du DFARS (février 2014).

Informations sur les marques commerciales

NETAPP, le logo NETAPP et les marques citées sur le site <http://www.netapp.com/TM> sont des marques déposées ou des marques commerciales de NetApp, Inc. Les autres noms de marques et de produits sont des marques commerciales de leurs propriétaires respectifs.