Utilisez Trident

Trident

NetApp
January 15, 2026

Sommaire

Utilisez Trident

Préparer le nœud de travail

Tous les nœuds de travail du cluster Kubernetes doivent pouvoir monter les volumes que vous avez provisionnés pour vos pods. Pour préparer les nœuds de travail, vous devez installer les outils NFS, iSCSI, NVMe/TCP ou FC en fonction du pilote que vous avez sélectionné.

Choisir les bons outils

Si vous utilisez une combinaison de pilotes, vous devez installer tous les outils requis pour vos pilotes. Les versions récentes de Red Hat Enterprise Linux CoreOS (RHCOS) intègrent ces outils par défaut.

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` .

Outils NVMe

"Installez les outils NVMe"si vous utilisez `ontap-san` pour le protocole NVMe/TCP (Nonvolatile Memory Express sur TCP).

NetApp recommande ONTAP 9.12 ou une version ultérieure pour NVMe/TCP.

Outils SCSI sur FC

Se référer à"Méthodes de configuration des hôtes SAN FC et FC-NVMe" pour plus d'informations sur la configuration de vos hôtes SAN FC et FC-NVMe.

"Installez les outils FC"si vous utilisez `ontap-san` avec sanType `fcp` (SCSI sur FC).

Points à prendre en compte : * SCSI sur FC est pris en charge dans les environnements OpenShift et KubeVirt. * Le protocole SCSI sur FC n'est pas pris en charge sur Docker. * L'auto-réparation iSCSI n'est pas applicable à SCSI sur FC.

Découverte de services de nœuds

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

La découverte de services de nœuds identifie les services découverts, mais ne garantit pas que ces services soient correctement configurés. Inversement, l'absence de service détecté ne garantit pas l'échec du montage du volume.

Revoir les événements

Trident crée des événements pour que le nœud puisse identifier les services découverts. Pour consulter ces événements, exécutez :

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

Avis sur les services découverts

Trident identifie les services activés pour chaque nœud sur le CR du nœud Trident . Pour afficher les services détectés, exécutez :

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

Volumes NFS

Installez les outils NFS en utilisant les commandes correspondant à votre système d'exploitation. Assurez-vous que le service NFS est démarré au démarrage du système.

RHEL 8+

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

Ubuntu

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

Redémarrez vos nœuds de travail après l'installation des outils NFS pour éviter les échecs lors de l'attachement des volumes aux conteneurs.

volumes iSCSI

Trident peut établir automatiquement une session iSCSI, analyser les LUN, découvrir les périphériques multipath, les formater et les monter sur un pod.

Capacités d'auto-réparation iSCSI

Pour les systèmes ONTAP , Trident exécute une auto-réparation iSCSI toutes les cinq minutes afin de :

1. **Identifier** l'état de session iSCSI souhaité et l'état de session iSCSI actuel.
2. **Comparer** l'état souhaité à l'état actuel pour identifier les réparations nécessaires. Trident détermine les priorités de réparation et les situations où il convient d'anticiper les réparations.
3. **Effectuer les réparations** nécessaires pour ramener l'état actuel de la session iSCSI à l'état souhaité.

Les journaux d'activité d'auto-guérison se trouvent dans le `trident-main` conteneur sur le pod Daemonset respectif. Pour consulter les journaux, vous devez avoir configuré `debug` à « vrai » lors de l'installation de Trident .

Les capacités d'auto-réparation de Trident iSCSI peuvent contribuer à prévenir :

- Sessions iSCSI obsolètes ou défaillantes pouvant survenir suite à un problème de connectivité réseau. En cas de session inactive, Trident attend sept minutes avant de se déconnecter afin de rétablir la connexion avec un portail.

> ⓘ Par exemple, si les secrets CHAP étaient renouvelés sur le contrôleur de stockage et que le réseau perdait sa connectivité, les anciens secrets CHAP (obsolètes) pourraient persister. L'auto-réparation peut détecter cela et rétablir automatiquement la session pour appliquer les secrets CHAP mis à jour.

- Sessions iSCSI manquantes
- LUN manquants

Points à prendre en compte avant de mettre à niveau Trident

- Si seuls les igroups par nœud (introduits dans la version 23.04 et suivantes) sont utilisés, l'auto-réparation iSCSI lancera des analyses SCSI pour tous les périphériques du bus SCSI.
- Si seuls les igroups à portée backend (dépréciés depuis la version 23.04) sont utilisés, l'auto-réparation iSCSI lancera des analyses SCSI pour les ID LUN exacts sur le bus SCSI.
- Si une combinaison d'igroups par nœud et d'igroups à portée dorsale est utilisée, l'auto-réparation iSCSI lancera des analyses SCSI pour les ID LUN exacts sur le bus SCSI.

Installez les outils iSCSI

Installez les outils iSCSI en utilisant les commandes correspondant à votre système d'exploitation.

Avant de commencer

- Chaque nœud du cluster Kubernetes doit avoir un IQN unique. **Ceci est une condition préalable nécessaire**.
- Si vous utilisez RHCOS version 4.5 ou ultérieure, ou une autre distribution Linux compatible RHEL, avec le `solidfire-san` Si le pilote et Element OS 12.5 ou antérieur sont installés, assurez-vous que l'algorithme d'authentification CHAP est défini sur MD5. `/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
```

- Lors de l'utilisation de nœuds de travail exécutant RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) avec des volumes persistants iSCSI, spécifiez le `discard` L'option mountOption dans StorageClass permet d'effectuer une récupération d'espace en ligne. Se référer à "Documentation Red Hat" .
- Assurez-vous d'avoir effectué la mise à jour vers la dernière version de `multipath-tools` .

RHEL 8+

1. Installez les paquets système suivants :

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

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

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

3. Configurer la numérisation en mode manuel :

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

4. Activer le multipathing :

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

Assurer `/etc/multipath.conf` contient `find_multipaths no` sous `defaults` .

5. Assurez-vous 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 paquets système suivants :

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

2. Vérifiez que la version d'open-iscsi est 2.0.874-5ubuntu2.10 ou ultérieure (pour bionic) ou 2.0.874-7.1ubuntu6.1 ou ultérieure (pour focal) :

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

3. Configurer la numérisation en mode manuel :

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

4. Activer le multipathing :

```
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
```

Assurer `/etc/multipath.conf` contient `find_multipaths no` sous `defaults` .

5. Assurez-vous que `open-iscsi` et `multipath-tools` sont activés 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.

Configurer ou désactiver l'autoréparation iSCSI

Vous pouvez configurer les paramètres d'auto-réparation iSCSI Trident suivants pour corriger les sessions obsolètes :

- **Intervalle d'auto-réparation iSCSI** : Détermine la fréquence à laquelle l'auto-réparation iSCSI est invoquée (par défaut : 5 minutes). Vous pouvez le configurer pour qu'il s'exécute plus fréquemment en définissant un nombre plus petit, ou moins fréquemment en définissant un nombre plus grand.

Définir l'intervalle d'auto-réparation iSCSI à 0 arrête complètement l'auto-réparation iSCSI. Nous ne recommandons pas de désactiver l'auto-réparation iSCSI ; elle ne doit être désactivée que dans certains cas, lorsque l'auto-réparation iSCSI ne fonctionne pas comme prévu ou à des fins de débogage.

- **Délai d'attente d'auto-réparation iSCSI** : Détermine la durée pendant laquelle l'auto-réparation iSCSI attend avant de se déconnecter d'une session défaillante et de tenter de se reconnecter (par défaut : 7 minutes). Vous pouvez le configurer sur un nombre plus élevé afin que les sessions identifiées comme non saines doivent attendre plus longtemps avant d'être déconnectées, puis qu'une tentative de reconnexion soit effectuée, ou sur un nombre plus petit pour se déconnecter et se reconnecter plus tôt.

Barre

Pour configurer ou modifier les paramètres d'auto-réparation iSCSI, transmettez le `iscsiSelfHealingInterval` et `iscsiSelfHealingWaitTime` paramètres lors de l'installation ou de la mise à jour de Helm.

L'exemple suivant configure l'intervalle d'auto-réparation iSCSI à 3 minutes et le délai d'attente d'auto-réparation à 6 minutes :

```
helm install trident trident-operator-100.2506.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

Pour configurer ou modifier les paramètres d'auto-réparation iSCSI, transmettez le `iscsi-self-healing-interval` et `iscsi-self-healing-wait-time` paramètres lors de l'installation ou de la mise à jour de tridentctl.

L'exemple suivant configure l'intervalle d'auto-réparation iSCSI à 3 minutes et le délai d'attente d'auto-réparation à 6 minutes :

```
tridentctl install --iscsi-self-healing-interval=3m0s --iscsi-self
-healing-wait-time=6m0s -n trident
```

Volumes NVMe/TCP

Installez les outils NVMe en utilisant les commandes correspondant à votre système d'exploitation.

- NVMe nécessite RHEL 9 ou une version ultérieure.
- Si la version du noyau de votre nœud Kubernetes est trop ancienne ou si le package NVMe n'est pas disponible pour votre version du noyau, vous devrez peut-être mettre à jour la version du noyau de votre nœud vers une version incluant le package NVMe.

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Vérifier l'installation

Après l'installation, vérifiez que chaque nœud du cluster Kubernetes possède un NQN unique à l'aide de la commande :

```
cat /etc/nvme/hostnqn
```

> Trident modifie le `ctrl_device_tmo` valeur permettant de s'assurer que NVMe ne renonce pas au chemin en cas de panne. Ne modifiez pas ce paramètre.

Volumes SCSI sur FC

Vous pouvez désormais utiliser le protocole Fibre Channel (FC) avec Trident pour provisionner et gérer les ressources de stockage sur le système ONTAP .

Prérequis

Configurez les paramètres réseau et de nœud requis pour FC.

Paramètres réseau

1. Obtenez le WWPN des interfaces cibles. Se référer à "affichage de l'interface réseau" pour plus d'informations.
2. Obtenez le WWPN pour les interfaces sur l'initiateur (hôte).

   Consultez les utilitaires correspondants du système d'exploitation hôte.

3. Configurez le zonage sur le commutateur FC en utilisant les WWPN de l'hôte et de la cible.

   Veuillez vous référer à la documentation du fournisseur du commutateur concerné pour plus d'informations.

   Pour plus de détails, veuillez consulter la documentation ONTAP suivante :

   - "Aperçu du zonage Fibre Channel et FCoE"

- "Méthodes de configuration des hôtes SAN FC et FC-NVMe"

Installez les outils FC

Installez les outils FC en utilisant les commandes correspondant à votre système d'exploitation.

- Lors de l'utilisation de nœuds de travail exécutant RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) avec des volumes persistants FC, spécifiez le `discard` L'option mountOption dans StorageClass permet d'effectuer une récupération d'espace en ligne. Se référer à "Documentation Red Hat" .

RHEL 8+

1. Installez les paquets système suivants :

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. Activer le multipathing :

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

Assurer `/etc/multipath.conf` contient `find_multipaths no` sous `defaults` .

3. Assurez-vous que `multipathd` est en cours d'exécution :

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

Ubuntu

1. Installez les paquets système suivants :

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

2. Activer le multipathing :

```
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
```

Assurer `/etc/multipath.conf` contient `find_multipaths no` sous `defaults` .

3. Assurez-vous que `multipath-tools` est activé et en cours d'exécution :

```
sudo systemctl status multipath-tools
```

Configurer et gérer les backends

Configurer les backends

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

Trident propose automatiquement des pools de stockage provenant de systèmes backend qui correspondent aux exigences définies par une classe de stockage. Apprenez à configurer le backend de votre système de stockage.

- "Configurer un backend Azure NetApp Files"
- "Configurer un backend Google Cloud NetApp Volumes"
- "Configurer un Cloud Volumes Service pour le backend de Google Cloud Platform"
- "Configurer un backend NetApp HCI ou SolidFire"
- "Configurez un backend avec les pilotes NAS ONTAP ou Cloud Volumes ONTAP."
- "Configurez un backend avec les pilotes SAN ONTAP ou Cloud Volumes ONTAP"
- "Utiliser Trident avec Amazon FSx for NetApp ONTAP"

Azure NetApp Files

Configurer un backend Azure NetApp Files

Vous pouvez configurer Azure NetApp Files comme backend pour Trident. Vous pouvez connecter des volumes NFS et SMB à l'aide d'un backend Azure NetApp Files . Trident prend également en charge la gestion des informations d'identification à l'aide d'identités gérées pour les clusters Azure Kubernetes Services (AKS).

Détails du pilote Azure NetApp Files

Trident fournit les pilotes de stockage Azure NetApp Files suivants pour communiquer avec le cluster. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`azure-netapp-files`	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Considérations

- Le service Azure NetApp Files ne prend pas en charge les volumes inférieurs à 50 Gio. Trident crée automatiquement des volumes de 50 Gio si un volume plus petit est demandé.
- Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows.

Gestion des identités pour AKS

Trident soutient"identités gérées" pour les clusters Azure Kubernetes Services. Pour bénéficier de la gestion simplifiée des identifiants offerte par les identités gérées, vous devez disposer de :

- Un cluster Kubernetes déployé à l'aide d'AKS
- Identités gérées configurées sur le cluster Kubernetes AKS
- Trident installé qui comprend le `cloudProvider` préciser `"Azure"` .

Opérateur Trident

Pour installer Trident à l'aide de l'opérateur Trident , modifiez `tridentorchestrator_cr.yaml` définir `cloudProvider` à `"Azure"` . Par exemple:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
```

Barre

L'exemple suivant installe les ensembles Trident `cloudProvider` vers Azure en utilisant la variable d'environnement `$CP` :

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

L'exemple suivant installe Trident et configure les `cloudProvider` drapeau à `Azure` :

```
tridentctl install --cloud-provider="Azure" -n trident
```

Identité cloud pour AKS

L'identité cloud permet aux pods Kubernetes d'accéder aux ressources Azure en s'authentifiant en tant qu'identité de charge de travail au lieu de fournir des informations d'identification Azure explicites.

Pour tirer parti de l'identité cloud dans Azure, vous devez disposer de :

- Un cluster Kubernetes déployé à l'aide d'AKS

- L’identité de la charge de travail et l’émetteur OIDC sont configurés sur le cluster Kubernetes AKS.
- Trident installé qui comprend le `cloudProvider` préciser `"Azure"` et `cloudIdentity` spécification de l’identité de la charge de travail

Opérateur Trident

Pour installer Trident à l'aide de l'opérateur Trident , modifiez `tridentorchestrator_cr.yaml` définir `cloudProvider` à `"Azure"` et ensemble `cloudIdentity` à `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Par exemple:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxx' # Edit
```

Barre

Définissez les valeurs des indicateurs **cloud-provider (CP)** et **cloud-identity (CI)** à l'aide des variables d'environnement suivantes :

```
export CP="Azure"
export CI="'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx'"
```

L'exemple suivant installe Trident et configure `cloudProvider` vers Azure en utilisant la variable d'environnement `$CP` et établit le `cloudIdentity` en utilisant la variable d'environnement `$CI` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>tridentctl</code>

Définissez les valeurs des indicateurs **cloud provider** et **cloud identity** à l'aide des variables d'environnement suivantes :

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx"
```

L'exemple suivant installe Trident et configure les `cloud-provider` drapeau à `$CP` , et `cloud-identity` à `$CI` :

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Préparez-vous à configurer un backend Azure NetApp Files

Avant de pouvoir configurer votre backend Azure NetApp Files , vous devez vous assurer que les exigences suivantes sont respectées.

Prérequis pour les volumes NFS et SMB

Si vous utilisez Azure NetApp Files pour la première fois ou dans un nouvel emplacement, une configuration initiale est nécessaire pour configurer Azure NetApp Files et créer un volume NFS. Se référer à "Azure : Configurez Azure NetApp Files et créez un volume NFS" .

Pour configurer et utiliser un "Azure NetApp Files" Côté serveur, vous avez besoin des éléments suivants :

- `subscriptionID`, `tenantID` , `clientID` , `location` , et `clientSecret` sont facultatives lors de l'utilisation d'identités gérées sur un cluster AKS.
- `tenantID`, `clientID` , et `clientSecret` sont facultatives lors de l'utilisation d'une identité cloud sur un cluster AKS.

- Un pool de capacité. Se référer à"Microsoft : Créer un pool de capacité pour Azure NetApp Files" .
- Un sous-réseau délégué à Azure NetApp Files. Se référer à"Microsoft : Déléguer un sous-réseau à Azure NetApp Files" .
- `subscriptionID`à partir d'un abonnement Azure avec Azure NetApp Files activé.
- `tenantID`, `clientID` , et `clientSecret` d'un"Inscription à l'application" dans Azure Active Directory avec les autorisations suffisantes pour le service Azure NetApp Files . L'enregistrement de l'application doit utiliser soit :
  - Le rôle de propriétaire ou de contributeur"prédéfini par Azure" .
  - UN"Rôle de contributeur personnalisé" au niveau de l'abonnement(`assignableScopes`) avec les autorisations suivantes, limitées à ce que Trident exige. Après avoir créé le rôle personnalisé,"Attribuez le rôle à l'aide du portail Azure" .

Rôle de contributeur personnalisé

```
{
  "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/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/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": []
        }
      ]
    }
}
```

- L'Azur `location` qui contient au moins un "sous-réseau délégué" . À partir de Trident 22.01, le `location` Ce paramètre est un champ obligatoire au niveau supérieur du fichier de configuration du backend. Les valeurs d'emplacement spécifiées dans les pools virtuels sont ignorées.
- À utiliser `Cloud Identity` , obtenez le `client ID` d'un "identité gérée attribuée par l'utilisateur" et spécifiez cet ID dans `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Exigences supplémentaires pour les volumes PME

Pour créer un volume SMB, vous devez disposer de :

- Active Directory configuré et connecté à Azure NetApp Files. Se référer à"Microsoft : Créer et gérer des connexions Active Directory pour Azure NetApp Files" .
- Un cluster Kubernetes avec un nœud contrôleur Linux et au moins un nœud de travail Windows exécutant Windows Server 2022. Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows.
- Au moins un secret Trident contenant vos informations d'identification Active Directory est nécessaire pour Azure NetApp Files puisse s'authentifier auprès d'Active Directory. Générer des secrets `smbcreds` :

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

- Un proxy CSI configuré comme un service Windows. Pour configurer un `csi-proxy` , se référer à"GitHub : CSI Proxy" ou"GitHub : CSI Proxy pour Windows" pour les nœuds Kubernetes exécutés sous Windows.

Options et exemples de configuration du backend Azure NetApp Files

Découvrez les options de configuration des serveurs NFS et SMB pour Azure NetApp Files et consultez des exemples de configuration.

options de configuration du backend

Trident utilise votre configuration backend (sous-réseau, réseau virtuel, niveau de service et emplacement) pour créer des volumes Azure NetApp Files sur des pools de capacité disponibles à l'emplacement demandé et correspondant au niveau de service et au sous-réseau demandés.

* À partir de la version NetApp Trident 25.06, les pools de capacité QoS manuels sont pris en charge en tant qu'aperçu technique.*

Les backends Azure NetApp Files offrent ces options de configuration.

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	"azure-netapp-files"
`backendName`	Nom personnalisé ou système de stockage	Nom du conducteur + "_" + caractères aléatoires
`subscriptionID`	L'ID d'abonnement de votre abonnement Azure (facultatif lorsque les identités gérées sont activées sur un cluster AKS).	
`tenantID`	L'identifiant du locataire issu d'un enregistrement d'application est facultatif lorsque des identités gérées ou une identité cloud sont utilisées sur un cluster AKS.	
`clientID`	L'identifiant client issu d'un enregistrement d'application est facultatif lorsque des identités gérées ou une identité cloud sont utilisées sur un cluster AKS.	
`clientSecret`	Le secret client issu d'un enregistrement d'application est facultatif lorsque des identités gérées ou une identité cloud sont utilisées sur un cluster AKS.	
`serviceLevel`	L'un des `Standard` , `Premium` , ou `Ultra`	"" (aléatoire)
`location`	Nom de l'emplacement Azure où les nouveaux volumes seront créés. Facultatif lorsque les identités gérées sont activées sur un cluster AKS.	

Paramètre	Description	Défaut
`resourceGroups`	Liste des groupes de ressources pour filtrer les ressources découvertes	"[]" (aucun filtre)
`netappAccounts`	Liste des comptes NetApp pour le filtrage des ressources découvertes	"[]" (aucun 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`	Ensemble de fonctionnalités VNet pour un volume, peut être `Basic` ou `Standard` . La fonctionnalité Réseau n'est pas disponible dans toutes les régions et peut nécessiter un abonnement pour être activée. Spécifier `networkFeatures` Lorsque cette fonctionnalité n'est pas activée, le provisionnement des volumes 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 séparées 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 du backend.	"nfsvers=3"
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur.	"" (non appliqué par défaut)
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, `\{"api": false, "method": true, "discovery": true}` . N'utilisez cette fonction que si vous effectuez un dépannage et avez besoin d'un journal de transactions détaillé.	nul

Paramètre	Description	Défaut
`nasType`	Configurer la création de volumes NFS ou SMB. Les options sont `nfs` , `smb` ou nul. La valeur nulle correspond par défaut aux volumes NFS.	`nfs`
`supportedTopologies`	Représente une liste des régions et zones prises en charge par ce serveur. Pour plus d'informations, veuillez consulter"Utiliser la topologie CSI" .	
`qosType`	Indique le type de QoS : Auto ou Manuel. **Aperçu technique de Trident 25.06**	Automatique
`maxThroughput`	Définit le débit maximal autorisé en Mio/s. Prise en charge uniquement pour les pools de capacité QoS manuels. **Aperçu technique de Trident 25.06**	`4 MiB/sec`

Pour plus d'informations sur les fonctionnalités réseau, consultez"Configurer les fonctionnalités réseau pour un volume Azure NetApp Files" .

Autorisations et ressources requises

Si vous recevez une erreur « Aucun pool de capacité trouvé » lors de la création d'un PVC, il est probable que l'enregistrement de votre application ne dispose pas des autorisations et des ressources requises (sous-réseau, réseau virtuel, pool de capacité) associées. Si le mode débogage est activé, Trident enregistrera les ressources Azure découvertes lors de la création du backend. Vérifiez qu'un rôle approprié est utilisé.

Les valeurs pour `resourceGroups` , `netappAccounts` , `capacityPools` , `virtualNetwork` , et `subnet` peuvent être spécifiés à l'aide de noms courts ou complets. Dans la plupart des situations, il est recommandé d'utiliser des noms complets, car les noms courts peuvent correspondre à plusieurs ressources portant le même nom.

Le `resourceGroups` , `netappAccounts` , et `capacityPools` Les valeurs sont des filtres qui limitent l'ensemble des ressources découvertes à celles disponibles pour ce système de stockage et peuvent être spécifiées dans n'importe quelle combinaison. Les noms complets suivent ce format :

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 des volumes par défaut en spécifiant les options suivantes dans une section spéciale du fichier de configuration. Se référer à Exemples de configurations pour plus de détails.

Paramètre	Description	Défaut
`exportRule`	Règles d'exportation pour les nouveaux volumes. `exportRule` doit être une liste séparée par des virgules de toute 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	« Vrai » pour NFSv4, « Faux » pour NFSv3
`size`	La taille par défaut des nouveaux volumes	"100G"
`unixPermissions`	Les permissions Unix des nouveaux volumes (4 chiffres octaux). Ignoré pour les volumes SMB.	"" (fonctionnalité en avant-première, nécessite une inscription sur la liste blanche dans l'abonnement)

Exemples de configurations

Les exemples suivants présentent des configurations de base qui laissent la plupart des paramètres par défaut. Voici la manière la plus simple de définir un backend.

Configuration minimale

Il s'agit de la configuration minimale absolue du backend. Avec cette configuration, Trident détecte tous vos comptes NetApp , pools de capacité et sous-réseaux délégués à Azure NetApp Files dans 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 dorsal provisionnera les volumes NFS.

Cette configuration est idéale lorsque vous débutez avec Azure NetApp Files et que vous faites des essais, mais en pratique, vous souhaiterez définir une portée supplémentaire pour les volumes que vous provisionnez.

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  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
```

Gestion des identités pour AKS

Cette configuration backend omet `subscriptionID` , `tenantID` , `clientID` , et `clientSecret` , qui sont facultatives lors de l'utilisation d'identités gérées.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
```

Identité cloud pour AKS

Cette configuration backend omet `tenantID` , `clientID` , et `clientSecret` , qui sont facultatives lors de l'utilisation d'une identité cloud.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

Configuration spécifique du niveau de service avec filtres de pool de capacité

Cette configuration backend place les volumes dans Azure. `eastus` emplacement dans un `Ultra` réserve de capacité. Trident détecte automatiquement tous les sous-réseaux délégués à Azure NetApp Files à cet emplacement et place un nouveau volume sur l'un d'eux de manière aléatoire.

```
---
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 de backend avec pools de capacité QoS manuels

Cette configuration backend place les volumes dans Azure. `eastus` emplacement avec pools de capacité QoS manuels. **Aperçu technique dans NetApp Trident 25.06**.

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

Configuration avancée

Cette configuration backend réduit encore la portée du placement des volumes à un seul sous-réseau et modifie également certains paramètres par défaut de 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"
```

Configuration du pool virtuel

Cette configuration backend définit plusieurs pools de stockage dans un seul fichier. Ceci est utile lorsque vous disposez de plusieurs pools de capacité prenant en charge différents niveaux de service et que vous souhaitez créer des classes de stockage dans Kubernetes qui les représentent. Des étiquettes virtuelles ont été utilisées pour différencier les bassins 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
```

Configuration des topologies prises en charge

Trident facilite la mise à disposition de volumes pour les charges de travail en fonction des régions et des zones de disponibilité. Le `supportedTopologies` Dans cette configuration backend, le bloc `block` sert à fournir une liste de régions et de zones par backend. Les valeurs de région et de zone spécifiées ici doivent correspondre aux valeurs de région et de zone des étiquettes de chaque nœud du cluster Kubernetes. Ces régions et zones représentent la liste des valeurs autorisées qui peuvent être fournies dans une classe de stockage. Pour les classes de stockage qui contiennent un sous-ensemble des régions et zones fournies dans un backend, Trident crée des volumes dans la région et la zone mentionnées. Pour plus d'informations, veuillez consulter"Utiliser la topologie CSI" .

```
---
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
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

Définitions des classes de stockage

Ce qui suit `StorageClass` Les définitions font référence aux pools de stockage ci-dessus.

Exemples de définitions utilisant `parameter.selector` champ

En utilisant `parameter.selector` vous pouvez spécifier pour chaque `StorageClass` le pool virtuel utilisé pour héberger un volume. Le volume comprendra les aspects définis dans le pool choisi.

```
---
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

En utilisant `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.

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"
```

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}
```

Utiliser 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 prenant en charge les volumes SMB. `nasType: nfs` ou `nasType: null` Filtres pour les pools NFS.

Créer le backend

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

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

Si la création du backend échoue, c'est qu'il y a un problème avec la configuration du backend. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs
```

Une fois le problème du fichier de configuration identifié et corrigé, vous pouvez exécuter à nouveau la commande de création.

Google Cloud NetApp Volumes

Configurer un backend Google Cloud NetApp Volumes

Vous pouvez désormais configurer Google Cloud NetApp Volumes comme backend pour Trident. Vous pouvez connecter des volumes NFS et SMB à l'aide d'un backend Google Cloud NetApp Volumes .

Détails du pilote Google Cloud NetApp Volumes

Trident fournit le `google-cloud-netapp-volumes` pilote pour communiquer avec le cluster. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`google-cloud-netapp-volumes`	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Identité cloud pour GKE

L'identité cloud permet aux pods Kubernetes d'accéder aux ressources Google Cloud en s'authentifiant en tant qu'identité de charge de travail au lieu de fournir des informations d'identification Google Cloud explicites.

Pour tirer parti de l'identité cloud dans Google Cloud, vous devez disposer de :

- Un cluster Kubernetes déployé à l'aide de GKE.
- L'identité de la charge de travail est configurée sur le cluster GKE et le serveur de métadonnées GKE est configuré sur les pools de nœuds.

- Un compte de service GCP avec le rôle d’administrateur de Google Cloud NetApp Volumes (roles/netapp.admin) ou un rôle personnalisé.
- Trident installé, incluant le cloudProvider spécifiant « GCP » et le cloudIdentity spécifiant le nouveau compte de service GCP. Un exemple est donné ci-dessous.

Opérateur Trident

Pour installer Trident à l'aide de l'opérateur Trident , modifiez `tridentorchestrator_cr.yaml` définir `cloudProvider` à `"GCP"` et ensemble `cloudIdentity` à `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com` .

Par exemple:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com'
```

Barre

Définissez les valeurs des indicateurs **cloud-provider (CP)** et **cloud-identity (CI)** à l'aide des variables d'environnement suivantes :

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

L'exemple suivant installe Trident et configure `cloudProvider` à GCP en utilisant la variable d'environnement `$CP` et établit le `cloudIdentity` en utilisant la variable d'environnement `$ANNOTATION` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>tridentctl</code>

Définissez les valeurs des indicateurs **cloud provider** et **cloud identity** à l'aide des variables d'environnement suivantes :

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

L'exemple suivant installe Trident et configure les `cloud-provider` drapeau à `$CP` , et `cloud-identity` à `$ANNOTATION` :

```
tridentctl install --cloud-provider=$CP --cloud
-identity="$ANNOTATION" -n trident
```

Préparez-vous à configurer un backend Google Cloud NetApp Volumes

Avant de pouvoir configurer votre backend Google Cloud NetApp Volumes , vous devez vous assurer que les exigences suivantes sont respectées.

Prérequis pour les volumes NFS

Si vous utilisez Google Cloud NetApp Volumes pour la première fois ou dans un nouvel emplacement, une configuration initiale est nécessaire pour configurer Google Cloud NetApp Volumes et créer un volume NFS. Se référer à"Avant de commencer" .

Assurez-vous de disposer des éléments suivants avant de configurer le backend Google Cloud NetApp Volumes :

- Un compte Google Cloud configuré avec le service Google Cloud NetApp Volumes . Se référer à"Google Cloud NetApp Volumes" .
- Numéro de projet de votre compte Google Cloud. Se référer à"Identification des projets" .
- Un compte de service Google Cloud avec l'administrateur des volumes NetApp(`roles/netapp.admin`) rôle. Se référer à"Rôles et autorisations de gestion des identités et des accès" .
- Fichier de clé API pour votre compte GCNV. Se référer à"Créer une clé de compte de service"
- Un pool de stockage. Se référer à"Aperçu des pools de stockage" .

Pour plus d'informations sur la configuration de l'accès à Google Cloud NetApp Volumes, consultez la documentation."Configurer l'accès aux Google Cloud NetApp Volumes" .

Options et exemples de configuration du backend Google Cloud NetApp Volumes .

Découvrez les options de configuration backend pour Google Cloud NetApp Volumes et consultez des exemples de configuration.

options de configuration du backend

Chaque serveur dorsal provisionne des volumes dans une seule région Google Cloud. Pour créer des volumes dans d'autres régions, vous pouvez définir des serveurs backend supplémentaires.

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	La valeur de `storageDriverName` doit être spécifié comme « google-cloud-netapp-volumes ».

Paramètre	Description	Défaut
`backendName`	(Facultatif) Nom personnalisé du système de stockage	Nom du pilote + "_" + partie de la clé API
`storagePools`	Paramètre optionnel permettant de spécifier les pools de stockage pour la création de volumes.	
`projectNumber`	Numéro de projet du compte Google Cloud. Cette valeur se trouve sur la page d'accueil du portail Google Cloud.	
`location`	L'emplacement Google Cloud où Trident crée les volumes GCNV. Lors de la création de clusters Kubernetes interrégionaux, les volumes créés dans un `location` peut être utilisé dans des charges de travail planifiées sur des nœuds répartis sur plusieurs régions Google Cloud. Le trafic interrégional engendre des coûts supplémentaires.	
`apiKey`	Clé API pour le compte de service Google Cloud avec le `netapp.admin` rôle. Il comprend le contenu au format JSON du fichier de clé privée d'un compte de service Google Cloud (copié tel quel dans le fichier de configuration backend). Le `apiKey` doit inclure des paires clé-valeur pour les clés suivantes : `type` , `project_id` , `client_email` , `client_id` , `auth_uri` , `token_uri` , `auth_provider_x509_cert_url` , et `client_x509_cert_url` .	
`nfsMountOptions`	Contrôle précis des options de montage NFS.	"nfsvers=3"
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur.	"" (non appliqué par défaut)
`serviceLevel`	Le niveau de service d'un pool de stockage et ses volumes. Les valeurs sont `flex` , `standard` , `premium` , ou `extreme` .	
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	""
`network`	Le réseau Google Cloud est utilisé pour les volumes GCNV.	
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, `{"api":false, "method":true}` . N'utilisez cette fonction que si vous effectuez un dépannage et avez besoin d'un journal de transactions détaillé.	nul
`nasType`	Configurer la création de volumes NFS ou SMB. Les options sont `nfs` , `smb` ou nul. La valeur nulle correspond par défaut aux volumes NFS.	`nfs`

Paramètre	Description	Défaut

`supportedTopologies`	Représente une liste des régions et zones prises en charge par ce serveur. Pour plus d'informations, veuillez consulter"Utiliser la topologie CSI" . Par exemple:
	`supportedTopologies:`
	`- topology.kubernetes.io/region: asia-east1`
	`topology.kubernetes.io/zone: asia-east1-a`	

options de provisionnement de volume

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

Paramètre	Description	Défaut
`exportRule`	Les règles d'exportation pour les nouveaux volumes. Doit être une liste d'adresses IPv4, séparées par des virgules, pouvant contenir n'importe quelle combinaison d'adresses IPv4.	"0.0.0.0/0"
`snapshotDir`	L'accès à `.snapshot` annuaire	« Vrai » pour NFSv4, « Faux » pour NFSv3
`snapshotReserve`	Pourcentage du volume réservé aux instantanés	"" (accepter la valeur par défaut de 0)
`unixPermissions`	Les permissions Unix des nouveaux volumes (4 chiffres octaux).	""

Exemples de configurations

Les exemples suivants présentent des configurations de base qui laissent la plupart des paramètres par défaut. Voici la manière la plus simple de définir un backend.

Configuration minimale

Il s'agit de la configuration minimale absolue du backend. Avec cette configuration, Trident détecte tous vos pools de stockage délégués à Google Cloud NetApp Volumes dans l'emplacement configuré et place les nouveaux volumes sur l'un de ces pools de manière aléatoire. Parce que `nasType` est omis, le `nfs` La valeur par défaut s'applique et le système dorsal provisionnera les volumes NFS.

Cette configuration est idéale lorsque vous débutez avec Google Cloud NetApp Volumes et que vous faites des essais, mais en pratique, vous devrez probablement définir une portée supplémentaire pour les volumes que vous provisionnez.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    XsYg6gyxy4zq7OlwWgLwGa==\n
    -----END PRIVATE KEY-----\n

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuration pour les volumes SMB

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    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/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuration avec filtre StoragePools

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuration du pool virtuel

Cette configuration backend définit plusieurs pools virtuels dans un seul fichier. Les pools virtuels sont définis dans le `storage` section. Elles sont utiles lorsque vous disposez de plusieurs pools de stockage prenant en charge différents niveaux de service et que vous souhaitez créer des classes de stockage dans Kubernetes qui les représentent. Les étiquettes des pools virtuels servent à différencier les pools. Par exemple, dans l'exemple ci-dessous `performance` étiquette et `serviceLevel` Le type est utilisé pour différencier les pools virtuels.

Vous pouvez également définir des valeurs par défaut applicables à tous les pools virtuels et remplacer les valeurs par défaut de chaque pool virtuel. Dans l'exemple suivant, `snapshotReserve` et `exportRule` servent de valeurs par défaut pour tous les pools virtuels.

Pour plus d'informations, veuillez consulter"Piscines virtuelles" .

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
```

```
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Identité cloud pour GKE

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

Configuration des topologies prises en charge

Trident facilite la mise à disposition de volumes pour les charges de travail en fonction des régions et des zones de disponibilité. Le `supportedTopologies` Dans cette configuration backend, le bloc `block` sert à fournir une liste de régions et de zones par backend. Les valeurs de région et de zone spécifiées ici doivent correspondre aux valeurs de région et de zone des étiquettes de chaque nœud du cluster Kubernetes. Ces régions et zones représentent la liste des valeurs autorisées qui peuvent être fournies dans une classe de stockage. Pour les classes de stockage qui contiennent un sous-ensemble des régions et zones fournies dans un backend, Trident crée des volumes dans la région et la zone mentionnées. Pour plus d'informations, veuillez consulter"Utiliser la topologie CSI" .

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

Quelle est la prochaine étape ?

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

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

Pour vérifier que le backend a bien été créé, exécutez la commande suivante :

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE   STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound   Success
```

Si la création du backend échoue, c'est qu'il y a un problème avec la configuration du backend. Vous pouvez décrire le backend en utilisant le `kubectl get tridentbackendconfig <backend-name>` Vous pouvez également consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs
```

Une fois le problème du fichier de configuration identifié et corrigé, vous pouvez supprimer le backend et exécuter à nouveau la commande de création.

Définitions des classes de stockage

Voici un exemple de base `StorageClass` définition qui fait référence au backend ci-dessus.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

Exemples de définitions utilisant `parameter.selector` champ:

En utilisant `parameter.selector` vous pouvez spécifier pour chaque `StorageClass` le"piscine virtuelle" qui sert à héberger un volume. Le volume comprendra les aspects définis dans le pool choisi.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

Pour plus de détails sur les classes de stockage, veuillez consulter"Créer une classe de stockage" .

Exemples de définitions pour les volumes SMB

En utilisant `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. N'importe quel nom d'utilisateur/mot de passe Active Directory, avec ou sans autorisations, peut être utilisé comme secret de l'étape du nœud.

Configuration de base sur l'espace de noms par défaut

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

Utilisation de secrets différents par espace de noms

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

Utiliser différents secrets par volume

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  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 prenant en charge les volumes SMB. `nasType: nfs ou nasType: null Filtres pour les pools NFS.

Exemple de définition du PVC

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

Pour vérifier si le PVC est lié, exécutez la commande suivante :

```
kubectl get pvc gcnv-nfs-pvc

NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc  Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213  100Gi
RWX            gcnv-nfs-sc  1m
```

Configurer un Cloud Volumes Service pour le backend Google Cloud

Découvrez comment configurer NetApp Cloud Volumes Service pour Google Cloud comme backend pour votre installation Trident à l'aide des exemples de configuration fournis.

Détails du pilote Google Cloud

Trident fournit le `gcp-cvs` pilote pour communiquer avec le cluster. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`gcp-cvs`	NFS	Système de fichiers	RWO, ROX, RWX, RWOP	`nfs`

Découvrez la prise en charge par Trident du Cloud Volumes Service pour Google Cloud.

Trident peut créer des volumes Cloud Volumes Service dans l'un des deux modes suivants :"types de services" :

- **CVS-Performance** : Le type de service Trident par défaut. Ce type de service optimisé pour les performances est parfaitement adapté aux charges de travail de production qui privilégient la performance. 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'un des"trois niveaux de service" :
  - `standard`
  - `premium`
  - `extreme`
- **CVS** : Le type de service CVS offre une haute disponibilité zonale avec des niveaux de performance limités à modérés. Le type de service CVS est une option logicielle qui utilise des pools de stockage pour prendre en charge des volumes aussi petits que 1 Gio. Le pool de stockage peut contenir jusqu'à 50 volumes, tous partageant la capacité et les performances du pool. Vous pouvez choisir l'un des"deux niveaux de service" :
  - `standardsw`
  - `zoneredundantstandardsw`

Ce dont vous aurez besoin

Pour configurer et utiliser le "Cloud Volumes Service pour Google Cloud" Côté serveur, 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 de Cloud Volumes Service

options de configuration du backend

Chaque serveur dorsal provisionne des volumes dans une seule région Google Cloud. Pour créer des volumes dans d'autres régions, vous pouvez définir des serveurs backend supplémentaires.

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	"gcp-cvs"
`backendName`	Nom personnalisé ou système de stockage	Nom du pilote + "_" + partie de la clé API
`storageClass`	Paramètre optionnel utilisé pour spécifier le type de service CVS. Utiliser `software` pour sélectionner le type de service CVS. Sinon, Trident suppose le type de service CVS-Performance(`hardware`).	
`storagePools`	Service de type CVS uniquement. Paramètre optionnel permettant de spécifier les pools de stockage pour la création de volumes.	

Paramètre	Description	Défaut
`projectNumber`	Numéro de projet du compte Google Cloud. Cette valeur se trouve sur la page d'accueil du portail Google Cloud.	
`hostProjectNumber`	Requis si vous utilisez un réseau VPC partagé. Dans ce scénario, `projectNumber` est le projet de service, et `hostProjectNumber` est le projet hôte.	
`apiRegion`	La région Google Cloud où Trident crée des volumes Cloud Volumes Service . Lors de la création de clusters Kubernetes interrégionaux, les volumes créés dans un `apiRegion` peut être utilisé dans des charges de travail planifiées sur des nœuds répartis sur plusieurs régions Google Cloud. Le trafic interrégional engendre des coûts supplémentaires.	
`apiKey`	Clé API pour le compte de service Google Cloud avec le `netappcloudvolumes.admin` rôle. Il comprend le contenu au format JSON du fichier de clé privée d'un compte de service Google Cloud (copié tel quel dans le fichier de configuration backend).	
`proxyURL`	URL du proxy si un serveur proxy est requis pour se connecter au compte CVS. Le serveur proxy peut être soit un proxy HTTP, soit un proxy HTTPS. Pour un proxy HTTPS, la validation du certificat est ignorée afin de permettre l'utilisation de certificats auto-signés sur 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`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur.	"" (non appliqué par défaut)
`serviceLevel`	Le niveau de service CVS-Performance ou CVS pour les nouveaux volumes. Les valeurs CVS-Performance sont `standard` , `premium` , ou `extreme` . Les valeurs CVS sont `standardsw` ou `zoneredundantstandardsw` .	La valeur par défaut de CVS-Performance est « standard ». La valeur par défaut de CVS est « standardsw ».
`network`	Le réseau Google Cloud est utilisé pour les volumes du Cloud Volumes Service .	"défaut"
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, `\{"api":false, "method":true}` . N'utilisez cette fonction que si vous effectuez un dépannage et avez besoin d'un journal de transactions détaillé.	nul

Paramètre	Description	Défaut

`allowedTopologies`	Pour permettre l'accès interrégional, votre définition StorageClass pour `allowedTopologies` doit inclure toutes les régions. Par exemple:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

options de provisionnement de volume

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

Paramètre	Description	Défaut
`exportRule`	Les règles d'exportation pour les nouveaux volumes. Doit être une liste séparée par des virgules de toute combinaison d'adresses IPv4 ou de sous-réseaux IPv4 en notation CIDR.	"0.0.0.0/0"
`snapshotDir`	L'accès à `.snapshot` annuaire	"FAUX"
`snapshotReserve`	Pourcentage du volume réservé aux instantanés	"" (accepter la valeur par défaut de CVS : 0)
`size`	La taille des nouveaux volumes. La taille minimale requise pour CVS-Performance est de 100 Gio. La valeur minimale de CVS est de 1 Gio.	Le type de service CVS-Performance est par défaut de « 100 Gio ». Le type de service CVS ne définit pas de valeur par défaut, mais exige un minimum de 1 Gio.

Exemples de types de services CVS-Performance

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

Exemple 1 : Configuration minimale

Il s'agit de la configuration minimale du backend utilisant le type de service CVS-Performance par défaut avec le niveau de service « standard » par défaut.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: "012345678901"
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: <id_value>
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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
```

Exemple 2 : Configuration du niveau de service

Cet exemple illustre les options de configuration du backend, notamment le niveau de service et les valeurs par défaut du volume.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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
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
```

Exemple 3 : Configuration du pool virtuel

Cet échantillon utilise `storage` pour configurer les pools virtuels et le `StorageClasses` qui renvoient à eux. Se référer àDéfinitions des classes de stockage pour voir comment les classes de stockage étaient définies.

Ici, des valeurs par défaut spécifiques sont définies pour tous les pools virtuels, qui définissent les `snapshotReserve` à 5 % et le `exportRule` à 0.0.0.0/0. Les pools virtuels sont définis dans le `storage` section. Chaque pool virtuel individuel définit ses propres `serviceLevel` et certaines pools écrasent les valeurs par défaut. Des étiquettes virtuelles ont été utilisées pour différencier les bassins en fonction de `performance` et `protection` .

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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 StorageClass suivantes s'appliquent à l'exemple de configuration de pool virtuel. En utilisant `parameters.selector` Vous pouvez spécifier pour chaque StorageClass le pool virtuel utilisé pour héberger un volume. Le volume comprendra les aspects définis dans le pool choisi.

Exemple de classe de stockage

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

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

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

Exemples de types de services CVS

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

Exemple 1 : Configuration minimale

Voici la configuration minimale du backend utilisant `storageClass` pour spécifier le type de service CVS et la valeur par défaut `standardsw` niveau de service.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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
serviceLevel: standardsw
```

Exemple 2 : Configuration du pool de stockage

Cette configuration backend d'exemple utilise `storagePools` 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-----
    <key_value>
    -----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
```

Quelle est la prochaine étape ?

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

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

Si la création du backend échoue, c'est qu'il y a un problème avec la configuration du backend. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs
```

Une fois le problème du fichier de configuration identifié et corrigé, vous pouvez exécuter à nouveau la commande de création.

Configurer un backend NetApp HCI ou SolidFire

Apprenez à créer et à utiliser un backend Element avec votre installation Trident .

Détails du pilote Element

Trident fournit le `solidfire-san` Pilote de stockage pour communiquer avec le cluster. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Le `solidfire-san` Le pilote de stockage prend en charge les modes de volume *file* et *block*. Pour le `Filesystem` En mode volume, Trident crée un volume et un système de fichiers. Le type de système de fichiers est spécifié par la StorageClass.

Conducteur	Protocole	Mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`solidfire-san`	iSCSI	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers. Périphérique de bloc brut.
`solidfire-san`	iSCSI	Système de fichiers	RWO, RWOP	`xfs`, `ext3` , `ext4`

Avant de commencer

Vous aurez besoin des éléments suivants avant de créer un backend Element.

- Un système de stockage compatible qui exécute le logiciel Element.
- Identifiants d'un administrateur ou d'un utilisateur locataire d'un cluster NetApp HCI/ SolidFire pouvant gérer des volumes.
- Tous vos nœuds de travail Kubernetes doivent avoir les outils iSCSI appropriés installés. Se référer à"Informations de préparation des nœuds de travail" .

options de configuration du backend

Consultez le tableau suivant pour connaître les options de configuration du backend :

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	Toujours "solidfire-san"

Paramètre	Description	Défaut
`backendName`	Nom personnalisé ou système de stockage	"solidfire_" + adresse IP de stockage (iSCSI)
`Endpoint`	MVIP pour le cluster SolidFire avec identifiants de locataire	
`SVIP`	Adresse IP et port de stockage (iSCSI)	
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes.	""
`TenantName`	Nom du locataire à utiliser (créé s'il n'est pas trouvé)	
`InitiatorIFace`	Limiter le trafic iSCSI à une interface hôte spécifique	"défaut"
`UseCHAP`	Utilisez CHAP pour authentifier iSCSI. Trident utilise CHAP.	true
`AccessGroups`	Liste des ID de groupes d'accès à utiliser	Recherche l'ID d'un groupe d'accès nommé « trident ».
`Types`	Spécifications QoS	
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur.	"" (non appliqué par défaut)
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, {"api":false, "method":true}	nul

Ne pas utiliser `debugTraceFlags` sauf si vous effectuez un dépannage et avez besoin d'un journal détaillé.

Exemple 1 : Configuration du backend pour `solidfire-san` pilote avec trois types de volume

Cet exemple montre un fichier backend utilisant l'authentification CHAP et modélisant trois types de volumes avec des garanties QoS spécifiques. Vous définiriez alors très probablement des classes de stockage pour consommer chacune d'elles en utilisant `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 backend et de la classe de stockage pour `solidfire-san` pilote avec pools virtuels

Cet exemple montre le fichier de définition du backend configuré avec des pools virtuels ainsi que les StorageClasses qui y font référence.

Lors de la mise en service, Trident copie les étiquettes présentes sur un pool de stockage vers le LUN de stockage backend. Pour plus de commodité, les administrateurs de stockage peuvent définir des étiquettes par pool virtuel et regrouper les volumes par étiquette.

Dans l'exemple de fichier de définition de backend présenté ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, qui définissent les `type` à 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 StorageClass suivantes font référence aux pools virtuels ci-dessus. En utilisant le

`parameters.selector` Dans ce champ, chaque StorageClass indique quel(s) pool(s) virtuel(s) peuvent être utilisés pour héberger un volume. Le volume aura les aspects définis dans le pool virtuel choisi.

La première classe de stockage(`solidfire-gold-four`) sera associé au premier pool virtuel. Il s'agit de la seule piscine offrant des performances de niveau or avec un `Volume Type QoS` d'or. La dernière classe de stockage(`solidfire-silver`) désigne tout pool de stockage offrant une performance de niveau argent. Trident déterminera quel pool virtuel sera sélectionné et s'assurera 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
```

Trouver plus d'informations

- "groupes d'accès au volume"

Pilotes SAN ONTAP

Présentation du pilote ONTAP SAN

Découvrez comment configurer un backend ONTAP avec les pilotes SAN ONTAP et Cloud Volumes ONTAP .

Détails du pilote ONTAP SAN

Trident fournit les pilotes de stockage SAN suivants pour communiquer avec le cluster ONTAP . Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-san`	iSCSI SCSI sur FC	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers ; périphérique de stockage brut
`ontap-san`	iSCSI SCSI sur FC	Système de fichiers	RWO, RWOP Les modes ROX et RWX ne sont pas disponibles en mode volume du système de fichiers.	`xfs`, `ext3` , `ext4`

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-san`	NVMe/TCP Se référer àConsidérations supplémentaires concernant NVMe/TCP.	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers ; périphérique de stockage brut
`ontap-san`	NVMe/TCP Se référer àConsidérations supplémentaires concernant NVMe/TCP.	Système de fichiers	RWO, RWOP Les modes ROX et RWX ne sont pas disponibles en mode volume du système de fichiers.	`xfs`, `ext3` , `ext4`
`ontap-san-economy`	iSCSI	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers ; périphérique de stockage brut
`ontap-san-economy`	iSCSI	Système de fichiers	RWO, RWOP Les modes ROX et RWX ne sont pas disponibles en mode volume du système de fichiers.	`xfs`, `ext3` , `ext4`

- Utiliser `ontap-san-economy` uniquement si le nombre d'utilisations de volume persistantes devrait être supérieur à"limites de volume ONTAP prises en charge" .
- Utiliser `ontap-nas-economy` uniquement si le nombre d'utilisations de volume persistantes devrait être supérieur à"limites de volume ONTAP prises en charge" et le `ontap-san-economy` Le pilote ne peut pas être utilisé.
- Ne pas utiliser `ontap-nas-economy` si vous prévoyez un besoin en matière de protection des données, de reprise après sinistre ou de mobilité.
- NetApp ne recommande pas l'utilisation de la croissance automatique Flexvol dans tous les pilotes ONTAP , à l'exception de ontap-san. En guise de solution de contournement, Trident prend en charge l'utilisation de la réserve de snapshots et adapte les volumes Flexvol en conséquence.

Autorisations de l'utilisateur

Trident s'attend à être exécuté en tant qu'administrateur ONTAP ou SVM, généralement en utilisant `admin` utilisateur de cluster ou un `vsadmin` Utilisateur SVM, ou un utilisateur portant un nom différent mais ayant le même rôle. Pour les déploiements Amazon FSx for NetApp ONTAP , Trident s'attend à être exécuté en tant qu'administrateur ONTAP ou SVM, en utilisant le cluster. `fsxadmin` utilisateur ou un `vsadmin` Utilisateur SVM, ou un utilisateur portant un nom différent mais ayant le même rôle. Le `fsxadmin` L'utilisateur est un remplaçant limité pour l'utilisateur administrateur du cluster.

> Si vous utilisez le `limitAggregateUsage` Les paramètres suivants sont requis : autorisations d'administrateur de cluster. Lors de l'utilisation Amazon FSx for NetApp ONTAP avec Trident, `limitAggregateUsage` Le paramètre ne fonctionnera pas avec le `vsadmin` et `fsxadmin` comptes utilisateurs. L'opération de configuration échouera si vous spécifiez ce paramètre.

Bien qu'il soit possible de créer un rôle plus restrictif au sein ONTAP qu'un pilote Trident puisse utiliser, nous ne le recommandons pas. La plupart des nouvelles versions de Trident feront appel à des API supplémentaires dont il faudra tenir compte, ce qui rendra les mises à niveau difficiles et sujettes aux erreurs.

Considérations supplémentaires concernant NVMe/TCP

Trident prend en charge le protocole NVMe (Non-Volatile Memory Express) en utilisant le `ontap-san` conducteur, y compris :

- IPv6
- Instantanés et clones de volumes NVMe
- Redimensionnement d'un volume NVMe
- Importer un volume NVMe créé en dehors de Trident afin que son cycle de vie puisse être géré par Trident
- Multipathing natif NVMe
- Arrêt progressif ou brutal des nœuds K8s (24.06)

Trident ne prend pas en charge :

- DH-HMAC-CHAP pris en charge nativement par NVMe
- multipathing du mappeur de périphériques (DM)
- cryptage LUKS

> NVMe est pris en charge uniquement avec les API REST ONTAP et n'est pas pris en charge avec ONTAPI (ZAPI).

Préparez-vous à configurer le backend avec les pilotes SAN ONTAP

Comprendre les exigences et les options d'authentification pour configurer un backend ONTAP avec des pilotes SAN ONTAP .

Exigences

Pour tous les backends ONTAP , Trident exige qu'au moins un agrégat soit affecté au SVM.

"Systèmes ASA r2"diffèrent des autres systèmes ONTAP (ASA, AFF et FAS) dans la mise en œuvre de leur couche de stockage. Dans les systèmes ASA r2, on utilise des zones de disponibilité de stockage au lieu d'agrégats. Se référer à"ce" Article de la base de connaissances sur la manière d'attribuer des agrégats aux SVM dans les systèmes ASA r2.

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 la `ontap-san` conducteur et un `san-default` classe qui utilise la `ontap-san-economy` un.

Tous vos nœuds de travail Kubernetes doivent avoir les outils iSCSI appropriés installés. Se référer à "Préparer le nœud de travail" pour plus de détails.

Authentifier le backend ONTAP

Trident propose deux modes d'authentification pour un système dorsal ONTAP .

- Authentification par identifiants : nom d'utilisateur et mot de passe d'un utilisateur ONTAP disposant des autorisations requises. Il est recommandé d'utiliser un rôle de connexion de sécurité prédéfini, tel que `admin` ou `vsadmin` pour assurer une compatibilité maximale avec les versions ONTAP .
- Authentification par certificat : Trident peut également communiquer avec un cluster ONTAP en utilisant un certificat installé sur le serveur. Ici, la définition du backend doit contenir les valeurs encodées en Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance si utilisé (recommandé).

Vous pouvez mettre à jour les systèmes d'arrière-plan existants pour passer d'une méthode basée sur les identifiants à une méthode basée sur les certificats. Cependant, une seule méthode d'authentification est prise en charge à la fois. Pour passer à une autre méthode d'authentification, vous devez supprimer la méthode actuelle de la configuration du serveur.

Si vous tentez de fournir **à la fois des identifiants et des certificats**, la création du backend échouera avec une erreur indiquant que plusieurs méthodes d'authentification ont été fournies dans le fichier de configuration.

Activer l'authentification par identifiants

Trident a besoin des identifiants d'un administrateur au niveau SVM/cluster pour communiquer avec le backend ONTAP . Il est recommandé d'utiliser des rôles standard prédéfinis tels que `admin` ou `vsadmin` . Cela garantit la compatibilité ascendante avec les futures versions ONTAP qui pourraient exposer des API de fonctionnalités utilisables par les futures versions de Trident . Il est possible de créer et d'utiliser un rôle de connexion de sécurité personnalisé avec Trident, mais cela n'est pas recommandé.

Voici un exemple de définition de backend :

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON

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

N'oubliez pas que la définition du backend est le seul endroit où les identifiants sont stockés en clair. Une fois le backend créé, les noms d'utilisateur et les mots de passe sont encodés en Base64 et stockés en tant que secrets Kubernetes. La création ou la mise à jour d'un backend est la seule étape qui nécessite la connaissance des identifiants. Il s'agit donc d'une opération réservée aux administrateurs, qui doit être effectuée par l'administrateur Kubernetes/stockage.

Activer l'authentification basée sur les certificats

Les nouveaux et les existants serveurs dorsaux peuvent utiliser un certificat et communiquer avec le serveur dorsal ONTAP . Trois paramètres sont requis dans la définition du backend.

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

Un flux de travail typique comprend les étapes suivantes.

Étapes

1. Générer un certificat client et une clé. Lors de la génération, définissez le nom commun (CN) sur l'utilisateur ONTAP sous lequel s'authentifier.

```
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. Ajouter un certificat d'autorité de certification de confiance au cluster ONTAP . Cela est peut-être déjà géré par l'administrateur du stockage. Ignorer si aucune autorité de certification de confiance 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 (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 prend en charge `cert` méthode 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 du certificat généré. Remplacez < ONTAP Management LIF> et <nom du serveur virtuel> par l'adresse IP de l'interface de gestion LIF et le nom du SVM.

```
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 d'autorité de certification de confiance au format 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 backend en utilisant les valeurs obtenues à l'étape précédente.

```
cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Mettez à jour les méthodes d'authentification ou changez les identifiants.

Vous pouvez mettre à jour un système dorsal existant pour utiliser une méthode d'authentification différente ou pour renouveler ses identifiants. Cela fonctionne dans les deux sens : les systèmes d'arrière-plan qui utilisent un nom d'utilisateur/mot de passe peuvent être mis à jour pour utiliser des certificats ; les systèmes d'arrière-plan qui utilisent des certificats peuvent être mis à jour pour utiliser un 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 pour 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 étape est suivie d'une mise à jour du système dorsal. Lors de la rotation des certificats, plusieurs certificats peuvent être ajoutés à l'utilisateur. Le système dorsal est ensuite mis à jour pour utiliser le nouveau certificat, après quoi l'ancien certificat peut être supprimé du cluster ONTAP .

La mise à jour d'un système dorsal n'interrompt pas l'accès aux volumes déjà créés et n'a aucun impact sur les connexions de volumes effectuées ultérieurement. Une mise à jour réussie du système dorsal indique que Trident peut communiquer avec le système dorsal ONTAP et gérer les futures opérations de volume.

Créer un rôle ONTAP personnalisé pour Trident

Vous pouvez créer un rôle de cluster ONTAP avec des privilèges minimaux afin de ne pas avoir à utiliser le rôle d'administrateur ONTAP pour effectuer des opérations dans Trident. Lorsque vous incluez le nom d'utilisateur dans une configuration backend Trident , Trident utilise le rôle de cluster ONTAP que vous avez créé pour effectuer les opérations.

Se référer à"Générateur de rôles personnalisés Trident" pour plus d'informations sur la création de rôles personnalisés Trident .

Utilisation de l'interface de ligne de commande ONTAP

1. Créez un nouveau rôle à l'aide de la commande suivante :

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Créez un nom d'utilisateur pour l'utilisateur Trident :

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Associer le rôle à l'utilisateur :

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilisation du gestionnaire système

Effectuez les étapes suivantes dans ONTAP System Manager :

1. **Créer un rôle personnalisé** :
   a. Pour créer un rôle personnalisé au niveau du cluster, sélectionnez **Cluster > Paramètres**.

      (Ou) Pour créer un rôle personnalisé au niveau de la SVM, sélectionnez **Stockage > Machines virtuelles de stockage > `required SVM` > Paramètres > Utilisateurs et rôles**.

   b. Sélectionnez l'icône flèche (→) à côté de **Utilisateurs et rôles**.
   c. Sélectionnez **+Ajouter** sous **Rôles**.
   d. Définissez les règles du rôle et cliquez sur **Enregistrer**.
2. **Associer le rôle à l'utilisateur Trident * : + Effectuez les étapes suivantes sur la page *Utilisateurs et rôles** :
   a. Sélectionnez l'icône Ajouter **+** sous **Utilisateurs**.
   b. Sélectionnez le nom d'utilisateur requis, puis sélectionnez un rôle dans le menu déroulant **Rôle**.
   c. Cliquez sur **Enregistrer**.

Pour plus d'informations, veuillez consulter les pages suivantes :

- "Rôles personnalisés pour l'administration d' ONTAP"ou"Définir des rôles personnalisés"
- "Collaborer avec les rôles et les utilisateurs"

Authentifier les connexions avec CHAP bidirectionnel

Trident peut authentifier les sessions iSCSI avec CHAP bidirectionnel pour le `ontap-san` et `ontap-san-economy` conducteurs. Cela nécessite d'activer `useCHAP` option dans votre définition de backend. Lorsqu'il est réglé sur `true` Trident configure la sécurité par défaut de l'initiateur SVM en CHAP bidirectionnel et définit le nom d'utilisateur et les secrets à partir du fichier backend. NetApp recommande l'utilisation du protocole CHAP bidirectionnel pour authentifier les connexions. Voici un exemple de configuration :

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

Le `useCHAP` Ce paramètre est une option booléenne qui ne peut être configurée qu'une seule fois. Cette option est désactivée par défaut. Une fois que vous l'avez défini sur vrai, vous ne pouvez plus le définir sur faux.

En plus de `useCHAP=true` , le `chapInitiatorSecret` , `chapTargetInitiatorSecret` , `chapTargetUsername` , et `chapUsername` Les champs doivent être inclus dans la définition du backend. Les secrets peuvent être modifiés après la création d'un backend en exécutant la commande suivante : `tridentctl update` .

Comment ça marche

En fixant `useCHAP` Si la valeur est « true », l'administrateur du stockage demande à Trident de configurer CHAP sur le système de stockage dorsal. Cela comprend les éléments suivants :

- Configuration de CHAP sur la SVM :
  - Si le type de sécurité par défaut de l'initiateur du SVM est « aucun » (paramètre par défaut) **et** qu'aucun LUN n'est déjà présent dans le volume, Trident définira le type de sécurité par défaut sur `CHAP` et procédez à la configuration du nom d'utilisateur et des secrets de l'initiateur et de la cible CHAP.
  - Si le SVM contient des LUN, Trident n'activera pas CHAP sur le SVM. Cela garantit que l'accès aux LUN déjà présentes sur la SVM n'est pas restreint.
- Configuration du nom d'utilisateur et des secrets de l'initiateur et de la cible CHAP ; ces options doivent être spécifiées dans la configuration du backend (comme indiqué ci-dessus).

Une fois le backend créé, Trident crée un correspondant `tridentbackend` CRD et stocke les secrets CHAP et les noms d'utilisateur en tant que secrets Kubernetes. Tous les PV créés par Trident sur ce backend seront montés et attachés via CHAP.

Faire pivoter les informations d'identification et mettre à jour les backends

Vous pouvez mettre à jour les informations d'identification CHAP en modifiant les paramètres CHAP dans le `backend.json` déposer. Cela nécessitera la mise à jour des secrets CHAP et l'utilisation du `tridentctl update` commande pour refléter ces changements.

Lors de la mise à jour des secrets CHAP pour un serveur dorsal, vous devez utiliser `tridentctl` mettre à jour le système dorsal. Ne mettez pas à jour les informations d'identification sur le cluster de stockage à l'aide de l'interface de ligne de commande ONTAP ou du gestionnaire système ONTAP , car Trident ne pourra pas prendre en compte 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",
    "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-4378-8a33-1a472ffbeb5c |
online |       7 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

Les connexions existantes resteront inchangées ; elles resteront actives si les informations d'identification sont mises à jour par Trident sur le SVM. Les nouvelles connexions utilisent les informations d'identification mises à jour et les connexions existantes restent actives. Déconnecter puis reconnecter les anciens PV leur permettra d'utiliser les identifiants mis à jour.

Options et exemples de configuration SAN ONTAP

Apprenez à créer et à utiliser des pilotes ONTAP SAN avec votre installation Trident . Cette section fournit des exemples de configuration backend et des détails sur le mappage des backends aux StorageClasses.

"Systèmes ASA r2"diffère des autres systèmes ONTAP (ASA, AFF et FAS) dans la mise en œuvre de sa couche de stockage. Ces variations ont une incidence sur l'utilisation de certains paramètres, comme indiqué.

"Apprenez-en davantage sur les différences entre les systèmes ASA r2 et les autres systèmes ONTAP.".

> Seuls les `ontap-san` Le pilote (avec les protocoles iSCSI et NVMe/TCP) est pris en charge pour les systèmes ASA r2.

Dans la configuration du backend Trident , il n'est pas nécessaire de préciser que votre système est un ASA r2. Lorsque vous sélectionnez `ontap-san` comme le `storageDriverName` Trident détecte automatiquement le ASA r2 ou le système ONTAP traditionnel. Certains paramètres de configuration du backend ne sont pas applicables aux systèmes ASA r2, comme indiqué dans le tableau ci-dessous.

options de configuration du backend

Consultez le tableau suivant pour connaître les options de configuration du backend :

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	`ontap-san`ou `ontap-san-economy`
`backendName`	Nom personnalisé ou système de stockage	Nom du conducteur + "_" + dataLIF
`managementLIF`	Adresse IP d'une interface logique de gestion de cluster ou de SVM. Un nom de domaine pleinement qualifié (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme ceci : `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Pour une transition MetroCluster sans interruption, consultez la documentation.Exemple de MetroCluster . > Si vous utilisez les identifiants « vsadmin », `managementLIF` doit être celle du SVM ; si vous utilisez les identifiants « administrateur », `managementLIF` doit être celui du groupe.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Paramètre	Description	Défaut
`dataLIF`	Adresse IP du protocole LIF. Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme ceci : `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Ne pas spécifier pour iSCSI.** Trident utilise"Carte LUN sélective ONTAP" pour découvrir les LIF iSCSI nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si `dataLIF` est explicitement défini. **Omettre pour Metrocluster.** Voir leExemple de MetroCluster .	Dérivé par le SVM
`svm`	Machine virtuelle de stockage à utiliser **Omettre pour Metrocluster.** Voir leExemple de MetroCluster .	Dérivé d'un SVM `managementLIF` est spécifié
`useCHAP`	Utilisez CHAP pour authentifier iSCSI pour les pilotes SAN ONTAP [Booléen]. Réglé sur `true` pour que Trident configure et utilise CHAP bidirectionnel comme authentification par défaut pour la SVM fournie dans le backend. Se référer à "Préparez-vous à configurer le backend avec les pilotes SAN ONTAP" pour plus de détails. **Non pris en charge pour FCP ou NVMe/TCP.**	`false`
`chapInitiatorSecret`	Secret de l'initiateur CHAP. Obligatoire si `useCHAP=true`	""
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	""
`chapTargetInitiatorSecret`	Secret de l'initiateur de la cible CHAP. Obligatoire si `useCHAP=true`	""
`chapUsername`	Nom d'utilisateur entrant. Obligatoire si `useCHAP=true`	""
`chapTargetUsername`	Nom d'utilisateur cible. Obligatoire 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 d'autorité de certification de confiance. Facultatif. Utilisé pour l'authentification par certificat.	""
`username`	Nom d'utilisateur nécessaire pour communiquer avec le cluster ONTAP . Utilisé pour l'authentification basée sur les informations d'identification. Pour l'authentification Active Directory, voir "Authentifier Trident auprès d'une SVM principale à l'aide des informations d'identification Active Directory".	""

Paramètre	Description	Défaut
`password`	Mot de passe nécessaire pour communiquer avec le cluster ONTAP . Utilisé pour l'authentification basée sur les informations d'identification. Pour l'authentification Active Directory, voir "Authentifier Trident auprès d'une SVM principale à l'aide des informations d'identification Active Directory".	""
`svm`	machine virtuelle de stockage à utiliser	Dérivé d'un SVM `managementLIF` est spécifié
`storagePrefix`	Préfixe utilisé lors de la mise en service de nouveaux volumes dans la SVM. Ne peut être modifié ultérieurement. Pour mettre à jour ce paramètre, vous devrez créer un nouveau backend.	`trident`
`aggregate`	Agrégat pour le provisionnement (facultatif ; s'il est défini, il doit être affecté au SVM). Pour le `ontap-nas-flexgroup` conducteur, cette option est ignorée. Si aucun agrégat n'est attribué, n'importe lequel des agrégats disponibles peut être utilisé pour provisionner un volume FlexGroup . Lorsque l'agrégat est mis à jour dans SVM, il est automatiquement mis à jour dans Trident par interrogation de SVM sans qu'il soit nécessaire de redémarrer le contrôleur Trident . Lorsque vous avez configuré un agrégat spécifique dans Trident pour provisionner des volumes, si l'agrégat est renommé ou déplacé hors du SVM, le backend passera à l'état d'échec dans Trident lors de l'interrogation de l'agrégat SVM. Vous devez soit modifier l'agrégat pour qu'il soit présent sur la SVM, soit le supprimer complètement pour remettre le serveur en ligne. **Ne pas spécifier pour les systèmes ASA r2**.	""
`limitAggregateUsage`	L'approvisionnement échouera si l'utilisation dépasse ce pourcentage. Si vous utilisez un backend Amazon FSx for NetApp ONTAP , ne spécifiez pas `limitAggregateUsage` . Le fourni `fsxadmin` et `vsadmin` ne contiennent pas les autorisations requises pour récupérer l'utilisation agrégée et la limiter à l'aide de Trident. **Ne pas spécifier pour les systèmes ASA r2**.	"" (non appliqué par défaut)

Paramètre	Description	Défaut
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur. Il limite également la taille maximale des volumes qu'il gère pour les LUN.	"" (non appliqué par défaut)
`lunsPerFlexvol`	Nombre maximal de LUN par Flexvol, doit être compris entre 50 et 200.	`100`
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple : {"api":false, "method":true} À n'utiliser que si vous effectuez un dépannage et avez besoin d'un journal détaillé.	`null`
`useREST`	Paramètre booléen pour utiliser les API REST ONTAP . `` `useREST`Lorsqu'il est réglé sur `true` Trident utilise les API REST ONTAP pour communiquer avec le système dorsal ; lorsqu'il est configuré pour `false` Trident utilise des appels ONTAPI (ZAPI) pour communiquer avec le backend. Cette fonctionnalité nécessite ONTAP 9.11.1 et versions ultérieures. De plus, le rôle de connexion ONTAP utilisé doit avoir accès à `ontapi` application. Ceci est satisfait par la définition prédéfinie `vsadmin` et `cluster-admin` rôles. À compter de la version Trident 24.06 et ONTAP 9.15.1 ou ultérieure, `useREST` est réglé sur `true` par défaut ; modifier `useREST` à `false` utiliser les appels ONTAPI (ZAPI). `` `useREST`est entièrement compatible NVMe/TCP. NVMe est pris en charge uniquement avec les API REST ONTAP et n'est pas pris en charge avec ONTAPI (ZAPI). **Si spécifié, toujours définir sur `true` pour les systèmes ASA r2**.	`true`pour ONTAP 9.15.1 ou version ultérieure, sinon `false`.

Paramètre	Description	Défaut
`sanType`	Utiliser pour sélectionner `iscsi` pour iSCSI, `nvme` pour NVMe/TCP ou `fcp` pour SCSI sur Fibre Channel (FC).	`iscsi`si vide
`formatOptions`	Utiliser `formatOptions` pour spécifier les arguments de ligne de commande pour le `mkfs` commande, qui sera appliquée à chaque formatage d'un volume. Cela vous permet de formater le volume selon vos préférences. Veillez à spécifier les options de formatage similaires à celles de la commande mkfs, en excluant le chemin du périphérique. Exemple : "-E nodiscard" **Prise en charge pour `ontap-san` et `ontap-san-economy` pilotes avec protocole iSCSI. De plus, cette fonctionnalité est prise en charge pour les systèmes ASA r2 lors de l'utilisation des protocoles iSCSI et NVMe/TCP.**	
`limitVolumePoolSize`	Taille FlexVol maximale requise lors de l'utilisation de LUN dans le backend ontap-san-economy.	"" (non appliqué par défaut)
`denyNewVolumePools`	Restreint `ontap-san-economy` les backends de création de nouveaux volumes FlexVol pour contenir leurs LUN. Seuls les Flexvols préexistants sont utilisés pour provisionner de nouveaux PV.	

Recommandations pour l'utilisation de formatOptions

Trident recommande l'option suivante pour accélérer le processus de mise en forme :

-E nodiscard:

- Conservez les blocs, n'essayez pas de les supprimer lors de la création du système de fichiers MKFS (la suppression initiale des blocs est utile sur les périphériques à semi-conducteurs et le stockage clairsemé/à provisionnement fin). Cela remplace l'option obsolète « -K » et s'applique à tous les systèmes de fichiers (xfs, ext3 et ext4).

Authentifier Trident auprès d'une SVM principale à l'aide des informations d'identification Active Directory

Vous pouvez configurer Trident pour s'authentifier auprès d'une SVM principale à l'aide des informations d'identification Active Directory (AD). Avant qu'un compte AD puisse accéder au SVM, vous devez configurer l'accès du contrôleur de domaine AD au cluster ou au SVM. Pour l'administration du cluster avec un compte AD, vous devez créer un tunnel de domaine. Se référer à "Configurer l'accès au contrôleur de domaine Active Directory dans ONTAP" pour plus de détails.

mesures

1. Configurer les paramètres du système de noms de domaine (DNS) pour un SVM backend :

```
vserver services dns create -vserver <svm_name> -dns-servers
<dns_server_ip1>,<dns_server_ip2>
```

2. Exécutez la commande suivante pour créer un compte d'ordinateur pour le SVM dans Active Directory :

```
vserver active-directory create -vserver DataSVM -account-name ADSERVER1
-domain demo.netapp.com
```

3. Utilisez cette commande pour créer un utilisateur ou un groupe AD pour gérer le cluster ou le SVM

```
security login create -vserver <svm_name> -user-or-group-name
<ad_user_or_group> -application <application> -authentication-method domain
-role vsadmin
```

4. Dans le fichier de configuration du backend Trident , définissez le `username` et `password` paramètres au nom d'utilisateur ou de groupe AD et au mot de passe, respectivement.

Options de configuration backend pour les volumes de provisionnement

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

Paramètre	Description	Défaut
`spaceAllocation`	Allocation d'espace pour les LUN	"true" **Si spécifié, définir sur `true` pour les systèmes ASA r2**.
`spaceReserve`	Mode de réservation d'espace ; « aucun » (mince) ou « volume » (épais). **Réglé sur `none` pour les systèmes ASA r2**.	"aucun"
`snapshotPolicy`	Politique d'instantané à utiliser. **Réglé sur `none` pour les systèmes ASA r2**.	"aucun"
`qosPolicy`	Groupe de stratégie QoS à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage/backend. L'utilisation des groupes de politiques QoS avec Trident nécessite ONTAP 9.8 ou une version ultérieure. Vous devez utiliser un groupe de stratégies QoS non partagé et vous assurer que ce groupe de stratégies est appliqué individuellement à chaque composant. Un groupe de politiques QoS partagé impose un plafond au débit total de toutes les charges de travail.	""
`adaptiveQosPolicy`	Groupe de stratégie QoS adaptatif à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage/backend.	""
`snapshotReserve`	Pourcentage du volume réservé aux instantanés. **Ne pas spécifier pour les systèmes ASA r2**.	"0" si `snapshotPolicy` est « aucun », sinon « »
`splitOnClone`	Séparer un clone de son parent lors de sa création	"FAUX"

Paramètre	Description	Défaut
`encryption`	Activez le chiffrement de volume NetApp (NVE) sur le nouveau volume ; la valeur par défaut est `false` . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NAE. Pour plus d'informations, veuillez consulter :"Comment Trident fonctionne avec NVE et NAE" .	"false" **Si spécifié, définir sur `true` pour les systèmes ASA r2**.
`luksEncryption`	Activer le chiffrement LUKS. Se référer à"Utiliser Linux Unified Key Setup (LUKS)" .	"" **Réglé sur `false` pour les systèmes ASA r2**.
`tieringPolicy`	Politique de hiérarchisation à utiliser « aucune » **Ne pas spécifier pour les systèmes ASA r2** .	
`nameTemplate`	Modèle pour créer des noms de volumes personnalisés.	""

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
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` Le pilote Trident ajoute 10 % de capacité supplémentaire au FlexVol pour prendre en charge les métadonnées LUN. Le LUN sera configuré avec la taille exacte demandée par l'utilisateur dans le PVC. Trident ajoute 10 % au FlexVol (affiché comme taille disponible dans ONTAP). Les utilisateurs recevront désormais la capacité utilisable qu'ils ont demandée. Cette modification empêche également les LUN de devenir en lecture seule à moins que l'espace disponible ne soit entièrement utilisé. Ceci ne s'applique pas à ontap-san-economy.

Pour les backends qui définissent `snapshotReserve` Trident calcule la taille des volumes comme suit :

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

Le 1.1 correspond aux 10 % supplémentaires ajoutés par Trident au FlexVol pour prendre en charge les métadonnées LUN Pour `snapshotReserve` = 5 %, et demande PVC = 5 Gio, la taille totale du volume est de 5,79 Gio et la taille disponible est de 5,5 Gio. Le `volume show` La commande devrait afficher des résultats similaires à cet exemple :

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

Exemples de configuration minimale

Les exemples suivants présentent des configurations de base qui laissent la plupart des paramètres par défaut. Voici la manière la plus simple de définir un backend.

Si vous utilisez Amazon FSx sur NetApp ONTAP avec Trident, NetApp recommande de spécifier les noms DNS des LIF au lieu des adresses IP.

Exemple ONTAP SAN

Il s'agit d'une configuration de base utilisant le `ontap-san` conducteur.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

Exemple de MetroCluster

Vous pouvez configurer le système dorsal pour éviter d'avoir à mettre à jour manuellement sa définition après un basculement et un retour en arrière."Réplication et récupération SVM" .

Pour une transition et un retour en arrière sans interruption, spécifiez le SVM en utilisant `managementLIF` et omettre le `svm` paramètres. Par exemple:

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Exemple d'économie ONTAP SAN

```
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

Exemple d'authentification par certificat

Dans cet exemple de configuration de base `clientCertificate` , `clientPrivateKey` , et `trustedCACertificate` (facultatif, si vous utilisez une autorité de certification de confiance) sont renseignés dans `backend.json` et prendre respectivement les valeurs encodées en base64 du certificat client, de la clé privée et du certificat d'autorité de certification de confiance.

```
---
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
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Exemples CHAP bidirectionnels

Ces exemples créent un backend avec `useCHAP` défini à `true` .

Exemple ONTAP SAN CHAP

```
---
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
username: vsadmin
password: <password>
```

Exemple d'économie ONTAP SAN CHAP

```
---
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
username: vsadmin
password: <password>
```

Exemple NVMe/TCP

Vous devez disposer d'une SVM configurée avec NVMe sur votre serveur ONTAP . Il s'agit d'une configuration de base pour le backend NVMe/TCP.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

Exemple SCSI sur FC (FCP)

Vous devez disposer d'une SVM configurée avec FC sur votre backend ONTAP . Voici une configuration backend de base pour FC.

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

Exemple de configuration backend avec nameTemplate

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Exemple d'options de formatage pour le pilote ontap-san-economy

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

Exemples de serveurs backend avec pools virtuels

Dans ces exemples de fichiers de définition de backend, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, telles que : `spaceReserve` à aucun, `spaceAllocation` à faux, et `encryption` à faux. Les pools virtuels sont définis dans la section stockage.

Trident définit les étiquettes de provisionnement dans le champ « Commentaires ». Les commentaires sont définis sur le FlexVol volume. Trident copie toutes les étiquettes présentes sur un pool virtuel vers le volume de stockage lors de la mise en service. Pour plus de commodité, les administrateurs de stockage peuvent définir des étiquettes par pool virtuel et regrouper les volumes par étiquette.

Dans ces exemples, certains pools de stockage définissent leurs propres paramètres `spaceReserve` , `spaceAllocation` , et `encryption` valeurs, et certains pools remplacent les valeurs par défaut.

Exemple ONTAP SAN

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
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"
```

Exemple d'économie ONTAP SAN

```
---
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
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"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1c
```

```
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

Exemple NVMe/TCP

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: "false"
  encryption: "true"
storage:
  - labels:
      app: testApp
      cost: "20"
    defaults:
      spaceAllocation: "false"
      encryption: "false"
```

Associer les backends aux StorageClasses

Les définitions de StorageClass suivantes font référence àExemples de serveurs backend avec pools virtuels . En utilisant le `parameters.selector` Dans ce champ, chaque StorageClass indique quels pools virtuels peuvent être utilisés pour héberger un volume. Le volume aura les aspects définis dans le pool virtuel choisi.

- Le `protection-gold` StorageClass sera associé au premier pool virtuel dans le `ontap-san` backend. Il s'agit de la seule piscine offrant une protection de niveau or.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- Le `protection-not-gold` StorageClass sera associé au deuxième et au troisième pool virtuel dans `ontap-san` backend. Ce sont les seuls pools offrant un niveau de protection autre que l'or.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- Le `app-mysqldb` StorageClass sera associé au troisième pool virtuel dans `ontap-san-economy` backend. Il s'agit du seul pool offrant une configuration de pool de stockage pour les applications de type mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- Le `protection-silver-creditpoints-20k` StorageClass sera associé au deuxième pool virtuel dans `ontap-san` backend. Il s'agit du seul fonds de placement offrant une protection de niveau argent et 20 000 points de crédit.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- Le `creditpoints-5k` StorageClass sera associé au troisième pool virtuel dans `ontap-san` le backend et le quatrième pool virtuel dans le `ontap-san-economy` backend. Ce sont les seules offres de piscine avec 5000 points de crédit.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

- Le `my-test-app-sc` StorageClass sera associé à `testAPP` piscine virtuelle dans le `ontap-san` conducteur avec `sanType: nvme` . C'est la seule piscine proposée `testApp` .

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-test-app-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=testApp"
  fsType: "ext4"
```

Trident déterminera quel pool virtuel sera sélectionné et s'assurera que les besoins en stockage sont satisfaits.

Pilotes ONTAP NAS

Présentation du pilote ONTAP NAS

Découvrez comment configurer un backend ONTAP avec les pilotes NAS ONTAP et Cloud Volumes ONTAP .

Détails du pilote ONTAP NAS

Trident fournit les pilotes de stockage NAS suivants pour communiquer avec le cluster ONTAP . Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-nas`	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`
`ontap-nas-economy`	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`

Conducteur	Protocole	mode de volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-nas-flexgroup`	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`

- Utiliser `ontap-san-economy` uniquement si le nombre d'utilisations de volume persistantes devrait être supérieur à"limites de volume ONTAP prises en charge" .
- Utiliser `ontap-nas-economy` uniquement si le nombre d'utilisations de volume persistantes devrait être supérieur à"limites de volume ONTAP prises en charge" et le `ontap-san-economy` Le pilote ne peut pas être utilisé.
- Ne pas utiliser `ontap-nas-economy` si vous prévoyez un besoin en matière de protection des données, de reprise après sinistre ou de mobilité.
- NetApp ne recommande pas l'utilisation de la croissance automatique Flexvol dans tous les pilotes ONTAP , à l'exception de ontap-san. En guise de solution de contournement, Trident prend en charge l'utilisation de la réserve de snapshots et adapte les volumes Flexvol en conséquence.

Autorisations de l'utilisateur

Trident s'attend à être exécuté en tant qu'administrateur ONTAP ou SVM, généralement en utilisant `admin` utilisateur de cluster ou un `vsadmin` Utilisateur SVM, ou un utilisateur portant un nom différent mais ayant le même rôle.

Pour les déploiements Amazon FSx for NetApp ONTAP , Trident s'attend à être exécuté en tant qu'administrateur ONTAP ou SVM, en utilisant le cluster. `fsxadmin` utilisateur ou un `vsadmin` Utilisateur SVM, ou un utilisateur portant un nom différent mais ayant le même rôle. Le `fsxadmin` L'utilisateur est un remplaçant limité pour l'utilisateur administrateur du cluster.

Si vous utilisez le `limitAggregateUsage` Les paramètres suivants sont requis : autorisations d'administrateur de cluster. Lors de l'utilisation Amazon FSx for NetApp ONTAP avec Trident, `limitAggregateUsage` Le paramètre ne fonctionnera pas avec le `vsadmin` et `fsxadmin` comptes utilisateurs. L'opération de configuration échouera si vous spécifiez ce paramètre.

Bien qu'il soit possible de créer un rôle plus restrictif au sein ONTAP qu'un pilote Trident puisse utiliser, nous ne le recommandons pas. La plupart des nouvelles versions de Trident feront appel à des API supplémentaires dont il faudra tenir compte, ce qui rendra les mises à niveau difficiles et sujettes aux erreurs.

Préparez-vous à configurer un serveur dorsal avec des pilotes NAS ONTAP.

Comprendre les exigences, les options d'authentification et les politiques d'exportation pour configurer un backend ONTAP avec les pilotes ONTAP NAS.

Exigences

- Pour tous les backends ONTAP , Trident exige qu'au moins un agrégat soit affecté au SVM.
- Vous pouvez 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 `ontap-nas` pilote et une classe Bronze qui utilise le `ontap-nas-economy` un.

- Tous vos nœuds de travail Kubernetes doivent avoir les outils NFS appropriés installés. Se référer à"ici" pour plus de détails.
- Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows. Se référer à Préparez-vous à provisionner des volumes PME pour plus de détails.

Authentifier le backend ONTAP

Trident propose deux modes d'authentification pour un système dorsal ONTAP .

- Authentification par identifiants : ce mode nécessite des autorisations suffisantes sur le serveur ONTAP . Il est recommandé d'utiliser un compte associé à un rôle de connexion de sécurité prédéfini, tel que `admin` ou `vsadmin` pour assurer une compatibilité maximale avec les versions ONTAP .
- Mode basé sur un certificat : ce mode nécessite un certificat installé sur le serveur pour que Trident puisse communiquer avec un cluster ONTAP . Ici, la définition du backend doit contenir les valeurs encodées en Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance si utilisé (recommandé).

Vous pouvez mettre à jour les systèmes d'arrière-plan existants pour passer d'une méthode basée sur les identifiants à une méthode basée sur les certificats. Cependant, une seule méthode d'authentification est prise en charge à la fois. Pour passer à une autre méthode d'authentification, vous devez supprimer la méthode actuelle de la configuration du serveur.

> Si vous tentez de fournir **à la fois des identifiants et des certificats**, la création du backend échouera avec une erreur indiquant que plusieurs méthodes d'authentification ont été fournies dans le fichier de configuration.

Activer l'authentification par identifiants

Trident a besoin des identifiants d'un administrateur au niveau SVM/cluster pour communiquer avec le backend ONTAP . Il est recommandé d'utiliser des rôles standard prédéfinis tels que `admin` ou `vsadmin` . Cela garantit la compatibilité ascendante avec les futures versions ONTAP qui pourraient exposer des API de fonctionnalités utilisables par les futures versions de Trident . Il est possible de créer et d'utiliser un rôle de connexion de sécurité personnalisé avec Trident, mais cela n'est pas recommandé.

Voici un exemple de définition de backend :

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

N'oubliez pas que la définition du backend est le seul endroit où les identifiants sont stockés en clair. Une fois le backend créé, les noms d'utilisateur et les mots de passe sont encodés en Base64 et stockés en tant que secrets Kubernetes. La création/mise à jour d'un backend est la seule étape qui nécessite la connaissance des identifiants. Il s'agit donc d'une opération réservée aux administrateurs, qui doit être effectuée par l'administrateur Kubernetes/stockage.

Activer l'authentification par certificat

Les nouveaux et les existants serveurs dorsaux peuvent utiliser un certificat et communiquer avec le serveur dorsal ONTAP . Trois paramètres sont requis dans la définition du backend.

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

Un flux de travail typique comprend les étapes suivantes.

Étapes

1. Générer un certificat client et une clé. Lors de la génération, définissez le nom commun (CN) sur l'utilisateur ONTAP sous lequel s'authentifier.

```
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. Ajouter un certificat d'autorité de certification de confiance au cluster ONTAP . Cela est peut-être déjà géré par l'administrateur du stockage. Ignorer si aucune autorité de certification de confiance 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 (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 prend en charge `cert` méthode 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 du certificat généré. Remplacez < ONTAP Management LIF> et <nom du serveur virtuel> par l'adresse IP de l'interface de gestion LIF et le nom du SVM. Vous devez vous assurer que le LIF a sa politique de service configurée comme suit : `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 d'autorité de certification de confiance au format 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 backend en utilisant les valeurs obtenues à 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...Vaaalllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Mettez à jour les méthodes d'authentification ou changez les identifiants.

Vous pouvez mettre à jour un système dorsal existant pour utiliser une méthode d'authentification différente ou pour renouveler ses identifiants. Cela fonctionne dans les deux sens : les systèmes d'arrière-plan qui utilisent un nom d'utilisateur/mot de passe peuvent être mis à jour pour utiliser des certificats ; les systèmes d'arrière-plan qui utilisent des certificats peuvent être mis à jour pour utiliser un 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 pour 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 étape est suivie d'une mise à jour du système dorsal. Lors de la rotation des certificats, plusieurs certificats peuvent être ajoutés à l'utilisateur. Le système dorsal est ensuite mis à jour pour utiliser le nouveau certificat, après quoi l'ancien certificat peut être supprimé du cluster ONTAP .

La mise à jour d'un système dorsal n'interrompt pas l'accès aux volumes déjà créés et n'a aucun impact sur les connexions de volumes effectuées ultérieurement. Une mise à jour réussie du système dorsal indique que Trident peut communiquer avec le système dorsal ONTAP et gérer les futures opérations de volume.

Créer un rôle ONTAP personnalisé pour Trident

Vous pouvez créer un rôle de cluster ONTAP avec des privilèges minimaux afin de ne pas avoir à utiliser le rôle d'administrateur ONTAP pour effectuer des opérations dans Trident. Lorsque vous incluez le nom d'utilisateur dans une configuration backend Trident , Trident utilise le rôle de cluster ONTAP que vous avez créé pour effectuer les opérations.

Se référer à"Générateur de rôles personnalisés Trident" pour plus d'informations sur la création de rôles personnalisés Trident .

Utilisation de l'interface de ligne de commande ONTAP

1. Créez un nouveau rôle à l'aide de la commande suivante :

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Créez un nom d'utilisateur pour l'utilisateur Trident :

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Associer le rôle à l'utilisateur :

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilisation du gestionnaire système

Effectuez les étapes suivantes dans ONTAP System Manager :

1. **Créer un rôle personnalisé** :
   a. Pour créer un rôle personnalisé au niveau du cluster, sélectionnez **Cluster > Paramètres**.

      (Ou) Pour créer un rôle personnalisé au niveau de la SVM, sélectionnez **Stockage > Machines virtuelles de stockage > `required SVM` > Paramètres > Utilisateurs et rôles**.

   b. Sélectionnez l'icône flèche (→) à côté de **Utilisateurs et rôles**.
   c. Sélectionnez **+Ajouter** sous **Rôles**.
   d. Définissez les règles du rôle et cliquez sur **Enregistrer**.
2. **Associer le rôle à l'utilisateur Trident * : + Effectuez les étapes suivantes sur la page *Utilisateurs et rôles** :
   a. Sélectionnez l'icône Ajouter **+** sous **Utilisateurs**.
   b. Sélectionnez le nom d'utilisateur requis, puis sélectionnez un rôle dans le menu déroulant **Rôle**.
   c. Cliquez sur **Enregistrer**.

Pour plus d'informations, veuillez consulter les pages suivantes :

- "Rôles personnalisés pour l'administration d' ONTAP"ou"Définir des rôles personnalisés"
- "Collaborer avec les rôles et les utilisateurs"

Gérer les politiques d'exportation NFS

Trident utilise des politiques d'exportation NFS pour contrôler l'accès aux volumes qu'il provisionne.

Trident propose deux options pour la gestion des politiques d'exportation :

- Trident peut gérer dynamiquement la politique d'exportation elle-même ; dans ce mode de fonctionnement, l'administrateur de stockage spécifie une liste de blocs CIDR qui représentent des adresses IP admissibles. Trident ajoute automatiquement à la politique d'exportation, lors de la publication, les adresses IP des nœuds concernés qui se trouvent dans ces plages. Sinon, lorsqu'aucun CIDR n'est spécifié, toutes les adresses IP unicast à portée globale trouvées sur le nœud sur lequel le volume est publié seront ajoutées à la politique d'exportation.
- Les administrateurs de stockage peuvent créer une politique d'exportation et ajouter des règles manuellement. Trident utilise la politique d'exportation par défaut, sauf si un nom de politique d'exportation différent est spécifié dans la configuration.

Gérer dynamiquement les politiques d'exportation

Trident offre la possibilité de gérer dynamiquement les politiques d'exportation pour les systèmes backend ONTAP . Cela permet à l'administrateur du stockage de spécifier un espace d'adressage autorisé pour les adresses IP des nœuds de travail, plutôt que de définir manuellement des règles explicites. Cela simplifie considérablement la gestion des politiques d'exportation ; les modifications apportées à la politique d'exportation ne nécessitent plus d'intervention manuelle sur le cluster de stockage. De plus, cela permet de limiter l'accès au cluster de stockage aux seuls nœuds de travail qui montent des volumes et dont les adresses IP se trouvent dans la plage spécifiée, ce qui permet une gestion précise et automatisée.

> ⓘ N'utilisez pas la traduction d'adresses réseau (NAT) lorsque vous utilisez des politiques d'exportation dynamiques. Avec NAT, le contrôleur de stockage voit l'adresse NAT frontale et non l'adresse IP réelle de l'hôte ; l'accès sera donc refusé si aucune correspondance n'est trouvée dans les règles d'exportation.

Exemple

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

```
---
version: 1
storageDriverName: ontap-nas-economy
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 racine de votre SVM dispose d'une stratégie d'exportation préalablement créée avec une règle d'exportation autorisant le bloc CIDR du nœud (telle que la stratégie d'exportation par défaut). Suivez toujours les bonnes pratiques recommandées par NetApp pour dédier une SVM à Trident.

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

- `autoExportPolicy`est réglé sur `true`. Cela indique que Trident crée une politique d'exportation pour chaque volume provisionné avec ce backend pour le `svm1` SVM et gestion de l'ajout et de la suppression de règles à l'aide de `autoexportCIDRs` blocs d'adresses. Tant qu'un volume n'est pas attaché à un nœud, le volume utilise une politique d'exportation vide, sans aucune règle pour empêcher les accès non autorisés à ce volume. Lorsqu'un volume est publié sur un nœud, Trident crée une politique d'exportation portant le même nom que l'arbre qtree sous-jacent contenant l'adresse IP du nœud dans le bloc CIDR spécifié. Ces adresses IP seront également ajoutées à la politique d'exportation utilisée par le FlexVol volume parent.
  - Par exemple:
    - UUID du serveur dorsal : 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy`défini à `true`
    - préfixe de stockage `trident`
    - UUID PVC a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - L'arbre qtree nommé trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c crée une politique d'exportation pour le FlexVol nommé `trident-403b5326-8482-40db96d0-d83fb3f4daec`, une politique d'exportation pour le qtree nommé `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` et une politique d'exportation vide nommée `trident_empty` sur la SVM. Les règles de la politique d'exportation FlexVol seront un sur-ensemble de toutes les règles contenues dans les politiques d'exportation qtree. La stratégie d'exportation vide sera réutilisée par tous les volumes non attachés.
- `autoExportCIDRs`contient une liste de blocs d'adresses. Ce champ est facultatif et sa valeur par défaut est ["0.0.0.0/0", "::/0"]. Si aucune adresse n'est définie, Trident ajoute toutes les adresses unicast à portée globale trouvées sur les nœuds de travail comportant des publications.

Dans cet exemple, le `192.168.0.0/24` Un espace d'adressage est prévu. Cela indique que les adresses IP des nœuds Kubernetes qui se trouvent dans cette plage d'adresses et qui contiennent des publications seront ajoutées à la politique d'exportation créée par Trident . Lorsque Trident enregistre un nœud sur lequel il s'exécute, il récupère les adresses IP du nœud et les compare aux blocs d'adresses fournis dans `autoExportCIDRs` Au moment de la publication, après avoir filtré les adresses IP, Trident crée les règles de stratégie d'exportation pour les adresses IP clientes du nœud sur lequel il publie.

Vous pouvez mettre à jour `autoExportPolicy` et `autoExportCIDRs` pour les backends une fois que vous les avez créés. Vous pouvez ajouter de nouveaux CIDR pour un backend géré automatiquement ou supprimer les CIDR existants. Soyez prudent lors de la suppression des CIDR afin de vous assurer que les connexions existantes ne sont pas interrompues. Vous pouvez également choisir de désactiver `autoExportPolicy` pour un système dorsal et recourir à une politique d'exportation créée manuellement en cas de besoin. Cela nécessitera de paramétrer le `exportPolicy` paramètre dans votre configuration backend.

Une fois que Trident a créé ou mis à jour un backend, vous pouvez le vérifier 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>
      fileSystemType: ext4
```

Lorsqu'un nœud est supprimé, Trident vérifie toutes les politiques d'exportation afin de supprimer les règles d'accès correspondant à ce nœud. En supprimant cette adresse IP de nœud des politiques d'exportation des backends gérés, Trident empêche les montages non autorisés, sauf si cette adresse IP est réutilisée par un nouveau nœud du cluster.

Pour les backends existants, la mise à jour du backend avec `tridentctl update backend` garantit que Trident gère automatiquement les politiques d'exportation. Cela crée deux nouvelles politiques d'exportation nommées d'après l'UUID et le nom qtree du backend lorsque cela est nécessaire. Les volumes présents sur le système dorsal utiliseront les politiques d'exportation nouvellement créées après avoir été démontés puis remontés.

> La suppression d'un backend avec des politiques d'exportation gérées automatiquement supprimera la politique d'exportation créée dynamiquement. Si le système dorsal est recréé, il est traité comme un nouveau système dorsal et entraînera la création d'une nouvelle politique d'exportation.

Si l'adresse IP d'un nœud en production est mise à jour, vous devez redémarrer le pod Trident sur ce nœud. Trident mettra ensuite à jour sa politique d'exportation pour les serveurs backend qu'elle gère afin de refléter ce changement d'adresse IP.

Préparez-vous à provisionner des volumes PME

Avec un peu de préparation supplémentaire, vous pouvez provisionner des volumes SMB en utilisant `ontap-nas` conducteurs.

Vous devez configurer les protocoles NFS et SMB/CIFS sur la SVM pour créer un `ontap-nas-economy` Volume SMB pour les clusters ONTAP sur site. Le défaut de configuration de l'un ou l'autre de ces protocoles entraînera l'échec de la création du volume SMB.

`autoExportPolicy`La prise en charge des volumes SMB n'est pas assurée.

Avant de commencer

Avant de pouvoir provisionner des volumes SMB, vous devez disposer des éléments suivants.

- Un cluster Kubernetes avec un nœud contrôleur Linux et au moins un nœud de travail Windows exécutant Windows Server 2022. Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows.
- Au moins un secret Trident contenant vos informations d'identification Active Directory. Générer des secrets `smbcreds` :

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

- Un proxy CSI configuré comme un service Windows. Pour configurer un `csi-proxy` , se référer à"GitHub : CSI Proxy" ou"GitHub : CSI Proxy pour Windows" pour les nœuds Kubernetes exécutés sous Windows.

Étapes

1. Pour ONTAP sur site, vous pouvez créer un partage SMB en option ou Trident peut en créer un pour vous.

   Les partages SMB sont requis pour Amazon FSx pour ONTAP.

   Vous pouvez créer les partages d'administration SMB de deux manières : soit en utilisant…"Console de gestion Microsoft" composant logiciel enfichable Dossiers partagés ou via l'interface de ligne de commande ONTAP . Pour créer les partages SMB à l'aide de l'interface de ligne de commande ONTAP :

   a. Si nécessaire, créez la structure de chemin d'accès au répertoire partagé.

      Le `vserver cifs share create` Cette 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 à la SVM spécifiée :

```
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 bien été créé :

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

Se référer à"Créer un partage SMB" pour plus de détails.

2. Lors de la création du backend, vous devez configurer les éléments suivants pour spécifier les volumes SMB. Pour connaître toutes les options de configuration du backend FSx pour ONTAP , veuillez vous référer à"Options et exemples de configuration de FSx pour ONTAP" .

Paramètre	Description	Exemple
`smbShare`	Vous pouvez spécifier l'un des éléments suivants : le nom d'un partage SMB créé à l'aide de la console de gestion Microsoft ou de l'interface de ligne de commande ONTAP ; un nom permettant à Trident de créer le partage SMB ; ou vous pouvez laisser le paramètre vide pour empêcher l'accès aux volumes via un partage commun. Ce paramètre est facultatif pour ONTAP sur site. Ce paramètre est obligatoire pour les serveurs backend Amazon FSx for ONTAP et ne peut pas être vide.	`smb-share`
`nasType`	**Doit être réglé sur `smb` .** Si la valeur est nulle, la valeur par défaut est `nfs` .	`smb`
`securityStyle`	Style de sécurité pour les nouveaux volumes. **Doit être réglé 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.**	""

Activer le SMB sécurisé

À partir de la version 25.06, NetApp Trident prend en charge le provisionnement sécurisé des volumes SMB créés à l'aide de `ontap-nas` et `ontap-nas-economy` backends. Lorsque le protocole SMB sécurisé est activé, vous pouvez fournir un accès contrôlé aux partages SMB pour les utilisateurs et les groupes d'utilisateurs Active Directory (AD) à l'aide de listes de contrôle d'accès (ACL).

Points à retenir

- Importer `ontap-nas-economy` Les volumes ne sont pas pris en charge.
- Seuls les clones en lecture seule sont pris en charge pour `ontap-nas-economy` volumes.
- Si le protocole SMB sécurisé est activé, Trident ignorera le partage SMB mentionné dans le système dorsal.
- La mise à jour de l'annotation PVC, de l'annotation de classe de stockage et du champ backend ne met pas à jour la liste de contrôle d'accès (ACL) du partage SMB.
- La liste de contrôle d'accès (ACL) de partage SMB spécifiée dans l'annotation du PVC cloné aura priorité sur celle du PVC source.
- Veillez à fournir des utilisateurs AD valides tout en activant le protocole SMB sécurisé. Les utilisateurs non valides ne seront pas ajoutés à la liste de contrôle d'accès (ACL).
- Si vous fournissez le même utilisateur AD dans le backend, la classe de stockage et le PVC avec des autorisations différentes, la priorité des autorisations sera la suivante : PVC, classe de stockage, puis backend.

- Le protocole SMB sécurisé est pris en charge pour `ontap-nas` S'applique aux importations de volumes gérés et non aux importations de volumes non gérés.

Étapes

1. Spécifiez adAdminUser dans TridentBackendConfig comme indiqué dans l'exemple suivant :

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Ajoutez l'annotation dans la classe de stockage.

   Ajoutez le `trident.netapp.io/smbShareAdUser` Annotation à la classe de stockage pour activer le protocole SMB sécurisé sans erreur. La valeur utilisateur spécifiée pour l'annotation `trident.netapp.io/smbShareAdUser` doit être identique au nom d'utilisateur spécifié dans le `smbcreds` secrète. Vous pouvez choisir l'une des options suivantes pour `smbShareAdUserPermission` : `full_control` , `change` , ou `read` . L'autorisation par défaut est `full_control` .

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. Créer un PVC.

   L'exemple suivant crée un PVC :

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Options et exemples de configuration ONTAP NAS

Apprenez à créer et à utiliser des pilotes ONTAP NAS avec votre installation Trident . Cette section fournit des exemples de configuration backend et des détails sur le mappage des backends aux StorageClasses.

options de configuration du backend

Consultez le tableau suivant pour connaître les options de configuration du backend :

Paramètre	Description	Défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	`ontap-nas`, `ontap-nas-economy` , ou `ontap-nas-flexgroup`
`backendName`	Nom personnalisé ou système de stockage	Nom du conducteur + "_" + dataLIF

Paramètre	Description	Défaut
`managementLIF`	Adresse IP d'une interface de gestion de cluster ou SVM (LIF) Un nom de domaine pleinement qualifié (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme ceci : `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Pour une transition MetroCluster sans interruption, consultez la documentation.Exemple de MetroCluster .	"10.0.0.1", "[2001:1234:abcd::fefe]"
`dataLIF`	Adresse IP du protocole LIF. NetApp recommande de spécifier `dataLIF` . Si aucune donnée n'est fournie, Trident récupère les dataLIF à partir du SVM. Vous pouvez spécifier un nom de domaine pleinement qualifié (FQDN) à utiliser pour les opérations de montage NFS, ce qui vous permet de créer un DNS à répartition circulaire pour équilibrer la charge sur plusieurs dataLIF. Peut être modifié après la configuration initiale. Se référer à . Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme ceci : `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Omettre pour Metrocluster.** Voir leExemple de MetroCluster .	Adresse spécifiée ou dérivée de la SVM, si non spécifiée (non recommandé)
`svm`	Machine virtuelle de stockage à utiliser **Omettre pour Metrocluster.** Voir leExemple de MetroCluster .	Dérivé d'un SVM `managementLIF` est spécifié
`autoExportPolicy`	Activer la création et la mise à jour automatiques de la politique d'exportation [Booléen]. En utilisant le `autoExportPolicy` et `autoExportCIDRs` Avec certaines options, Trident peut gérer automatiquement les politiques d'exportation.	FAUX
`autoExportCIDRs`	Liste des CIDR à utiliser pour filtrer les adresses IP des nœuds Kubernetes lorsque `autoExportPolicy` est activé. En utilisant le `autoExportPolicy` et `autoExportCIDRs` Avec certaines options, Trident peut gérer automatiquement les politiques d'exportation.	["0.0.0.0/0", "::/0"]`
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	""
`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 d'autorité de certification de confiance. Facultatif. Utilisé pour l'authentification par certificat	""

Paramètre	Description	Défaut
`username`	Nom d'utilisateur pour se connecter au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants. Pour l'authentification Active Directory, voir "Authentifier Trident auprès d'une SVM principale à l'aide des informations d'identification Active Directory".	
`password`	Mot de passe pour se connecter au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants. Pour l'authentification Active Directory, voir "Authentifier Trident auprès d'une SVM principale à l'aide des informations d'identification Active Directory".	
`storagePrefix`	Préfixe utilisé lors de la mise en service de nouveaux volumes dans la SVM. Impossible de le mettre à jour après l'avoir configuré. Lors de l'utilisation d'ontap-nas-economy et d'un préfixe de stockage de 24 caractères ou plus, les qtrees n'auront pas le préfixe de stockage intégré, bien qu'il soit présent dans le nom du volume.	"trident"
`aggregate`	Agrégat pour le provisionnement (facultatif ; s'il est défini, il doit être affecté au SVM). Pour le `ontap-nas-flexgroup` conducteur, cette option est ignorée. Si aucun agrégat n'est attribué, n'importe lequel des agrégats disponibles peut être utilisé pour provisionner un volume FlexGroup . Lorsque l'agrégat est mis à jour dans SVM, il est automatiquement mis à jour dans Trident par interrogation de SVM sans qu'il soit nécessaire de redémarrer le contrôleur Trident . Lorsque vous avez configuré un agrégat spécifique dans Trident pour provisionner des volumes, si l'agrégat est renommé ou déplacé hors du SVM, le backend passera à l'état d'échec dans Trident lors de l'interrogation de l'agrégat SVM. Vous devez soit modifier l'agrégat pour qu'il soit présent sur la SVM, soit le supprimer complètement pour remettre le serveur en ligne.	""

Paramètre	Description	Défaut
`limitAggregateUsage`	L'approvisionnement échouera si l'utilisation dépasse ce pourcentage. **Ne s'applique pas à Amazon FSx pour ONTAP**.	"" (non appliqué par défaut)
liste d'agrégation flexgroup	Liste des agrégats à provisionner (facultatif ; si défini, doit être affecté au SVM). Tous les agrégats affectés au SVM sont utilisés pour provisionner un volume FlexGroup . Pris en charge par le pilote de stockage **ontap-nas-flexgroup**. Lorsque la liste agrégée est mise à jour dans SVM, la liste est automatiquement mise à jour dans Trident par interrogation de SVM sans qu'il soit nécessaire de redémarrer le contrôleur Trident . Lorsque vous avez configuré une liste d'agrégats spécifique dans Trident pour provisionner des volumes, si la liste d'agrégats est renommée ou déplacée hors de SVM, le backend passera à l'état d'échec dans Trident lors de l'interrogation de l'agrégat SVM. Vous devez soit modifier la liste agrégée pour utiliser une liste présente sur le SVM, soit la supprimer complètement pour remettre le serveur en ligne.	""
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur. Il limite également la taille maximale des volumes qu'il gère pour les qtrees, et le `qtreesPerFlexvol` Cette option permet de personnaliser le nombre maximal d'arbres qtree par FlexVol volume	"" (non appliqué par défaut)
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple : {"api":false, "method":true} Ne pas utiliser `debugTraceFlags` sauf si vous effectuez un dépannage et avez besoin d'un journal détaillé.	nul
`nasType`	Configurer la création de volumes NFS ou SMB. Les options sont `nfs` , `smb` ou nul. La valeur nulle correspond par défaut aux volumes NFS.	`nfs`

Paramètre	Description	Défaut
`nfsMountOptions`	Liste des options de montage NFS séparées par des virgules. Les options de montage des volumes persistants Kubernetes sont normalement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Trident utilisera les options de montage spécifiées dans le fichier de configuration du backend de stockage. Si aucune option de montage n'est spécifiée dans la classe de stockage ou dans le fichier de configuration, Trident ne définira aucune option de montage sur un volume persistant associé.	""
`qtreesPerFlexvol`	Nombre maximal d'arbres Q par FlexVol, doit être compris entre 50 et 300.	"200"
`smbShare`	Vous pouvez spécifier l'un des éléments suivants : le nom d'un partage SMB créé à l'aide de la console de gestion Microsoft ou de l'interface de ligne de commande ONTAP ; un nom permettant à Trident de créer le partage SMB ; ou vous pouvez laisser le paramètre vide pour empêcher l'accès aux volumes via un partage commun. Ce paramètre est facultatif pour ONTAP sur site. Ce paramètre est obligatoire pour les serveurs backend Amazon FSx for ONTAP et ne peut pas être vide.	`smb-share`
`useREST`	Paramètre booléen pour utiliser les API REST ONTAP . `useREST` `Lorsqu'il est réglé sur` `true` Trident utilise les API REST ONTAP pour communiquer avec le système dorsal ; lorsqu'il est configuré pour `false` Trident utilise des appels ONTAPI (ZAPI) pour communiquer avec le backend. Cette fonctionnalité nécessite ONTAP 9.11.1 et versions ultérieures. De plus, le rôle de connexion ONTAP utilisé doit avoir accès à `ontapi` application. Ceci est satisfait par la définition prédéfinie `vsadmin` et `cluster-admin` rôles. À compter de la version Trident 24.06 et ONTAP 9.15.1 ou ultérieure, `useREST` est réglé sur `true` par défaut ; modifier `useREST` à `false` utiliser les appels ONTAPI (ZAPI).	`true` `pour ONTAP 9.15.1 ou version ultérieure, sinon` `false .`
`limitVolumePoolSize`	Taille maximale de FlexVol pouvant être demandée lors de l'utilisation de Qtrees dans le backend ontap-nas-economy.	"" (non appliqué par défaut)
`denyNewVolumePools`	Restreint `ontap-nas-economy` les backends créant de nouveaux volumes FlexVol pour contenir leurs Qtrees. Seuls les Flexvols préexistants sont utilisés pour provisionner de nouveaux PV.	

Paramètre	Description	Défaut
`adAdminUser`	Utilisateur administrateur Active Directory ou groupe d'utilisateurs disposant d'un accès complet aux partages SMB. Utilisez ce paramètre pour accorder des droits d'administrateur sur le partage SMB avec un contrôle total.	

Options de configuration backend pour les volumes de provisionnement

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

Paramètre	Description	Défaut
`spaceAllocation`	Allocation d'espace pour les Qtrees	"vrai"
`spaceReserve`	Mode de réservation d'espace ; « aucun » (fin) ou « volume » (épais)	"aucun"
`snapshotPolicy`	Politique d'instantané à utiliser	"aucun"
`qosPolicy`	Groupe de stratégie QoS à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage/backend.	""
`adaptiveQosPolicy`	Groupe de stratégie QoS adaptatif à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage/backend. Non pris en charge par ontap-nas-economy.	""
`snapshotReserve`	Pourcentage du volume réservé aux instantanés	"0" si `snapshotPolicy` est « aucun », sinon « »
`splitOnClone`	Séparer un clone de son parent lors de sa création	"FAUX"
`encryption`	Activez le chiffrement de volume NetApp (NVE) sur le nouveau volume ; la valeur par défaut est `false` . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NAE. Pour plus d'informations, veuillez consulter :"Comment Trident fonctionne avec NVE et NAE" .	"FAUX"
`tieringPolicy`	Politique de hiérarchisation : utiliser « aucun »	
`unixPermissions`	Mode pour les nouveaux volumes	« 777 » pour les volumes NFS ; vide (non applicable) pour les volumes SMB
`snapshotDir`	Contrôle l'accès à `.snapshot` annuaire	« Vrai » pour NFSv4, « Faux » pour NFSv3

Paramètre	Description	Défaut
`exportPolicy`	Politique d'exportation à utiliser	"défaut"
`securityStyle`	Style de sécurité pour les nouveaux volumes. NFS prend en charge `mixed` et `unix` Styles de sécurité. Les PME prennent en charge `mixed` et `ntfs` Styles de sécurité.	La valeur par défaut de NFS est `unix` . La valeur par défaut de SMB est `ntfs` .
`nameTemplate`	Modèle pour créer des noms de volumes personnalisés.	""

L'utilisation des groupes de politiques QoS avec Trident nécessite ONTAP 9.8 ou une version ultérieure. Vous devez utiliser un groupe de stratégies QoS non partagé et vous assurer que ce groupe de stratégies est appliqué individuellement à chaque composant. Un groupe de politiques QoS partagé impose un plafond au 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` Trident utilise désormais un nouveau calcul pour garantir que le FlexVol est correctement dimensionné avec le pourcentage snapshotReserve et le PVC. Lorsqu'un

utilisateur demande un PVC, Trident crée le FlexVol d'origine avec plus d'espace grâce à ce nouveau calcul. Ce calcul garantit que l'utilisateur reçoit l'espace inscriptible demandé dans le PVC, et non un espace inférieur. Avant la version 21.07, lorsqu'un utilisateur demandait un PVC (par exemple, 5 Gio), avec un snapshotReserve à 50 %, il ne recevait que 2,5 Gio d'espace inscriptible. En effet, l'utilisateur a demandé le volume entier et `snapshotReserve` est un pourcentage de cela. Avec Trident 21.07, ce que l'utilisateur demande, c'est l'espace inscriptible, et Trident définit cet espace. `snapshotReserve` nombre en pourcentage du volume total. Cela ne s'applique pas à `ontap-nas-economy` . Consultez l'exemple suivant pour voir comment cela fonctionne :

Le calcul est le suivant :

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

Pour snapshotReserve = 50 % et une demande PVC = 5 Gio, la taille totale du volume est de 5/0,5 = 10 Gio et la taille disponible est de 5 Gio, ce qui correspond à ce que l'utilisateur a demandé dans la demande PVC Le `volume show` La commande devrait afficher des résultats similaires à cet exemple :

Les backends existants des installations précédentes provisionneront les volumes comme expliqué ci-dessus lors de la mise à niveau de Trident. Pour les volumes créés avant la mise à niveau, vous devez les redimensionner afin que la modification soit prise en compte. Par exemple, un PVC de 2 Gio avec `snapshotReserve=50` Cela a précédemment abouti à un volume offrant 1 Gio d'espace inscriptible. Par exemple, le redimensionnement à 3 Gio permet à l'application de disposer de 3 Gio d'espace inscriptible sur un volume de 6 Gio.

Exemples de configuration minimale

Les exemples suivants présentent des configurations de base qui laissent la plupart des paramètres par défaut. Voici la manière la plus simple de définir un backend.

> Si vous utilisez Amazon FSx sur NetApp ONTAP avec Trident, il est recommandé de spécifier les noms DNS des LIF au lieu des adresses IP.

Exemple d'économie NAS ONTAP

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

Exemple de groupe flexible ONTAP NAS

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

Exemple de MetroCluster

Vous pouvez configurer le système dorsal pour éviter d'avoir à mettre à jour manuellement sa définition après un basculement et un retour en arrière."Réplication et récupération SVM" .

Pour une transition et un retour en arrière sans interruption, spécifiez le SVM en utilisant `managementLIF` et omettre le `dataLIF` et `svm` paramètres. Par exemple:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Exemple de volumes SMB

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Exemple d'authentification par certificat

Voici un exemple de configuration minimale du backend. `clientCertificate` , `clientPrivateKey` , et `trustedCACertificate` (facultatif, si vous utilisez une autorité de certification de confiance) sont renseignés dans `backend.json` et prendre respectivement les valeurs encodées en base64 du certificat client, de la clé privée et du certificat d'autorité de certification de confiance.

```
---
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_
```

Exemple de politique d'exportation automatique

Cet exemple vous montre comment configurer Trident pour qu'il utilise des politiques d'exportation dynamiques afin de créer et de gérer automatiquement la politique d'exportation. Cela fonctionne de la même manière pour le `ontap-nas-economy` et `ontap-nas-flexgroup` conducteurs.

```
---
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
```

Exemple d'adresses IPv6

Cet exemple montre `managementLIF` en utilisant 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
```

Exemple d'utilisation Amazon FSx pour ONTAP avec des volumes SMB

Le `smbShare` Ce paramètre est requis pour FSx for 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
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Exemple de configuration backend avec nameTemplate

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Exemples de serveurs backend avec pools virtuels

Dans les exemples de fichiers de définition de backend présentés ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, telles que : `spaceReserve` à aucun, `spaceAllocation` à faux, et `encryption` à faux. Les pools virtuels sont définis dans la section stockage.

Trident définit les étiquettes de provisionnement dans le champ « Commentaires ». Des commentaires sont disponibles sur FlexVol pour `ontap-nas` ou FlexGroup pour `ontap-nas-flexgroup` . Lors de la mise en

service, Trident copie toutes les étiquettes présentes sur un pool virtuel vers le volume de stockage. Pour plus de commodité, les administrateurs de stockage peuvent définir des étiquettes par pool virtuel et regrouper les volumes par étiquette.

Dans ces exemples, certains pools de stockage définissent leurs propres paramètres `spaceReserve` , `spaceAllocation` , et `encryption` valeurs, et certains pools remplacent les valeurs par défaut.

Exemple de NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
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:
      department: legal
      creditpoints: "5000"
    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"
```

Exemple de FlexGroup NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
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"
```

Exemple d'économie NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
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:
      protection: bronze
      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"
```

Associer les backends aux StorageClasses

Les définitions de StorageClass suivantes font référence àExemples de serveurs backend avec pools virtuels . En utilisant le `parameters.selector` Dans ce champ, chaque StorageClass indique quels pools virtuels peuvent être utilisés pour héberger un volume. Le volume aura les aspects définis dans le pool virtuel choisi.

- Le `protection-gold` StorageClass sera associé au premier et au deuxième pool virtuel dans le `ontap-nas-flexgroup` backend. Ce sont les seules piscines à offrir une protection de niveau or.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- Le `protection-not-gold` StorageClass sera associé au troisième et au quatrième pool virtuel dans le `ontap-nas-flexgroup` backend. Ce sont les seuls pools offrant un niveau de protection autre que l'or.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- Le `app-mysqldb` StorageClass sera associé au quatrième pool virtuel dans le `ontap-nas` backend. Il s'agit du seul pool offrant une configuration de pool de stockage pour les applications de type mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- Le `protection-silver-creditpoints-20k` StorageClass sera associé au troisième pool virtuel dans le `ontap-nas-flexgroup` backend. Il s'agit du seul fonds de placement offrant une protection de niveau argent et 20 000 points de crédit.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- Le `creditpoints-5k` StorageClass sera associé au troisième pool virtuel dans le `ontap-nas` le backend et le deuxième pool virtuel dans le `ontap-nas-economy` backend. Ce sont les seules offres de piscine avec 5000 points de crédit.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Trident déterminera quel pool virtuel sera sélectionné et s'assurera que les besoins en stockage sont satisfaits.

Mise à jour `dataLIF` après la configuration initiale

Vous pouvez modifier le dataLIF après la configuration initiale en exécutant la commande suivante pour fournir le nouveau fichier JSON backend avec le dataLIF mis à jour.

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

Si des PVC sont connectés à un ou plusieurs pods, vous devez mettre hors service tous les pods correspondants, puis les remettre en service pour que la nouvelle interface dataLIF prenne effet.

Exemples de PME sécurisées

Configuration du backend avec le pilote ontap-nas

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuration du backend avec le pilote ontap-nas-economy

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuration du backend avec pool de stockage

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Exemple de classe de stockage avec le pilote ontap-nas

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Assurez-vous d'ajouter `annotations` pour activer le SMB sécurisé. Le protocole SMB sécurisé ne fonctionne pas sans les annotations, quelles que soient les configurations définies dans le backend ou le PVC.

Exemple de classe de stockage avec le pilote ontap-nas-economy

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Exemple de PVC avec un seul utilisateur AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Exemple de PVC avec plusieurs utilisateurs AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Amazon FSx for NetApp ONTAP

Utiliser Trident avec Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"est un service AWS entièrement géré qui permet aux clients de lancer et d'exécuter des systèmes de fichiers alimentés par le système d'exploitation de stockage NetApp ONTAP . FSx for ONTAP vous permet de tirer parti des fonctionnalités, des performances et des capacités d'administration de NetApp que vous connaissez, tout en bénéficiant de la simplicité, de l'agilité, de la sécurité et de l'évolutivité du stockage des données sur AWS. FSx pour ONTAP prend en charge les fonctionnalités du système de fichiers ONTAP et les API d'administration.

Vous pouvez intégrer votre système de fichiers Amazon FSx for NetApp ONTAP avec Trident pour garantir que les clusters Kubernetes exécutés dans Amazon Elastic Kubernetes Service (EKS) peuvent provisionner des volumes persistants de blocs et de fichiers pris en charge par ONTAP.

Dans Amazon FSx, le système de fichiers est la ressource principale, analogue à un cluster ONTAP sur site. Au sein de chaque SVM, vous pouvez créer un ou plusieurs volumes, qui sont des conteneurs de données

stockant les fichiers et les dossiers de votre système de fichiers. Avec Amazon FSx for NetApp ONTAP, un système de fichiers géré dans le cloud sera fourni. Le nouveau type de système de fichiers s'appelle * NetApp ONTAP*.

En utilisant Trident avec Amazon FSx for NetApp ONTAP, vous pouvez garantir que les clusters Kubernetes exécutés dans Amazon Elastic Kubernetes Service (EKS) peuvent provisionner des volumes persistants de blocs et de fichiers pris en charge par ONTAP.

Exigences

En plus de"exigences de Trident" Pour intégrer FSx for ONTAP à Trident, vous avez besoin de :

- Un cluster Amazon EKS existant ou un cluster Kubernetes autogéré avec `kubectl` installé.
- Un système de fichiers Amazon FSx for NetApp ONTAP et une machine virtuelle de stockage (SVM) existants, accessibles depuis les nœuds de travail de votre cluster.
- Les nœuds de travail qui sont préparés pour"NFS ou iSCSI" .

> Veillez à suivre les étapes de préparation des nœuds requises pour Amazon Linux et Ubuntu. "Images de machines Amazon" (AMI) en fonction de votre type d'AMI EKS.

Considérations

- Volumes SMB :
  - Les volumes SMB sont pris en charge à l'aide de `ontap-nas` conducteur seulement.
  - Les volumes SMB ne sont pas pris en charge par l'extension Trident EKS.
  - Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows. Se référer à "Préparez-vous à provisionner des volumes PME" pour plus de détails.
- Avant Trident 24.02, les volumes créés sur des systèmes de fichiers Amazon FSx dont les sauvegardes automatiques sont activées ne pouvaient pas être supprimés par Trident. Pour éviter ce problème dans Trident 24.02 ou version ultérieure, spécifiez le `fsxFilesystemID` , AWS `apiRegion` , AWS `apikey` et AWS `secretKey` dans le fichier de configuration backend pour AWS FSx pour ONTAP.

> Si vous spécifiez un rôle IAM pour Trident, vous pouvez omettre de spécifier le `apiRegion` , `apiKey` , et `secretKey` champs à Trident explicitement. Pour plus d'informations, veuillez consulter"Options et exemples de configuration de FSx pour ONTAP" .

Utilisation simultanée des pilotes Trident SAN/iSCSI et EBS-CSI

Si vous prévoyez d'utiliser des pilotes ontap-san (par exemple, iSCSI) avec AWS (EKS, ROSA, EC2 ou toute autre instance), la configuration multi-chemin requise sur les nœuds peut entrer en conflit avec le pilote CSI Amazon Elastic Block Store (EBS). Pour garantir que le multivoie fonctionne sans interférer avec les disques EBS sur le même nœud, vous devez exclure EBS dans votre configuration de multivoie. Cet exemple montre un `multipath.conf` fichier contenant les paramètres Trident requis tout en excluant les disques EBS du multipathing :

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

Authentification

Trident propose deux modes d'authentification.

- Authentification par identifiants (recommandée) : stocke les identifiants en toute sécurité dans AWS Secrets Manager. Vous pouvez utiliser le `fsxadmin` utilisateur pour votre système de fichiers ou le `vsadmin` utilisateur configuré pour votre SVM.

> Trident prévoit d'être géré comme un `vsadmin` Utilisateur SVM ou en tant qu'utilisateur avec un nom différent mais ayant le même rôle. Amazon FSx for NetApp ONTAP possède un `fsxadmin` utilisateur qui remplace partiellement l' ONTAP `admin` utilisateur du cluster. Nous recommandons fortement d'utiliser `vsadmin` avec Trident.

- Communication par certificat : Trident communiquera avec la SVM de votre système de fichiers FSx à l'aide d'un certificat installé sur votre SVM.

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

- "Authentification ONTAP NAS"
- "Authentification SAN ONTAP"

Images machine Amazon (AMI) testées

Le cluster EKS prend en charge divers systèmes d'exploitation, mais AWS a optimisé certaines images de machine Amazon (AMI) pour les conteneurs et EKS. Les AMI suivantes ont été testées avec NetApp Trident 25.02.

AMI	NAS	NAS-économie	iSCSI	économie iSCSI
AL2023_x86_64_STANDARD	Oui	Oui	Oui	Oui
AL2_x86_64	Oui	Oui	Oui*	Oui*
BOTTLEROCKET_x86_64	Oui**	Oui	S/O	S/O
AL2023_ARM_64_STANDARD	Oui	Oui	Oui	Oui
AL2_ARM_64	Oui	Oui	Oui*	Oui*

BOTTLEROCKET_ARM_64	Oui**	Oui	S/O	S/O

- * Impossible de supprimer le PV sans redémarrer le nœud
- ** Ne fonctionne pas avec NFSv3 avec Trident version 25.02.

> Si l'AMI que vous recherchez ne figure pas dans cette liste, cela ne signifie pas qu'elle n'est pas prise en charge ; cela signifie simplement qu'elle n'a pas été testée. Cette liste sert de guide pour les AMI connues pour fonctionner.

Tests effectués avec :

- Version EKS : 1.32
- Méthode d'installation : Helm 25.06 et en tant que module complémentaire AWS 25.06
- Pour le NAS, les protocoles NFSv3 et NFSv4.1 ont été testés.
- Pour le SAN, seul l'iSCSI a été testé, et non le NVMe-oF.

Tests effectués :

- Créer : Classe de stockage, PVC, capsule
- Supprimer : pod, pvc (standard, qtree/lun – économique, NAS avec sauvegarde AWS)

Trouver plus d'informations

- "Documentation Amazon FSx for NetApp ONTAP"
- "Article de blog sur Amazon FSx for NetApp ONTAP"

Créez un rôle IAM et un secret AWS.

Vous pouvez configurer les pods Kubernetes pour accéder aux ressources AWS en s'authentifiant en tant que rôle AWS IAM au lieu de fournir des informations d'identification AWS explicites.

> Pour vous authentifier à l'aide d'un rôle AWS IAM, vous devez disposer d'un cluster Kubernetes déployé à l'aide d'EKS.

Créer un secret AWS Secrets Manager

Étant donné que Trident utilisera des API sur un serveur virtuel FSx pour gérer le stockage à votre place, il aura besoin d'identifiants pour ce faire. La méthode la plus sûre pour transmettre ces informations d'identification consiste à utiliser un secret AWS Secrets Manager. Par conséquent, si vous n'en possédez pas déjà un, vous devrez créer un secret AWS Secrets Manager contenant les informations d'identification du compte vsadmin.

Cet exemple crée un secret AWS Secrets Manager pour stocker les informations d'identification Trident CSI :

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\":\"vsadmin\",\"password\":\"<svmpassword>\"}"
```

Créer une stratégie IAM

Trident a également besoin des autorisations AWS pour fonctionner correctement. Vous devez donc créer une politique qui accorde à Trident les autorisations nécessaires.

Les exemples suivants créent une stratégie IAM à l'aide de l'interface de ligne de commande AWS :

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

Exemple de JSON de politique :

```
{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx:DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}
```

Créer une identité de pod ou un rôle IAM pour l'association du compte de service (IRSA)

Vous pouvez configurer un compte de service Kubernetes pour assumer un rôle AWS Identity and Access Management (IAM) avec EKS Pod Identity ou un rôle IAM pour l'association de compte de service (IRSA). Tous les pods configurés pour utiliser le compte de service peuvent alors accéder à n'importe quel service AWS auquel le rôle a accès.

Identité du pod

Les associations d'identité de pod Amazon EKS offrent la possibilité de gérer les informations d'identification de vos applications, de la même manière que les profils d'instance Amazon EC2 fournissent des informations d'identification aux instances Amazon EC2.

Installer Pod Identity sur votre cluster EKS :

Vous pouvez créer une identité de pod via la console AWS ou en utilisant la commande AWS CLI suivante :

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

Pour plus d'informations, veuillez consulter"Configurer l'agent d'identité du pod Amazon EKS" .

Créer trust-relationship.json :

Créez un fichier trust-relationship.json pour permettre au principal de service EKS d'assumer ce rôle pour l'identité du pod. Créez ensuite un rôle avec cette politique de confiance :

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

Fichier trust-relationship.json :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Associez la stratégie de rôle au rôle IAM :

Associez la stratégie de rôle de l'étape précédente au rôle IAM qui a été créé :

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \
  --role-name fsxn-csi-role
```

Créer une association d'identité de pod :

Créer une association d'identité de pod entre le rôle IAM et le compte de service Trident (trident-controller).

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

Rôle IAM pour l'association de comptes de service (IRSA)

Utilisation de l'interface de ligne de commande AWS :

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \
  --assume-role-policy-document file://trust-relationship.json
```

Fichier trust-relationship.json :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

Mettez à jour les valeurs suivantes dans le `trust-relationship.json` déposer:

- **<account_id>** - Votre ID de compte AWS
- **<oidc_provider>** - L'OIDC de votre cluster EKS. Vous pouvez obtenir le fournisseur oidc en exécutant :

```
aws eks describe-cluster --name my-cluster --query
"cluster.identity.oidc.issuer"\
  --output text | sed -e "s/^https:\/\///"
```

Associez le rôle IAM à la stratégie IAM :

Une fois le rôle créé, associez la stratégie (créée à l'étape précédente) au rôle à l'aide de cette commande :

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

Vérifiez que le fournisseur OICD est associé :

Vérifiez que votre fournisseur OIDC est associé à votre cluster. Vous pouvez le vérifier à l'aide de cette commande :

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Si le résultat est vide, utilisez la commande suivante pour associer IAM OIDC à votre cluster :

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

Si vous utilisez eksctl, utilisez l'exemple suivant pour créer un rôle IAM pour le compte de service dans EKS :

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Installer Trident

Trident simplifie la gestion du stockage Amazon FSx for NetApp ONTAP dans Kubernetes afin de permettre à vos développeurs et administrateurs de se concentrer sur le déploiement des applications.

Vous pouvez installer Trident en utilisant l'une des méthodes suivantes :

- Barre
- Module complémentaire EKS

Si vous souhaitez utiliser la fonctionnalité de capture d'instantané, installez l'extension CSI snapshot controller. Se référer à"Activer la fonctionnalité de snapshot pour les volumes CSI" pour plus d'informations.

Installez Trident via Helm.

Identité du pod

1. Ajouter le dépôt Trident Helm :

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Installez Trident en utilisant l'exemple suivant :

```
helm install trident-operator netapp-trident/trident-operator --version 100.2502.1 --namespace trident --create-namespace
```

Vous pouvez utiliser le `helm list` commande permettant de consulter les détails d'installation tels que le nom, l'espace de noms, le graphique, l'état, la version de l'application et le numéro de révision.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                           APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2502.0    25.02.0
```

Association de compte de service (IRSA)

1. Ajouter le dépôt Trident Helm :

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Définissez les valeurs de **fournisseur de cloud** et **identité cloud** :

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

Vous pouvez utiliser le `helm list` commande permettant de consulter les détails d'installation tels que le nom, l'espace de noms, le graphique, l'état, la version de l'application et le numéro de révision.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2506.0    25.06.0
```

Si vous prévoyez d'utiliser iSCSI, assurez-vous que iSCSI est activé sur votre machine cliente. Si vous utilisez le système d'exploitation du nœud de travail AL2023, vous pouvez automatiser l'installation du client iSCSI en ajoutant le paramètre node prep dans l'installation helm :

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

Installez Trident via l'extension EKS

Le module complémentaire Trident EKS inclut les derniers correctifs de sécurité et de bogues, et est validé par AWS pour fonctionner avec Amazon EKS. Le module complémentaire EKS vous permet de garantir en permanence la sécurité et la stabilité de vos clusters Amazon EKS et de réduire le travail nécessaire à l'installation, à la configuration et à la mise à jour des modules complémentaires.

Prérequis

Assurez-vous de disposer des éléments suivants avant de configurer le module complémentaire Trident pour AWS EKS :

- Un compte de cluster Amazon EKS avec abonnement complémentaire
- Autorisations AWS pour la place de marché AWS :
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Type d'AMI : Amazon Linux 2 (AL2_x86_64) ou Amazon Linux 2 Arm (AL2_ARM_64)
- Type de nœud : AMD ou ARM
- Un système de fichiers Amazon FSx for NetApp ONTAP

Activez le module complémentaire Trident pour AWS

Console de gestion

1. Ouvrez la console Amazon EKS à https://console.aws.amazon.com/eks/home#/clusters .
2. Dans le volet de navigation de gauche, sélectionnez **Clusters**.
3. Sélectionnez le nom du cluster pour lequel vous souhaitez configurer le module complémentaire NetApp Trident CSI.
4. Sélectionnez **Modules complémentaires** puis **Obtenir plus de modules complémentaires**.
5. Suivez ces étapes pour sélectionner le module complémentaire :
   a. Faites défiler vers le bas jusqu'à la section **modules complémentaires AWS Marketplace** et tapez **"Trident"** dans la zone de recherche.
   b. Cochez la case située dans le coin supérieur droit de la boîte Trident by NetApp .
   c. Sélectionnez **Suivant**.
6. Sur la page des paramètres **Configurer les modules complémentaires sélectionnés**, procédez comme suit :

   Ignorez ces étapes si vous utilisez l'association d'identité de pod.

   a. Sélectionnez la **Version** que vous souhaitez utiliser.
   b. Si vous utilisez l'authentification IRSA, assurez-vous de définir les valeurs de configuration disponibles dans les paramètres de configuration optionnels :
      - Sélectionnez la **Version** que vous souhaitez utiliser.
      - Suivez le **schéma de configuration du module complémentaire** et définissez le paramètre **configurationValues** dans la section **Valeurs de configuration** sur le rôle-ARN que vous avez créé à l'étape précédente (la valeur doit être au format suivant) :

```
{

  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
  "cloudProvider": "AWS"

}
```

   +
   Si vous sélectionnez Remplacer comme méthode de résolution des conflits, un ou plusieurs paramètres du module complémentaire existant peuvent être remplacés par les paramètres du module complémentaire Amazon EKS. Si vous n'activez pas cette option et qu'il y a un conflit avec vos paramètres existants, l'opération échouera. Vous pouvez utiliser le message d'erreur généré pour résoudre le conflit. Avant de sélectionner cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas des paramètres que vous devez gérer vous-même.

7. Choisissez **Suivant**.
8. Sur la page **Vérifier et ajouter**, choisissez **Créer**.

   Une fois l'installation du module complémentaire terminée, vous verrez le module complémentaire installé.

AWS CLI

1. Créez le `add-on.json` déposer:

Pour l'identité du pod, utilisez le format suivant :

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

Pour l'authentification IRSA, utilisez le format suivant :

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

Remplacer `<role ARN>` avec l'ARN du rôle créé à l'étape précédente.

2. Installez le module complémentaire Trident EKS.

```
aws eks create-addon --cli-input-json file://add-on.json
```

eksctl

La commande suivante permet d'installer le module complémentaire Trident EKS :

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

Mettre à jour le module complémentaire Trident EKS

Console de gestion

1. Ouvrez la console Amazon EKS https://console.aws.amazon.com/eks/home#/clusters .
2. Dans le volet de navigation de gauche, sélectionnez **Clusters**.
3. Sélectionnez le nom du cluster pour lequel vous souhaitez mettre à jour le module complémentaire NetApp Trident CSI.
4. Sélectionnez l'onglet **Modules complémentaires**.
5. Sélectionnez * Trident by NetApp* puis sélectionnez **Modifier**.
6. Sur la page **Configurer Trident by NetApp**, procédez comme suit :
   a. Sélectionnez la **Version** que vous souhaitez utiliser.
   b. Développez la section **Paramètres de configuration optionnels** et modifiez-les selon vos besoins.
   c. Sélectionnez **Enregistrer les modifications**.

AWS CLI

L'exemple suivant met à jour le module complémentaire EKS :

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- Vérifiez la version actuelle de votre module complémentaire FSxN Trident CSI. Remplacer `my-cluster` avec le nom de votre cluster.

  ```
  eksctl get addon --name netapp_trident-operator --cluster my-cluster
  ```

  Exemple de résultat :

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.6.0-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- Mettez à jour le module complémentaire avec la version renvoyée sous MISE À JOUR DISPONIBLE dans le résultat de l'étape précédente.

```
eksctl update addon --name netapp_trident-operator --version
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

Si vous retirez le `--force` Si une option et l'un des paramètres du module complémentaire Amazon EKS entrent en conflit avec vos paramètres existants, la mise à jour du module complémentaire Amazon EKS échoue ; vous recevez un message d'erreur pour vous aider à résoudre le conflit. Avant de spécifier cette option, assurez-vous que le module complémentaire Amazon EKS ne gère pas des paramètres que vous devez gérer, car ces paramètres seront écrasés par cette option. Pour plus d'informations sur les autres options de ce paramètre, consultez"Modules complémentaires" . Pour plus d'informations sur la gestion des champs Amazon EKS Kubernetes, consultez"Gestion des champs Kubernetes" .

Désinstallez/supprimez le module complémentaire Trident EKS.

Vous avez deux options pour supprimer un module complémentaire Amazon EKS :

- **Conserver les logiciels complémentaires sur votre cluster** – Cette option supprime la gestion des paramètres par Amazon EKS. Cela supprime également la possibilité pour Amazon EKS de vous informer des mises à jour et de mettre à jour automatiquement le module complémentaire Amazon EKS après que vous ayez lancé une mise à jour. Toutefois, il préserve les logiciels complémentaires sur votre cluster. Cette option transforme l'extension en une installation autogérée, plutôt qu'en une extension Amazon EKS. Avec cette option, l'extension ne nécessite aucune interruption de service. Conservez le `--preserve` option dans la commande pour conserver le module complémentaire.
- **Supprimez complètement le logiciel complémentaire de votre cluster** – NetApp recommande de supprimer le module complémentaire Amazon EKS de votre cluster uniquement si aucune ressource de votre cluster n'en dépend. Retirez le `--preserve` l'option de l' `delete` commande pour supprimer l'extension.

Si le module complémentaire est associé à un compte IAM, ce compte IAM n'est pas supprimé.

Console de gestion

1. Ouvrez la console Amazon EKS à https://console.aws.amazon.com/eks/home#/clusters .
2. Dans le volet de navigation de gauche, sélectionnez **Clusters**.
3. Sélectionnez le nom du cluster pour lequel vous souhaitez supprimer le module complémentaire NetApp Trident CSI.
4. Sélectionnez l'onglet **Modules complémentaires** puis * Trident by NetApp*.
5. Sélectionnez **Supprimer**.
6. Dans la boîte de dialogue **Confirmation de suppression de netapp_trident-operator**, procédez comme suit :
   a. Si vous souhaitez qu'Amazon EKS cesse de gérer les paramètres de l'extension, sélectionnez **Conserver sur le cluster**. Faites ceci si vous souhaitez conserver le logiciel complémentaire sur votre cluster afin de pouvoir gérer vous-même tous les paramètres de ce module.
   b. Saisissez **netapp_trident-operator**.
   c. Sélectionnez **Supprimer**.

AWS CLI

Remplacer `my-cluster` avec le nom de votre cluster, puis exécutez la commande suivante.

```
aws eks delete-addon --cluster-name my-cluster --addon-name
netapp_trident-operator --preserve
```

eksctl

La commande suivante désinstalle le module complémentaire Trident EKS :

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Configurer le système de stockage dorsal

Intégration des pilotes ONTAP SAN et NAS

Pour créer un système de stockage, vous devez créer un fichier de configuration au format JSON ou YAML. Le fichier doit préciser le type de stockage souhaité (NAS ou SAN), le système de fichiers, le SVM à partir duquel le récupérer et la méthode d'authentification. L'exemple suivant montre comment définir un stockage basé sur un NAS et utiliser un secret AWS pour stocker les informations d'identification de la SVM que vous souhaitez utiliser :

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFilesystemID: fs-xxxxxxxxxx
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFilesystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

Exécutez les commandes suivantes pour créer et valider la configuration du backend Trident (TBC) :

- Créez une configuration backend Trident (TBC) à partir d'un fichier yaml et exécutez la commande suivante :

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- Vérifiez que la configuration du backend Trident (TBC) a été créée avec succès :

```
Kubectl get tbc -n trident
```

```
NAME                          BACKEND NAME          BACKEND UUID
PHASE   STATUS

backend-tbc-ontap-nas         tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9   Bound   Success
```

Détails du pilote FSx pour ONTAP

Vous pouvez intégrer Trident à Amazon FSx for NetApp ONTAP à l'aide des pilotes suivants :

- `ontap-san`Chaque PV provisionné est un LUN au sein de son propre volume Amazon FSx for NetApp ONTAP . Recommandé pour le stockage par blocs.
- `ontap-nas`Chaque PV provisionné est un volume Amazon FSx for NetApp ONTAP . Recommandé pour NFS et SMB.
- `ontap-san-economy`Chaque PV provisionné est un LUN avec un nombre configurable de LUN par volume Amazon FSx for NetApp ONTAP .
- `ontap-nas-economy`Chaque PV provisionné est un qtree, avec un nombre configurable de qtrees par volume Amazon FSx for NetApp ONTAP .
- `ontap-nas-flexgroup`Chaque PV provisionné est un volume complet Amazon FSx for NetApp ONTAP FlexGroup .

Pour plus de détails sur le conducteur, veuillez consulter"Pilotes NAS" et"Pilotes SAN" .

Une fois le fichier de configuration créé, exécutez cette commande pour le créer dans votre EKS :

```
kubectl create -f configuration_file
```

Pour vérifier l'état, exécutez la commande suivante :

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

```
NAME                    BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

Configuration avancée et exemples du backend

Consultez le tableau suivant pour connaître les options de configuration du backend :

Paramètre	Description	Exemple
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	`ontap-nas`, `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy`
`backendName`	Nom personnalisé ou système de stockage	Nom du conducteur + "_" + dataLIF
`managementLIF`	Adresse IP d'une interface de gestion de cluster ou SVM (LIF) Un nom de domaine pleinement qualifié (FQDN) peut être spécifié. Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Si vous fournissez le `fsxFilesystemID` sous le `aws` champ, vous n'avez pas besoin de le fournir `managementLIF` car Trident récupère le SVM `managementLIF` Informations provenant d'AWS. Vous devez donc fournir les identifiants d'un utilisateur sous SVM (par exemple : vsadmin), et cet utilisateur doit disposer des `vsadmin` rôle.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Paramètre	Description	Exemple
`dataLIF`	Adresse IP du protocole LIF. * Pilotes NAS ONTAP * : NetApp recommande de spécifier dataLIF. Si aucune donnée n'est fournie, Trident récupère les dataLIF à partir du SVM. Vous pouvez spécifier un nom de domaine pleinement qualifié (FQDN) à utiliser pour les opérations de montage NFS, ce qui vous permet de créer un DNS à répartition circulaire pour équilibrer la charge sur plusieurs dataLIF. Peut être modifié après la configuration initiale. Se référer à . * Pilotes SAN ONTAP * : Ne pas spécifier pour iSCSI. Trident utilise ONTAP Selective LUN Map pour découvrir les LIF iSCI nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si dataLIF est explicitement défini. Peut être configuré pour utiliser des adresses IPv6 si Trident a été installé avec l'option IPv6. Les adresses IPv6 doivent être définies entre crochets, comme [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Activer la création et la mise à jour automatiques de la politique d'exportation [Booléen]. En utilisant le `autoExportPolicy` et `autoExportCIDRs` Avec certaines options, Trident peut gérer automatiquement les politiques d'exportation.	`false`
`autoExportCIDRs`	Liste des CIDR à utiliser pour filtrer les adresses IP des nœuds Kubernetes lorsque `autoExportPolicy` est activé. En utilisant le `autoExportPolicy` et `autoExportCIDRs` Avec certaines options, Trident peut gérer automatiquement les politiques 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 d'autorité de certification de confiance. Facultatif. Utilisé pour l'authentification par certificat.	""
`username`	Nom d'utilisateur pour se connecter au cluster ou à la SVM. Utilisé pour l'authentification basée sur les informations d'identification. Par exemple, vsadmin.	
`password`	Mot de passe pour se connecter au cluster ou à la SVM. Utilisé pour l'authentification basée sur les informations d'identification.	
`svm`	machine virtuelle de stockage à utiliser	Dérivé si un LIF de gestion SVM est spécifié.
`storagePrefix`	Préfixe utilisé lors de la mise en service de nouveaux volumes dans la SVM. Ne peut être modifié après sa création. Pour mettre à jour ce paramètre, vous devrez créer un nouveau backend.	`trident`
`limitAggregateUsage`	**Ne pas spécifier pour Amazon FSx for NetApp ONTAP.** Le fourni `fsxadmin` et `vsadmin` ne contiennent pas les autorisations requises pour récupérer l'utilisation agrégée et la limiter à l'aide de Trident.	Ne pas utiliser.
`limitVolumeSize`	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur. Il limite également la taille maximale des volumes qu'il gère pour les qtrees et les LUN, et le `qtreesPerFlexvol` Cette option permet de personnaliser le nombre maximal d'arbres qtree par FlexVol volume	"" (non appliqué par défaut)
`lunsPerFlexvol`	Le nombre maximal de LUN par volume Flexvol doit être compris entre 50 et 200. SAN uniquement.	"100"

Paramètre	Description	Exemple
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple : {"api":false, "method":true} Ne pas utiliser `debugTraceFlags` sauf si vous effectuez un dépannage et avez besoin d'un journal détaillé.	nul
`nfsMountOptions`	Liste des options de montage NFS séparées par des virgules. Les options de montage des volumes persistants Kubernetes sont normalement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Trident utilisera les options de montage spécifiées dans le fichier de configuration du backend de stockage. Si aucune option de montage n'est spécifiée dans la classe de stockage ou dans le fichier de configuration, Trident ne définira aucune option de montage sur un volume persistant associé.	""
`nasType`	Configurer la création de volumes NFS ou SMB. Les options sont `nfs` , `smb` , ou nul. **Doit être réglé sur `smb` pour les volumes SMB.** La valeur nulle correspond par défaut aux volumes NFS.	`nfs`
`qtreesPerFlexvol`	Le nombre maximal d'arbres Q par FlexVol volume doit être compris entre 50 et 300.	`"200"`
`smbShare`	Vous pouvez spécifier l'un des éléments suivants : le nom d'un partage SMB créé à l'aide de la console de gestion Microsoft ou de l'interface de ligne de commande ONTAP , ou un nom permettant à Trident de créer le partage SMB. Ce paramètre est requis pour les serveurs backend Amazon FSx pour ONTAP .	`smb-share`

Paramètre	Description	Exemple
`useREST`	Paramètre booléen pour utiliser les API REST ONTAP . Lorsqu'il est réglé sur `true` Trident utilisera les API REST ONTAP pour communiquer avec le système dorsal. Cette fonctionnalité nécessite ONTAP 9.11.1 et versions ultérieures. De plus, le rôle de connexion ONTAP utilisé doit avoir accès à `ontap` application. Ceci est satisfait par la définition prédéfinie `vsadmin` et `cluster-admin` rôles.	`false`
`aws`	Vous pouvez spécifier les éléments suivants dans le fichier de configuration d'AWS FSx pour ONTAP: - `fsxFilesystemID` : Spécifiez l'ID du système de fichiers AWS FSx. - `apiRegion` : Nom de la région de l'API AWS. - `apikey` : Clé API AWS. - `secretKey` : Clé secrète AWS.	`""` `""` `""`
`credentials`	Spécifiez les informations d'identification FSx SVM à stocker dans AWS Secrets Manager. - `name` : Nom de ressource Amazon (ARN) du secret, qui contient les informations d'identification de la SVM. - `type` : Définir sur `awsarn` . Se référer à"Créer un secret AWS Secrets Manager" pour plus d'informations.	

Options de configuration backend pour les volumes de provisionnement

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

Paramètre	Description	Défaut
`spaceAllocation`	Allocation d'espace pour les LUN	`true`
`spaceReserve`	Mode de réservation d'espace ; « aucun » (fin) ou « volume » (épais)	`none`
`snapshotPolicy`	Politique d'instantané à utiliser	`none`

Paramètre	Description	Défaut
`qosPolicy`	Groupe de stratégie QoS à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage ou backend. L'utilisation des groupes de politiques QoS avec Trident nécessite ONTAP 9.8 ou une version ultérieure. Vous devez utiliser un groupe de stratégies QoS non partagé et vous assurer que ce groupe de stratégies est appliqué individuellement à chaque composant. Un groupe de politiques QoS partagé impose un plafond au débit total de toutes les charges de travail.	""
`adaptiveQosPolicy`	Groupe de stratégie QoS adaptatif à attribuer aux volumes créés. Choisissez l'une des options qosPolicy ou adaptiveQosPolicy par pool de stockage ou backend. Non pris en charge par ontap-nas-economy.	""
`snapshotReserve`	Pourcentage du volume réservé aux instantanés « 0 »	Si `snapshotPolicy` est `none` , `else` ""
`splitOnClone`	Séparer un clone de son parent lors de sa création	`false`
`encryption`	Activez le chiffrement de volume NetApp (NVE) sur le nouveau volume ; la valeur par défaut est `false` . Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NAE. Pour plus d'informations, veuillez consulter :"Comment Trident fonctionne avec NVE et NAE" .	`false`
`luksEncryption`	Activer le chiffrement LUKS. Se référer à"Utiliser Linux Unified Key Setup (LUKS)" . SAN uniquement.	""
`tieringPolicy`	Politique de hiérarchisation à utiliser `none`	
`unixPermissions`	Mode pour les nouveaux volumes. **Laisser vide pour les volumes SMB.**	""

Paramètre	Description	Défaut
`securityStyle`	Style de sécurité pour les nouveaux volumes. NFS prend en charge `mixed` et `unix` Styles de sécurité. Les PME prennent en charge `mixed` et `ntfs` Styles de sécurité.	La valeur par défaut de NFS est `unix` . La valeur par défaut de SMB est `ntfs` .

Préparez-vous à provisionner des volumes PME

Vous pouvez provisionner des volumes SMB à l'aide de `ontap-nas` conducteur. Avant de terminerIntégration des pilotes ONTAP SAN et NAS Veuillez suivre les étapes suivantes.

Avant de commencer

Avant de pouvoir provisionner des volumes SMB à l'aide de `ontap-nas` Conducteur, vous devez avoir les éléments suivants.

- Un cluster Kubernetes avec un nœud contrôleur Linux et au moins un nœud de travail Windows exécutant Windows Server 2019. Trident prend uniquement en charge les volumes SMB montés sur des pods exécutés sur des nœuds Windows.
- Au moins un secret Trident contenant vos informations d'identification Active Directory. Générer des secrets `smbcreds` :

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

- Un proxy CSI configuré comme un service Windows. Pour configurer un `csi-proxy` , se référer à"GitHub : CSI Proxy" ou"GitHub : CSI Proxy pour Windows" pour les nœuds Kubernetes exécutés sous Windows.

Étapes

1. Créer des partages SMB. Vous pouvez créer les partages d'administration SMB de deux manières : soit en utilisant…"Console de gestion Microsoft" composant logiciel enfichable Dossiers partagés ou via l'interface de ligne de commande ONTAP . Pour créer les partages SMB à l'aide de l'interface de ligne de commande ONTAP :
   a. Si nécessaire, créez la structure de chemin d'accès au répertoire partagé.

      Le `vserver cifs share create` Cette 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 à la SVM spécifiée :

```
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 bien été créé :

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

Se référer à"Créer un partage SMB" pour plus de détails.

2. Lors de la création du backend, vous devez configurer les éléments suivants pour spécifier les volumes SMB. Pour connaître toutes les options de configuration du backend FSx pour ONTAP , veuillez vous référer à"Options et exemples de configuration de FSx pour ONTAP" .

Paramètre	Description	Exemple
`smbShare`	Vous pouvez spécifier l'un des éléments suivants : le nom d'un partage SMB créé à l'aide de la console de gestion Microsoft ou de l'interface de ligne de commande ONTAP , ou un nom permettant à Trident de créer le partage SMB. Ce paramètre est requis pour les serveurs backend Amazon FSx pour ONTAP .	`smb-share`
`nasType`	**Doit être réglé sur `smb` .** Si la valeur est nulle, la valeur par défaut est `nfs` .	`smb`
`securityStyle`	Style de sécurité pour les nouveaux volumes. **Doit être réglé 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.**	""

Configurez une classe de stockage et un PVC.

Configurez un objet StorageClass Kubernetes et créez la classe de stockage pour indiquer à Trident comment provisionner les volumes. Créez une PersistentVolumeClaim (PVC) qui utilise la StorageClass Kubernetes configurée pour demander l'accès au PV. Vous pouvez ensuite monter le panneau photovoltaïque sur un support.

Créer une classe de stockage

Configurer un objet StorageClass Kubernetes

Le "Objet StorageClass Kubernetes" L'objet identifie Trident comme le provisionneur utilisé pour cette classe et indique à Trident comment provisionner un volume. Utilisez cet exemple pour configurer Storageclass pour les volumes utilisant NFS (reportez-vous à la section Attribut Trident ci-dessous pour la liste complète des attributs) :

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

Utilisez cet exemple pour configurer Storageclass pour les volumes utilisant iSCSI :

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  provisioningType: "thin"
  snapshots: "true"
```

Pour provisionner des volumes NFSv3 sur AWS Bottlerocket, ajoutez les éléments requis. `mountOptions` à la classe de stockage :

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock
```

Se référer à"Objets Kubernetes et Trident" pour plus de détails sur la manière dont les classes de stockage interagissent avec le `PersistentVolumeClaim` et des paramètres permettant de contrôler les volumes de provisionnement de Trident .

Créer une classe de stockage

Étapes

1. Il s'agit d'un objet Kubernetes, donc utilisez-le `kubectl` pour le créer dans Kubernetes.

```
kubectl create -f storage-class-ontapnas.yaml
```

2. Vous devriez maintenant voir une classe de stockage **basic-csi** dans Kubernetes et Trident, et Trident devrait avoir détecté les pools sur le backend.

```
kubectl get sc basic-csi
```

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

Créer le PVC

UN "*PersistentVolumeClaim*" (PVC) est une demande d'accès au PersistentVolume sur le cluster.

Le PVC peut être configuré pour demander un stockage d'une certaine taille ou un certain mode d'accès. En utilisant la StorageClass associée, l'administrateur du cluster peut contrôler bien plus que la taille et le mode d'accès du PersistentVolume, comme par exemple les performances ou le niveau de service.

Une fois le PVC créé, vous pouvez monter le volume dans un boîtier.

Exemples de manifestes

Manifestes d'exemple de PersistentVolumeClaim

Ces exemples illustrent les options de configuration de base pour les installations en PVC.

PVC avec accès RWX

Cet exemple montre un PVC de base avec accès RWX associé à une StorageClass nommée `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

Exemple de PVC utilisant iSCSI

Cet exemple montre un PVC de base pour iSCSI avec accès RWO associé à une StorageClass nommée `protection-gold`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 1Gi
storageClassName: protection-gold
```

Créer PVC

Étapes

1. Créez le PVC.

```
kubectl create -f pvc.yaml
```

2. Vérifier l'état du PVC.

```
kubectl get pvc
```

```
NAME          STATUS VOLUME     CAPACITY ACCESS MODES STORAGECLASS AGE
pvc-storage Bound  pv-name 2Gi      RWO                           5m
```

Se référer à"Objets Kubernetes et Trident" pour plus de détails sur la manière dont les classes de stockage interagissent avec le `PersistentVolumeClaim` et des paramètres permettant de contrôler les volumes de provisionnement de Trident .

attributs du Trident

Ces paramètres déterminent quels pools de stockage gérés par Trident doivent être utilisés pour provisionner des volumes d'un type donné.

Attribut	Type	Valeurs	Offre	Demande	Soutenu par
médias[1]	chaîne	disque dur, hybride, SSD	La piscine contient des médias de ce type ; hybride signifie à la fois	Type de média spécifié	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
type de provisionnement	chaîne	mince, épais	Pool prend en charge cette méthode d'approvisionnement	Méthode de provisionnement spécifiée	Épais : tous les produits Ontap ; mince : tous les produits Ontap et Solidfire-San
Type de backend	chaîne	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool appartient à ce type de backend	Backend spécifié	Tous les conducteurs
instantanés	booléen	vrai, faux	Pool prend en charge les volumes avec instantanés	Volume avec instantanés activés	ontap-nas, ontap-san, solidfire-san, gcp-cvs
clones	booléen	vrai, faux	Pool prend en charge les volumes de clonage	Volume avec clones activés	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Attribut	Type	Valeurs	Offre	Demande	Soutenu par
cryptage	booléen	vrai, faux	Pool prend en charge les volumes chiffrés	Volume avec chiffrement activé	ontap-nas, ontap-nas-économie, ontap-nas-groupes flexibles, ontap-san
Op E/S par sec	int	entier positif	Pool est capable de garantir des IOPS dans cette plage.	Volume garanti pour ces IOPS	solidefire-san

[1] : Non pris en charge par les systèmes ONTAP Select

Déployer l'application exemple

Une fois le compartiment de stockage et le PVC créés, vous pouvez monter le PV sur un module. Cette section présente un exemple de commande et de configuration pour associer le PV à un pod.

Étapes

1. Montez le volume dans un boîtier.

```
kubectl create -f pv-pod.yaml
```

Ces exemples montrent des configurations de base pour fixer le PVC à un module : **Configuration de base** :

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

Vous pouvez suivre les progrès en utilisant `kubectl get pod --watch`.

2. Vérifiez que le volume est monté sur `/my/mount/path`.

```
kubectl exec -it pv-pod -- df -h /my/mount/path
```

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

Vous pouvez maintenant supprimer le Pod. L'application Pod n'existera plus, mais le volume restera.

```
kubectl delete pod pv-pod
```

Configurez le module complémentaire Trident EKS sur un cluster EKS.

NetApp Trident simplifie la gestion du stockage Amazon FSx for NetApp ONTAP dans Kubernetes afin de permettre à vos développeurs et administrateurs de se concentrer sur le déploiement des applications. Le module complémentaire NetApp Trident EKS inclut les derniers correctifs de sécurité et de bogues, et est validé par AWS pour fonctionner avec Amazon EKS. Le module complémentaire EKS vous permet de garantir en

permanence la sécurité et la stabilité de vos clusters Amazon EKS et de réduire le travail nécessaire à l'installation, à la configuration et à la mise à jour des modules complémentaires.

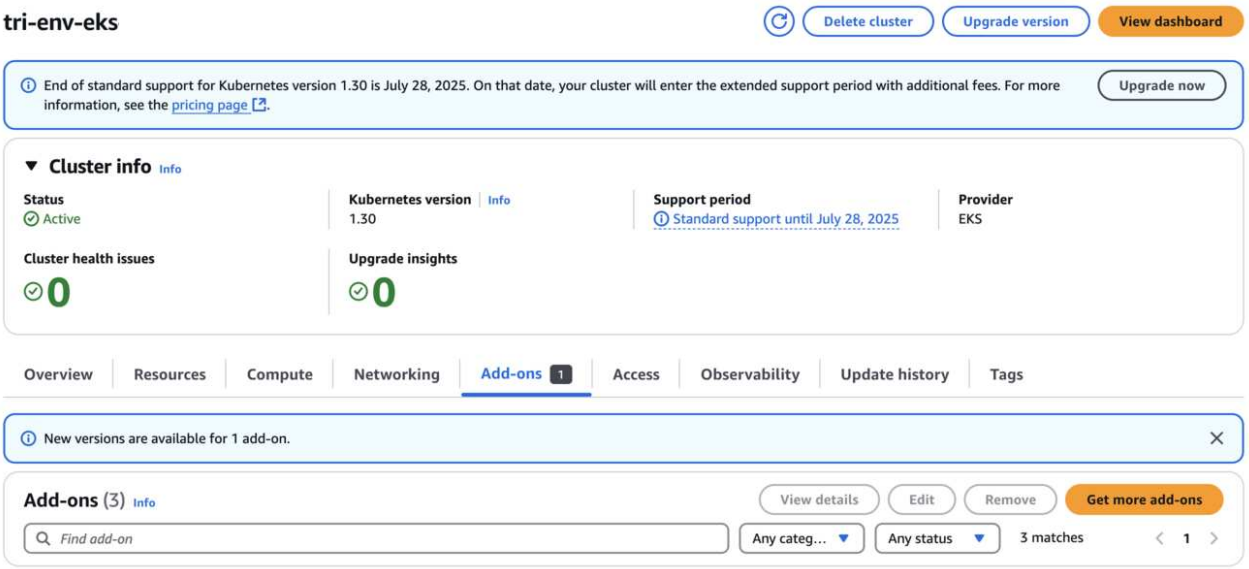
Prérequis

Assurez-vous de disposer des éléments suivants avant de configurer le module complémentaire Trident pour AWS EKS :

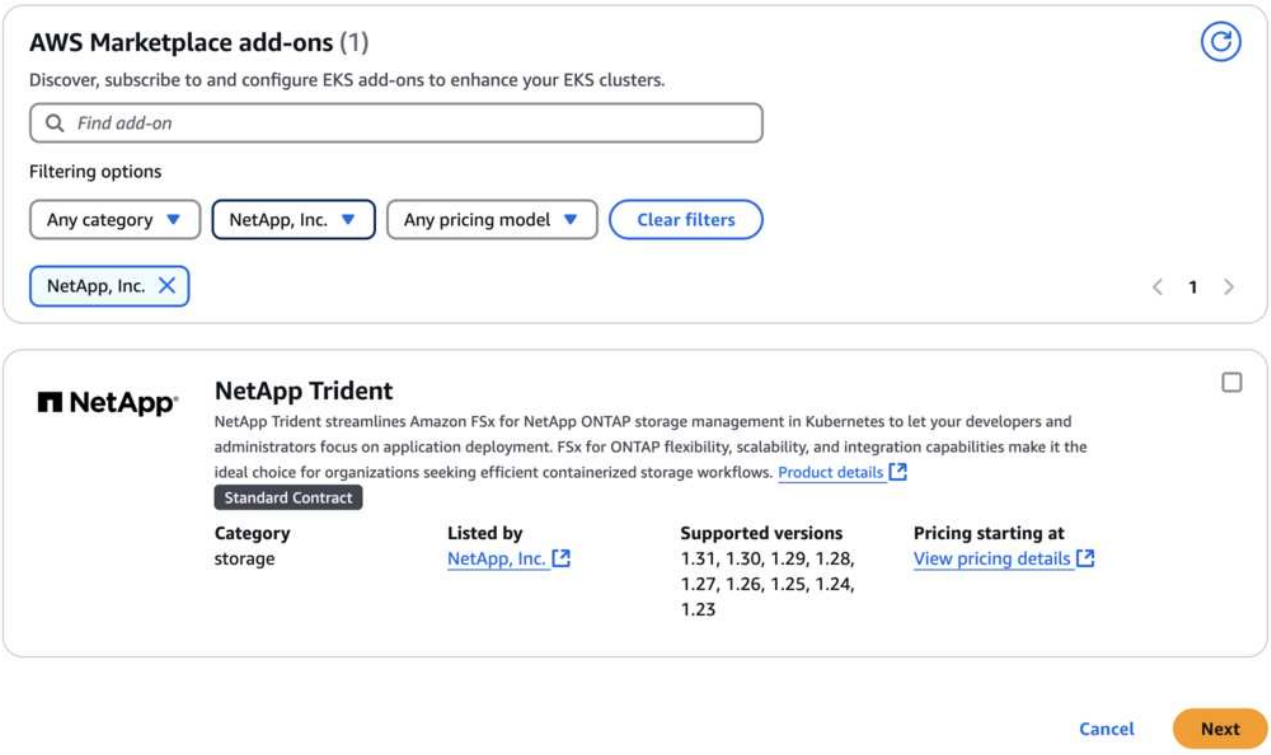
- Un compte de cluster Amazon EKS disposant des autorisations nécessaires pour utiliser des modules complémentaires. Se référer à"Modules complémentaires Amazon EKS" .
- Autorisations AWS pour la place de marché AWS :
  ```
  "aws-marketplace:ViewSubscriptions",
  "aws-marketplace:Subscribe",
  "aws-marketplace:Unsubscribe
  ```
- Type d'AMI : Amazon Linux 2 (AL2_x86_64) ou Amazon Linux 2 Arm (AL2_ARM_64)
- Type de nœud : AMD ou ARM
- Un système de fichiers Amazon FSx for NetApp ONTAP

Étapes

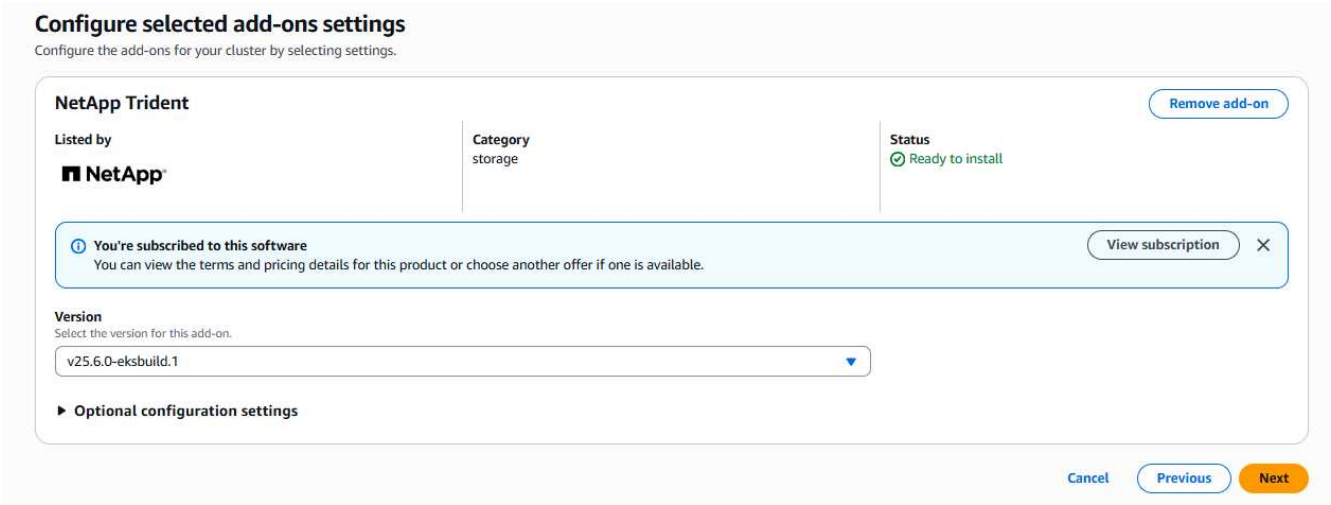
1. Veillez à créer un rôle IAM et un secret AWS pour permettre aux pods EKS d'accéder aux ressources AWS. Pour les instructions, voir"Créez un rôle IAM et un secret AWS." .
2. Sur votre cluster Kubernetes EKS, accédez à l'onglet **Modules complémentaires**.



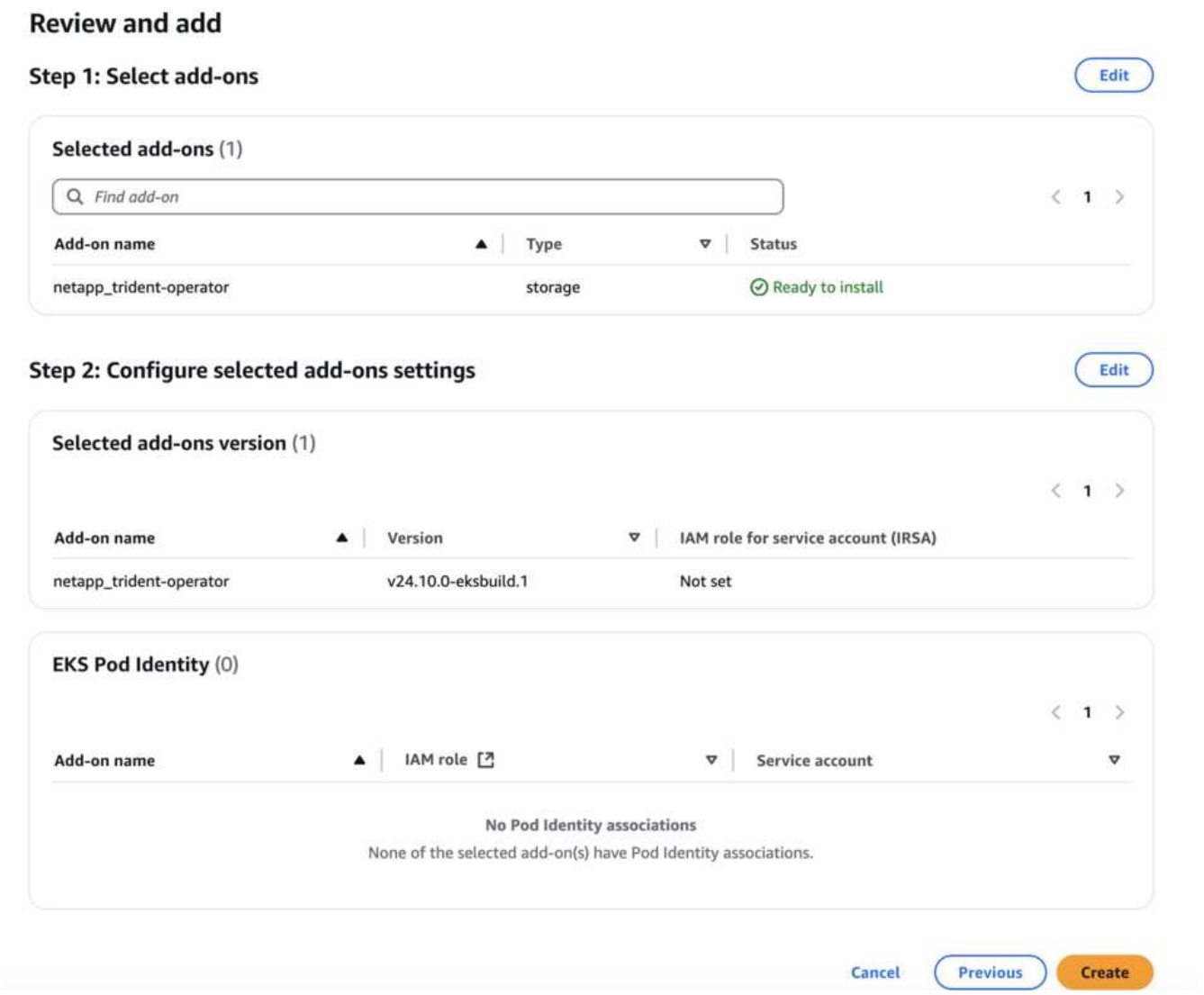
3. Accédez à **AWS Marketplace add-ons** et choisissez la catégorie *stockage*.



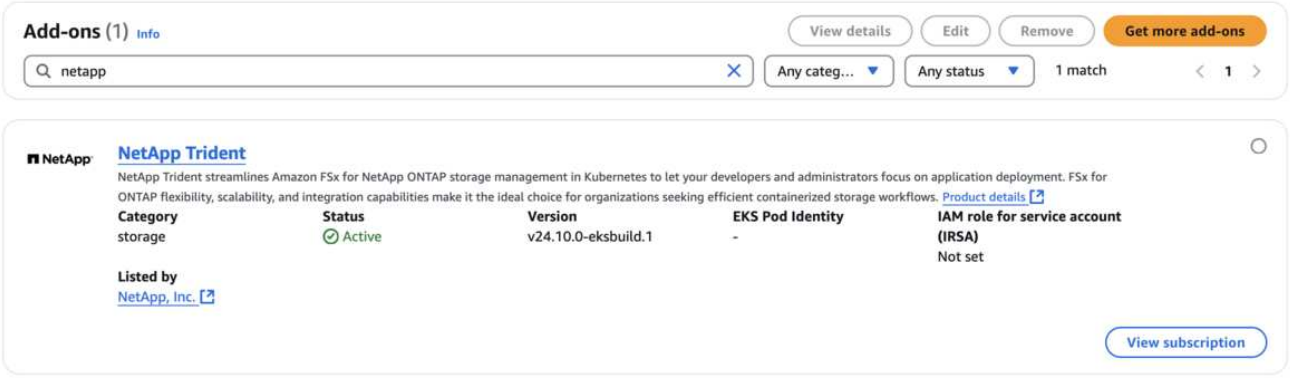
4. Localisez * NetApp Trident* et cochez la case correspondant au module complémentaire Trident , puis cliquez sur **Suivant**.
5. Choisissez la version souhaitée du module complémentaire.



6. Configurez les paramètres du module complémentaire requis.



7. Si vous utilisez IRSA (rôles IAM pour compte de service), reportez-vous aux étapes de configuration supplémentaires."ici" .
8. Sélectionnez **Créer**.
9. Vérifiez que le statut du module complémentaire est *Actif*.



10. Exécutez la commande suivante pour vérifier que Trident est correctement installé sur le cluster :

```
kubectl get pods -n trident
```

11. Poursuivez l'installation et configurez le système de stockage. Pour plus d'informations, voir"Configurer le système de stockage dorsal" .

Installez/désinstallez l'extension Trident EKS via l'interface de ligne de commande (CLI).

Installez le module complémentaire NetApp Trident EKS à l'aide de l'interface de ligne de commande :

La commande suivante permet d'installer le module complémentaire Trident EKS : `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (avec une version dédiée)

Désinstallez le module complémentaire NetApp Trident EKS à l'aide de l'interface de ligne de commande :

La commande suivante désinstalle le module complémentaire Trident EKS :

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Créer des backends avec kubectl

Un backend définit la relation entre Trident et un système de stockage. Il indique à Trident comment communiquer avec ce système de stockage et comment Trident doit provisionner des volumes à partir de celui-ci. Une fois Trident installé, l'étape suivante consiste à créer un backend. Le `TridentBackendConfig` La définition de ressource personnalisée (CRD) vous permet de créer et de gérer des backends 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 une CRD frontale et organisée en espaces de noms qui vous permet de gérer les backends Trident à l'aide de `kubectl` . Les administrateurs Kubernetes et de stockage peuvent désormais créer et gérer des backends directement via l'interface de ligne de commande Kubernetes, sans avoir besoin d'un utilitaire de ligne de commande dédié.(`tridentctl`).

Lors de la création d'un `TridentBackendConfig` objet, le scénario suivant se produit :

- Trident crée automatiquement un backend en fonction de la configuration que vous fournissez. Ceci est représenté en interne comme un `TridentBackend` (`tbe` , `tridentbackend`) CR.
- Le `TridentBackendConfig` est lié de manière unique à un `TridentBackend` qui a été créé par Trident.

Chaque `TridentBackendConfig` maintient une correspondance un-à-un avec un `TridentBackend` La première est l'interface fournie à l'utilisateur pour concevoir et configurer les backends ; la seconde est la manière dont Trident représente l'objet backend proprement dit.

`TridentBackend`Les CR sont créées automatiquement par Trident. Vous ne devriez **pas** les modifier. Si vous souhaitez mettre à jour les backends, procédez en modifiant le `TridentBackendConfig` objet.

Voir l'exemple suivant pour 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 dans le "installateur de trident" Répertoire contenant des exemples de configurations pour la plateforme/le service de stockage souhaité.

Le `spec` prend des paramètres de configuration spécifiques au backend. Dans cet exemple, le backend utilise le `ontap-san` Le pilote de stockage utilise les paramètres de configuration qui sont présentés dans le tableau ci-dessous. Pour obtenir la liste des options de configuration de votre pilote de stockage, reportez-vous à la documentation."Informations de configuration du backend pour votre pilote de stockage" .

Le `spec` Cette section comprend également `credentials` et `deletionPolicy` domaines, qui sont nouvellement 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. Il s'agit d'un secret Kubernetes créé par l'utilisateur. Les identifiants ne peuvent pas être transmis en clair et entraîneront une erreur.
- `deletionPolicy`Ce champ définit ce qui doit se produire lorsque `TridentBackendConfig` est supprimé. Elle peut prendre l'une des deux valeurs suivantes :
  - `delete`Cela entraîne la suppression des deux `TridentBackendConfig` CR et le système dorsal associé. Il s'agit de la valeur par défaut.
  - `retain`: Quand un `TridentBackendConfig` La CR est supprimée, mais la définition du backend reste présente et peut être gérée avec `tridentctl` . Définir la politique de suppression sur `retain` permet aux utilisateurs de revenir à une version antérieure (avant la 21.04) tout en conservant les backends 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` . Si aucun nom n'est spécifié, le nom du backend est défini sur le nom du `TridentBackendConfig` objet (metadata.name). Il est recommandé de définir explicitement les noms des backends en utilisant `spec.backendName` .

Des backends créés avec `tridentctl` n'ont pas d'association `TridentBackendConfig` objet. Vous pouvez choisir de gérer ces backends avec `kubectl` en créant un `TridentBackendConfig` CR. Il convient de veiller à spécifier des paramètres de configuration identiques (tels que `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , et ainsi de suite). Trident liera automatiquement le compte nouvellement créé `TridentBackendConfig` avec le système dorsal préexistant.

Aperçu des étapes

Pour créer un nouveau backend en utilisant `kubectl` , vous devriez faire ce qui suit :

1. Créer un "Secret de Kubernetes" Ce secret contient les informations d'identification dont Trident a besoin pour communiquer avec le cluster/service de stockage.
2. Créer un `TridentBackendConfig` objet. Ce fichier contient des informations spécifiques sur le cluster/service de stockage et fait référence au secret créé à l'étape précédente.

Une fois le backend créé, vous pouvez observer son état en utilisant `kubectl get tbc <tbc-name> -n <trident-namespace>` et recueillir des informations supplémentaires.

Étape 1 : Créer un secret Kubernetes

Créez un secret contenant les identifiants d'accès au serveur. Cela est propre 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 qui doivent figurer dans le secret pour chaque plateforme de stockage :

Description des champs secrets de la plateforme de stockage	Secrète	Description des champs
Azure NetApp Files	clientID	L'identifiant client issu de l'enregistrement d'une application
Cloud Volumes Service pour GCP	clé_privée_id	Identifiant de la clé privée. Partie de la clé API pour un compte de service GCP avec rôle d'administrateur CVS
Cloud Volumes Service pour GCP	clé privée	Clé privée. Partie de la clé API pour un compte de service GCP avec rôle d'administrateur CVS
Élément (NetApp HCI/ SolidFire)	Point final	MVIP pour le cluster SolidFire avec identifiants de locataire
ONTAP	nom d'utilisateur	Nom d'utilisateur pour se connecter au cluster/SVM. Utilisé pour l'authentification par identifiants
ONTAP	mot de passe	Mot de passe pour se connecter au cluster/SVM. Utilisé pour l'authentification par identifiants
ONTAP	clé privée du client	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat
ONTAP	Nom d'utilisateur du chapitre	Nom d'utilisateur entrant. Requis si useCHAP=true. Pour `ontap-san` et `ontap-san-economy`
ONTAP	chapitreInitiateurSecret	Secret de l'initiateur CHAP. Requis si useCHAP=true. Pour `ontap-san` et `ontap-san-economy`
ONTAP	nom d'utilisateur cible du chapitre	Nom d'utilisateur cible. Requis si useCHAP=true. Pour `ontap-san` et `ontap-san-economy`
ONTAP	chapCibleInitiateurSecret	Secret de l'initiateur de la cible CHAP. Requis si useCHAP=true. Pour `ontap-san` et `ontap-san-economy`

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

Étape 2 : Créer le `TridentBackendConfig` CR

Vous êtes maintenant prêt à créer votre `TridentBackendConfig` CR. Dans cet exemple, un backend qui utilise le `ontap-san` Le pilote est créé en utilisant le `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 de `TridentBackendConfig` CR

Maintenant que vous avez créé le `TridentBackendConfig` CR, vous pouvez vérifier le statut. 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 backend a été créé et lié avec succès au `TridentBackendConfig` CR.

La phase peut prendre l'une des valeurs suivantes :

- `Bound`: Le `TridentBackendConfig` CR est associé à un backend, et ce backend contient `configRef` réglé sur le `TridentBackendConfig` L'UID de CR.
- `Unbound`: Représenté à l'aide de `""` . Le `TridentBackendConfig` L'objet n'est pas lié à un serveur dorsal. Tous les nouveaux créés `TridentBackendConfig` Les CR se trouvent par défaut dans cette phase. Après les changements de phase, il ne peut plus revenir à l'état non lié.
- `Deleting`: Le `TridentBackendConfig` CR `deletionPolicy` était configuré pour être supprimé. Quand le `TridentBackendConfig` Lorsque le CR est supprimé, il passe à l'état « Suppression en cours ».

    - S'il n'existe aucune revendication de volume persistante (PVC) sur le système dorsal, la suppression de `TridentBackendConfig` Cela entraînera la suppression du backend par Trident ainsi que du `TridentBackendConfig` CR.
    - Si un ou plusieurs PVC sont présents sur le serveur, celui-ci passe en état de suppression. Le `TridentBackendConfig` CR entre ensuite également en phase de suppression. Le backend et `TridentBackendConfig` ne sont supprimées qu'une fois toutes les PVC supprimées.
- `Lost`: Le backend associé à `TridentBackendConfig` CR a été supprimé accidentellement ou délibérément et le `TridentBackendConfig` CR conserve une référence au backend supprimé. Le `TridentBackendConfig` CR peut toujours être supprimé indépendamment du `deletionPolicy` valeur.
- `Unknown`Trident est incapable de déterminer l'état ou l'existence du serveur dorsal associé à `TridentBackendConfig` CR. Par exemple, si le serveur API ne répond pas ou si le `tridentbackends.trident.netapp.io` Le CRD est manquant. Cela pourrait nécessiter une intervention.

À ce stade, le backend est créé avec succès ! Plusieurs opérations supplémentaires peuvent être prises en charge, telles que :"mises à jour et suppressions du backend" .

(Facultatif) Étape 4 : Obtenir plus de détails

Vous pouvez exécuter la commande suivante pour obtenir plus d'informations sur votre serveur :

```
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
```

De plus, vous pouvez également obtenir un dump 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 backend qui a été créé en réponse à `TridentBackendConfig` CR. Le `lastOperationStatus` Le champ représente l'état de la dernière opération de la `TridentBackendConfig` CR, qui peut être déclenché par l'utilisateur (par exemple, lorsque l'utilisateur a modifié quelque chose dans `spec`) ou déclenché par Trident (par exemple, lors des redémarrages de Trident). Cela peut être soit un succès, soit un échec. `phase` représente l'état de la relation entre les `TridentBackendConfig` CR et le backend. Dans l'exemple ci-dessus, `phase` a la valeur Bound, ce qui signifie que le `TridentBackendConfig` CR est associé au backend.

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

> Vous ne pouvez pas mettre à jour ni supprimer un backend contenant un élément associé `TridentBackendConfig` objet utilisant `tridentctl` . Pour comprendre les étapes impliquées dans le passage d'un mode de vie à un autre `tridentctl` et `TridentBackendConfig` ,"voir ici" .

Gérer les backends

Effectuez la gestion du backend avec kubectl

Découvrez comment effectuer des opérations de gestion du backend en utilisant `kubectl` .

Supprimer un backend

En supprimant un `TridentBackendConfig` , vous demandez à Trident de supprimer/conserver les backends (en fonction de `deletionPolicy`). Pour supprimer un backend, assurez-vous que `deletionPolicy` est configuré pour être supprimé. Pour supprimer uniquement le `TridentBackendConfig` , assurez-vous que `deletionPolicy` est prévu pour conserver. Cela garantit que le système dorsal est toujours présent et peut être géré à l'aide de `tridentctl` .

Exécutez la commande suivante :

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

Trident ne supprime pas les secrets Kubernetes qui étaient utilisés par `TridentBackendConfig` . L'utilisateur Kubernetes est responsable du nettoyage des secrets. Il convient d'être prudent lors de la suppression de secrets. Vous ne devez supprimer les secrets que s'ils ne sont pas utilisés par les systèmes backend.

Afficher les backends existants

Exécutez la commande suivante :

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

Vous pouvez également courir `tridentctl get backend -n trident` ou `tridentctl get backend -o yaml -n trident` pour obtenir la liste de tous les serveurs backend existants. Cette liste inclura également les backends créés avec `tridentctl` .

Mettre à jour un backend

Il peut exister plusieurs raisons de mettre à jour un backend :

- Les identifiants d'accès au système de stockage ont changé. Pour mettre à jour les informations d'identification, le secret Kubernetes utilisé dans le `TridentBackendConfig` L'objet doit être mis à jour. Trident mettra automatiquement à jour le système dorsal avec les dernières informations d'identification fournies. Exécutez la commande suivante pour mettre à jour le secret Kubernetes :

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

- Les paramètres (tels que le nom de la SVM ONTAP utilisée) doivent être mis à jour.
  - Vous pouvez mettre à jour `TridentBackendConfig` objets directement via Kubernetes à l'aide de la commande suivante :

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

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

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

- En cas d'échec d'une mise à jour du système dorsal, celui-ci reste dans sa dernière configuration connue. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante : `kubectl get tbc <tbc-name> -o yaml -n trident` ou `kubectl describe tbc <tbc-name> -n trident` .
- Une fois le problème du fichier de configuration identifié et corrigé, vous pouvez relancer la commande de mise à jour.

Effectuez la gestion du backend avec tridentctl

Découvrez comment effectuer des opérations de gestion du backend en utilisant `tridentctl` .

Créer un backend

Après avoir créé un"fichier de configuration du backend" , exécutez la commande suivante :

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

Si la création du backend échoue, c'est qu'il y a un problème avec sa configuration. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Après avoir identifié et corrigé le problème du fichier de configuration, vous pouvez simplement exécuter la commande suivante : `create` commandez à nouveau.

Supprimer un backend

Pour supprimer un backend de Trident, procédez comme suit :

1. Récupérer le nom du serveur :

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

2. Supprimer le backend :

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

Si Trident a provisionné des volumes et des instantanés à partir de ce backend qui existent encore, la suppression du backend empêche le provisionnement de nouveaux volumes par celui-ci. Le système dorsal restera dans un état « Suppression ».

Afficher les backends existants

Pour consulter les serveurs backend connus de Trident , procédez comme suit :

- Pour obtenir un résumé, 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 backend

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

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

Si la mise à jour du serveur échoue, cela signifie qu'il y a un problème avec la configuration du serveur ou que vous avez tenté une mise à jour invalide. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Après avoir identifié et corrigé le problème du fichier de configuration, vous pouvez simplement exécuter la commande suivante : `update` commandez à nouveau.

Identifiez les classes de stockage qui utilisent un backend

Voici un exemple du type de questions auxquelles vous pouvez répondre avec le JSON. `tridentctl` Sorties pour les objets backend. Cela 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 backends créés à l'aide de `TridentBackendConfig` .

Passer d'une option de gestion du backend à une autre

Découvrez les différentes manières de gérer les backends dans Trident.

Options pour la gestion des backends

Avec l'introduction de `TridentBackendConfig` Les administrateurs disposent désormais de deux méthodes uniques pour gérer les systèmes d'arrière-plan. Cela soulève les questions suivantes :

- Les backends peuvent-ils être créés à l'aide de `tridentctl` être géré avec `TridentBackendConfig` ?
- Les backends peuvent-ils être créés à l'aide de `TridentBackendConfig` être géré à l'aide de `tridentctl` ?

Gérer `tridentctl` backends utilisant `TridentBackendConfig`

Cette section décrit les étapes nécessaires à la gestion des backends créés à l'aide de `tridentctl` directement via l'interface Kubernetes en créant `TridentBackendConfig` objets.

Cela s'appliquera aux scénarios suivants :

- Les systèmes backend préexistants, qui n'ont pas de `TridentBackendConfig` parce qu'ils ont été créés avec `tridentctl` .
- De nouveaux backends créés avec `tridentctl` , tandis que d'autres `TridentBackendConfig` Les objets existent.

Dans les deux cas, les serveurs d'arrière-plan resteront en place, Trident assurant la planification et le traitement des volumes. Les administrateurs ont ici l'un des deux choix suivants :

- Continuer à utiliser `tridentctl` pour gérer les backends créés à l'aide de celui-ci.
- Liaison des backends créés à l'aide de `tridentctl` à un nouveau `TridentBackendConfig` objet. Cela impliquerait que les backends seraient gérés à l'aide de `kubectl` et non `tridentctl` .

Pour gérer un backend préexistant en utilisant `kubectl` , vous devrez créer un `TridentBackendConfig` qui se lie au système dorsal existant. Voici un aperçu de son fonctionnement :

1. Créer un secret Kubernetes. Ce secret contient les identifiants dont Trident a besoin pour communiquer avec le cluster/service de stockage.
2. Créer un `TridentBackendConfig` objet. Ce fichier contient des informations spécifiques sur le cluster/service de stockage et fait référence au secret créé à l'étape précédente. Il convient de veiller à spécifier des paramètres de configuration identiques (tels que `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , et ainsi de suite). `spec.backendName` doit être défini sur le nom du backend existant.

Étape 0 : Identifier le backend

Pour créer un `TridentBackendConfig` Pour que cette fonctionnalité se lie à un système dorsal existant, vous devrez obtenir la configuration de ce système dorsal. Dans cet exemple, supposons qu'un backend ait 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 : Créer un secret Kubernetes

Créez un secret contenant les identifiants du serveur, 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 liera automatiquement à la CR préexistante `ontap-nas-backend` (comme dans cet exemple). Assurez-vous que les exigences suivantes sont respectées :

- Le même nom de backend est défini dans `spec.backendName` .
- Les paramètres de configuration sont identiques à ceux du système dorsal d'origine.
- Les pools virtuels (le cas échéant) doivent conserver le même ordre que dans le backend d'origine.
- Les informations d'identification sont fournies via un secret Kubernetes et non en clair.

Dans ce cas, le `TridentBackendConfig` Cela ressemblera à ceci :

```
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 de `TridentBackendConfig` CR

Après le `TridentBackendConfig` a été créée, sa phase doit être `Bound` . Il doit également refléter le même nom de backend et le même UUID que ceux du backend existant.

```
kubectl 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 backend sera désormais entièrement géré à l'aide de `tbc-ontap-nas-backend` `TridentBackendConfig` objet.

Gérer `TridentBackendConfig` backends utilisant `tridentctl`

```
`tridentctl`peut être utilisé pour lister les backends qui ont été créés à
l'aide de `TridentBackendConfig` .  De plus, les administrateurs peuvent
également choisir de gérer entièrement ces backends via `tridentctl` en
supprimant `TridentBackendConfig` et en veillant à `spec.deletionPolicy`
est réglé sur `retain` .
```

Étape 0 : Identifier le backend

Par exemple, supposons que le backend suivant ait 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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Les résultats montrent que `TridentBackendConfig` a été créé avec succès et est lié à un backend [observer l'UUID du backend].

Étape 1 : Confirmer `deletionPolicy` **est réglé sur** `retain`

Examinons la valeur de `deletionPolicy` . Il faut régler cela sur `retain` . Cela garantit que lorsqu'un `TridentBackendConfig` La CR est supprimée, mais la définition du backend reste 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 passez pas à l'étape suivante sauf si `deletionPolicy` est réglé sur `retain` .

Étape 2 : Supprimer le `TridentBackendConfig` CR

La dernière étape consiste à supprimer le `TridentBackendConfig` CR. Après avoir confirmé le `deletionPolicy` est réglé sur `retain` , vous pouvez procéder à 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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Suite à la suppression de `TridentBackendConfig` Trident supprime simplement l'objet sans supprimer réellement le backend lui-même.

Créer et gérer des classes de stockage

Créer une classe de stockage

Configurez un objet StorageClass Kubernetes et créez la classe de stockage pour indiquer à Trident comment provisionner les volumes.

Configurer un objet StorageClass Kubernetes

Le "Objet StorageClass Kubernetes" identifie Trident comme le provisionneur utilisé pour cette classe et indique à Trident comment provisionner un volume. Par exemple:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Se référer à"Objets Kubernetes et Trident" pour plus de détails sur la manière dont les classes de stockage interagissent avec le `PersistentVolumeClaim` et des paramètres permettant de contrôler les volumes de provisionnement de Trident .

Créer une classe de stockage

Une fois l'objet StorageClass créé, vous pouvez créer la classe de stockage. échantillons de classe stockage fournit quelques exemples de base que vous pouvez utiliser ou modifier.

Étapes

1. Il s'agit d'un objet Kubernetes, donc utilisez-le `kubectl` pour le créer dans Kubernetes.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. Vous devriez maintenant voir une classe de stockage **basic-csi** dans Kubernetes et Trident, et Trident devrait avoir détecté les pools sur le backend.

```
kubectl get sc basic-csi
```

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

```
./tridentctl -n trident get storageclass basic-csi -o json
```

```
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}
```


échantillons de classe stockage

Trident fournit "Définitions simples de classes de stockage pour des backends spécifiques" .

Vous pouvez également modifier `sample-input/storage-class-csi.yaml.templ` fichier fourni avec le programme d'installation et remplacer *`BACKEND_TYPE`* avec le nom du pilote de stockage.

```
./tridentctl -n trident get backend
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

Gérer les classes de stockage

Vous pouvez consulter les classes de stockage existantes, définir une classe de stockage par défaut, identifier le système de stockage dorsal et supprimer des classes de stockage.

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 d'une classe de stockage Kubernetes, exécutez la commande suivante :

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

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

```
tridentctl get storageclass
```

- Pour afficher les détails de la classe de stockage synchronisée de 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. Il s'agit de la classe de stockage qui sera utilisée pour provisionner un volume persistant si un utilisateur n'en spécifie pas un dans une revendication de volume persistant (PVC).

- Définissez une classe de stockage par défaut en configurant l'annotation `storageclass.kubernetes.io/is-default-class` à vrai dans la définition de la classe de stockage. Conformément aux spécifications, toute autre valeur ou absence d'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 en utilisant la commande suivante :

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

On trouve également des exemples dans le package d'installation de Trident qui incluent cette annotation.

> Il ne devrait y avoir qu'une seule classe de stockage par défaut dans votre cluster à la fois. Techniquement, Kubernetes ne vous empêche pas d'en avoir plusieurs, mais il se comportera comme s'il n'existait aucune classe de stockage par défaut.

Identifier le backend d'une classe de stockage

Voici un exemple du type de questions auxquelles vous pouvez répondre avec le JSON. `tridentctl` Sorties pour les objets backend Trident . Cela utilise le `jq` utilitaire, que vous devrez peut-être installer au préalable.

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

Supprimer une classe de stockage

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

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

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

Tous les volumes persistants créés via cette classe de stockage resteront intacts et Trident continuera de les gérer.

> Trident impose un blanc `fsType` pour les volumes qu'elle génère. Pour les backends iSCSI, il est recommandé d'appliquer `parameters.fsType` dans la classe de stockage. Vous devez supprimer les StorageClasses existantes et les recréer avec `parameters.fsType` spécifié.

Provisionner et gérer les volumes

Provisionnez un volume

Créez une PersistentVolumeClaim (PVC) qui utilise la StorageClass Kubernetes configurée pour demander l'accès au PV. Vous pouvez ensuite monter le panneau photovoltaïque sur un support.

Aperçu

UN "*PersistentVolumeClaim*" (PVC) est une demande d'accès au PersistentVolume sur le cluster.

Le PVC peut être configuré pour demander un stockage d'une certaine taille ou un certain mode d'accès. En utilisant la StorageClass associée, l'administrateur du cluster peut contrôler bien plus que la taille et le mode d'accès du PersistentVolume, comme par exemple les performances ou le niveau de service.

Une fois le PVC créé, vous pouvez monter le volume dans un boîtier.

Créer le PVC

Étapes

1. Créez le PVC.

```
kubectl create -f pvc.yaml
```

2. Vérifier l'état du PVC.

```
kubectl get pvc
```

```
NAME          STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS AGE
pvc-storage Bound    pv-name    1Gi        RWO                    5m
```

1. Montez le volume dans un boîtier.

```
kubectl create -f pv-pod.yaml
```

Vous pouvez suivre les progrès en utilisant `kubectl get pod --watch`.

2. Vérifiez que le volume est monté sur `/my/mount/path`.

```
kubectl exec -it task-pv-pod -- df -h /my/mount/path
```

3. Vous pouvez maintenant supprimer le Pod. L'application Pod n'existera plus, mais le volume restera.

```
kubectl delete pod pv-pod
```

Exemples de manifestes

Manifestes d'exemple de PersistentVolumeClaim

Ces exemples illustrent les options de configuration de base pour les installations en PVC.

PVC avec accès RWO

Cet exemple montre un PVC de base avec accès RWO associé à une StorageClass nommée `basic-csi` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

PVC avec NVMe/TCP

Cet exemple montre un PVC de base pour NVMe/TCP avec accès RWO associé à une StorageClass nommée `protection-gold` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

Exemples de manifestes Pod

Ces exemples illustrent des configurations de base pour fixer le PVC à un support.

configuration de base

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

Configuration NVMe/TCP de base

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

Se référer à"Objets Kubernetes et Trident" pour plus de détails sur la manière dont les classes de stockage interagissent avec le `PersistentVolumeClaim` et des paramètres permettant de contrôler les volumes de provisionnement de Trident .

Augmenter les volumes

Trident offre aux utilisateurs de Kubernetes la possibilité d'étendre leurs volumes après leur création. Recherchez des informations sur les configurations requises pour étendre les volumes iSCSI, NFS, SMB, NVMe/TCP et FC.

Étendre un volume iSCSI

Vous pouvez étendre un volume persistant iSCSI (PV) en utilisant le provisionneur CSI.

> L'extension de volume iSCSI est prise en charge par `ontap-san` , `ontap-san-economy` , `solidfire-san` pilotes et nécessite Kubernetes 1.16 et versions ultérieures.

Étape 1 : Configurer la StorageClass pour prendre en charge l'extension de volume

Modifiez la définition de StorageClass pour définir le `allowVolumeExpansion` champ à `true` .

```
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 StorageClass existante, modifiez-la pour inclure le `allowVolumeExpansion` paramètre.

Étape 2 : Créez un PVC avec la StorageClass que vous avez créée.

Modifiez la définition du PVC et mettez à jour le `spec.resources.requests.storage` pour refléter la nouvelle taille souhaitée, qui doit être supérieure à la taille d'origine.

```
cat pvc-ontapsan.yaml
```

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

Trident crée un volume persistant (PV) et l'associe à cette revendication de volume persistant (PVC).

```
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 : Définir un module auquel se fixe le PVC

Fixez le PV à un module pour qu'il puisse être redimensionné. Il existe deux scénarios lors du redimensionnement d'un PV iSCSI :

- Si le PV est attaché à un pod, Trident étend le volume sur le backend de stockage, analyse à nouveau le périphérique et redimensionne le système de fichiers.
- Lors de la tentative de redimensionnement d'un PV non attaché, Trident étend le volume sur le système de stockage dorsal. Une fois le PVC lié à un pod, Trident analyse à nouveau le périphérique et redimensionne le système de fichiers. Kubernetes met ensuite à jour la taille du PVC une fois l'opération d'expansion terminée avec succès.

Dans cet exemple, un pod est créé qui 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 : Développer le PV

Pour redimensionner le PV créé de 1 Gio à 2 Gio, modifiez la définition du PVC et mettez à jour le `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'expansion

Vous pouvez vérifier que l'extension a fonctionné correctement en contrôlant la taille du PVC, du PV et le volume du Trident :

```
kubectl 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
kubectl 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Développer un volume FC

Vous pouvez étendre un volume persistant FC (PV) en utilisant le provisionneur CSI.

> L'expansion du volume des FC est prise en charge par le `ontap-san` pilote et nécessite Kubernetes 1.16 et versions ultérieures.

Étape 1 : Configurer la StorageClass pour prendre en charge l'extension de volume

Modifiez la définition de StorageClass pour définir le `allowVolumeExpansion` champ à `true` .

```
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 StorageClass existante, modifiez-la pour inclure le `allowVolumeExpansion` paramètre.

Étape 2 : Créez un PVC avec la StorageClass que vous avez créée.

Modifiez la définition du PVC et mettez à jour le `spec.resources.requests.storage` pour refléter la nouvelle taille souhaitée, qui doit être supérieure à la taille d'origine.

```
cat pvc-ontapsan.yaml
```

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

Trident crée un volume persistant (PV) et l'associe à cette revendication de volume persistant (PVC).

```
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 : Définir un module auquel se fixe le PVC

Fixez le PV à un module pour qu'il puisse être redimensionné. Il existe deux scénarios lors du redimensionnement d'un PV FC :

- Si le PV est attaché à un pod, Trident étend le volume sur le backend de stockage, analyse à nouveau le périphérique et redimensionne le système de fichiers.
- Lors de la tentative de redimensionnement d'un PV non attaché, Trident étend le volume sur le système de stockage dorsal. Une fois le PVC lié à un pod, Trident analyse à nouveau le périphérique et redimensionne le système de fichiers. Kubernetes met ensuite à jour la taille du PVC une fois l'opération d'expansion

terminée avec succès.

Dans cet exemple, un pod est créé qui 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 : Développer le PV

Pour redimensionner le PV créé de 1 Gio à 2 Gio, modifiez la définition du PVC et mettez à jour le `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'expansion

Vous pouvez vérifier que l'extension a fonctionné correctement en contrôlant la taille du PVC, du PV et le volume du Trident :

```
kubectl 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
kubectl 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Étendre un volume NFS

Trident prend en charge l'extension de volume pour les PV NFS provisionnés sur `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `gcp-cvs` , et `azure-netapp-files` backends.

Étape 1 : Configurer la StorageClass pour prendre en charge l'extension de volume

Pour redimensionner un volume persistant NFS, l'administrateur doit d'abord configurer la classe de stockage pour autoriser l'extension du volume en définissant 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 `kubectl edit storageclass` pour permettre la dilatation du volume.

Étape 2 : Créez un PVC avec la StorageClass que vous avez créée.

```
cat pvc-ontapnas.yaml
```

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

Trident doit créer un PV NFS de 20 Mio pour ce PVC :

```
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 : Développer le PV

Pour redimensionner le PV nouvellement créé de 20 Mio à 1 Gio, modifiez le PVC et définissez `spec.resources.requests.storage` jusqu'à 1 Gio :

```
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'expansion

Vous pouvez vérifier que le redimensionnement a fonctionné correctement en contrôlant la taille du PVC, du PV et du volume Trident :

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

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   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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

volumes d’importation

Vous pouvez importer des volumes de stockage existants en tant que PV Kubernetes à l’aide de `tridentctl import` .

Aperçu et considérations

Vous pouvez importer un volume dans Trident pour :

- Conteneurisez une application et réutilisez son ensemble de données existant
- Utiliser un clone d’un ensemble de données pour une application éphémère
- Reconstruire un cluster Kubernetes défaillant
- Migrer les données d’application lors d’une reprise après sinistre

Considérations

Avant d’importer un volume, veuillez prendre en compte les points suivants.

- Trident peut importer uniquement des volumes ONTAP de type RW (lecture-écriture). Les volumes de type DP (protection des données) sont des volumes de destination SnapMirror . Vous devez rompre la relation miroir avant d’importer le volume dans Trident.

- Nous vous suggérons d'importer les volumes sans connexions actives. Pour importer un volume activement utilisé, clonez le volume puis effectuez l'importation.

  Ceci est particulièrement important pour les volumes de blocs, car Kubernetes ne serait pas au courant de la connexion précédente et pourrait facilement associer un volume actif à un pod. Cela peut entraîner une corruption des données.

- Cependant `StorageClass` Ce paramètre doit être spécifié sur un PVC ; Trident ne l'utilise pas lors de l'importation. Les classes de stockage sont utilisées lors de la création de volumes pour sélectionner les pools disponibles en fonction des caractéristiques de stockage. Le volume existant déjà, aucune sélection de pool n'est requise lors de l'importation. Par conséquent, l'importation ne connaîtra pas d'échec même si le volume existe sur un backend ou un pool qui ne correspond pas à la classe de stockage spécifiée dans le PVC.
- Le volume existant est déterminé et fixé dans le PVC. Une fois le volume importé par le pilote de stockage, le PV est créé avec une référence ClaimRef vers le PVC.
  - La politique de réclamation est initialement définie pour `retain` dans le PV. Une fois que Kubernetes a correctement lié le PVC et le PV, la politique de récupération est mise à jour pour correspondre à la politique de récupération de la classe de stockage.
  - Si la politique de récupération de la classe de stockage est `delete` Le volume de stockage sera supprimé lorsque le PV sera supprimé.
- Par défaut, Trident gère le PVC et renomme le FlexVol volume et le LUN en arrière-plan. Vous pouvez réussir le `--no-manage` Indiquez si vous souhaitez importer un volume non géré. Si vous utilisez `--no-manage` Trident n'effectue aucune opération supplémentaire sur le PVC ou le PV pendant le cycle de vie des objets. Le volume de stockage n'est pas supprimé lorsque le PV est supprimé, et d'autres opérations telles que le clonage et le redimensionnement de volume sont également ignorées.

  Cette option est utile si vous souhaitez utiliser Kubernetes pour les charges de travail conteneurisées, mais que vous souhaitez par ailleurs gérer le cycle de vie du volume de stockage en dehors de Kubernetes.

- Une annotation est ajoutée au PVC et au PV qui sert un double objectif : indiquer que le volume a été importé et si le PVC et le PV sont gérés. Cette annotation ne doit pas être modifiée ni supprimée.

Importer un volume

Vous pouvez utiliser `tridentctl import` importer un volume.

Étapes

1. Créez le fichier de revendication de volume persistant (PVC) (par exemple, `pvc.yaml`) qui servira à créer le PVC. Le fichier PVC doit inclure `name` , `namespace` , `accessModes` , et `storageClassName` . Vous pouvez éventuellement spécifier `unixPermissions` dans votre définition du PVC.

   Voici un exemple de spécification minimale :

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

N'incluez pas de paramètres supplémentaires tels que le nom du PV ou la taille du volume. Cela peut entraîner l'échec de la commande d'importation.

2. Utilisez le `tridentctl import` commande permettant de spécifier le nom du backend Trident contenant le volume et le nom qui identifie de manière unique le volume sur le stockage (par exemple : ONTAP FlexVol, Element Volume, chemin du Cloud Volumes Service). Le `-f` Un argument est requis pour spécifier le chemin d'accès au fichier PVC.

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

Exemples

Consultez les exemples d'importation de volumes suivants pour connaître les pilotes pris en charge.

ONTAP NAS et ONTAP NAS FlexGroup

Trident prend en charge l'importation de volumes via `ontap-nas` et `ontap-nas-flexgroup` conducteurs.

- Trident ne prend pas en charge l'importation de volume à l'aide de `ontap-nas-economy` conducteur.
- Le `ontap-nas` et `ontap-nas-flexgroup` Les pilotes n'autorisent pas les noms de volume en double.

Chaque volume créé avec le `ontap-nas` Le pilote est un FlexVol volume sur le cluster ONTAP . Importer des volumes FlexVol avec le `ontap-nas` Le pilote fonctionne de la même manière. Un volume FlexVol existant déjà sur un cluster ONTAP peut être importé en tant que `ontap-nas` PVC. De même, les volumes FlexGroup peuvent être importés en tant que `ontap-nas-flexgroup` PVC.

Exemples de NAS ONTAP

L'exemple suivant illustre l'importation d'un volume géré et d'un volume non géré.

Volume géré

L'exemple suivant importe un volume nommé `managed_volume` sur un backend nommé `ontap_nas` :

```
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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+---------+---------+
```

Volume non géré

Lors de l'utilisation du `--no-manage` argument, Trident ne renomme pas le volume.

L'exemple suivant importe `unmanaged_volume` sur le `ontap_nas` backend :

```
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   |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+---------+---------+
```

ONTAP SAN

Trident prend en charge l'importation de volumes à l'aide du `ontap-san` (iSCSI, NVMe/TCP et FC) et `ontap-san-economy` conducteurs.

Trident peut importer des volumes ONTAP SAN FlexVol contenant un seul LUN. Ceci est cohérent avec le `ontap-san` pilote, qui crée un FlexVol volume pour chaque PVC et un LUN dans le FlexVol volume. Trident importe le FlexVol volume et l'associe à la définition PVC. Trident peut importer `ontap-san-economy` volumes contenant plusieurs LUN.

Exemples ONTAP SAN

L'exemple suivant illustre l'importation d'un volume géré et d'un volume non géré.

Volume géré

Pour les volumes gérés, Trident renomme le FlexVol volume en `pvc-<uuid>` format et le LUN dans le FlexVol volume à `lun0` .

L'exemple suivant importe le `ontap-san-managed` FlexVol volume présent sur le `ontap_san_default` backend :

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
```

Volume non géré

L'exemple suivant importe `unmanaged_example_volume` sur le `ontap_san` backend :

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

Si vous avez des LUN mappés à des igroups qui partagent un IQN avec un IQN de nœud Kubernetes, comme indiqué dans l'exemple suivant, vous recevrez l'erreur suivante : `LUN already mapped to initiator(s) in this group` . Vous devrez supprimer l'initiateur ou déconnecter le LUN pour importer le volume.

```
Vserver   Igroup       Protocol OS Type  Initiators
--------- ------------ -------- -------- ------------------------------------
svm0      k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3
                       iscsi    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
svm0      unmanaged-example-igroup
                       mixed    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
```

Élément

Trident prend en charge le logiciel NetApp Element et l'importation de volumes NetApp HCI à l'aide de `solidfire-san` conducteur.

> Le pilote Element prend en charge les noms de volumes en double. Toutefois, Trident renvoie une erreur en cas de noms de volumes en double. Pour contourner ce problème, clonez le volume, donnez-lui un nom unique, puis importez le volume cloné.

Exemple d'élément

L'exemple suivant importe un `element-managed` volume sur le backend `element_default` .

```
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    |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
```

Plateforme Google Cloud

Trident prend en charge l'importation de volumes via `gcp-cvs` conducteur.

> Pour importer un volume basé sur le service NetApp Cloud Volumes Service dans Google Cloud Platform, identifiez le volume par son chemin d'accès. Le chemin du volume est la portion du chemin d'exportation du volume qui suit le `:/` . Par exemple, si le chemin d'exportation est `10.0.0.1:/adroit-jolly-swift` , le chemin du volume est `adroit-jolly-swift` .

Exemple de Google Cloud Platform

L'exemple suivant importe un `gcp-cvs` volume sur le backend `gcpcvs_YEppr` avec le chemin de volume de `adroit-jolly-swift` .

```
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    |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
```

Azure NetApp Files

Trident prend en charge l'importation de volumes via `azure-netapp-files` conducteur.

> Pour importer un volume Azure NetApp Files , identifiez le volume par son chemin d'accès. Le chemin du volume est la portion du chemin d'exportation du volume qui suit le `:/` . Par exemple, si le chemin de montage est `10.0.0.2:/importvol1` , le chemin du volume est `importvol1` .

Exemple de Azure NetApp Files

L'exemple suivant importe un `azure-netapp-files` volume sur le backend `azurenetappfiles_40517` avec le chemin de volume `importvol1` .

```
tridentctl import volume azurenetappfiles_40517 importvol1 -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    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

Google Cloud NetApp Volumes

Trident prend en charge l'importation de volumes via `google-cloud-netapp-volumes` conducteur.

Exemple de Google Cloud NetApp Volumes

L'exemple suivant importe un `google-cloud-netapp-volumes` volume sur le backend `backend-tbc-gcnv1` avec le volume `testvoleasiaeast1` .

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

L'exemple suivant importe un `google-cloud-netapp-volumes` volume lorsque deux volumes sont présents dans la même région :

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                  NAME                    |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personnaliser les noms et les étiquettes des volumes

Avec Trident, vous pouvez attribuer des noms et des étiquettes significatifs aux volumes que vous créez. Cela vous aide à identifier et à associer facilement les volumes à leurs ressources Kubernetes respectives (PVC). Vous pouvez également définir des modèles au niveau du backend pour créer des noms de volumes et des étiquettes personnalisés ; tous les volumes que vous créez, importez ou clonez respecteront ces modèles.

Avant de commencer

Prise en charge des noms et étiquettes de volume personnalisables :

1. Opérations de création, d'importation et de clonage de volumes.
2. Dans le cas du pilote ontap-nas-economy, seul le nom du volume Qtree est conforme au modèle de nom.
3. Dans le cas du pilote ontap-san-economy, seul le nom du LUN est conforme au modèle de nom.

Limites

1. Les noms de volumes personnalisables sont compatibles uniquement avec les pilotes ONTAP sur site.
2. Les noms de volumes personnalisables ne s'appliquent pas aux volumes existants.

Comportements clés des noms de volumes personnalisables

1. Si une erreur survient en raison d'une syntaxe invalide dans un modèle de nom, la création du backend échoue. Toutefois, si l'application du modèle échoue, le volume sera nommé conformément à la convention d'appellation existante.
2. Le préfixe de stockage n'est pas applicable lorsqu'un volume est nommé à l'aide d'un modèle de nom

provenant de la configuration du système dorsal. Toute valeur de préfixe souhaitée peut être directement ajoutée au modèle.

Exemples de configuration backend avec modèle de nom et étiquettes

Des modèles de noms personnalisés peuvent être définis au niveau racine et/ou au niveau du pool.

Exemple de niveau racine

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "defaults": {
    "nameTemplate":
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.volume.Requ
estName}}"
  },
  "labels": {
    "cluster": "ClusterA",
    "PVC": "{{.volume.Namespace}}_{{.volume.RequestName}}"
  }
}
```

Exemple de niveau de piscine

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels": {
        "labelname": "label1",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels": {
        "cluster": "label2",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

Exemples de modèles de noms

Exemple 1 :

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Exemple 2 :

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Points à considérer

1. Dans le cas des importations de volumes, les étiquettes ne sont mises à jour que si le volume existant possède des étiquettes dans un format spécifique. Par exemple: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. Dans le cas des importations de volumes gérés, le nom du volume suit le modèle de nom défini au niveau racine dans la définition du backend.
3. Trident ne prend pas en charge l'utilisation d'un opérateur de découpage avec le préfixe de stockage.
4. Si les modèles ne produisent pas de noms de volumes uniques, Trident ajoutera quelques caractères aléatoires pour créer des noms de volumes uniques.
5. Si le nom personnalisé d'un volume économique NAS dépasse 64 caractères, Trident nommera les volumes conformément à la convention d'appellation existante. Pour tous les autres pilotes ONTAP , si le nom du volume dépasse la limite de noms, le processus de création du volume échoue.

Partager un volume NFS entre espaces de noms

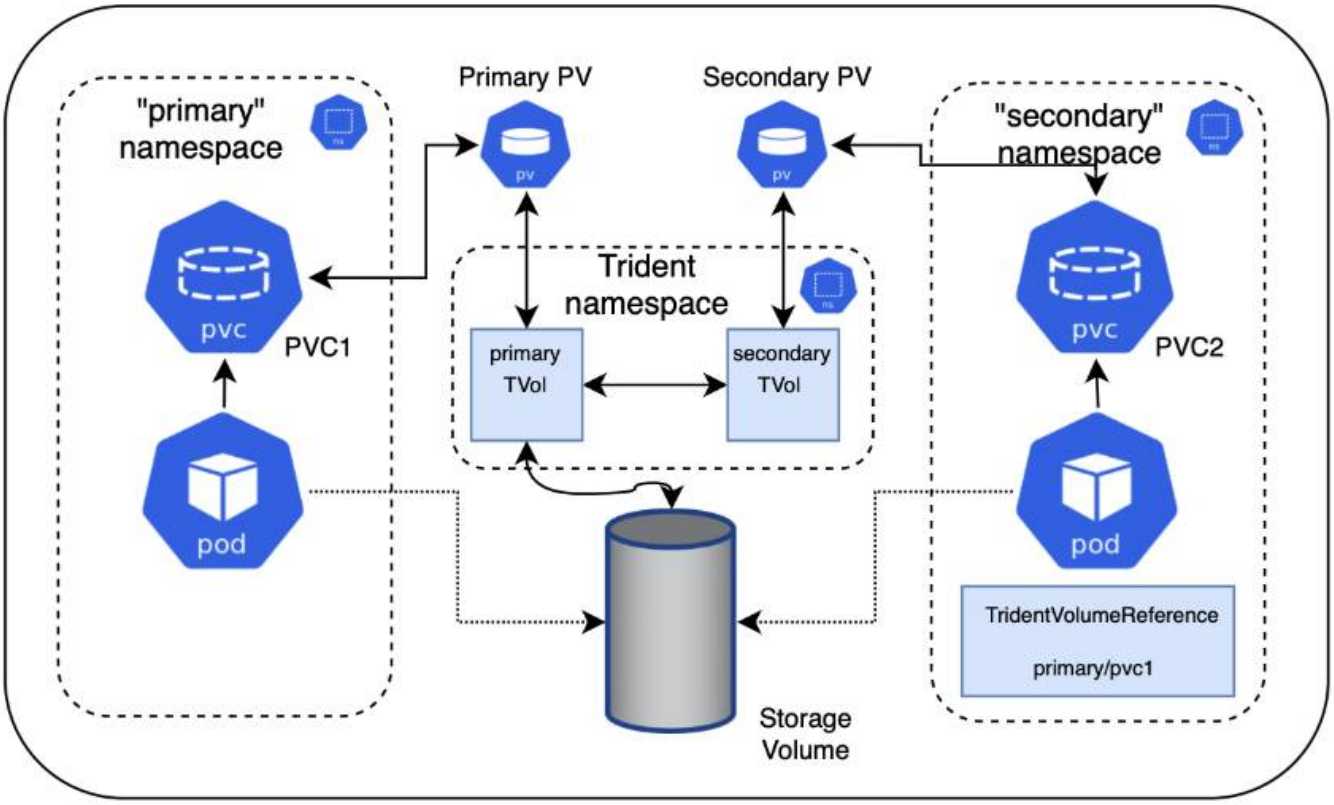
Avec Trident, vous pouvez créer un volume dans un espace de noms principal et le partager dans un ou plusieurs espaces de noms secondaires.

Caractéristiques

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

- Plusieurs niveaux de contrôle d'accès pour garantir la sécurité
- Compatible avec tous les pilotes de volume NFS Trident
- Aucune dépendance à tridentctl ni à aucune autre fonctionnalité non native de Kubernetes

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



Démarrage rapide

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

1 **Configurez le PVC source pour partager le volume.**

Le propriétaire de l'espace de noms source accorde l'autorisation d'accéder aux données du PVC source.

2 **Autoriser la création d'une demande de changement dans l'espace de noms de destination**

L'administrateur du cluster autorise le propriétaire de l'espace de noms de destination à créer la ressource personnalisée TridentVolumeReference.

3 **Créez une référence de volume Trident dans l'espace de noms de destination.**

Le propriétaire de l'espace de noms de destination crée la ressource personnalisée TridentVolumeReference pour faire référence au volume persistant source.

4 **Créez le PVC subordonné dans l'espace de noms de destination**

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

Configurez les espaces de noms source et de destination

Pour garantir la sécurité, le partage entre espaces de noms nécessite une collaboration et une action de la part du propriétaire de l'espace de noms source, de l'administrateur du cluster et du propriétaire de l'espace de noms de destination. Le rôle de l'utilisateur est défini à chaque étape.

Étapes

1. **Propriétaire de l'espace de noms source :** Créer le PVC(`pvc1`) dans l'espace de noms source qui autorise le partage avec l'espace de noms de destination(`namespace2`) en utilisant `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
```

   Trident crée le PV et son volume de stockage NFS backend.

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

2. **Administrateur de cluster :** assurez-vous qu'un RBAC approprié est en place 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 une ressource personnalisée 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`) en utilisant `shareFromPVC` annotation pour désigner le 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
```

Le diamètre du tuyau PVC de destination doit être inférieur ou égal à celui du tuyau PVC source.

Résultats

Trident lit le `shareFromPVC` annotation sur le PVC de destination et crée le PV de destination en tant que volume subordonné sans ressource de stockage propre qui pointe vers le PV source et partage la ressource de stockage du PV source. Les tubes PVC et PV de destination semblent être liés normalement.

Supprimer un volume partagé

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

Utiliser `tridentctl get` interroger les volumes subordonnés

En utilisant le[`tridentctl` utilitaire, vous pouvez exécuter le `get` commande pour obtenir les volumes

subordonnés. Pour plus d'informations, consultez le lien :../trident-reference/tridentctl.html[`tridentctl` commandes et options].

```
Usage:
  tridentctl get [option]
```

Drapeaux:

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

Limites

- Trident ne peut pas empêcher les espaces de noms de destination d'écrire sur le volume partagé. Vous devriez utiliser le verrouillage de fichiers ou d'autres processus pour empêcher l'écrasement des données du volume partagé.
- Vous ne pouvez pas révoquer l'accès à la PVC source en retirant le `shareToNamespace` ou `shareFromNamespace` annotations ou suppression des `TridentVolumeReference` CR. Pour révoquer l'accès, vous devez supprimer le PVC subordonné.
- Les instantanés, les clones et la mise en miroir ne sont pas possibles sur les volumes subordonnés.

Pour plus d'informations

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

- Visite"Partage de volumes entre espaces de noms : découvrez l'accès aux volumes inter-espaces de noms" .
- Regardez la démo sur"NetAppTV" .

Cloner des volumes entre espaces de noms

Avec Trident, vous pouvez créer de nouveaux volumes à partir de volumes existants ou d'instantanés de volumes provenant d'un espace de noms différent au sein du même cluster Kubernetes.

Prérequis

Avant de cloner des volumes, assurez-vous que les systèmes de stockage source et de destination sont du même type et ont la même classe de stockage.

> Le clonage entre espaces de noms est pris en charge uniquement pour le `ontap-san` et `ontap-nas` pilotes de stockage. Les clones en lecture seule ne sont pas pris en charge.

Démarrage rapide

Vous pouvez configurer le clonage de volumes en quelques étapes seulement.

1 Configurez le PVC source pour cloner le volume.

Le propriétaire de l'espace de noms source accorde l'autorisation d'accéder aux données du PVC source.

2 Autoriser la création d'une demande de changement dans l'espace de noms de destination

L'administrateur du cluster autorise le propriétaire de l'espace de noms de destination à créer la ressource personnalisée TridentVolumeReference.

3 Créez une référence de volume Trident dans l'espace de noms de destination.

Le propriétaire de l'espace de noms de destination crée la ressource personnalisée TridentVolumeReference pour faire référence au volume persistant source.

4 Créez le PVC cloné dans l'espace de noms de destination

Le propriétaire de l'espace de noms de destination crée un PVC pour cloner le PVC de l'espace de noms source.

Configurez les espaces de noms source et de destination

Pour garantir la sécurité, le clonage de volumes entre espaces de noms nécessite la collaboration et l'action du propriétaire de l'espace de noms source, de l'administrateur du cluster et du propriétaire de l'espace de noms de destination. Le rôle de l'utilisateur est défini à chaque étape.

Étapes

1. **Propriétaire de l'espace de noms source :** Créer le PVC(`pvc1`) dans l'espace de noms source(`namespace1`) qui autorise le partage avec l'espace de noms de destination(`namespace2`) en utilisant `cloneToNamespace` annotation.

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

Trident crée le PV et son volume de stockage backend.

- Vous pouvez partager le PVC avec plusieurs espaces de noms en utilisant une liste séparée par des virgules. Par exemple, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4` .
- Vous pouvez partager avec tous les espaces de noms en utilisant `*` . Par exemple, `trident.netapp.io/cloneToNamespace: *`
- Vous pouvez mettre à jour le PVC pour inclure le `cloneToNamespace` annotation à tout moment.

2. **Administrateur du cluster :** Assurez-vous que le contrôle d'accès basé sur les rôles (RBAC) est correctement configuré pour autoriser le propriétaire de l'espace de noms de destination à créer la ressource personnalisée TridentVolumeReference dans l'espace de noms de destination.(`namespace2`).
3. **Propriétaire de l'espace de noms de destination :** Créez une ressource personnalisée 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`) en utilisant `cloneFromPVC` ou `cloneFromSnapshot` , et `cloneFromNamespace` annotations pour désigner le PVC source.

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

Limites

- Pour les PVC provisionnés à l'aide des pilotes ontap-nas-economy, les clones en lecture seule ne sont pas pris en charge.

Répliquez des volumes à l'aide de SnapMirror

Trident prend en charge les relations de miroir entre un volume source sur un cluster et le volume de destination sur le cluster homologue pour la réplication des données en vue de la reprise après sinistre. Vous pouvez utiliser une définition de ressource personnalisée (CRD) avec espace de noms, appelée relation miroir Trident (TMR), pour effectuer les opérations suivantes :

- Créer des relations de miroir entre les volumes (PVC)
- Supprimer les relations de miroir entre les volumes
- Briser les relations miroir
- Promouvoir le volume secondaire en cas de sinistre (basculements)
- Effectuer une transition sans perte des applications d'un cluster à l'autre (lors de basculements ou de migrations planifiés).

Prérequis de réplication

Veuillez vous assurer que les conditions préalables suivantes sont remplies avant de commencer :

Clusters ONTAP

- * Trident*: La version 22.10 ou ultérieure de Trident doit être présente sur les clusters Kubernetes source et de destination qui utilisent ONTAP comme backend.
- **Licences** : Les licences asynchrones ONTAP SnapMirror utilisant le module de protection des données doivent être activées sur les clusters ONTAP source et de destination. Se référer à "Présentation des licences SnapMirror dans ONTAP" pour plus d'informations.

  À partir d' ONTAP 9.10.1, toutes les licences sont fournies sous forme de fichier de licence NetApp (NLF), qui est un fichier unique permettant d'activer plusieurs fonctionnalités. Se référer à"Licences incluses avec ONTAP One" pour plus d'informations.

  Seule la protection asynchrone SnapMirror est prise en charge.

Interconnexion

- **Cluster et SVM** : Les backends de stockage ONTAP doivent être appariés. Se référer à "Aperçu du peering de clusters et de SVM" pour plus d'informations.

  Assurez-vous que les noms SVM utilisés dans la relation de réplication entre deux clusters ONTAP sont uniques.

- * Trident et SVM * : Les SVM distants appariés doivent être disponibles pour Trident sur le cluster de destination.

Pilotes pris en charge

NetApp Trident prend en charge la réplication de volumes avec la technologie NetApp SnapMirror utilisant des

classes de stockage prises en charge par les pilotes suivants : **`ontap-nas` : NFS** `ontap-san` : iSCSI **`ontap-san` : FC** `ontap-san` : NVMe/TCP (nécessite au minimum la version ONTAP 9.15.1)

> La réplication de volumes à l'aide de SnapMirror n'est pas prise en charge pour les systèmes ASA r2. Pour plus d'informations sur les systèmes ASA r2, consultez"En savoir plus sur les systèmes de stockage ASA r2" .

Créer un PVC miroir

Suivez ces étapes et utilisez les exemples CRD pour créer une relation miroir entre les volumes primaires et secondaires.

Étapes

1. Effectuez les étapes suivantes sur le cluster Kubernetes principal :
   a. Créez un objet StorageClass avec le `trident.netapp.io/replication: true` paramètre.

      Exemple

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        backendType: "ontap-nas"
        fsType: "nfs"
        trident.netapp.io/replication: "true"
      ```

   b. Créez un PVC avec la StorageClass précédemment créée.

      Exemple

      ```
      kind: PersistentVolumeClaim
      apiVersion: v1
      metadata:
        name: csi-nas
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        storageClassName: csi-nas
      ```

   c. Créez une ressource personnalisée MirrorRelationship avec des informations locales.

Exemple

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Trident récupère les informations internes du volume et l'état actuel de protection des données (DP) du volume, puis remplit le champ d'état de la MirrorRelationship.

d. Obtenez le CR TridentMirrorRelationship pour obtenir le nom interne et le SVM du PVC.

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. Effectuez les étapes suivantes sur le cluster Kubernetes secondaire :

   a. Créez une StorageClass avec le paramètre trident.netapp.io/replication : true.

Exemple

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

b. Créez une ressource personnalisée MirrorRelationship avec les informations de destination et de source.

Exemple

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
  - localPVCName: csi-nas
    remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

Trident créera une relation SnapMirror avec le nom de stratégie de relation configuré (ou par défaut pour ONTAP) et l'initialisera.

c. Créez un PVC avec la StorageClass précédemment créée pour servir de destination secondaire (SnapMirror).

Exemple

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Trident vérifiera l'existence de la définition de ressource personnalisée (CRD) TridentMirrorRelationship et ne parviendra pas à créer le volume si la relation n'existe pas. Si la relation existe, Trident s'assurera que le nouveau FlexVol volume est placé sur une SVM appariée avec la SVM distante définie dans la MirrorRelationship.

États de réplication du volume

Une relation miroir Trident (TMR) est un CRD qui représente une extrémité d'une relation de réplication entre PVC. Le TMR de destination possède un état qui indique à Trident quel est l'état souhaité. Le TMR de destination présente les états suivants :

- **Établi** : le PVC local est le volume de destination d'une relation miroir, et il s'agit d'une nouvelle relation.
- **Promu** : le PVC local est lisible en écriture et montable, sans relation miroir actuellement en vigueur.
- **Rétabli** : le PVC local est le volume de destination d'une relation miroir et faisait également partie auparavant de cette relation miroir.
  - L'état rétabli doit être utilisé si le volume de destination a déjà été en relation avec le volume source, car il écrase le contenu du volume de destination.
  - L'état rétabli échouera si le volume n'était pas auparavant en relation avec la source.

Favoriser la mise en place d'un PVC secondaire lors d'un basculement imprévu

Effectuez l'étape suivante sur le cluster Kubernetes secondaire :

- Mettez à jour le champ *spec.state* de TridentMirrorRelationship en `promoted` .

Favoriser le PVC secondaire lors d'un basculement planifié

Lors d'un basculement (migration) planifié, effectuez les étapes suivantes pour promouvoir le PVC secondaire :

Étapes

1. Sur le cluster Kubernetes principal, créez un instantané du PVC et attendez que l'instantané soit créé.

2. Sur le cluster Kubernetes principal, créez la ressource personnalisée SnapshotInfo pour obtenir des détails internes.

   Exemple

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. Sur le cluster Kubernetes secondaire, mettez à jour le champ *spec.state* de la ressource personnalisée *TridentMirrorRelationship* à *promoted* et *spec.promotedSnapshotHandle* à internalName du snapshot.
4. Sur le cluster Kubernetes secondaire, vérifiez que l'état (champ status.state) de TridentMirrorRelationship est bien promu.

Rétablir une relation miroir après un basculement

Avant de rétablir une relation miroir, choisissez le côté que vous souhaitez définir comme nouveau côté principal.

Étapes

1. Sur le cluster Kubernetes secondaire, assurez-vous que les valeurs du champ *spec.remoteVolumeHandle* de la relation TridentMirrorRelationship sont mises à jour.
2. Sur le cluster Kubernetes secondaire, mettez à jour le champ *spec.mirror* de TridentMirrorRelationship en `reestablished`.

Opérations supplémentaires

Trident prend en charge les opérations suivantes sur les volumes primaires et secondaires :

Répliquez le PVC primaire sur un nouveau PVC secondaire.

Assurez-vous d'avoir déjà un PVC primaire et un PVC secondaire.

Étapes

1. Supprimez les CRD PersistentVolumeClaim et TridentMirrorRelationship du cluster secondaire (destination) établi.
2. Supprimez le CRD TridentMirrorRelationship du cluster principal (source).
3. Créez une nouvelle CRD TridentMirrorRelationship sur le cluster principal (source) pour le nouveau PVC secondaire (destination) que vous souhaitez établir.

Redimensionner un PVC miroir, primaire ou secondaire

Le PVC peut être redimensionné normalement ; ONTAP étendra automatiquement les flevxols de destination si la quantité de données dépasse la taille actuelle.

Supprimer la réplication d'un PVC

Pour supprimer la réplication, effectuez l'une des opérations suivantes sur le volume secondaire actuel :

- Supprimez la relation MirrorRelationship sur le PVC secondaire. Cela rompt la relation de réplication.
- Ou bien, mettez à jour le champ spec.state à *promue*.

Supprimer un PVC (qui était précédemment dupliqué)

Trident vérifie la présence de PVC répliqués et libère la relation de réplication avant de tenter de supprimer le volume.

Supprimer un TMR

La suppression d'un TMR d'un côté d'une relation en miroir entraîne la transition du TMR restant vers l'état *promu* avant que Trident ne termine la suppression. Si le TMR sélectionné pour suppression est déjà à l'état *promu*, il n'existe aucune relation miroir et le TMR sera supprimé et Trident promouvra le PVC local à *Lecture/Écriture*. Cette suppression libère les métadonnées SnapMirror pour le volume local dans ONTAP. Si ce volume est utilisé dans une relation miroir à l'avenir, il doit utiliser un nouveau TMR avec un état de réplication de volume *établi* lors de la création de la nouvelle relation miroir.

Mettez à jour les relations miroir lorsque ONTAP est en ligne.

Les relations miroir peuvent être mises à jour à tout moment après leur établissement. Vous pouvez utiliser le `state: promoted` ou `state: reestablished` champs pour mettre à jour les relations. Lors de la promotion d'un volume de destination en un volume ReadWrite standard, vous pouvez utiliser *promotedSnapshotHandle* pour spécifier un instantané spécifique dans lequel restaurer le volume actuel.

Mettre à jour les relations miroir lorsque ONTAP est hors ligne

Vous pouvez utiliser une CRD pour effectuer une mise à jour SnapMirror sans que Trident ait une connectivité directe avec le cluster ONTAP . Veuillez vous référer à l'exemple de format suivant pour TridentActionMirrorUpdate :

Exemple

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state`reflète l'état du CRD TridentActionMirrorUpdate. Elle peut prendre la valeur *Réussi*, *En cours* ou *Échec*.

Utiliser la topologie CSI

Trident peut créer et attacher sélectivement des volumes aux nœuds présents dans un cluster Kubernetes en utilisant "Fonctionnalité de topologie CSI" .

Aperçu

À l'aide de la fonctionnalité 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 de cloud permettent aujourd'hui aux administrateurs Kubernetes de créer des nœuds basés sur des zones. Les nœuds peuvent être situés dans différentes zones de disponibilité au sein d'une région ou dans plusieurs régions. Pour faciliter le provisionnement des volumes pour les charges de travail dans une architecture multizone, Trident utilise la topologie CSI.

Découvrez plus d'informations sur la fonctionnalité de topologie CSI "ici" .

Kubernetes propose deux modes de liaison de volumes uniques :

- Avec `VolumeBindingMode` défini à `Immediate` Trident crée le volume sans aucune connaissance de la topologie. La liaison de volumes et le provisionnement dynamique sont gérés lors de la création du PVC. Il s'agit de la valeur par défaut. `VolumeBindingMode` et convient aux clusters qui n'imposent pas de contraintes de topologie. Les volumes persistants sont créés sans aucune dépendance aux exigences de planification du pod demandeur.
- Avec `VolumeBindingMode` défini à `WaitForFirstConsumer` La création et la liaison d'un volume persistant pour un PVC sont retardées jusqu'à ce qu'un pod utilisant le PVC soit planifié et créé. De cette façon, les volumes sont créés pour répondre aux contraintes de planification imposées par les exigences de topologie.

Le `WaitForFirstConsumer` Le mode de liaison ne nécessite pas d'étiquettes de topologie. Cette fonctionnalité peut être utilisée indépendamment de la fonction de topologie CSI.

Ce dont vous aurez besoin

Pour utiliser la topologie CSI, vous avez besoin 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:"1e11e4a2108024935ecfcb2912226cedeafd99df",
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:"1e11e4a2108024935ecfcb2912226cedeafd99df",
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 de prendre en compte la topologie.(`topology.kubernetes.io/region` et `topology.kubernetes.io/zone`). Ces étiquettes **doivent être présentes sur les nœuds du cluster** avant l'installation de Trident pour que ce Trident puisse prendre en compte 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","kube
rnetes.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","kube
rnetes.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","kube
rnetes.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 : Créer un backend prenant en compte la topologie

Les systèmes de stockage Trident peuvent être conçus pour provisionner sélectivement des volumes en fonction des zones de disponibilité. Chaque serveur dorsal peut contenir une option `supportedTopologies` Bloc représentant la liste des zones et régions prises en charge. Pour les StorageClasses qui utilisent un tel backend, un volume ne sera créé que s'il est demandé par une application planifiée dans une région/zone prise en charge.

Voici un exemple de définition de backend :

YAML

```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

```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 serveur dorsal. Ces régions et zones représentent la liste des valeurs autorisées qui peuvent être fournies dans une StorageClass. Pour les StorageClasses qui contiennent un sous-ensemble des régions et zones fournies dans un backend, Trident crée un volume sur le backend.

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

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-b
```

Dans cet exemple, le `region` et `zone` Les étiquettes indiquent l'emplacement du bassin de stockage. `topology.kubernetes.io/region` et `topology.kubernetes.io/zone` indiquer d'où les pools de stockage peuvent être utilisés.

Étape 2 : Définir des StorageClasses prenant en compte la topologie

En fonction des étiquettes de topologie fournies aux nœuds du cluster, les StorageClasses peuvent être définies pour contenir des informations de topologie. Cela déterminera les pools de stockage qui peuvent servir de candidats pour les demandes de PVC effectuées, ainsi que 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: null
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions: null
  - 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 StorageClass fournie ci-dessus, `volumeBindingMode` est réglé sur `WaitForFirstConsumer` . Les PVC demandés avec cette StorageClass ne seront pas traités tant qu'ils ne seront pas référencés dans un pod. Et, `allowedTopologies` indique les zones et la région à utiliser. Le `netapp-san-us-east1` StorageClass crée des PVC sur le `san-backend-us-east1` Le backend est défini ci-dessus.

Étape 3 : Créer et utiliser un PVC

Une fois la StorageClass créée et associée à un backend, vous pouvez désormais créer des PVC.

Voir l'exemple `spec` ci-dessous:

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

La création d'un PVC à l'aide de ce manifeste donnerait le résultat suivant :

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl 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                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

Pour que Trident puisse créer un volume et le lier au PVC, utilisez le PVC dans une capsule. 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: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Ce podSpec indique à Kubernetes de planifier le pod sur les nœuds présents dans le `us-east1` et choisissez parmi tous les nœuds présents dans la région `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
```

Mettre à jour les backends pour inclure `supportedTopologies`

Les backends préexistants peuvent être mis à jour pour inclure une liste de `supportedTopologies` en utilisant `tridentctl backend update` . Cela n'affectera pas les volumes déjà provisionnés et ne sera utilisé que pour les PVC ultérieurs.

Trouver plus d'informations

- "Gérer les ressources pour les conteneurs"
- "sélecteur de nœud"
- "Affinité et anti-affinité"
- "Souillures et tolérances"

Travailler avec des instantanés

Les snapshots de volumes persistants (PV) de Kubernetes permettent de créer des copies ponctuelles de volumes. Vous pouvez créer un instantané d'un volume créé à l'aide de Trident, importer un instantané créé en dehors de Trident, créer un nouveau volume à partir d'un instantané existant et récupérer des données de volume à partir d'instantanés.

Aperçu

La capture instantanée de volume est prise en charge par `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , et `google-cloud-netapp-volumes` conducteurs.

Avant de commencer

Vous devez disposer d'un contrôleur de snapshots externe et de définitions de ressources personnalisées (CRD) pour travailler avec les snapshots. C'est la responsabilité de l'orchestrateur Kubernetes (par exemple : Kubeadm, GKE, OpenShift).

Si votre distribution Kubernetes n'inclut pas le contrôleur de snapshots et les CRD, reportez-vous à la documentation.Déployer un contrôleur d'instantané 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 snapshots intégré et caché.

Créer un instantané de volume

Étapes

1. Créer un `VolumeSnapshotClass` Pour plus d'informations, veuillez consulter"Classe d'instantané de volume" .
   - Le `driver` indique le pilote Trident CSI.
   - `deletionPolicy`peut être `Delete ou `Retain` . Lorsqu'il est réglé sur `Retain` , l'instantané physique sous-jacent sur le cluster de stockage est conservé même lorsque le `VolumeSnapshot` L'objet a été supprimé.

     Exemple

     ```
     cat snap-sc.yaml
     ```

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

2. Créer un instantané d'un PVC existant.

   Exemples

   - Cet exemple crée un instantané d'un PVC existant.

     ```
     cat snap.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Cet exemple crée un objet instantané de volume pour un PVC nommé `pvc1` et le nom de l'instantané est défini sur `pvc1-snap` . Un VolumeSnapshot est analogue à un PVC et est associé à un `VolumeSnapshotContent` objet représentant l'instantané réel.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Vous pouvez identifier le `VolumeSnapshotContent` l'objet pour le `pvc1-snap` VolumeSnapshot en le décrivant. Le `Snapshot Content Name` identifie l'objet VolumeSnapshotContent qui sert de support à cet instantané. Le `Ready To Use` Ce paramètre indique que l'instantané peut être utilisé pour créer un nouveau PVC.

```
kubectl describe volumesnapshots pvc1-snap
Name:         pvc1-snap
Namespace:    default
...
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:       PersistentVolumeClaim
    Name:       pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
...
```

Créer un PVC à partir d'un instantané de volume

Vous pouvez utiliser `dataSource` pour créer un PVC à l'aide d'un VolumeSnapshot nommé `<pvc-name>` comme source des données. Une fois le tube PVC créé, il peut être fixé à une capsule et utilisé comme n'importe quel autre tube PVC.

> Le PVC sera créé dans le même système de traitement que le volume source. Se référer à"KB : La création d'un PVC à partir d'un instantané de PVC Trident ne peut pas être effectuée dans un autre backend." .

L'exemple suivant crée le PVC à l'aide de `pvc1-snap` comme source de données.

```
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: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importer un instantané de volume

Trident soutient"Processus de snapshot pré-provisionné Kubernetes" pour permettre à l'administrateur du cluster de créer un `VolumeSnapshotContent` objet et instantanés d'importation créés en dehors de Trident.

Avant de commencer

Trident doit avoir créé ou importé le volume parent de l'instantané.

Étapes

1. **Administrateur du cluster :** Créer un `VolumeSnapshotContent` objet qui référence l'instantané du backend. Cela lance le flux de travail de capture d'instantané dans Trident.
   - Spécifiez le nom de l'instantané du backend dans `annotations` comme `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">` .
   - Spécifier `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` dans `snapshotHandle` Il s'agit de la seule information fournie à Trident par le dispositif de capture d'instantanés externe. `ListSnapshots` appel.

     > Le `<volumeSnapshotContentName>` Il est impossible de toujours faire correspondre le nom de l'instantané du backend en raison des contraintes de dénomination des CR.

     Exemple

     L'exemple suivant crée un `VolumeSnapshotContent` objet qui référence un instantané du backend `snap-01` .

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01"  # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Administrateur du cluster :** Créez le `VolumeSnapshot` CR qui fait référence à la `VolumeSnapshotContent` objet. Cette demande d'accès à l'utilisation du `VolumeSnapshot` dans un espace de noms donné.

   Exemple

   L'exemple suivant crée un `VolumeSnapshot` CR nommé `import-snap` qui fait référence à `VolumeSnapshotContent` nommé `import-snap-content` .

   ```
   apiVersion: snapshot.storage.k8s.io/v1
   kind: VolumeSnapshot
   metadata:
     name: import-snap
   spec:
     # volumeSnapshotClassName: csi-snapclass (not required for pre-
   provisioned or imported snapshots)
     source:
       volumeSnapshotContentName: import-snap-content
   ```

3. **Traitement interne (aucune action requise) :** Le gestionnaire de snapshots externe reconnaît le snapshot nouvellement créé `VolumeSnapshotContent` et dirige le `ListSnapshots` appel. Trident crée le `TridentSnapshot` .
   - Le dispositif de capture d'écran externe définit le `VolumeSnapshotContent` à `readyToUse` et le `VolumeSnapshot` à `true` .
   - Trident revient `readyToUse=true` .
4. **Tout utilisateur :** Créez un `PersistentVolumeClaim` pour faire référence au nouveau `VolumeSnapshot` , où le `spec.dataSource` (ou `spec.dataSourceRef`) le nom est le

`VolumeSnapshot` nom.

Exemple

L'exemple suivant crée un PVC faisant référence à `VolumeSnapshot` nommé `import-snap` .

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Récupérer des données de volume à l'aide d'instantanés

Le répertoire des instantanés est masqué par défaut afin d'assurer une compatibilité maximale des volumes provisionnés à l'aide de `ontap-nas` et `ontap-nas-economy` conducteurs. Activer `.snapshot` répertoire permettant de récupérer directement les données à partir des instantanés.

Utilisez la commande ONTAP snapshot restore pour restaurer un volume à un état enregistré dans un instantané précédent.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Lorsque vous restaurez une copie instantanée, la configuration de volume existante est écrasée. Les modifications apportées aux données du volume après la création de la copie instantanée sont perdues.

Restauration de volume sur place à partir d'un instantané

Trident permet une restauration volumétrique rapide et in situ à partir d'une image instantanée grâce à `TridentActionSnapshotRestore` (TASR) CR. Cette demande de modification (CR) fonctionne comme une action impérative Kubernetes et ne persiste pas une fois l'opération terminée.

Trident prend en charge la restauration des instantanés sur le `ontap-san` , `ontap-san-economy` , `ontap-nas` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` , `google-cloud-netapp-volumes` , et `solidfire-san` conducteurs.

Avant de commencer

Vous devez disposer d'un PVC relié et d'un instantané du volume disponible.

- Vérifiez que le statut du PVC est lié.

```
kubectl get pvc
```

- Vérifiez que l'instantané du volume est prêt à être utilisé.

```
kubectl get vs
```

Étapes

1. Créer le TASR CR. Cet exemple crée un CR pour PVC `pvc1` et instantané de volume `pvc1-snapshot` .

   Le TASR CR doit se trouver dans un espace de noms où existent le PVC et le VS.

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Appliquez la demande de restauration (CR) pour restaurer à partir de l'instantané. Cet exemple effectue une restauration à partir d'un instantané. `pvc1` .

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Résultats

Trident restaure les données à partir de l'instantané. Vous pouvez vérifier l'état de la restauration de l'instantané :

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- Dans la plupart des cas, Trident ne réessaiera pas automatiquement l'opération en cas d'échec. Vous devrez répéter l'opération.
- Les utilisateurs de Kubernetes ne disposant pas d'un accès administrateur peuvent devoir obtenir l'autorisation de l'administrateur pour créer une ressource personnalisée TASR dans l'espace de noms de leur application.

Supprimer un PV avec ses instantanés associés

Lors de la suppression d'un volume persistant avec des instantanés associés, le volume Trident correspondant est mis à jour à l'état « Suppression en cours ». Supprimez les instantanés de volume pour supprimer le volume Trident .

Déployer un contrôleur d'instantané de volume

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

Étapes

1. Créer des CRD d'instantané 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.yam
l
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Créez le contrôleur d'instantané.

```
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
```

Si nécessaire, ouvrez `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` et mise à jour `namespace` à votre espace de noms.

Liens connexes

- "Instantanés de volume"
- "Classe d'instantané de volume"

Travailler avec les instantanés de groupes de volumes

Instantanés de groupe de volumes Kubernetes de volumes persistants (PV) NetApp Trident offre la possibilité de créer des instantanés de plusieurs volumes (un groupe d'instantanés de volumes). Cet instantané de groupe de volumes représente des copies de plusieurs volumes prises au même moment.

VolumeGroupSnapshot est une fonctionnalité bêta de Kubernetes avec des API bêta. Kubernetes 1.32 est la version minimale requise pour VolumeGroupSnapshot.

Créer des instantanés de groupes de volumes

La prise en charge des instantanés de groupes de volumes est assurée par `ontap-san` pilote uniquement pour le protocole iSCSI, non encore pris en charge avec Fibre Channel (FCP) ni NVMe/TCP. Avant de commencer

- Assurez-vous que votre version de Kubernetes est K8s 1.32 ou supérieure.
- Vous devez disposer d'un contrôleur de snapshots externe et de définitions de ressources personnalisées (CRD) pour travailler avec les snapshots. C'est la responsabilité de l'orchestrateur Kubernetes (par exemple : Kubeadm, GKE, OpenShift).

  Si votre distribution Kubernetes n'inclut pas le contrôleur de snapshots externe et les CRD, reportez-vous à la documentation.Déployer un contrôleur d'instantané de volume .

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

- Dans le fichier YAML du contrôleur d'instantané, définissez le `CSIVolumeGroupSnapshot` Définissez la fonctionnalité « true » sur cette porte pour garantir l'activation de l'instantané du groupe de volumes.
- Créez les classes d'instantané de groupe de volumes requises avant de créer un instantané de groupe de volumes.
- Assurez-vous que tous les PVC/volumes se trouvent sur le même SVM pour pouvoir créer un VolumeGroupSnapshot.

Étapes

- Créez une classe VolumeGroupSnapshotClass avant de créer un VolumeGroupSnapshot. Pour plus d'informations, veuillez consulter"Classe d'instantané de groupe de volume" .

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

- Créez des PVC avec les étiquettes requises en utilisant les classes de stockage existantes, ou ajoutez ces étiquettes aux PVC existants.

  L'exemple suivant crée le PVC à l'aide de `pvc1-group-snap` comme source de données et étiquette `consistentGroupSnapshot: groupA` . Définissez la clé et la valeur de l'étiquette en fonction de vos besoins.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- Créez un VolumeGroupSnapshot avec la même étiquette(`consistentGroupSnapshot: groupA`) spécifié dans le PVC.

  Cet exemple crée un instantané de groupe de volumes :

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Récupérer des données de volume à l'aide d'un instantané de groupe

Vous pouvez restaurer des volumes persistants individuels à l'aide des instantanés individuels créés dans le cadre de l'instantané du groupe de volumes. Vous ne pouvez pas récupérer l'instantané du groupe de volumes en tant qu'unité.

Utilisez la commande ONTAP snapshot restore pour restaurer un volume à un état enregistré dans un instantané précédent.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Lorsque vous restaurez une copie instantanée, la configuration de volume existante est écrasée. Les modifications apportées aux données du volume après la création de la copie instantanée sont perdues.

Restauration de volume sur place à partir d'un instantané

Trident permet une restauration volumétrique rapide et in situ à partir d'une image instantanée grâce à `TridentActionSnapshotRestore` (TASR) CR. Cette demande de modification (CR) fonctionne comme une action impérative Kubernetes et ne persiste pas une fois l'opération terminée.

Pour plus d'informations, voir "Restauration de volume sur place à partir d'un instantané".

Supprimer un PV avec des instantanés de groupe associés

Lors de la suppression d'un instantané de volume de groupe :

- Vous pouvez supprimer l'ensemble des VolumeGroupSnapshots, et non les snapshots individuels qui le composent.
- Si des PersistentVolumes sont supprimés alors qu'un snapshot existe pour ce PersistentVolume, Trident déplacera ce volume vers un état « en cours de suppression » car le snapshot doit être supprimé avant que le volume puisse être supprimé en toute sécurité.
- Si un clone a été créé à partir d'un instantané groupé et que le groupe doit ensuite être supprimé, une opération de division sur le clone sera lancée et le groupe ne pourra pas être supprimé tant que la division n'est pas terminée.

Déployer un contrôleur d'instantané de volume

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

Étapes

1. Créer des CRD d'instantané de volume.

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.
yaml
```

2. Créez le contrôleur d'instantané.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

Si nécessaire, ouvrez `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` et mise à jour `namespace` à votre espace de noms.

Liens connexes

- "Classe d'instantané de groupe de volume"
- "Instantanés de volume"