



Documentation Trident 25.06

Trident

NetApp
March 05, 2026

Sommaire

Documentation Trident 25.06	1
Notes de version	2
Quoi de neuf	2
Quoi de neuf dans la version 25.06.2	2
Modifications dans la version 25.06.1	2
Modifications dans la version 25.06	2
Modifications dans la version 25.02.1	5
Modifications dans la version 25.02	5
Modifications dans la version 24.10.1	7
Modifications dans la version 24.10	7
Modifications dans la version 24.06	9
Modifications dans la version 24.02	10
Modifications dans la version 23.10	10
Modifications dans la version 23.07.1	11
Modifications dans la version 23.07	11
Modifications dans la version 23.04	12
Modifications dans la version 23.01.1	13
Modifications dans la version 23.01	14
Modifications dans la version 22.10	14
Modifications dans la version 22.07	16
Modifications dans la version 22.04	17
Modifications dans la version 22.01.1	17
Modifications dans la version 22.01.0	18
Modifications dans la version 21.10.1	18
Modifications dans la version 21.10.0	19
Problèmes connus	20
Trouver plus d'informations	21
Versions antérieures de la documentation	21
Problèmes connus	21
La restauration des sauvegardes Restic de fichiers volumineux peut échouer	21
Commencer	22
Découvrez Trident	22
Découvrez Trident	22
Architecture Trident	23
Concepts	26
Démarrage rapide pour Trident	30
Quelle est la prochaine étape ?	31
Exigences	31
Informations essentielles sur Trident	31
Interfaces prises en charge (orchestrateurs)	32
Systèmes de stockage pris en charge	32
Prise en charge de Trident pour la virtualisation KubeVirt et OpenShift	33
Exigences fonctionnelles	33

Systèmes d'exploitation hôtes testés	34
Configuration de l'hôte	34
Configuration du système de stockage	34
Ports Trident	35
Images de conteneurs et versions Kubernetes correspondantes	35
Installer Trident	36
Installation via l'opérateur Trident	36
Installer à l'aide de tridentctl	36
Installation effectuée par un opérateur certifié OpenShift	36
Utilisez Trident	37
Préparer le nœud de travail	37
Choisir les bons outils	37
Découverte de services de nœuds	37
Volumes NFS	38
volumes iSCSI	38
Volumes NVMe/TCP	42
Volumes SCSI sur FC	43
Configurer et gérer les backends	46
Configurer les backends	46
Azure NetApp Files	46
Google Cloud NetApp Volumes	66
Configurer un Cloud Volumes Service pour le backend Google Cloud	83
Configurer un backend NetApp HCI ou SolidFire	95
Pilotes SAN ONTAP	100
Pilotes ONTAP NAS	130
Amazon FSx for NetApp ONTAP	168
Créer des backends avec kubectl	205
Gérer les backends	212
Créer et gérer des classes de stockage	222
Créer une classe de stockage	222
Gérer les classes de stockage	225
Provisionner et gérer les volumes	227
Provisionnez un volume	227
Augmenter les volumes	231
volumes d'importation	242
Personnaliser les noms et les étiquettes des volumes	253
Partager un volume NFS entre espaces de noms	256
Cloner des volumes entre espaces de noms	260
Répliquez des volumes à l'aide de SnapMirror	263
Utiliser la topologie CSI	269
Travailler avec des instantanés	277
Travailler avec les instantanés de groupes de volumes	285
Gérer et surveiller Trident	290
Amélioration du Trident	290
Amélioration du Trident	290

Mise à niveau avec l'opérateur	291
Mise à niveau avec tridentctl	296
Gérez Trident à l'aide de tridentctl	297
Commandes et drapeaux globaux	297
Options de commande et drapeaux	299
Prise en charge des plug-ins	305
Monitor Trident	305
Aperçu	305
Étape 1 : Définir une cible Prometheus	305
Étape 2 : Créer un moniteur de service Prometheus	306
Étape 3 : Interroger les métriques Trident avec PromQL	306
Découvrez la télémétrie de Trident AutoSupport	307
Désactiver les métriques Trident	308
Désinstaller Trident	309
Déterminer la méthode d'installation d'origine	309
Désinstaller une installation d'opérateur Trident	309
Désinstaller un tridentctl installation	310
Trident pour Docker	311
Prérequis pour le déploiement	311
Vérifiez les exigences	311
Outils NVMe	313
Outils FC	314
Déployer Trident	316
Méthode de plugin géré par Docker (version 1.13/17.03 et ultérieures)	316
Méthode traditionnelle (version 1.12 ou antérieure)	318
Démarrage de Trident au démarrage du système	319
Mettez à jour ou désinstallez Trident	320
Mise à niveau	321
Désinstaller	322
Travailler avec des volumes	322
Créer un volume	322
Supprimer un volume	323
Cloner un volume	323
Accéder aux volumes créés en externe	325
Options de volume spécifiques au conducteur	325
Collecter les bûches	331
Collecter les journaux pour le dépannage	331
Conseils généraux de dépannage	332
Gérer plusieurs instances Trident	332
Étapes pour le plugin géré par Docker (version 1.13/17.03 ou ultérieure)	332
Étapes pour la version traditionnelle (1.12 ou antérieure)	333
options de configuration du stockage	333
options de configuration globale	333
Configuration ONTAP	334
Configuration du logiciel Element	342

Problèmes connus et limitations	344
La mise à niveau du plugin Trident Docker Volume vers la version 20.10 et ultérieures à partir de versions plus anciennes entraîne un échec de mise à niveau avec l'erreur « fichier ou répertoire introuvable ».	344
Les noms de volumes doivent comporter au minimum 2 caractères.	345
Docker Swarm présente certains comportements qui empêchent Trident de le prendre en charge avec toutes les combinaisons de stockage et de pilotes.	345
Lors de la mise en service d'un FlexGroup , ONTAP ne met pas en service un deuxième FlexGroup si ce dernier a un ou plusieurs FlexGroup en commun avec le FlexGroup en cours de mise en service.	345
Meilleures pratiques et recommandations	346
Déploiement	346
Déployer dans un espace de noms dédié	346
Utilisez des quotas et des limites de plage pour contrôler la consommation de stockage.	346
Configuration de stockage	346
Présentation de la plateforme	346
ONTAP et Cloud Volumes ONTAP	347
Bonnes pratiques SolidFire	351
Où trouver plus d'informations ?	353
Intégrer Trident	354
Sélection et déploiement des conducteurs	354
Conception de classe de stockage	357
Conception de piscine virtuelle	358
Opérations de volume	359
Service de métriques	363
Protection des données et reprise après sinistre	364
Réplication et récupération du Trident	364
Réplication et récupération SVM	365
Réplication et récupération de volume	366
Protection des données instantanées	366
Sécurité	366
Sécurité	366
Configuration unifiée des clés Linux (LUKS)	368
Chiffrement Kerberos en transit	374
Protégez vos applications avec Trident Protect	382
Découvrez Trident Protect	382
Quelle est la prochaine étape ?	382
Installer Trident Protect	382
Exigences de Trident Protect	382
Installez et configurez Trident Protect	386
Installez le plugin CLI Trident Protect	389
Personnaliser l'installation de Trident Protect	393
Gérer Trident Protect	398
Gérer l'autorisation et le contrôle d'accès de Trident Protect	398
Surveiller les ressources de Trident Protect	405
Générer un ensemble de support Trident Protect	410

Amélioration de Trident Protect	412
Gérer et protéger les applications	413
Utilisez les objets Trident Protect AppVault pour gérer les compartiments	413
Définissez une application de gestion avec Trident Protect	427
Protégez les applications à l'aide de Trident Protect	431
Restaurer les applications	441
Répliquez les applications à l'aide de NetApp SnapMirror et Trident Protect	459
Migrer les applications à l'aide de Trident Protect	475
Gérer les hooks d'exécution de Trident Protect	479
Désinstallez Trident Protect	491
Blogs de Trident et Trident Protect	492
Blogs de Trident	492
Blogs de Trident Protect	492
Connaissances et soutien	494
Foire aux questions	494
Questions générales	494
Installer et utiliser Trident sur un cluster Kubernetes	494
Dépannage et assistance	496
Amélioration du Trident	497
Gérer les backends et les volumes	497
Dépannage	501
Dépannage général	501
Déploiement Trident infructueux avec l'opérateur	503
Déploiement Trident infructueux utilisant <code>tridentctl</code>	505
Supprimer complètement Trident et CRD	505
Échec du déstockage des nœuds NVMe avec les espaces de noms de blocs bruts RWX sous Kubernetes 1.26	506
Les clients NFSv4.2 signalent un « argument non valide » après la mise à niveau ONTAP alors qu'ils s'attendent à ce que « v4.2-xattrs » soit activé	507
Support	507
Soutien Trident	507
Autonomie	508
soutien communautaire	508
Assistance technique NetApp	508
Pour plus d'informations	508
Référence	509
Ports Trident	509
Ports Trident	509
API REST Trident	509
Quand utiliser l'API REST	509
Utilisation de l'API REST	509
Options de ligne de commande	510
Enregistrement	510
Kubernetes	510
Docker	511

REPOS	511
Objets Kubernetes et Trident	511
Comment les objets interagissent-ils entre eux ?	511
Kubernetes PersistentVolumeClaim objets	512
Kubernetes PersistentVolume objets	513
Kubernetes StorageClass objets	514
Kubernetes VolumeSnapshotClass objets	518
Kubernetes VolumeSnapshot objets	518
Kubernetes VolumeSnapshotContent objets	519
Kubernetes VolumeGroupSnapshotClass objets	519
Kubernetes VolumeGroupSnapshot objets	520
Kubernetes VolumeGroupSnapshotContent objets	520
Kubernetes CustomResourceDefinition objets	521
Trident StorageClass objets	521
Objets backend Trident	521
Trident StoragePool objets	522
Trident Volume objets	522
Trident Snapshot objets	523
Trident ResourceQuota objet	524
Normes de sécurité des modules (PSS) et contraintes de contexte de sécurité (SCC)	525
Contexte de sécurité Kubernetes requis et champs associés	526
Normes de sécurité des capsules (PSS)	526
Politiques de sécurité des pods (PSP)	527
Contraintes de contexte de sécurité (CCS)	528
Mentions légales	531
Copyright	531
Marques de commerce	531
Brevets	531
Politique de confidentialité	531
Open source	531

Documentation Trident 25.06

Notes de version

Quoi de neuf

Les notes de version fournissent des informations sur les nouvelles fonctionnalités, les améliorations et les corrections de bogues de la dernière version de NetApp Trident.



Le `tridentctl` Le fichier binaire pour Linux fourni dans le fichier zip d'installation est la version testée et prise en charge. Sachez que le `macos` binaire fourni dans le `/extras` Une partie du fichier zip n'est ni testée ni prise en charge.

Quoi de neuf dans la version 25.06.2

Le résumé des nouveautés fournit des détails sur les améliorations, les correctifs et les suppressions pour les versions de Trident et de Trident Protect.

Trident

Corrections

- **Kubernetes** : problème critique résolu où des périphériques iSCSI incorrects étaient découverts lors du détachement de volumes des nœuds Kubernetes.

Modifications dans la version 25.06.1

Trident



Pour les clients utilisant SolidFire, veuillez ne pas effectuer la mise à niveau vers la version 25.06.1 en raison d'un problème connu lors de la dépublication des volumes. La version 25.06.2 sera bientôt publiée pour résoudre ce problème.

Corrections

- **Kubernetes** :
 - Correction d'un problème où les NQN n'étaient pas vérifiés avant d'être démappés des sous-systèmes.
 - Correction d'un problème où plusieurs tentatives de fermeture d'un périphérique LUKS entraînaient des échecs lors du détachement des volumes.
 - Correction du volume iSCSI non mis en scène lorsque le chemin du périphérique a changé depuis sa création.
 - Clonage de blocs de volumes sur plusieurs classes de stockage.
- **OpenShift** : un problème a été résolu où la préparation du nœud iSCSI échouait avec OCP 4.19.
- Augmentation du délai d'attente lors du clonage d'un volume utilisant des backends SolidFire ("[Numéro 1008](#)").

Modifications dans la version 25.06

Trident

Améliorations

• Kubernetes :

- Ajout de la prise en charge des instantanés de groupes de volumes CSI avec `v1beta1` API Kubernetes de capture d'instantané de groupe de volumes pour le pilote iSCSI ONTAP-SAN. Voir "[Travailler avec les instantanés de groupes de volumes](#)".



VolumeGroupSnapshot est une fonctionnalité bêta de Kubernetes avec des API bêta. Kubernetes 1.32 est la version minimale requise pour VolumeGroupSnapshot.

- Ajout de la prise en charge d'ONTAP ASA r2 pour NVMe/TCP en plus d'iSCSI. Voir [link:"Options et exemples de configuration SAN ONTAP"](#).
- Ajout d'une prise en charge SMB sécurisée pour les volumes ONTAP-NAS et ONTAP-NAS-Economy. Les utilisateurs et les groupes Active Directory peuvent désormais être utilisés avec les volumes SMB pour une sécurité renforcée. Voir "[Activer le SMB sécurisé](#)".
- Amélioration de la concurrence des nœuds Trident pour une meilleure évolutivité des opérations sur les nœuds pour les volumes iSCSI.
- Ajouté `--allow-discards` lors de l'ouverture des volumes LUKS pour autoriser les commandes discard/TRIM en vue de la récupération d'espace.
- Performances améliorées lors du formatage des volumes chiffrés LUKS.
- Nettoyage LUKS amélioré pour les périphériques LUKS défectueux mais partiellement formatés.
- Idempotence améliorée des nœuds Trident pour l'attachement et le détachement de volumes NVMe.
- Ajouté `internalID` champ à la configuration de volume Trident pour le pilote ONTAP-SAN-Economy.
- Ajout de la prise en charge de la réplication de volumes avec SnapMirror pour les backends NVMe. Voir "[Répliquez des volumes à l'aide de SnapMirror](#)".

Améliorations expérimentales



Ne pas utiliser en environnement de production.

- [Aperçu technique] Activation des opérations simultanées des contrôleurs Trident via le `--enable-concurrency` indicateur de fonctionnalité. Cela permet aux opérations du contrôleur de s'exécuter en parallèle, améliorant ainsi les performances dans les environnements chargés ou de grande taille.



Cette fonctionnalité est expérimentale et prend actuellement en charge des flux de travail parallèles limités avec le pilote ONTAP-SAN (protocoles iSCSI et FCP).

- [Aperçu technique] Ajout de la prise en charge manuelle de la QoS avec le pilote ANF.

Corrections

• Kubernetes :

- Correction d'un problème avec CSI NodeExpandVolume où les périphériques multipath pouvaient se retrouver avec des tailles incohérentes lorsque le ou les disques SCSI sous-jacents étaient indisponibles.
- Correction d'un problème de nettoyage des politiques d'exportation en double pour les pilotes ONTAP-

NAS et ONTAP-NAS-Economy.

- Les volumes GCNV corrigés utilisent par défaut NFSv3 lorsque `nfsMountOptions` n'est pas défini ; les protocoles NFSv3 et NFSv4 sont désormais tous deux pris en charge. Si `nfsMountOptions` si aucune version NFS n'est fournie, la version NFS par défaut de l'hôte (NFSv3 ou NFSv4) sera utilisée.
- Correction d'un problème de déploiement lors de l'installation de Trident à l'aide de Kustomize ("Numéro 831").
- Correction des politiques d'exportation manquantes pour les PVC créées à partir d'instantanés ("Numéro 1016").
- Correction d'un problème où les tailles de volume ANF ne s'alignaient pas automatiquement par incréments de 1 Gio.
- Problème résolu lors de l'utilisation de NFSv3 avec Bottlerocket.
- Correction du délai d'attente lors du clonage d'un volume utilisant des backends SolidFire ("Numéro 1008").
- Correction d'un problème lié à l'extension des volumes ONTAP-NAS-Economy jusqu'à 300 To malgré des échecs de redimensionnement.
- Correction d'un problème où les opérations de division de clones étaient effectuées de manière synchrone lors de l'utilisation de l'API REST ONTAP .

Dépréciations :

- **Kubernetes** : Mise à jour de la version minimale de Kubernetes prise en charge à la v1.27.

Protection Trident

NetApp Trident Protect offre des fonctionnalités avancées de gestion des données d'application qui améliorent la fonctionnalité et la disponibilité des applications Kubernetes avec état, prises en charge par les systèmes de stockage NetApp ONTAP et le provisionneur de stockage NetApp Trident CSI.

Améliorations

- Des temps de restauration améliorés, offrant la possibilité d'effectuer des sauvegardes complètes plus fréquentes.
- Amélioration de la granularité de la définition des applications et de la restauration sélective grâce au filtrage Group-Version-Kind (GVK).
- Resynchronisation et réplication inverse efficaces lors de l'utilisation d'AppMirrorRelationship (AMR) avec NetApp SnapMirror, pour éviter la réplication complète des PVC.
- Ajout de la possibilité d'utiliser EKS Pod Identity pour créer des compartiments AppVault, supprimant ainsi la nécessité de spécifier un secret avec les informations d'identification du compartiment pour les clusters EKS.
- Ajout de la possibilité d'ignorer la restauration des étiquettes et des annotations dans l'espace de noms de restauration, si nécessaire.
- AppMirrorRelationship (AMR) va maintenant vérifier l'expansion du PVC source et effectuer l'expansion appropriée sur le PVC de destination si nécessaire.

Corrections

- Correction d'un bug où les valeurs d'annotation des instantanés précédents étaient appliquées aux instantanés plus récents. Toutes les annotations d'instantané sont désormais correctement appliquées.

- Un secret a été défini par défaut pour le chiffrement du transporteur de données (Kopia / Restic), s'il n'est pas défini...
- Ajout de messages de validation et d'erreur améliorés pour la création de coffres-forts d'applications S3.
- AppMirrorRelationship (AMR) ne réplique désormais que les PV dans l'état lié, afin d'éviter les tentatives infructueuses.
- Correction d'un problème d'affichage d'erreurs lors de la récupération d'AppVaultContent sur un AppVault contenant un grand nombre de sauvegardes.
- Les snapshots de machines virtuelles KubeVirt sont exclus des opérations de restauration et de basculement afin d'éviter les pannes.
- Correction d'un problème avec Kopia où les instantanés étaient supprimés prématurément en raison du calendrier de rétention par défaut de Kopia qui remplaçait celui défini par l'utilisateur.

Modifications dans la version 25.02.1

Trident

Corrections

- **Kubernetes :**
 - Correction d'un problème dans l'opérateur trident où les noms et versions des images sidecar étaient incorrectement renseignés lors de l'utilisation d'un registre d'images non standard ("[Numéro 983](#)").
 - Correction du problème où les sessions multipath ne parviennent pas à récupérer lors d'un basculement ONTAP ("[Numéro 961](#)").

Modifications dans la version 25.02

À partir de Trident 25.02, le résumé des nouveautés fournit des détails sur les améliorations, les correctifs et les dépréciations pour les versions de Trident et de Trident Protect.

Trident

Améliorations

- **Kubernetes :**
 - Ajout de la prise en charge d' ONTAP ASA r2 pour iSCSI.
 - Ajout de la prise en charge du détachement forcé pour les volumes ONTAP-NAS lors de scénarios d'arrêt brutal des nœuds. Les nouveaux volumes ONTAP-NAS utiliseront désormais des politiques d'exportation par volume gérées par Trident. Nous avons fourni une voie de mise à niveau pour permettre aux volumes existants de passer au nouveau modèle de politique d'exportation lors de leur dépublication, sans affecter les charges de travail actives.
 - Ajout de l'annotation cloneFromSnapshot.
 - Ajout de la prise en charge du clonage de volumes entre espaces de noms.
 - Amélioration des corrections d'analyse d'auto-réparation iSCSI pour lancer de nouvelles analyses par hôte, canal, cible et ID LUN exacts.
 - Ajout de la prise en charge de Kubernetes 1.32.
- **OpenShift :**
 - Ajout de la prise en charge de la préparation automatique des nœuds iSCSI pour RHCOS sur les

clusters ROSA.

- Ajout de la prise en charge de la virtualisation OpenShift pour les pilotes ONTAP .
- Ajout de la prise en charge de Fibre Channel sur le pilote ONTAP-SAN.
- Ajout de la prise en charge NVMe LUKS.
- Utilisation d'images temporaires pour toutes les images de base.
- Ajout de la découverte et de la journalisation de l'état de connexion iSCSI lorsque les sessions iSCSI devraient être connectées mais ne le sont pas ("[Numéro 961](#)").
- Ajout de la prise en charge des volumes SMB avec le pilote google-cloud-netapp-volumes.
- Ajout d'une prise en charge permettant aux volumes ONTAP d'ignorer la file d'attente de récupération lors de leur suppression.
- Ajout de la possibilité de remplacer les images par défaut à l'aide de SHA au lieu de balises.
- Ajout de l'option image-pull-secrets à l'installateur tridentctl.

Corrections

- **Kubernetes :**
 - Correction des adresses IP de nœuds manquantes dans les politiques d'exportation automatique ("[Numéro 965](#)").
 - Correction d'un problème de basculement prématuré des politiques d'exportation automatique vers une politique par volume pour ONTAP-NAS-Economy.
 - Correction des identifiants de configuration du backend pour prendre en charge toutes les partitions ARN AWS disponibles ("[Numéro 913](#)").
 - Ajout d'une option pour désactiver la réconciliation du configurateur automatique dans l'opérateur Trident ("[Numéro 924](#)").
 - Ajout d'un contexte de sécurité pour le conteneur csi-resizer ("[Numéro 976](#)").

Protection Trident

NetApp Trident Protect offre des fonctionnalités avancées de gestion des données d'application qui améliorent la fonctionnalité et la disponibilité des applications Kubernetes avec état, prises en charge par les systèmes de stockage NetApp ONTAP et le provisionneur de stockage NetApp Trident CSI.

Améliorations

- Ajout de la prise en charge de la sauvegarde et de la restauration pour les machines virtuelles KubeVirt / OpenShift Virtualization pour les modes de stockage volumeMode : File et volumeMode : Block (périphérique brut). Cette prise en charge est compatible avec tous les pilotes Trident et améliore les fonctionnalités de protection existantes lors de la réplication du stockage à l'aide de NetApp SnapMirror avec Trident Protect.
- Ajout de la possibilité de contrôler le comportement de gel au niveau de l'application pour les environnements Kubevirt.
- Ajout de la prise en charge de la configuration des connexions proxy AutoSupport .
- Ajout de la possibilité de définir un secret pour le chiffrement du transporteur de données (Kopia / Restic).
- Ajout de la possibilité d'exécuter manuellement un hook d'exécution.
- Ajout de la possibilité de configurer les contraintes de contexte de sécurité (SCC) lors de l'installation de Trident Protect.

- Ajout de la prise en charge de la configuration de nodeSelector lors de l'installation de Trident Protect.
- Ajout de la prise en charge du proxy de sortie HTTP/HTTPS pour les objets AppVault.
- Filtre de ressources étendu pour permettre l'exclusion des ressources à l'échelle du cluster.
- Ajout de la prise en charge du jeton de session AWS dans les informations d'identification S3 AppVault.
- Ajout de la prise en charge de la collecte des ressources après les hooks d'exécution pré-instantané.

Corrections

- Amélioration de la gestion des volumes temporaires pour éviter la file d'attente de récupération des volumes ONTAP .
- Les annotations SCC ont retrouvé leurs valeurs d'origine.
- Amélioration de l'efficacité de la restauration grâce à la prise en charge des opérations parallèles.
- Prise en charge améliorée des délais d'expiration des hooks d'exécution pour les applications plus volumineuses.

Modifications dans la version 24.10.1

Améliorations

- **Kubernetes** : Ajout de la prise en charge de Kubernetes 1.32.
- Ajout de la découverte et de la journalisation de l'état de connexion iSCSI lorsque les sessions iSCSI devraient être connectées mais ne le sont pas ("[Numéro 961](#)").

Corrections

- Correction des adresses IP de nœuds manquantes dans les politiques d'exportation automatique ("[Numéro 965](#)").
- Correction d'un problème de basculement prématuré des politiques d'exportation automatique vers une politique par volume pour ONTAP-NAS-Economy.
- Mise à jour des dépendances de Trident et Trident-ASUP pour corriger les vulnérabilités CVE-2024-45337 et CVE-2024-45310.
- Suppression des déconnexions pour les portails non-CHAP présentant des dysfonctionnements intermittents pendant l'auto-réparation iSCSI ("[Numéro 961](#)").

Modifications dans la version 24.10

Améliorations

- Le pilote Google Cloud NetApp Volumes est désormais disponible pour tous les volumes NFS et prend en charge le provisionnement prenant en compte les zones.
- L'identité de charge de travail GCP sera utilisée comme identité cloud pour les Google Cloud NetApp Volumes avec GKE.
- Ajouté `formatOptions` Paramètre de configuration des pilotes ONTAP-SAN et ONTAP-SAN-Economy permettant aux utilisateurs de spécifier les options de format LUN.
- Réduction de la taille minimale des volumes Azure NetApp Files à 50 Gio. La nouvelle taille minimale d'Azure devrait être disponible pour tous en novembre.

- Ajouté `denyNewVolumePools` Paramètre de configuration permettant de limiter les pilotes ONTAP-NAS-Economy et ONTAP-SAN-Economy aux pools Flexvol préexistants.
- Ajout d'une détection pour l'ajout, la suppression ou le renommage d'agrégats du SVM sur tous les pilotes ONTAP .
- Ajout de 18 Mio de surcharge aux LUN LUKS pour garantir que la taille du PVC signalée est utilisable.
- Amélioration de la gestion des erreurs de mise en scène et de désinstallation des nœuds ONTAP-SAN et ONTAP-SAN-Economy pour permettre la désinstallation de supprimer des périphériques après une étape ayant échoué.
- Ajout d'un générateur de rôles personnalisé permettant aux clients de créer un rôle minimaliste pour Trident dans ONTAP.
- Ajout d'une journalisation supplémentaire pour le dépannage `lsscsi` ("[Numéro 792](#)").

Kubernetes

- Ajout de nouvelles fonctionnalités Trident pour les flux de travail natifs de Kubernetes :
 - Protection des données
 - Migration des données
 - Reprise après sinistre
 - Mobilité des applications

["Apprenez-en davantage sur Trident Protect"](#).
- Ajout d'un nouveau drapeau `--k8s-api-qps` aux installateurs de définir la valeur QPS utilisée par Trident pour communiquer avec le serveur API Kubernetes.
- Ajouté `--node-prep` Indicateur à fournir aux installateurs pour la gestion automatique des dépendances du protocole de stockage sur les nœuds du cluster Kubernetes. Compatibilité testée et vérifiée avec le protocole de stockage iSCSI Amazon Linux 2023
- Ajout de la prise en charge du détachement forcé pour les volumes ONTAP-NAS-Economy lors de scénarios d'arrêt de nœud non progressif.
- Les nouveaux volumes NFS ONTAP-NAS-Economy utiliseront des politiques d'exportation par `qtree` lors de leur utilisation. `autoExportPolicy` Option backend. Les `Qtrees` ne seront associés aux politiques d'exportation restrictives des nœuds qu'au moment de la publication afin d'améliorer le contrôle d'accès et la sécurité. Les `qtrees` existants seront basculés vers le nouveau modèle de politique d'exportation lorsque Trident dépubliera le volume de tous les nœuds afin de le faire sans impacter les charges de travail actives.
- Ajout de la prise en charge de Kubernetes 1.31.

Améliorations expérimentales

- Ajout d'un aperçu technique pour la prise en charge de Fibre Channel sur le pilote ONTAP-SAN.

Corrections

- **Kubernetes :**
 - Correction d'un webhook d'admission Rancher empêchant les installations de Trident Helm ("[Numéro 839](#)").
 - Clé d'affinité fixe dans les valeurs du graphique Helm ("[Numéro 898](#)").

- Le composant `tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector` ne fonctionne pas avec la valeur « true » ("[Numéro 899](#)").
- Suppression des instantanés éphémères créés lors du clonage ("[Numéro 901](#)").
- Ajout de la prise en charge de Windows Server 2019.
- Correction de `go mod tidy` dans le dépôt Trident ("[Numéro 767](#)").

Dépréciations

- **Kubernetes :**
 - Version minimale de Kubernetes prise en charge mise à jour à 1.25.
 - Suppression de la prise en charge de la politique de sécurité POD.

repositionnement de produit

À partir de la version 24.10, Astra Trident est renommé Trident (Netapp Trident). Ce changement de marque n'affecte aucune fonctionnalité, plateforme prise en charge ou interopérabilité de Trident.

Modifications dans la version 24.06

Améliorations

- **IMPORTANT** : `limitVolumeSize` Ce paramètre limite désormais la taille des qtrees/LUN dans les pilotes économiques ONTAP . Utilisez le nouveau `limitVolumePoolSize` paramètre permettant de contrôler les tailles Flexvol dans ces pilotes. ("[Numéro 341](#)").
- Ajout d'une fonctionnalité d'auto-réparation iSCSI permettant de lancer des analyses SCSI par ID LUN exact si des igroups obsolètes sont utilisés ("[Numéro 883](#)").
- Ajout de la prise en charge des opérations de clonage et de redimensionnement de volumes, même lorsque le serveur dorsal est en mode suspendu.
- Ajout de la possibilité de propager les paramètres de journalisation configurés par l'utilisateur pour le contrôleur Trident aux pods de nœuds Trident .
- Ajout de la prise en charge dans Trident de l'utilisation de REST par défaut au lieu d'ONTAPI (ZAPI) pour les versions ONTAP 9.15.1 et ultérieures.
- Ajout de la prise en charge des noms de volumes personnalisés et des métadonnées sur les backends de stockage ONTAP pour les nouveaux volumes persistants.
- Amélioration `azure-netapp-files` (ANF) pilote pour activer automatiquement le répertoire de snapshot par défaut lorsque les options de montage NFS sont configurées pour utiliser la version NFS 4.x.
- Ajout de la prise en charge de Bottlerocket pour les volumes NFS.
- Ajout d'une prise en charge en avant-première technique pour Google Cloud NetApp Volumes.

Kubernetes

- Ajout de la prise en charge de Kubernetes 1.30.
- Ajout de la possibilité pour Trident DaemonSet de nettoyer les montures zombies et les fichiers de suivi résiduels au démarrage ("[Numéro 883](#)").
- Ajout d'une annotation PVC `trident.netapp.io/luksEncryption` pour l'importation dynamique de volumes LUKS ("[Numéro 849](#)").

- Ajout d'une prise en compte de la topologie au pilote ANF.
- Ajout de la prise en charge des nœuds Windows Server 2022.

Corrections

- Correction des échecs d'installation de Trident dus à des transactions obsolètes.
- Correction de tridentctl pour ignorer les messages d'avertissement de Kubernetes ("[Numéro 892](#)").
- Contrôleur Trident modifié `SecurityContextConstraint` priorité à 0 ("[Numéro 887](#)").
- Les pilotes ONTAP acceptent désormais les tailles de volume inférieures à 20 Mio ("[Problème n° 885](#)").
- Correction de Trident pour empêcher la réduction des volumes FlexVol lors de l'opération de redimensionnement pour le pilote ONTAP-SAN.
- Correction d'un problème d'importation de volumes ANF avec NFS v4.1.

Modifications dans la version 24.02

Améliorations

- Ajout de la prise en charge de Cloud Identity.
 - AKS avec ANF - Azure Workload Identity sera utilisé comme identité cloud.
 - EKS avec FSxN - Le rôle AWS IAM sera utilisé comme identité cloud.
- Ajout de la possibilité d'installer Trident en tant qu'extension sur un cluster EKS depuis la console EKS.
- Ajout de la possibilité de configurer et de désactiver l'auto-réparation iSCSI ("[Numéro 864](#)").
- Ajout de la personnalité Amazon FSx aux pilotes ONTAP pour permettre l'intégration avec AWS IAM et SecretsManager, et pour permettre à Trident de supprimer les volumes FSx avec des sauvegardes ("[Numéro 453](#)").

Kubernetes

- Ajout de la prise en charge de Kubernetes 1.29.

Corrections

- Correction des messages d'avertissement ACP lorsque l'ACP n'est pas activé ("[Numéro 866](#)").
- Ajout d'un délai de 10 secondes avant d'effectuer une division de clone lors de la suppression d'un instantané pour les pilotes ONTAP, lorsqu'un clone est associé à l'instantané.

Dépréciations

- Suppression intégrale du cadre d'attestations des manifestes d'images multiplateformes.

Modifications dans la version 23.10

Corrections

- Extension de volume fixe si la nouvelle taille demandée est inférieure à la taille totale du volume pour les pilotes de stockage `ontap-nas` et `ontap-nas-flexgroup` ("[Numéro 834](#)").
- Taille de volume fixe pour n'afficher que la taille utilisable du volume lors de l'importation pour les pilotes de stockage `ontap-nas` et `ontap-nas-flexgroup` ("[Numéro 722](#)").

- Correction de la conversion du nom FlexVol pour ONTAP-NAS-Economy.
- Correction d'un problème d'initialisation de Trident sur un nœud Windows lors du redémarrage de ce dernier.

Améliorations

Kubernetes

Ajout de la prise en charge de Kubernetes 1.28.

Trident

- Ajout de la prise en charge de l'utilisation des identités managées Azure (AMI) avec le pilote de stockage azure-netapp-files.
- Ajout de la prise en charge de NVMe sur TCP pour le pilote ONTAP-SAN.
- Ajout de la possibilité de suspendre le provisionnement d'un volume lorsque le backend est mis en état suspendu par l'utilisateur ("[Numéro 558](#)").

Modifications dans la version 23.07.1

Kubernetes : Correction de la suppression du DaemonSet pour prendre en charge les mises à niveau sans interruption de service ("[Numéro 740](#)").

Modifications dans la version 23.07

Corrections

Kubernetes

- Correction de la mise à jour Trident pour ignorer les anciens pods bloqués en état de terminaison ("[Numéro 740](#)").
- Ajout d'une tolérance à la définition de « transient-trident-version-pod » ("[Numéro 795](#)").

Trident

- Requêtes ONTAPI (ZAPI) corrigées pour garantir que les numéros de série LUN sont interrogés lors de l'obtention des attributs LUN afin d'identifier et de corriger les périphériques iSCSI fantômes lors des opérations de préparation des nœuds.
- Correction de la gestion des erreurs dans le code du pilote de stockage ("[Numéro 816](#)").
- Redimensionnement corrigé des quotas lors de l'utilisation de pilotes ONTAP avec use-rest=true.
- Création de clones LUN corrigée dans ontap-san-economy.
- Rétablir le champ d'informations de publication `rawDevicePath` à `devicePath` ; ajout d'une logique de remplissage et de récupération (dans certains cas) `devicePath` champ.

Améliorations

Kubernetes

- Ajout de la prise en charge de l'importation d'instantanés préconfigurés.

- Déploiement minimal et permissions Linux pour les ensembles de démons ("[Numéro 817](#)").

Trident

- Le champ d'état n'est plus signalé pour les volumes et les instantanés « en ligne ».
- Met à jour l'état du backend si le backend ONTAP est hors ligne ("[Numéro 801](#)" , "[#543](#)").
- Le numéro de série LUN est toujours récupéré et publié lors du flux de travail ControllerVolumePublish.
- Ajout d'une logique supplémentaire pour vérifier le numéro de série et la taille du périphérique iSCSI multipath.
- Vérification supplémentaire des volumes iSCSI pour garantir que le périphérique multipath correct n'est pas configuré.

Amélioration expérimentale

Ajout d'une prise en charge en avant-première technique pour NVMe sur TCP pour le pilote ONTAP-SAN.

Documentation

De nombreuses améliorations ont été apportées à l'organisation et à la mise en forme.

Dépréciations

Kubernetes

- Suppression de la prise en charge des instantanés v1beta1.
- Suppression de la prise en charge des volumes et classes de stockage antérieurs à CSI.
- Version minimale de Kubernetes prise en charge mise à jour à 1.22.

Modifications dans la version 23.04



Le détachement forcé des volumes ONTAP-SAN-* n'est pris en charge qu'avec les versions de Kubernetes dont la fonctionnalité d'arrêt non gracieux des nœuds est activée. Le détachement forcé doit être activé lors de l'installation à l'aide de `--enable-force-detach` Drapeau de l'installateur Trident .

Corrections

- Correction de l'opérateur Trident pour utiliser l'hôte local IPv6 lors de l'installation lorsque cela est spécifié dans les spécifications.
- Les permissions du rôle de cluster de l'opérateur Trident ont été corrigées pour être synchronisées avec les permissions du bundle ("[Numéro 799](#)").
- Correction d'un problème lié à l'attachement d'un volume de blocs bruts sur plusieurs nœuds en mode RWX.
- Correction de la prise en charge du clonage FlexGroup et de l'importation de volumes SMB.
- Correction d'un problème où la manette Trident ne pouvait pas s'éteindre immédiatement ("[Numéro 811](#)").
- Ajout d'un correctif pour lister tous les noms d'igroup associés à un LUN spécifié provisionné avec les pilotes ontap-san-*
- Ajout d'un correctif permettant aux processus externes de s'exécuter jusqu'à leur terme.

- Correction d'une erreur de compilation pour l'architecture s390 ("[Numéro 537](#)").
- Correction du niveau de journalisation incorrect lors des opérations de montage de volume ("[Numéro 781](#)").
- Correction d'une erreur potentielle d'assertion de type ("[Numéro 802](#)").

Améliorations

- Kubernetes :
 - Ajout de la prise en charge de Kubernetes 1.27.
 - Ajout de la prise en charge de l'importation des volumes LUKS.
 - Ajout de la prise en charge du mode d'accès PVC ReadWriteOncePod.
 - Ajout de la prise en charge du détachement forcé pour les volumes ONTAP-SAN-* lors des scénarios d'arrêt non progressif des nœuds.
 - Tous les volumes ONTAP-SAN-* utiliseront désormais des igroups par nœud. Les LUN ne seront mappées aux igroups que lorsqu'elles seront activement publiées sur ces nœuds afin d'améliorer notre posture de sécurité. Les volumes existants seront transférés de manière opportuniste vers le nouveau système igroup lorsque Trident déterminera qu'il est possible de le faire en toute sécurité sans impacter les charges de travail actives ("[Numéro 758](#)").
 - Amélioration de la sécurité de Trident grâce au nettoyage des igroups gérés par Trident inutilisés sur les backends ONTAP-SAN-*.
- Ajout de la prise en charge des volumes SMB avec Amazon FSx aux pilotes de stockage ontap-nas-economy et ontap-nas-flexgroup.
- Ajout de la prise en charge des partages SMB avec les pilotes de stockage ontap-nas, ontap-nas-economy et ontap-nas-flexgroup.
- Ajout de la prise en charge des nœuds arm64 ("[Numéro 732](#)").
- Procédure d'arrêt Trident améliorée en désactivant d'abord les serveurs API ("[Numéro 811](#)").
- Ajout de la prise en charge de la compilation multiplateforme pour les hôtes Windows et arm64 au Makefile ; voir BUILD.md.

Dépréciations

Kubernetes : Les igroups à portée backend ne seront plus créés lors de la configuration des pilotes ontap-san et ontap-san-economy ("[Numéro 758](#)").

Modifications dans la version 23.01.1

Corrections

- Correction de l'opérateur Trident pour utiliser l'hôte local IPv6 lors de l'installation lorsque cela est spécifié dans les spécifications.
- Les permissions du rôle de cluster de l'opérateur Trident ont été corrigées afin d'être synchronisées avec les permissions du bundle."[Numéro 799](#)".
- Ajout d'un correctif permettant aux processus externes de s'exécuter jusqu'à leur terme.
- Correction d'un problème lié à l'attachement d'un volume de blocs bruts sur plusieurs nœuds en mode RWX.
- Correction de la prise en charge du clonage FlexGroup et de l'importation de volumes SMB.

Modifications dans la version 23.01



Kubernetes 1.27 est désormais pris en charge dans Trident. Veuillez mettre à niveau Trident avant de mettre à niveau Kubernetes.

Corrections

- Kubernetes : Ajout d'options pour exclure la création de politiques de sécurité de pod afin de corriger les installations de Trident via Helm ("[Numéros 783 et 794](#)").

Améliorations

Kubernetes

- Ajout de la prise en charge de Kubernetes 1.26.
- Amélioration de l'utilisation globale des ressources Trident RBAC ("[Numéro 757](#)").
- Ajout d'une automatisation pour détecter et corriger les sessions iSCSI interrompues ou obsolètes sur les nœuds hôtes.
- Ajout de la prise en charge de l'extension des volumes chiffrés LUKS.
- Kubernetes : Ajout de la prise en charge de la rotation des informations d'identification pour les volumes chiffrés LUKS.

Trident

- Ajout de la prise en charge des volumes SMB avec Amazon FSx for NetApp ONTAP au pilote de stockage `ontap-nas`.
- Ajout de la prise en charge des autorisations NTFS lors de l'utilisation de volumes SMB.
- Ajout de la prise en charge des pools de stockage pour les volumes GCP avec niveau de service CVS.
- Ajout de la prise en charge de l'utilisation optionnelle de `flexgroupAggregateList` lors de la création de FlexGroups avec le pilote de stockage `ontap-nas-flexgroup`.
- Amélioration des performances du pilote de stockage `ontap-nas-economy` lors de la gestion de plusieurs volumes FlexVol
- Activation des mises à jour `dataLIF` pour tous les pilotes de stockage NAS ONTAP .
- La convention d'appellation des déploiements Trident et des DaemonSets a été mise à jour pour refléter le système d'exploitation du nœud hôte.

Dépréciations

- Kubernetes : Mise à jour de la version minimale de Kubernetes prise en charge à 1.21.
- Les `DataLIF` ne doivent plus être spécifiés lors de la configuration. `ontap-san` ou `ontap-san-economy` conducteurs.

Modifications dans la version 22.10

Vous devez lire les informations essentielles suivantes avant de procéder à la mise à niveau vers Trident 22.10.

Informations essentielles concernant Trident 22.10



- Kubernetes 1.25 est désormais pris en charge dans Trident. Vous devez mettre à niveau Trident vers la version 22.10 avant de passer à Kubernetes 1.25.
- Trident impose désormais strictement l'utilisation de la configuration multipathing dans les environnements SAN, avec une valeur recommandée de `find_multipaths: no` dans le fichier `multipath.conf`.

Utilisation d'une configuration sans chemins multiples ou utilisation de `find_multipaths: yes` ou `find_multipaths: smart` La valeur dans le fichier `multipath.conf` entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version du 21 juillet.

Corrections

- Correction d'un problème spécifique au backend ONTAP créé à l'aide de `credentials` Le champ n'a pas pu être mis en ligne lors de la mise à niveau 22.07.0 ("Numéro 759").
- **Docker** : Correction d'un problème empêchant le démarrage du plugin de volume Docker dans certains environnements ("Numéro 548" et "Numéro 760").
- Correction d'un problème SLM spécifique aux backends SAN ONTAP afin de garantir que seul un sous-ensemble des dataLIF appartenant aux nœuds de reporting soit publié.
- Correction d'un problème de performances qui entraînait des analyses inutiles des LUN iSCSI lors de l'attachement d'un volume.
- Suppression des tentatives de nouvelle tentative granulaires dans le flux de travail Trident iSCSI pour détecter rapidement les échecs et réduire les intervalles de nouvelle tentative externes.
- Correction d'un problème où une erreur était renvoyée lors du vidage d'un périphérique iSCSI alors que le périphérique multipath correspondant avait déjà été vidé.

Améliorations

- Kubernetes :
 - Ajout de la prise en charge de Kubernetes 1.25. Vous devez mettre à niveau Trident vers la version 22.10 avant de passer à Kubernetes 1.25.
 - Ajout d'un ServiceAccount, d'un ClusterRole et d'un ClusterRoleBinding distincts pour le déploiement Trident et le DaemonSet afin de permettre de futures améliorations des autorisations.
 - Prise en charge supplémentaire pour "[partage de volumes entre espaces de noms](#)".
- Tous les Trident `ontap-*` Les pilotes de stockage fonctionnent désormais avec l'API REST ONTAP .
- Ajout d'un nouvel opérateur YAML(`bundle_post_1_25.yaml`) sans un PodSecurityPolicy pour prendre en charge Kubernetes 1.25.
- Ajouté "[prise en charge des volumes chiffrés LUKS](#)" pour `ontap-san` et `ontap-san-economy` pilotes de stockage.
- Ajout de la prise en charge des nœuds Windows Server 2019.
- Ajouté "[prise en charge des volumes SMB sur les nœuds Windows](#)" à travers le `azure-netapp-files` pilote de stockage.
- La détection automatique du basculement MetroCluster pour les pilotes ONTAP est désormais disponible pour tous.

Dépréciations

- **Kubernetes** : La version minimale de Kubernetes prise en charge a été mise à jour à la version 1.20.
- Suppression du pilote Astra Data Store (ADS).
- Suppression de la prise en charge de `yes` et `smart` options pour `find_multipaths` lors de la configuration du multipathing des nœuds de travail pour iSCSI.

Modifications dans la version 22.07

Corrections

Kubernetes

- Correction d'un problème de gestion des valeurs booléennes et numériques pour le sélecteur de nœuds lors de la configuration de Trident avec Helm ou l'opérateur Trident . (["Problème GitHub n° 700"](#))
- Correction d'un problème de gestion des erreurs provenant d'un chemin non-CHAP, de sorte que kubelet réessaiera en cas d'échec. (["Problème GitHub n° 736"](#))

Améliorations

- Transition de `k8s.gcr.io` à `registry.k8s.io` comme registre par défaut pour les images CSI
- Les volumes ONTAP-SAN utiliseront désormais des igroups par nœud et ne mapperont les LUN aux igroups que lorsqu'ils sont activement publiés sur ces nœuds afin d'améliorer notre posture de sécurité. Les volumes existants seront transférés de manière opportuniste vers le nouveau système igroup lorsque Trident déterminera qu'il est possible de le faire en toute sécurité sans impacter les charges de travail actives.
- Un quota de ressources a été inclus dans les installations de Trident afin de garantir que le DaemonSet de Trident soit planifié lorsque la consommation de PriorityClass est limitée par défaut.
- Ajout de la prise en charge des fonctionnalités réseau au pilote Azure NetApp Files . (["Problème GitHub n° 717"](#))
- Ajout d'une prévisualisation technique de la détection automatique du basculement MetroCluster aux pilotes ONTAP . (["Problème GitHub n° 228"](#))

Dépréciations

- **Kubernetes** : La version minimale de Kubernetes prise en charge a été mise à jour à la version 1.19.
- La configuration du backend n'autorise plus plusieurs types d'authentification dans une seule configuration.

Déménagements

- Le pilote AWS CVS (obsolète depuis la version 22.04) a été supprimé.
- Kubernetes
 - Suppression de la capacité `SYS_ADMIN` inutile des pods de nœud.
 - Réduit `nodeprep` à de simples informations sur l'hôte et à une découverte active des services pour confirmer au mieux que les services NFS/iSCSI sont disponibles sur les nœuds de travail.

Documentation

Un nouveau ["Normes de sécurité des capsules"](#) La section (PSS) a été ajoutée détaillant les permissions

activées par Trident lors de l'installation.

Modifications dans la version 22.04

NetApp améliore et optimise continuellement ses produits et services. Voici quelques-unes des dernières fonctionnalités de Trident. Pour les versions précédentes, veuillez vous référer à "[Versions antérieures de la documentation](#)".



Si vous effectuez une mise à niveau depuis une version antérieure de Trident et que vous utilisez Azure NetApp Files, `location` Le paramètre de configuration est désormais un champ obligatoire et unique.

Corrections

- Amélioration de l'analyse des noms d'initiateurs iSCSI. ("[Problème GitHub n° 681](#)")
- Correction d'un problème où les paramètres de classe de stockage CSI n'étaient pas autorisés. ("[Problème GitHub n° 598](#)")
- Correction d'une déclaration de clé dupliquée dans la CRD Trident. ("[Problème GitHub n° 671](#)")
- Correction des journaux d'instantanés CSI inexacts. ("[Problème GitHub n° 629](#)")
- Correction d'un problème lié à la dépublication des volumes sur les nœuds supprimés. ("[Problème GitHub n° 691](#)")
- Ajout de la gestion des incohérences du système de fichiers sur les périphériques de stockage par blocs. ("[Problème GitHub n° 656](#)")
- Correction d'un problème d'extraction des images de support automatique lors de la configuration `imageRegistry` signaler lors de l'installation. ("[Problème GitHub n° 715](#)")
- Correction d'un problème où le pilote Azure NetApp Files ne parvenait pas à cloner un volume comportant plusieurs règles d'exportation.

Améliorations

- Les connexions entrantes aux points de terminaison sécurisés de Trident nécessitent désormais au minimum le protocole TLS 1.3. ("[Problème GitHub n° 698](#)")
- Trident ajoute désormais des en-têtes HSTS aux réponses provenant de ses points de terminaison sécurisés.
- Trident tente désormais d'activer automatiquement la fonctionnalité d'autorisations Unix Azure NetApp Files.
- **Kubernetes** : Le démonset Trident s'exécute désormais avec la classe de priorité `system-node-critical`. ("[Problème GitHub n° 694](#)")

Déménagements

Le pilote de la série E (désactivé depuis le 20.07) a été supprimé.

Modifications dans la version 22.01.1

Corrections

- Correction d'un problème lié à la dépublication des volumes sur les nœuds supprimés. ("[Problème GitHub n° 691](#)")

- Correction d'un problème de panique lors de l'accès aux champs nuls pour l'espace agrégé dans les réponses de l'API ONTAP .

Modifications dans la version 22.01.0

Corrections

- **Kubernetes** : Augmenter le délai de nouvelle tentative d'enregistrement des nœuds pour les grands clusters.
- Correction d'un problème où le pilote azure-netapp-files pouvait être perturbé par plusieurs ressources portant le même nom.
- Les DataLIF ONTAP SAN IPv6 fonctionnent désormais si elles sont spécifiées entre crochets.
- Correction d'un problème où la tentative d'importation d'un volume déjà importé renvoyait une fin de fichier (EOF), laissant le volume persistant (PVC) en état d'attente. ("[Problème GitHub n° 489](#)")
- Correction d'un problème de ralentissement des performances de Trident lors de la création de plus de 32 snapshots sur un volume SolidFire .
- Remplacement de SHA-1 par SHA-256 lors de la création du certificat SSL.
- Correction du pilote Azure NetApp Files pour autoriser les noms de ressources en double et limiter les opérations à un seul emplacement.
- Correction du pilote Azure NetApp Files pour autoriser les noms de ressources en double et limiter les opérations à un seul emplacement.

Améliorations

- Améliorations de Kubernetes :
 - Ajout de la prise en charge de Kubernetes 1.23.
 - Ajouter des options de planification pour les pods Trident lors de leur installation via Trident Operator ou Helm. ("[Problème GitHub n° 651](#)")
- Autoriser les volumes interrégionaux dans le pilote GCP. ("[Problème GitHub n° 633](#)")
- Ajout de la prise en charge de l'option « unixPermissions » aux volumes Azure NetApp Files . ("[Problème GitHub n° 666](#)")

Dépréciations

L'interface REST de Trident peut écouter et répondre uniquement aux adresses 127.0.0.1 ou [::1].

Modifications dans la version 21.10.1



La version 21.10.0 présente un problème qui peut mettre le contrôleur Trident dans un état CrashLoopBackOff lorsqu'un nœud est supprimé puis rajouté au cluster Kubernetes. Ce problème est résolu dans la version 21.10.1 (problème GitHub 669).

Corrections

- Correction d'un problème de concurrence potentiel lors de l'importation d'un volume sur un backend CVS GCP, pouvant entraîner un échec d'importation.
- Correction d'un problème qui pouvait mettre le contrôleur Trident dans un état CrashLoopBackOff lorsqu'un nœud était supprimé puis rajouté au cluster Kubernetes (problème GitHub 669).

- Correction d'un problème où les SVM n'étaient plus détectés si aucun nom de SVM n'était spécifié (problème GitHub 612).

Modifications dans la version 21.10.0

Corrections

- Correction d'un problème où les clones de volumes XFS ne pouvaient pas être montés sur le même nœud que le volume source (problème GitHub 514).
- Correction d'un problème où Trident enregistrait une erreur fatale lors de l'arrêt (problème GitHub 597).
- Correctifs liés à Kubernetes :
 - Renvoyer l'espace utilisé d'un volume comme taille minimale de restauration lors de la création d'instantanés avec `ontap-nas` et `ontap-nas-flexgroup` pilotes (problème GitHub 645).
 - Problème résolu. `Failed to expand filesystem` Une erreur a été enregistrée après le redimensionnement du volume (problème GitHub 560).
 - Correction d'un problème où une capsule pouvait se bloquer. `Terminating` état (problème GitHub 572).
 - Correction du cas où un `ontap-san-economy` FlexVol pourrait être rempli de LUN instantanés (problème GitHub 533).
 - Correction d'un problème d'installation YAML personnalisé avec une image différente (problème GitHub 613).
 - Correction du calcul de la taille des instantanés (problème GitHub 611).
 - Correction d'un problème où tous les installateurs Trident pouvaient identifier Kubernetes simple comme OpenShift (problème GitHub 639).
 - Correction de l'opérateur Trident pour arrêter la réconciliation si le serveur API Kubernetes est inaccessible (problème GitHub 599).

Améliorations

- Prise en charge supplémentaire pour `unixPermissions` option pour les volumes de performance GCP-CVS.
- Ajout de la prise en charge des volumes CVS optimisés pour l'échelle dans GCP, dans la plage de 600 Gio à 1 Tio.
- Améliorations liées à Kubernetes :
 - Ajout de la prise en charge de Kubernetes 1.22.
 - Activation de l'opérateur Trident et du graphique Helm pour fonctionner avec Kubernetes 1.22 (problème GitHub 628).
 - Ajout d'une image de l'opérateur à `tridentctl` commande images (problème GitHub 570).

Améliorations expérimentales

- Ajout de la prise en charge de la réplication de volumes dans le `ontap-san` conducteur.
- Ajout de la prise en charge REST (aperçu technique) pour `ontap-nas-flexgroup`, `ontap-san`, et `ontap-nas-economy` conducteurs.

Problèmes connus

Les problèmes connus recensent les problèmes susceptibles de vous empêcher d'utiliser le produit correctement.

- Lors de la mise à niveau d'un cluster Kubernetes de la version 1.24 à la version 1.25 ou ultérieure sur lequel Trident est installé, vous devez mettre à jour le fichier `values.yaml` pour configurer `excludePodSecurityPolicy` à `true` ou ajouter `--set excludePodSecurityPolicy=true` au `helm upgrade` commande avant de pouvoir mettre à niveau le cluster.
- Trident applique désormais un blanc `fsType` (`fsType=""`) pour les volumes qui n'ont pas le `fsType` spécifié dans leur `StorageClass`. Lors de l'utilisation de Kubernetes 1.17 ou version ultérieure, Trident prend en charge la fourniture d'un fichier vide `fsType` pour les volumes NFS. Pour les volumes iSCSI, vous devez configurer le `fsType` sur votre `StorageClass` lors de l'application d'une `fsGroup` utilisation d'un contexte de sécurité.
- Lorsqu'un même serveur est utilisé sur plusieurs instances Trident, chaque fichier de configuration doit avoir une configuration différente. `storagePrefix` valeur pour les backends ONTAP ou utiliser une valeur différente `TenantName` pour les backends SolidFire. Trident ne peut pas détecter les volumes créés par d'autres instances de Trident. La tentative de création d'un volume existant sur les backends ONTAP ou SolidFire réussit, car Trident traite la création de volume comme une opération idempotente. Si `storagePrefix` ou `TenantName` ne diffèrent pas, il pourrait y avoir des conflits de noms pour les volumes créés sur le même système dorsal.
- Lors de l'installation de Trident (en utilisant `tridentctl` ou l'opérateur Trident) et en utilisant `tridentctl` Pour gérer Trident, vous devez vous assurer que `KUBECONFIG` La variable d'environnement est définie. Ceci est nécessaire pour indiquer le cluster Kubernetes qui `tridentctl` devrait fonctionner contre. Lorsque vous travaillez avec plusieurs environnements Kubernetes, vous devez vous assurer que `KUBECONFIG` Le fichier est sourcé avec précision.
- Pour effectuer une récupération d'espace en ligne pour les volumes persistants iSCSI, le système d'exploitation sous-jacent du nœud de travail peut nécessiter que des options de montage soient transmises au volume. Cela est vrai pour les instances RHEL/Red Hat Enterprise Linux CoreOS (RHCOS), qui nécessitent `discard` "option de montage"; assurez-vous que l'option de montage de rejet est incluse dans votre `[StorageClass ^]` pour prendre en charge la suppression des blocs en ligne.
- Si vous avez plusieurs instances de Trident par cluster Kubernetes, Trident ne peut pas communiquer avec les autres instances et ne peut pas découvrir les autres volumes qu'elles ont créés, ce qui entraîne un comportement inattendu et incorrect si plusieurs instances s'exécutent au sein d'un cluster. Il ne devrait y avoir qu'une seule instance de Trident par cluster Kubernetes.
- Si basé sur Trident `StorageClass` Lorsque Trident est hors ligne, les objets sont supprimés de Kubernetes, mais Trident ne supprime pas les classes de stockage correspondantes de sa base de données lorsqu'il est remis en ligne. Vous devriez supprimer ces classes de stockage en utilisant `tridentctl` ou l'API REST.
- Si un utilisateur supprime un PV provisionné par Trident avant de supprimer le PVC correspondant, Trident ne supprime pas automatiquement le volume sous-jacent. Vous devriez supprimer le volume via `tridentctl` ou l'API REST.
- ONTAP ne peut pas provisionner simultanément plus d'un FlexGroup à la fois, sauf si l'ensemble des agrégats est unique à chaque demande de provisionnement.
- Lorsque vous utilisez Trident sur IPv6, vous devez spécifier `managementLIF` et `dataLIF` dans la définition du backend entre crochets. Par exemple, `[fd20:8b1e:b258:2000:f816:3eff:feec:0]`.



Vous ne pouvez pas spécifier `dataLIF` sur un système dorsal SAN ONTAP . Trident détecte toutes les interfaces logiques iSCSI disponibles et les utilise pour établir la session multipath.

- Si vous utilisez le `solidfire-san` pilote avec OpenShift 4.5, assurez-vous que les nœuds de travail sous-jacents utilisent MD5 comme algorithme d'authentification CHAP. Les algorithmes CHAP sécurisés conformes à la norme FIPS SHA1, SHA-256 et SHA3-256 sont disponibles avec Element 12.7.

Trouver plus d'informations

- ["Trident GitHub"](#)
- ["Blogs de Trident"](#)

Versions antérieures de la documentation

Si vous n'utilisez pas Trident 25.06, la documentation des versions précédentes est disponible. ["cycle de vie du support Trident"](#) .

- ["Trident 25.02"](#)
- ["Trident 24.10"](#)
- ["Trident 24.06"](#)
- ["Trident 24.02"](#)
- ["Trident 23.10"](#)
- ["Trident 23.07"](#)
- ["Trident 23.04"](#)
- ["Trident 23.01"](#)
- ["Trident 22.10"](#)

Problèmes connus

Les problèmes connus identifient les problèmes qui pourraient vous empêcher d'utiliser cette version du produit avec succès.

Les problèmes connus suivants affectent la version actuelle :

La restauration des sauvegardes Restic de fichiers volumineux peut échouer.

Lors de la restauration de fichiers de 30 Go ou plus à partir d'une sauvegarde Amazon S3 effectuée à l'aide de Restic, l'opération de restauration peut échouer. En guise de solution de contournement, sauvegardez les données en utilisant Kopia comme outil de transfert de données (Kopia est l'outil de transfert de données par défaut pour les sauvegardes). Se référer à ["Protégez les applications à l'aide de Trident Protect"](#) pour les instructions.

Commencer

Découvrez Trident

Découvrez Trident

Trident est un projet open source entièrement pris en charge et maintenu par NetApp. Il a été conçu pour vous aider à répondre aux exigences de persistance de votre application conteneurisée en utilisant des interfaces standard de l'industrie, telles que l'interface de stockage de conteneurs (CSI).

Qu'est-ce que Trident?

NetApp Trident permet la consommation et la gestion des ressources de stockage sur toutes les plateformes de stockage NetApp populaires, dans le cloud public ou sur site, y compris les clusters ONTAP sur site (AFF, FAS et ASA), ONTAP Select, Cloud Volumes ONTAP, le logiciel Element (NetApp HCI, SolidFire), Azure NetApp Files, Amazon FSx for NetApp ONTAP et Cloud Volumes Service sur Google Cloud.

Trident est un orchestrateur de stockage dynamique conforme à l'interface de stockage de conteneurs (CSI) qui s'intègre nativement avec ["Kubernetes"](#). Trident s'exécute sous la forme d'un seul pod de contrôle et d'un pod de nœud sur chaque nœud de travail du cluster. Se référer à ["Architecture Trident"](#) pour plus de détails.

Trident assure également une intégration directe avec l'écosystème Docker pour les plateformes de stockage NetApp. Le plugin NetApp Docker Volume (nDVP) prend en charge le provisionnement et la gestion des ressources de stockage de la plateforme de stockage vers les hôtes Docker. Se référer à ["Déploiement de Trident pour Docker"](#) pour plus de détails.



Si vous utilisez Kubernetes pour la première fois, vous devriez vous familiariser avec...["Concepts et outils Kubernetes"](#).

Intégration de Kubernetes avec les produits NetApp

La gamme de produits de stockage NetApp s'intègre à de nombreux aspects d'un cluster Kubernetes, offrant des capacités avancées de gestion des données qui améliorent la fonctionnalité, les capacités, les performances et la disponibilité du déploiement Kubernetes.

Amazon FSx for NetApp ONTAP

["Amazon FSx for NetApp ONTAP"](#) est un service AWS entièrement géré qui vous permet de lancer et d'exécuter des systèmes de fichiers alimentés par le système d'exploitation de stockage NetApp ONTAP.

Azure NetApp Files

["Azure NetApp Files"](#) est un service de partage de fichiers Azure de niveau entreprise, optimisé par NetApp. Vous pouvez exécuter vos charges de travail basées sur des fichiers les plus exigeantes dans Azure de manière native, avec les performances et la gestion des données riches que vous attendez de NetApp.

Cloud Volumes ONTAP

"[Cloud Volumes ONTAP](#)" est un dispositif de stockage exclusivement logiciel qui exécute le logiciel de gestion de données ONTAP dans le cloud.

Google Cloud NetApp Volumes

"[Google Cloud NetApp Volumes](#)" est un service de stockage de fichiers entièrement géré sur Google Cloud qui offre un stockage de fichiers haute performance de niveau entreprise.

Logiciel Element

"[Élément](#)" permet à l'administrateur de stockage de consolider les charges de travail en garantissant les performances et en permettant une empreinte de stockage simplifiée et rationalisée.

NetApp HCI

"[NetApp HCI](#)" simplifie la gestion et l'évolution du centre de données en automatisant les tâches de routine et en permettant aux administrateurs d'infrastructure de se concentrer sur des fonctions plus importantes.

Trident peut provisionner et gérer des périphériques de stockage pour les applications conteneurisées directement sur la plateforme de stockage NetApp HCI sous-jacente.

NetApp ONTAP

"[NetApp ONTAP](#)" NetApp est un système d'exploitation de stockage multiprotocole et unifié qui offre des fonctionnalités avancées de gestion des données pour toutes les applications.

Les systèmes ONTAP proposent des configurations entièrement flash, hybrides ou entièrement HDD et offrent de nombreux modèles de déploiement différents : clusters FAS, AFA et ASA sur site, ONTAP Select et Cloud Volumes ONTAP. Trident prend en charge ces modèles de déploiement ONTAP .

Architecture Trident

Trident s'exécute sous la forme d'un seul pod de contrôleur et d'un pod de nœud sur chaque nœud de travail du cluster. Le pod du nœud doit être en cours d'exécution sur tout hôte sur lequel vous souhaitez potentiellement monter un volume Trident .

Comprendre les pods de contrôleur et les pods de nœud

Trident se déploie en une seule unité [Module de commande Trident](#) et un ou plusieurs [Modules de nœuds Trident](#) sur le cluster Kubernetes et utilise des conteneurs [CSI Sidecar](#) standard de Kubernetes pour simplifier le déploiement des plug-ins CSI. "[Conteneurs sidecar CSI Kubernetes](#)" sont maintenus par la communauté Kubernetes Storage.

Kubernetes "[sélecteurs de nœuds](#)" et "[tolérances et contaminations](#)" sont utilisées pour contraindre un pod à s'exécuter sur un nœud spécifique ou préféré. Vous pouvez configurer les sélecteurs de nœuds et les

tolérances pour les pods de contrôleur et de nœud lors de l'installation de Trident .

- Le plugin de contrôleur gère le provisionnement et la gestion des volumes, tels que les instantanés et le redimensionnement.
- Le plugin de nœud gère la connexion du stockage au nœud.

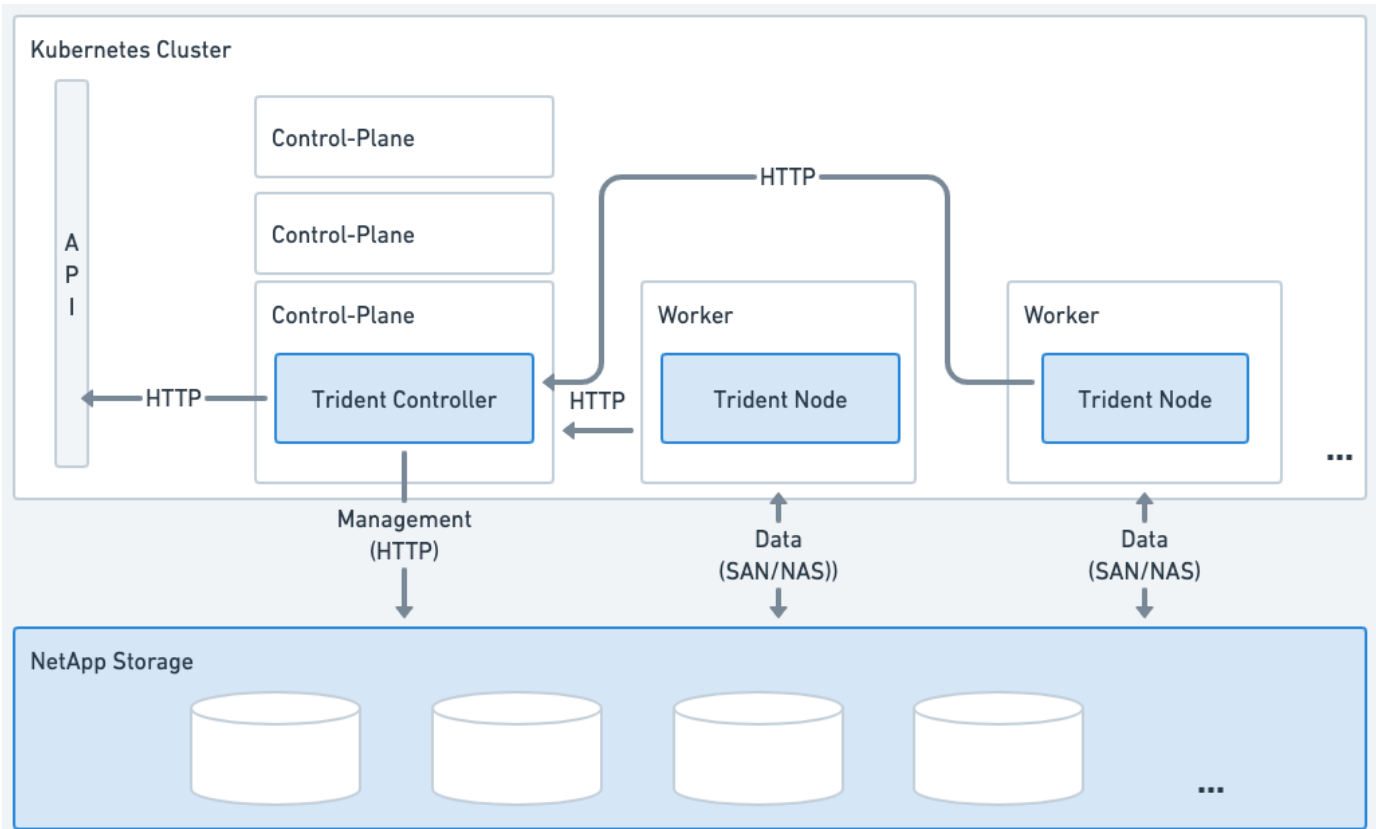


Figure 1. Trident déployé sur le cluster Kubernetes

Module de commande Trident

Le Trident Controller Pod est un pod unique exécutant le plugin CSI Controller.

- Responsable de la mise en service et de la gestion des volumes dans le stockage NetApp
- Géré par un déploiement Kubernetes
- Peut s'exécuter sur le plan de contrôle ou sur les nœuds de travail, selon les paramètres d'installation.

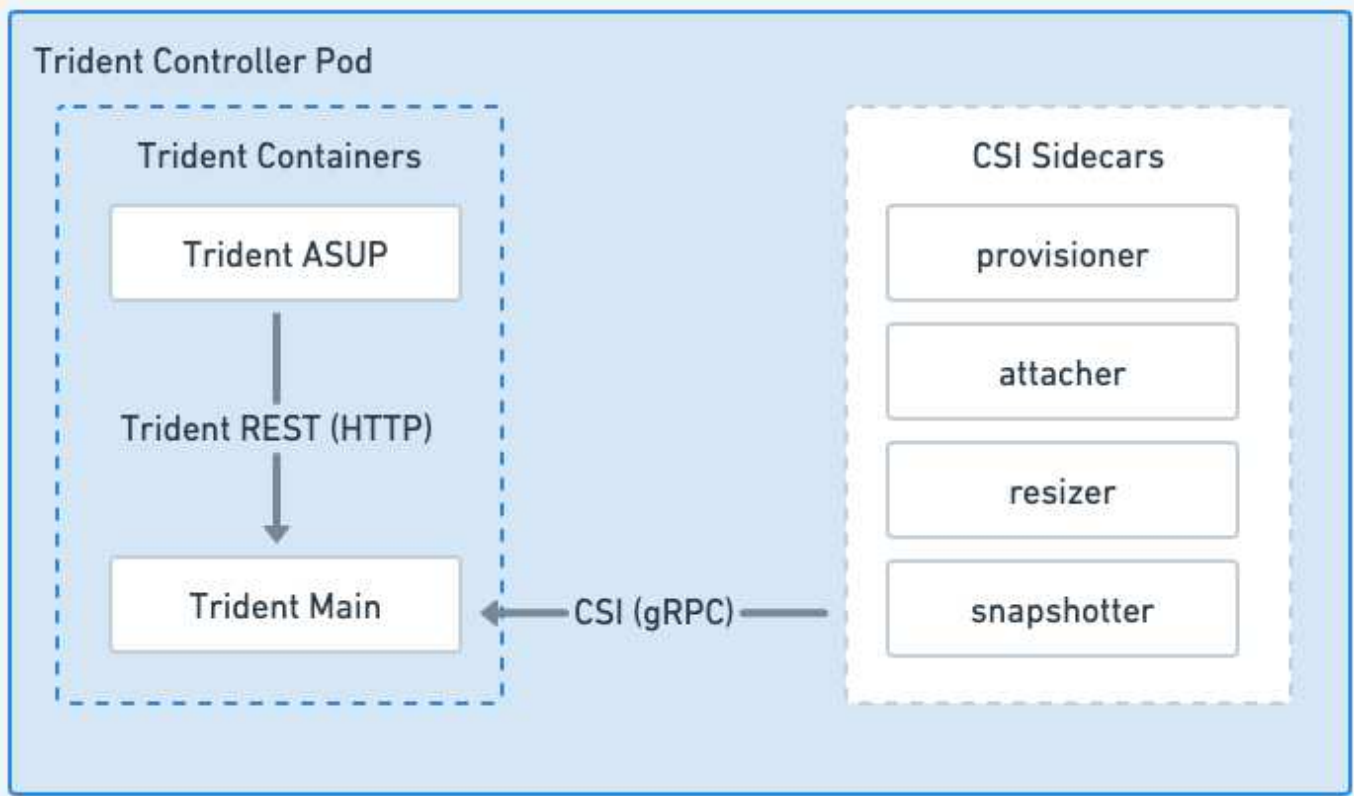


Figure 2. Schéma du module de commande Trident

Modules de nœuds Trident

Les pods Trident Node sont des pods privilégiés exécutant le plugin CSI Node.

- Responsable du montage et du démontage du stockage pour les Pods exécutés sur l'hôte
- Géré par un DaemonSet Kubernetes
- Doit être exécuté sur n'importe quel nœud capable de monter le stockage NetApp.

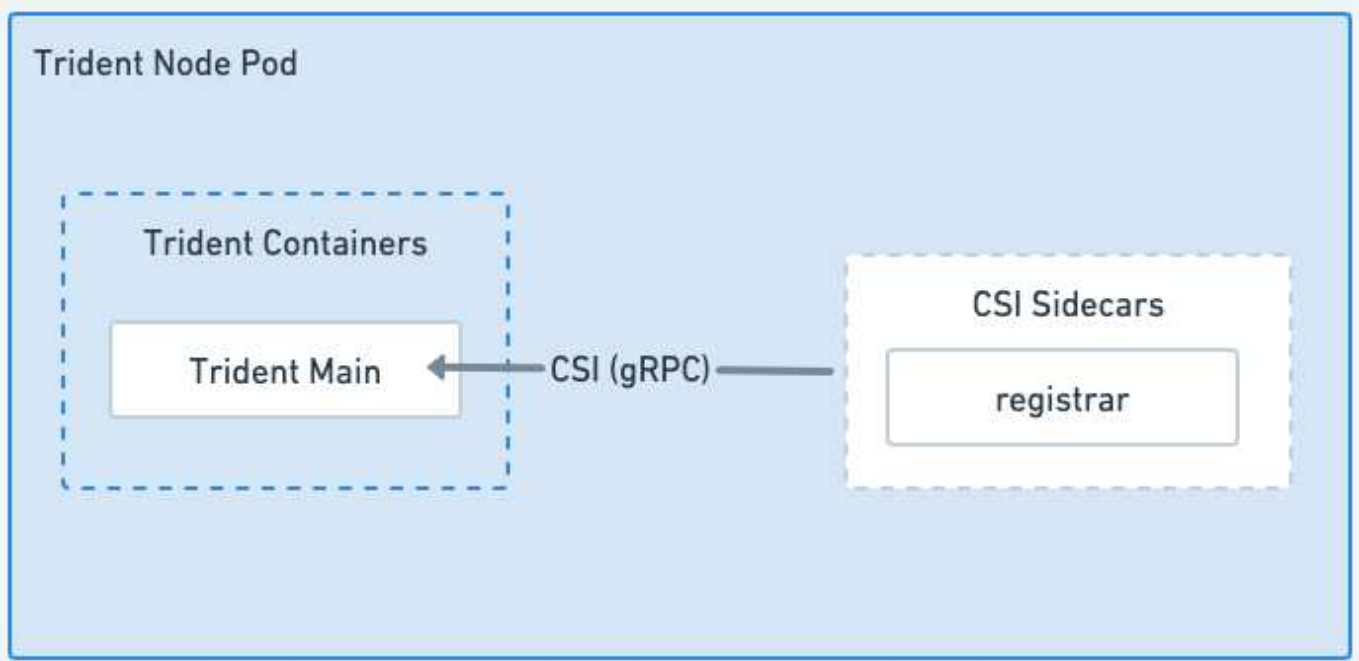


Figure 3. Diagramme du nœud Trident

Architectures de cluster Kubernetes prises en charge

Trident est compatible avec les architectures Kubernetes suivantes :

Architectures de clusters Kubernetes	Soutenu	Installation par défaut
Maître unique, calculer	Oui	Oui
Plusieurs maîtres, calcul	Oui	Oui
Maître, etcd , calculer	Oui	Oui
Maître, infrastructure, calcul	Oui	Oui

Concepts

Provisionnement

L'approvisionnement dans Trident comporte deux phases principales. La première phase associe une classe de stockage à l'ensemble des pools de stockage backend appropriés et intervient comme préparation nécessaire avant le provisionnement. La deuxième phase comprend la création du volume proprement dite et nécessite de choisir un pool de stockage parmi ceux associés à la classe de stockage du volume en attente.

Association de classe de stockage

L'association de pools de stockage backend à une classe de stockage dépend à la fois des attributs demandés par la classe de stockage et de ses `storagePools` , `additionalStoragePools` , et `excludeStoragePools` listes. Lorsque vous créez une classe de stockage, Trident compare les attributs et

les pools proposés par chacun de ses backends à ceux demandés par la classe de stockage. Si les attributs et le nom d'un pool de stockage correspondent à tous les attributs et noms de pool demandés, Trident ajoute ce pool de stockage à l'ensemble des pools de stockage adaptés à cette classe de stockage. De plus, Trident ajoute tous les pools de stockage répertoriés dans le `additionalStoragePools` ajouter à cette liste, même si leurs attributs ne répondent pas à la totalité ou à aucune des exigences de la classe de stockage. Vous devriez utiliser le `excludeStoragePools` liste permettant de remplacer et de supprimer des pools de stockage pour une classe de stockage. Trident effectue un processus similaire à chaque fois que vous ajoutez un nouveau backend, en vérifiant si ses pools de stockage correspondent à ceux des classes de stockage existantes et en supprimant ceux qui ont été marqués comme exclus.

Création de volume

Trident utilise ensuite les associations entre les classes de stockage et les pools de stockage pour déterminer où provisionner les volumes. Lorsque vous créez un volume, Trident récupère d'abord l'ensemble des pools de stockage correspondant à la classe de stockage de ce volume, puis, si vous spécifiez un protocole pour le volume, Trident supprime les pools de stockage qui ne peuvent pas fournir le protocole demandé (par exemple, un backend NetApp HCI/ SolidFire ne peut pas fournir un volume basé sur des fichiers, tandis qu'un backend ONTAP NAS ne peut pas fournir un volume basé sur des blocs). Trident randomise l'ordre de cet ensemble résultant, afin de faciliter une répartition uniforme des volumes, puis le parcourt en tentant de provisionner le volume sur chaque pool de stockage à tour de rôle. Si elle réussit une opération, elle se termine avec succès, en consignait toutes les erreurs rencontrées au cours du processus. Trident renvoie une erreur **uniquement si** il ne parvient pas à provisionner sur **tous** les pools de stockage disponibles pour la classe de stockage et le protocole demandés.

Instantanés de volume

Découvrez comment Trident gère la création d'instantanés de volume pour ses pilotes.

Découvrez la création d'instantanés de volume

- Pour le `ontap-nas`, `ontap-san`, `gcp-cvs`, et `azure-netapp-files` pilotes, chaque volume persistant (PV) est mappé à un FlexVol volume. Par conséquent, les instantanés de volume sont créés en tant qu'instantanés NetApp. La technologie de snapshots de NetApp offre une stabilité, une évolutivité, une capacité de récupération et des performances supérieures aux technologies de snapshots concurrentes. Ces copies instantanées sont extrêmement efficaces, tant en termes de temps de création que d'espace de stockage.
- Pour le `ontap-nas-flexgroup` pilote, chaque volume persistant (PV) correspond à un FlexGroup. Par conséquent, les instantanés de volume sont créés en tant qu'instantanés NetApp FlexGroup. La technologie de snapshots de NetApp offre une stabilité, une évolutivité, une capacité de récupération et des performances supérieures aux technologies de snapshots concurrentes. Ces copies instantanées sont extrêmement efficaces, tant en termes de temps de création que d'espace de stockage.
- Pour le `ontap-san-economy` Le pilote, les PV correspondent aux LUN créés sur des volumes FlexVol partagés. Les instantanés de volume des PV sont obtenus en effectuant des FlexClones du LUN associé. La technologie ONTAP FlexClone permet de créer des copies des ensembles de données les plus volumineux quasiment instantanément. Les copies partagent des blocs de données avec leurs parents, ne consommant aucun espace de stockage autre que celui nécessaire aux métadonnées.
- Pour le `solidfire-san` Chaque pilote PV correspond à un LUN créé sur le logiciel NetApp Element /cluster NetApp HCI. Les instantanés de volume sont représentés par des instantanés d'éléments du LUN sous-jacent. Ces instantanés sont des copies à un instant précis et n'occupent qu'une petite quantité de ressources système et d'espace.
- Lorsque vous travaillez avec le `ontap-nas` et `ontap-san` Les pilotes, les instantanés ONTAP sont des copies ponctuelles du FlexVol et consomment de l'espace sur le FlexVol lui-même. Cela peut entraîner une

réduction de l'espace inscriptible disponible dans le volume au fil du temps, à mesure que des instantanés sont créés/planifiés. Une solution simple consiste à augmenter le volume en le redimensionnant via Kubernetes. Une autre option consiste à supprimer les instantanés qui ne sont plus nécessaires. Lorsqu'un VolumeSnapshot créé via Kubernetes est supprimé, Trident supprime également le snapshot ONTAP associé. Les instantanés ONTAP qui n'ont pas été créés via Kubernetes peuvent également être supprimés.

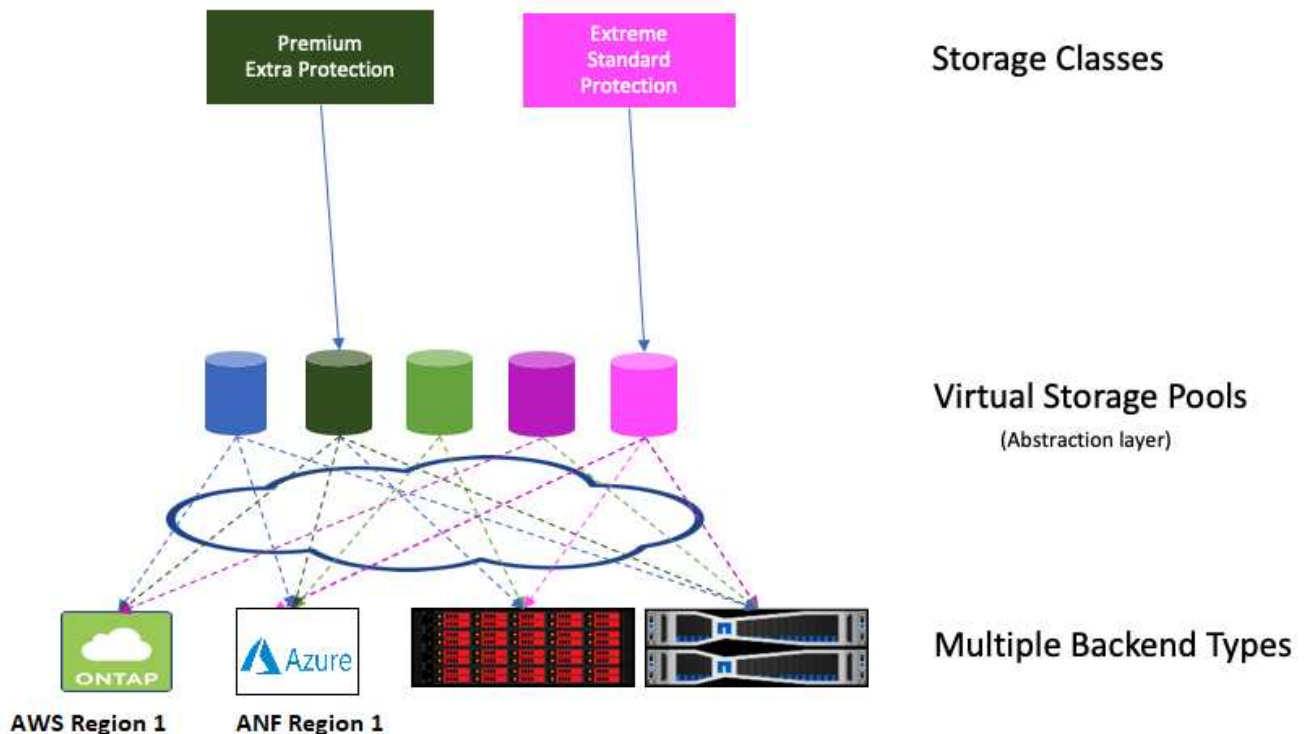
Avec Trident, vous pouvez utiliser VolumeSnapshots pour créer de nouveaux PV à partir de ceux-ci. La création de volumes persistants à partir de ces instantanés est réalisée à l'aide de la technologie FlexClone pour les systèmes backend ONTAP et CVS pris en charge. Lors de la création d'un PV à partir d'un instantané, le volume de support est un FlexClone du volume parent de l'instantané. Le `solidfire-san` Le pilote utilise les clones de volumes logiciels Element pour créer des PV à partir d'instantanés. Ici, il crée un clone à partir de l'instantané de l'élément.

Piscines virtuelles

Les pools virtuels fournissent une couche d'abstraction entre les backends de stockage Trident et Kubernetes. `StorageClasses`. Ils permettent à un administrateur de définir des aspects tels que l'emplacement, les performances et la protection de chaque serveur dorsal de manière commune et indépendante du serveur dorsal, sans avoir à effectuer de configuration spécifique. `StorageClass` Spécifiez le backend physique, le pool de backends ou le type de backend à utiliser pour répondre aux critères souhaités.

Découvrez les piscines virtuelles

L'administrateur de stockage peut définir des pools virtuels sur n'importe quel backend Trident dans un fichier de définition JSON ou YAML.



Tout aspect spécifié en dehors de la liste des pools virtuels est global au backend et s'appliquera à tous les pools virtuels, tandis que chaque pool virtuel peut spécifier un ou plusieurs aspects individuellement (remplaçant tout aspect global au backend).



- Lors de la définition des pools virtuels, ne tentez pas de réorganiser l'ordre des pools virtuels existants dans une définition de backend.
- Nous déconseillons de modifier les attributs d'une piscine virtuelle existante. Vous devez définir un nouveau pool virtuel pour effectuer les modifications.

La plupart des aspects sont spécifiés en termes propres au backend. Il est crucial de noter que les valeurs d'aspect ne sont pas exposées en dehors du pilote du backend et ne sont pas disponibles pour la mise en correspondance dans `StorageClasses`. L'administrateur définit plutôt une ou plusieurs étiquettes pour chaque pool virtuel. Chaque étiquette est une paire clé:valeur, et les étiquettes peuvent être communes à plusieurs systèmes backend différents. Comme les aspects, les étiquettes peuvent être spécifiées par pool ou globalement au niveau du backend. Contrairement aux aspects, qui ont des noms et des valeurs prédéfinis, l'administrateur a toute latitude pour définir les clés et les valeurs des étiquettes selon ses besoins. Pour plus de commodité, les administrateurs de stockage peuvent définir des étiquettes par pool virtuel et regrouper les volumes par étiquette.

Les étiquettes des pools virtuels peuvent être définies à l'aide de ces caractères :

- lettres majuscules A-Z
- lettres minuscules a-z
- Nombres 0-9
- souligne _
- traits d'union -

UN `StorageClass` identifie le pool virtuel à utiliser en faisant référence aux étiquettes d'un paramètre de sélection. Les sélecteurs de piscine virtuelle prennent en charge les opérateurs suivants :

Opérateur	Exemple	La valeur de l'étiquette d'un pool doit :
=	performance=premium	Correspondre
!=	performance ≠ extrême	Ne correspond pas
in	emplacement en (est, ouest)	Appartenir à l'ensemble des valeurs
notin	notation de performance (argent, bronze)	Ne pas faire partie de l'ensemble des valeurs
<key>	protection	Exister avec n'importe quelle valeur
!<key>	!protection	N'existe pas

groupes d'accès au volume

Apprenez-en davantage sur la façon dont Trident utilise ["groupes d'accès au volume"](#) .



Ignorez cette section si vous utilisez CHAP, ce qui est recommandé pour simplifier la gestion et éviter la limite de mise à l'échelle décrite ci-dessous. De plus, si vous utilisez Trident en mode CSI, vous pouvez ignorer cette section. Trident utilise CHAP lorsqu'il est installé en tant que fournisseur CSI amélioré.

Découvrez les groupes d'accès au volume

Trident peut utiliser des groupes d'accès aux volumes pour contrôler l'accès aux volumes qu'il provisionne. Si CHAP est désactivé, il s'attend à trouver un groupe d'accès appelé `trident` sauf si vous spécifiez un ou plusieurs ID de groupe d'accès dans la configuration.

Bien que Trident associe les nouveaux volumes aux groupes d'accès configurés, il ne crée ni ne gère les groupes d'accès eux-mêmes. Les groupes d'accès doivent exister avant que le système de stockage dorsal ne soit ajouté à Trident, et ils doivent contenir les IQN iSCSI de chaque nœud du cluster Kubernetes qui pourrait potentiellement monter les volumes provisionnés par ce système dorsal. Dans la plupart des installations, cela inclut chaque nœud de travail du cluster.

Pour les clusters Kubernetes comportant plus de 64 nœuds, il est recommandé d'utiliser plusieurs groupes d'accès. Chaque groupe d'accès peut contenir jusqu'à 64 IQN, et chaque volume peut appartenir à quatre groupes d'accès. Avec un maximum de quatre groupes d'accès configurés, n'importe quel nœud d'un cluster pouvant compter jusqu'à 256 nœuds pourra accéder à n'importe quel volume. Pour connaître les dernières limites applicables aux groupes d'accès au volume, veuillez consulter : ["ici"](#) .

Si vous modifiez la configuration à partir d'une configuration utilisant la configuration par défaut `trident` groupe d'accès à un autre qui utilise également d'autres groupes, incluez l'ID du groupe d'accès `trident` groupe d'accès dans la liste.

Démarrage rapide pour Trident

Vous pouvez installer Trident et commencer à gérer les ressources de stockage en quelques étapes. Avant de commencer, consultez ["exigences de Trident"](#) .



Pour Docker, reportez-vous à ["Trident pour Docker"](#) .

1

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.

["Préparer le nœud de travail"](#)

2

Installer Trident

Trident propose plusieurs méthodes et modes d'installation optimisés pour divers environnements et organisations.

["Installer Trident"](#)

3

Créer un backend

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.

["Configurer un backend"](#) pour votre système de stockage



Créer une classe de stockage Kubernetes

L'objet `StorageClass` de Kubernetes spécifie Trident comme provisionneur et vous permet de créer une classe de stockage pour provisionner des volumes avec des attributs personnalisables. Trident crée une classe de stockage correspondante pour les objets Kubernetes qui spécifient le provisionneur Trident .

["Créer une classe de stockage"](#)



Provisionnez un volume

Un *PersistentVolume* (PV) est une ressource de stockage physique provisionnée par l'administrateur du cluster sur un cluster Kubernetes. Le *PersistentVolumeClaim* (PVC) est une demande d'accès au `PersistentVolume` sur le cluster.

Créez un `PersistentVolume` (PV) et un `PersistentVolumeClaim` (PVC) qui utilise la classe de stockage Kubernetes configurée pour demander l'accès au PV. Vous pouvez ensuite monter le panneau photovoltaïque sur un support.

["Provisionnez un volume"](#)

Quelle est la prochaine étape ?

Vous pouvez désormais ajouter des serveurs backend supplémentaires, gérer les classes de stockage, gérer les serveurs backend et effectuer des opérations sur les volumes.

Exigences

Avant d'installer Trident , veuillez consulter la configuration système générale requise. Certains systèmes d'arrière-plan peuvent avoir des exigences supplémentaires.

Informations essentielles sur Trident

Vous devez lire les informations essentielles suivantes concernant Trident.

Informations essentielles concernant Trident

- Kubernetes 1.34 est désormais pris en charge dans Trident. Mettez à niveau Trident avant de mettre à niveau Kubernetes.
- Trident impose strictement l'utilisation de la configuration multipathing dans les environnements SAN, avec une valeur recommandée de `find_multipaths: no` dans le fichier `multipath.conf`.

Utilisation d'une configuration sans chemins multiples ou utilisation de `find_multipaths: yes` ou `find_multipaths: smart` La valeur dans le fichier `multipath.conf` entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version du 21 juillet.

Interfaces prises en charge (orchestrateurs)

Trident prend en charge plusieurs moteurs de conteneurs et orchestrateurs, notamment les suivants :

- Anthos On-Prem (VMware) et Anthos sur bare metal 1.16
- Kubernetes 1.27 - 1.34
- OpenShift 4.12, 4.14 - 4.19 (Si vous prévoyez d'utiliser la préparation de nœuds iSCSI avec OpenShift 4.19, la version Trident minimale prise en charge est 25.06.1.)



Trident continue de prendre en charge les anciennes versions d'OpenShift conformément aux directives. "[Cycle de vie des versions de Red Hat Extended Update Support \(EUS\)](#)", même si elles reposent sur des versions de Kubernetes qui ne sont plus officiellement prises en charge par le système en amont. Lors de l'installation de Trident dans de tels cas, vous pouvez ignorer en toute sécurité tous les messages d'avertissement concernant la version de Kubernetes.

- Rancher Kubernetes Engine 2 (RKE2) v1.27.x - 1.34.x



Alors que Trident est pris en charge sur les versions 1.27.x à 1.34.x de Rancher Kubernetes Engine 2 (RKE2), Trident est actuellement qualifié uniquement sur RKE2 v1.28.5+rke2r1.

Trident fonctionne également avec de nombreuses autres offres Kubernetes entièrement gérées et autogérées, notamment Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Mirantis Kubernetes Engine (MKE) et VMWare Tanzu Portfolio.

Trident et ONTAP peuvent être utilisés comme fournisseur de stockage pour "[KubeVirt](#)".



Avant de mettre à niveau un cluster Kubernetes de la version 1.25 à la version 1.26 ou ultérieure sur lequel Trident est installé, veuillez consulter la documentation. "[Mettre à niveau une installation Helm](#)".

Systemes de stockage pris en charge

Pour utiliser Trident, vous avez besoin d'un ou plusieurs des serveurs dorsaux pris en charge suivants :

- Amazon FSx for NetApp ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP
- Google Cloud NetApp Volumes
- Baie NetApp All SAN (ASA)
- FAS, AFF ou ASA r2 sur site (iSCSI, NVMe/TCP et FC) exécutant des versions d'ONTAP bénéficiant d'une prise en charge complète ou limitée par NetApp. Voir "[Prise en charge des versions logicielles](#)".
- Logiciel NetApp HCI/Element version 11 ou supérieure

Prise en charge de Trident pour la virtualisation KubeVirt et OpenShift

Pilotes de stockage pris en charge :

Trident prend en charge les pilotes ONTAP suivants pour la virtualisation KubeVirt et OpenShift :

- ontap-nas
- ontap-nas-économie
- ontap-san (iSCSI, FCP, NVMe sur TCP)
- ontap-san-economy (iSCSI uniquement)

Points à prendre en considération :

- Mettez à jour la classe de stockage pour avoir le `fsType` paramètre (par exemple : `fsType: "ext4"`) dans l'environnement de virtualisation OpenShift. Si nécessaire, configurez explicitement le mode de volume sur blocage à l'aide de `volumeMode=Block` paramètre dans le `dataVolumeTemplates` notifier CDI de créer des volumes de données par blocs.
- *Mode d'accès RWX pour les pilotes de stockage par blocs* : les pilotes `ontap-san` (iSCSI, NVMe/TCP, FC) et `ontap-san-economy` (iSCSI) ne sont pris en charge qu'avec « `volumeMode : Block` » (périphérique brut). Pour ces conducteurs, le `fsType` Ce paramètre ne peut pas être utilisé car les volumes sont fournis en mode périphérique brut.
- Pour les flux de travail de migration à chaud nécessitant le mode d'accès RWX, les combinaisons suivantes sont prises en charge :
 - NFS + `volumeMode=Filesystem`
 - iSCSI + `volumeMode=Block` (dispositif brut)
 - NVMe/TCP + `volumeMode=Block` (dispositif brut)
 - FC + `volumeMode=Block` (dispositif brut)

Exigences fonctionnelles

Le tableau ci-dessous récapitule les fonctionnalités disponibles avec cette version de Trident et les versions de Kubernetes qu'elle prend en charge.

Fonctionnalité	Version de Kubernetes	Portails fonctionnels requis ?
Trident	1,27 - 1,34	Non

Fonctionnalité	Version de Kubernetes	Portails fonctionnels requis ?
Instantanés de volume	1,27 - 1,34	Non
PVC de Volume Snapshots	1,27 - 1,34	Non
redimensionnement iSCSI PV	1,27 - 1,34	Non
CHAP bidirectionnel ONTAP	1,27 - 1,34	Non
Politiques d'exportation dynamiques	1,27 - 1,34	Non
Opérateur Trident	1,27 - 1,34	Non
Topologie CSI	1,27 - 1,34	Non

Systemes d'exploitation hôtes testés

Bien que Trident ne prenne pas officiellement en charge de systèmes d'exploitation spécifiques, les suivants sont connus pour fonctionner :

- Versions de Red Hat Enterprise Linux CoreOS (RHCOS) prises en charge par OpenShift Container Platform sur AMD64 et ARM64
- Red Hat Enterprise Linux (RHEL) 8 ou version ultérieure sur AMD64 et ARM64



NVMe/TCP nécessite RHEL 9 ou une version ultérieure.

- Ubuntu 22.04 LTS ou version ultérieure sur AMD64 et ARM64
- Windows Server 2022
- SUSE Linux Enterprise Server (SLES) 15 ou version ultérieure

Par défaut, Trident s'exécute dans un conteneur et fonctionnera donc sur n'importe quel nœud de calcul Linux. Cependant, ces travailleurs doivent pouvoir monter les volumes fournis par Trident à l'aide du client NFS standard ou de l'initiateur iSCSI, selon les systèmes de stockage utilisés.

Le `tridentctl` Cet utilitaire fonctionne également sur n'importe laquelle de ces distributions Linux.

Configuration de l'hôte

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 ou NVMe en fonction du pilote que vous avez sélectionné.

["Préparer le nœud de travail"](#)

Configuration du système de stockage

Trident peut nécessiter des modifications du système de stockage avant qu'une configuration backend puisse

l'utiliser.

["Configurer les backends"](#)

Ports Trident

Trident nécessite l'accès à des ports spécifiques pour communiquer.

["Ports Trident"](#)

Images de conteneurs et versions Kubernetes correspondantes

Pour les installations isolées du réseau, la liste suivante est une référence des images de conteneurs nécessaires à l'installation de Trident. Utilisez le `tridentctl images` commande permettant de vérifier la liste des images de conteneurs nécessaires.

Images de conteneur requises pour Trident 25.06.2

versions de Kubernetes	Image du conteneur
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	<ul style="list-style-type: none">• <code>docker.io/netapp/trident:25.06.2</code>• <code>docker.io/netapp/trident-autosupport:25.06</code>• <code>registry.k8s.io/sig-storage/csi-provisioner:v5.2.0</code>• <code>registry.k8s.io/sig-storage/csi-attacher:v4.8.1</code>• <code>registry.k8s.io/sig-storage/csi-resizer:v1.13.2</code>• <code>registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1</code>• <code>registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0</code>• <code>docker.io/netapp/trident-operator:25.06.2</code> (facultatif)

Images de conteneurs requises pour Trident 25.06

versions de Kubernetes	Image du conteneur
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	<ul style="list-style-type: none">• <code>docker.io/netapp/trident:25.06.0</code>• <code>docker.io/netapp/trident-autosupport:25.06</code>• <code>registry.k8s.io/sig-storage/csi-provisioner:v5.2.0</code>• <code>registry.k8s.io/sig-storage/csi-attacher:v4.8.1</code>• <code>registry.k8s.io/sig-storage/csi-resizer:v1.13.2</code>• <code>registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1</code>• <code>registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0</code>• <code>docker.io/netapp/trident-operator:25.06.0</code> (facultatif)

Installer Trident

Installation via l'opérateur Trident

Installer à l'aide de tridentctl

Installation effectuée par un opérateur certifié OpenShift

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.

Veillez 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
```

`tridentctl`

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-xxxxxxxxxxxxx`.

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-xxxxxxxxxxxxx' # 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-
xxxxxxxxxxxxx' "
```

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-
xxxxxxxxxxxxx"
```

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-xxxxxxxxxxxx` .

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, <pre>\{"api": false, "method": true, "discovery": true\}</pre> . 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 <code>nfs</code> , <code>smb</code> ou <code>nul</code> . La valeur nulle correspond par défaut aux volumes NFS.	<code>nfs</code>
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
<code>exportRule</code>	Règles d'exportation pour les nouveaux volumes. <code>exportRule</code> 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"
<code>snapshotDir</code>	Contrôle la visibilité du répertoire <code>.snapshot</code>	« Vrai » pour NFSv4, « Faux » pour NFSv3
<code>size</code>	La taille par défaut des nouveaux volumes	"100G"
<code>unixPermissions</code>	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 souhaitez 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: anfl
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
<code>google-cloud-netapp-volumes</code>	NFS SMB	Système de fichiers	RWO, ROX, RWX, RWOP	<code>nfs</code> , <code>smb</code>

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"
```

`tridentctl`

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
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	La valeur de <code>storageDriverName</code> doit être spécifié comme « <code>google-cloud-netapp-volumes</code> ».

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 <code>.snapshot</code> 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
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq70lwWgLwGa==
    -----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
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
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	"gcp-cvs"
<code>backendName</code>	Nom personnalisé ou système de stockage	Nom du pilote + "_" + partie de la clé API
<code>storageClass</code>	Paramètre optionnel utilisé pour spécifier le type de service CVS. Utiliser <code>software</code> pour sélectionner le type de service CVS. Sinon, Trident suppose le type de service CVS-Performance(<code>hardware</code>).	
<code>storagePools</code>	Service de type CVS uniquement. Paramètre optionnel permettant de spécifier les pools de stockage pour la création de volumes.	

Paramètre	Description	Défaut
<code>projectNumber</code>	Numéro de projet du compte Google Cloud. Cette valeur se trouve sur la page d'accueil du portail Google Cloud.	
<code>hostProjectNumber</code>	Requis si vous utilisez un réseau VPC partagé. Dans ce scénario, <code>projectNumber</code> est le projet de service, et <code>hostProjectNumber</code> est le projet hôte.	
<code>apiRegion</code>	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 <code>apiRegion</code> 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.	
<code>apiKey</code>	Clé API pour le compte de service Google Cloud avec le <code>netappcloudvolumes.admin</code> 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).	
<code>proxyURL</code>	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.	
<code>nfsMountOptions</code>	Contrôle précis des options de montage NFS.	"nfsvers=3"
<code>limitVolumeSize</code>	L'approvisionnement échouera si la taille du volume demandée est supérieure à cette valeur.	"" (non appliqué par défaut)
<code>serviceLevel</code>	Le niveau de service CVS-Performance ou CVS pour les nouveaux volumes. Les valeurs CVS-Performance sont <code>standard</code> , <code>premium</code> , ou <code>extreme</code> . Les valeurs CVS sont <code>standardsw</code> ou <code>zoneredundantstandardsw</code> .	La valeur par défaut de CVS-Performance est « standard ». La valeur par défaut de CVS est « standardsw ».
<code>network</code>	Le réseau Google Cloud est utilisé pour les volumes du Cloud Volumes Service .	"défaut"
<code>debugTraceFlags</code>	Indicateurs de débogage à utiliser lors du dépannage. Exemple, <code>\{"api":false, "method":true\}</code> . 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 <code>.snapshot</code> 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 à [Définitions 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
<code>solidfire-san</code>	iSCSI	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers. Périphérique de bloc brut.
<code>solidfire-san</code>	iSCSI	Système de fichiers	RWO, RWOP	<code>xfs</code> , <code>ext3</code> , <code>ext4</code>

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
<code>version</code>		Toujours 1
<code>storageDriverName</code>	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 associée 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.	xfst, 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 à Considérations supplémentaires concernant NVMe/TCP .	Bloc	RWO, ROX, RWX, RWOP	Aucun système de fichiers ; périphérique de stockage brut
ontap-san	NVMe/TCP Se référer à Considérations 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 mappeteur 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

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

JSON

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

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
```



Après avoir exécuté cette commande, ONTAP vous invite à saisir un certificat. Collez le contenu du `k8senv.pem` fichier généré à l'étape 1, puis appuyez sur END pour terminer l'installation.

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...Vaaallllluuuueeee",
  "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
<code>version</code>		Toujours 1
<code>storageDriverName</code>	Nom du pilote de stockage	<code>ontap-san</code> ou <code>ontap-san-economy</code>
<code>backendName</code>	Nom personnalisé ou système de stockage	Nom du conducteur + "_" + <code>dataLIF</code>
<code>managementLIF</code>	<p>Adresse IP d'une interface logique de gestion de cluster ou de SVM.</p> <p>Un nom de domaine pleinement qualifié (FQDN) peut être spécifié.</p> <p>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 :</p> <pre>[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]</pre> <p>Pour une transition MetroCluster sans interruption, consultez la documentation.Exemple de MetroCluster</p>	"10.0.0.1", "[2001:1234:abcd::fefe]"

Si vous utilisez les identifiants « `vsadmin` », `managementLIF` doit être celle du SVM ; si vous utilisez les identifiants « administrateur », `managementLIF` doit être celui du groupe.

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 le Exemple de MetroCluster .	Dérivé par le SVM
svm	Machine virtuelle de stockage à utiliser Omettre pour Metrocluster. Voir le Exemple 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	<p>Agrégat pour le provisionnement (facultatif ; s'il est défini, il doit être affecté au SVM). Pour le <code>ontap-nas-flexgroup</code> 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 .</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p> 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.</p> </div> <p>Ne pas spécifier pour les systèmes ASA r2.</p>	""
limitAggregateUsage	L'approvisionnement échouera si l'utilisation dépasse ce pourcentage. Si vous utilisez un backend Amazon FSx for NetApp ONTAP , ne spécifiez pas <code>limitAggregateUsage</code> . Le fourni <code>fsxadmin</code> et <code>vsadmin</code> 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	<p>Paramètre booléen pour utiliser les API REST ONTAP .</p> <div style="border: 1px solid gray; padding: 10px; margin: 10px 0;"> <pre> `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). </pre> </div> <p><code>`useREST`</code> est entièrement compatible NVMe/TCP.</p> <div style="border: 1px solid gray; padding: 5px; margin: 10px 0;">  <p>NVMe est pris en charge uniquement avec les API REST ONTAP et n'est pas pris en charge avec ONTAPI (ZAPI).</p> </div> <p>Si spécifié, toujours définir sur <code>true</code> pour les systèmes ASA r2.</p>	true`pour ONTAP 9.15.1 ou version ultérieure, sinon `false`.

Paramètre	Description	Défaut
sanType	Utiliser pour sélectionner <code>iscsi</code> pour iSCSI, <code>nvme</code> pour NVMe/TCP ou <code>fc</code> pour SCSI sur Fibre Channel (FC).	`iscsi` si vide
formatOptions	Utiliser <code>formatOptions</code> pour spécifier les arguments de ligne de commande pour le <code>mkfs</code> commande, qui sera appliquée à chaque formatage d'un volume. Cela vous permet de formater le volume selon vos préférences. Veuillez à spécifier les options de formatage similaires à celles de la commande <code>mkfs</code> , en excluant le chemin du périphérique. Exemple : "-E nodiscard" Prise en charge pour <code>ontap-san</code> et <code>ontap-san-economy</code> 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 <code>ontap-san-economy</code> .	"" (non appliqué par défaut)
denyNewVolumePools	Restreint <code>ontap-san-economy</code> 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 <code>false</code> . 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 " .	<code>"false"</code> Si spécifié, définir sur <code>true</code> pour les systèmes ASA r2.
luksEncryption	Activer le chiffrement LUKS. Se référer à " Utiliser Linux Unified Key Setup (LUKS) " .	<code>""</code> Réglé sur <code>false</code> 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.	<code>""</code>

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. La commande `volume show` devrait afficher des résultats similaires à cet exemple :

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

3 entries were displayed.

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

Exemples de configuration 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 à [Exemples 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

```
---
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

```
{
  "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...Vaaalllluuueeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "storagePrefix": "myPrefix_"
}

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

```

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>
      fileType: 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.

- 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 <code>nfs</code> .	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.
- Veuillez 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 le Exemple 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 le Exemple 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. <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;">  <p>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.</p> </div>	"trident"
aggregate	Agrégat pour le provisionnement (facultatif ; s'il est défini, il doit être affecté au SVM). Pour le <code>ontap-nas-flexgroup</code> 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 . <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;">  <p>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.</p> </div>	""

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	<p>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.</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p> 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.</p> </div>	""
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. <code>useREST</code> Lorsqu'il est réglé sur <code>true</code> Trident utilise les API REST ONTAP pour communiquer avec le système dorsal ; lorsqu'il est configuré pour <code>false</code> 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 <code>ontapi</code> application. Ceci est satisfait par la définition prédéfinie <code>vsadmin</code> et <code>cluster-admin</code> rôles. À compter de la version Trident 24.06 et ONTAP 9.15.1 ou ultérieure, <code>useREST</code> est réglé sur <code>true</code> par défaut ; modifier <code>useREST</code> à <code>false</code> utiliser les appels ONTAPI (ZAPI).	<code>true</code> pour ONTAP 9.15.1 ou version ultérieure, sinon <code>false</code> .
limitVolumePoolSize	Taille maximale de FlexVol pouvant être demandée lors de l'utilisation de Qtrees dans le backend <code>ontap-nas-economy</code> .	"" (non appliqué par défaut)
denyNewVolumePools	Restreint <code>ontap-nas-economy</code> 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 <code>false</code> . 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 <code>.snapshot</code> 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 <code>mixed</code> et <code>unix</code> Styles de sécurité. Les PME prennent en charge <code>mixed</code> et <code>ntfs</code> Styles de sécurité.	La valeur par défaut de NFS est <code>unix</code> . La valeur par défaut de SMB est <code>ntfs</code> .
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 :

$$\text{Total volume size} = (\text{PVC requested size}) / (1 - (\text{snapshotReserve percentage}) / 100)$$

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 :

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

2 entries were displayed.

Les 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 omettez 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: mysqlldb
  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 à [Exemples 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_ST ANDARD	Oui	Oui	Oui	Oui
AL2_x86_64	Oui	Oui	Oui*	Oui*
BOTTLEROCKET_x86_64	Oui**	Oui	S/O	S/O
AL2023_ARM_64_S TANDARD	Oui	Oui	Oui	Oui
AL2_ARM_64	Oui	Oui	Oui*	Oui*

BOTTLEROCKET_A RM_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 `nodePrep` 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 **Versión** 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:xxxxxxx: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:xxxxxxx: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 f2f4c87fa629 Bound	backend-fsx-ontap-nas Success	7a551921-997c-4c37-a1d1-

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 iSCSI 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 <code>nfs</code> , <code>smb</code> , ou <code>nul</code> . Doit être réglé sur <code>smb</code> pour les volumes SMB. La valeur nulle correspond par défaut aux volumes NFS.	<code>nfs</code>
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.	<code>smb-share</code>

Paramètre	Description	Exemple
useREST	Paramètre booléen pour utiliser les API REST ONTAP . Lorsqu'il est réglé sur <code>true</code> 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 <code>ontap application</code> . Ceci est satisfait par la définition prédéfinie <code>vsadmin</code> et <code>cluster-admin</code> rôles.	<code>false</code>
aws	Vous pouvez spécifier les éléments suivants dans le fichier de configuration d'AWS FSx pour ONTAP: - <code>fsxFilesystemID</code> : Spécifiez l'ID du système de fichiers AWS FSx. - <code>apiRegion</code> : Nom de la région de l'API AWS. - <code>apikey</code> : Clé API AWS. - <code>secretKey</code> : Clé secrète AWS.	"" "" ""
credentials	Spécifiez les informations d'identification FSx SVM à stocker dans AWS Secrets Manager. - <code>name</code> : Nom de ressource Amazon (ARN) du secret, qui contient les informations d'identification de la SVM. - <code>type</code> : Définir sur <code>awsarn</code> . 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
<code>spaceAllocation</code>	Allocation d'espace pour les LUN	<code>true</code>
<code>spaceReserve</code>	Mode de réservation d'espace ; « aucun » (fin) ou « volume » (épais)	<code>none</code>
<code>snapshotPolicy</code>	Politique d'instantané à utiliser	<code>none</code>

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 terminer [Intégration 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.

- 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 <code>nfs</code> .	smb
securityStyle	Style de sécurité pour les nouveaux volumes. Doit être réglé sur <code>ntfs</code> ou <code>mixed</code> pour les volumes SMB.	<code>ntfs</code> ou <code>mixed</code> 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 un 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-ontapas.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**.

The screenshot shows the AWS EKS console interface for a cluster named 'tri-env-eks'. At the top, there are buttons for 'Delete cluster', 'Upgrade version', and 'View dashboard'. Below this, a notification bar indicates that standard support for Kubernetes version 1.30 ends on July 28, 2025, with an 'Upgrade now' button. The main content area is divided into sections: 'Cluster info' (Status: Active, Kubernetes version: 1.30, Support period: Standard support until July 28, 2025, Provider: EKS), 'Cluster health issues' (0 issues), and 'Upgrade insights' (0 insights). A navigation bar below these sections includes 'Overview', 'Resources', 'Compute', 'Networking', 'Add-ons' (selected, with a '1' badge), 'Access', 'Observability', 'Update history', and 'Tags'. A notification bar below the navigation bar states 'New versions are available for 1 add-on.' The 'Add-ons' section shows 'Add-ons (3)' with buttons for 'View details', 'Edit', and 'Remove', and a 'Get more add-ons' button. A search bar is present with the text 'Find add-on' and filters for 'Any categ...', 'Any status', and '3 matches'.

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

AWS Marketplace add-ons (1) ↻

Discover, subscribe to and configure EKS add-ons to enhance your EKS clusters.

Filtering options

Any category ▾ NetApp, Inc. ▾ Any pricing model ▾ [Clear filters](#)

NetApp, Inc. ✕ < 1 >

NetApp **NetApp Trident** ☐

NetApp Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Standard Contract

Category storage	Listed by NetApp, Inc.	Supported versions 1.31, 1.30, 1.29, 1.28, 1.27, 1.26, 1.25, 1.24, 1.23	Pricing starting at View pricing details
----------------------------	--	---	--

[Cancel](#) [Next](#)

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

Configure selected add-ons settings
Configure the add-ons for your cluster by selecting settings.

NetApp Trident [Remove add-on](#)

Listed by NetApp	Category storage	Status 🟢 Ready to install
-----------------------------------	----------------------------	-------------------------------------

You're subscribed to this software [View subscription](#) ✕

You can view the terms and pricing details for this product or choose another offer if one is available.

Version
Select the version for this add-on.

▶ **Optional configuration settings**

[Cancel](#) [Previous](#) [Next](#)

- Configurez les paramètres du module complémentaire requis.

Review and add

Step 1: Select add-ons

[Edit](#)

Selected add-ons (1)

Find add-on

Add-on name	Type	Status
netapp_trident-operator	storage	Ready to install

Step 2: Configure selected add-ons settings

[Edit](#)

Selected add-ons version (1)

Add-on name	Version	IAM role for service account (IRSA)
netapp_trident-operator	v24.10.0-eksbuild.1	Not set

EKS Pod Identity (0)

Add-on name	IAM role	Service account
No Pod Identity associations None of the selected add-on(s) have Pod Identity associations.		

[Cancel](#)[Previous](#)[Create](#)

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

Add-ons (1) [Info](#)

View details Edit Remove [Get more add-ons](#)

netapp

Any categ... Any status 1 match

NetApp **NetApp Trident**

NetApp Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Category	Status	Version	EKS Pod Identity	IAM role for service account (IRSA)
storage	Active	v24.10.0-eksbuild.1	-	Not set

Listed by [NetApp, Inc.](#)

[View subscription](#)

- 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 <code>ontap-san</code> et <code>ontap-san-economy</code>
ONTAP	chapitreInitiateurSecret	Secret de l'initiateur CHAP. Requis si useCHAP=true. Pour <code>ontap-san</code> et <code>ontap-san-economy</code>
ONTAP	nom d'utilisateur cible du chapitre	Nom d'utilisateur cible. Requis si useCHAP=true. Pour <code>ontap-san</code> et <code>ontap-san-economy</code>
ONTAP	chapCibleInitiateurSecret	Secret de l'initiateur de la cible CHAP. Requis si useCHAP=true. Pour <code>ontap-san</code> et <code>ontap-san-economy</code>

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

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/conservé 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: mysqlldb
      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 :

```

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

```

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 :

```

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

```

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 :

```

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

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

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

```

volumes d'importation

Vous pouvez importer des volumes de stockage existants en tant que PV Kubernetes en utilisant `tridentctl import` ou en créant une revendication de volume persistant (PVC) avec des annotations d'importation Trident.

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 importer un volume en utilisant soit `tridentctl import` soit en créant un PVC avec des annotations d'importation Trident.



Si vous utilisez des annotations PVC, vous n'avez pas besoin de télécharger ou d'utiliser `tridentctl` pour importer le volume.

Utilisation de tridentctl

Étapes

1. Créez un fichier PVC (par exemple, `pvc.yaml`) qui sera utilisé pour créer le PVC. Le fichier PVC doit inclure `name`, `namespace`, `accessModes` et `storageClassName`. Vous pouvez éventuellement spécifier `unixPermissions` dans la définition de votre 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'indiquez que les paramètres requis. Les paramètres supplémentaires tels que le nom du PV ou la taille du volume peuvent 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>
```

Utilisation des annotations PVC

Étapes

1. Créez un fichier YAML PVC (par exemple, `pvc.yaml`) avec les annotations d'importation Trident requises. Le fichier PVC doit inclure :
 - `name` et `namespace` dans les métadonnées
 - `accessModes`, `resources.requests.storage`, et `storageClassName` dans les spécifications
 - Annotations :
 - `trident.netapp.io/importOriginalName`: Nom du volume sur le backend
 - `trident.netapp.io/importBackendUUID`: UUID du backend où le volume existe
 - `trident.netapp.io/notManaged` (*Facultatif*) : Définir sur `"true"` pour les volumes non gérés. La valeur par défaut est `"false"`.

Voici un exemple de spécification pour l'importation d'un volume géré :

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <pvc-name>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "<volume-name>"
    trident.netapp.io/importBackendUUID: "<backend-uuid>"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: <size>
    storageClassName: <storage-class-name>
```

2. Appliquez le fichier YAML PVC à votre cluster Kubernetes :

```
kubectl apply -f <pvc-file>.yaml
```

Trident importera automatiquement le volume et le liera au PVC.

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 ONTAP NAS utilisant `tridentctl`

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>
```

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

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

Exemples ONTAP NAS utilisant des annotations PVC

Les exemples suivants montrent comment importer des volumes gérés et non gérés à l'aide d'annotations PVC.

Volume géré

L'exemple suivant importe un volume de 1 GiB ontap-nas nommé `ontap_volume1` à partir du backend `81abcb27-ea63-49bb-b606-0a5315ac5f21` avec le mode d'accès RWO défini à l'aide d'annotations PVC :

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <managed-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap_volume1"
    trident.netapp.io/importBackendUUID: "81abcb27-ea63-49bb-b606-0a5315ac5f21"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

Volume non géré

L'exemple suivant importe 1GiB ontap-nas volume nommé `ontap-volume2` depuis le backend `34abcb27-ea63-49bb-b606-0a5315ac5f34` avec le mode d'accès RWO défini à l'aide d'annotations PVC :

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <unmanaged-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap-volume2"
    trident.netapp.io/importBackendUUID: "34abcb27-ea63-49bb-b606-0a5315ac5f34"
    trident.netapp.io/notManaged: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

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

Les exemples suivants montrent comment importer des volumes gérés et non gérés :

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.

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
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
block	pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe	d3ba047a-ea0b-43f9-9c42-e38e58301c49	10 GiB	online	basic-element	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 `gpcvcs_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:/importvoll` , le chemin du volume est `importvoll` .

L'exemple suivant importe un `azure-netapp-files` volume sur le backend `azurenetaappfiles_40517` avec le chemin de volume `importvoll` .

```
tridentctl import volume azurenetaappfiles_40517 importvoll -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  
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.

L'exemple suivant importe un volume sur le backend `backend-tbc-gcnv1` avec le volume `testvoleasiaeast1`.

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
file	pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0	8c18cdf1-0770-4bc0-bcc5-c6295fe6d837	10 GiB	online	gcnv-nfs-sc-identity	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
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
file	pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0	8c18cdf1-0770-4bc0-bcc5-c6295fe6d837	10 GiB	online	gcnv-nfs-sc-identity	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.RequestName}}"
  },
  "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

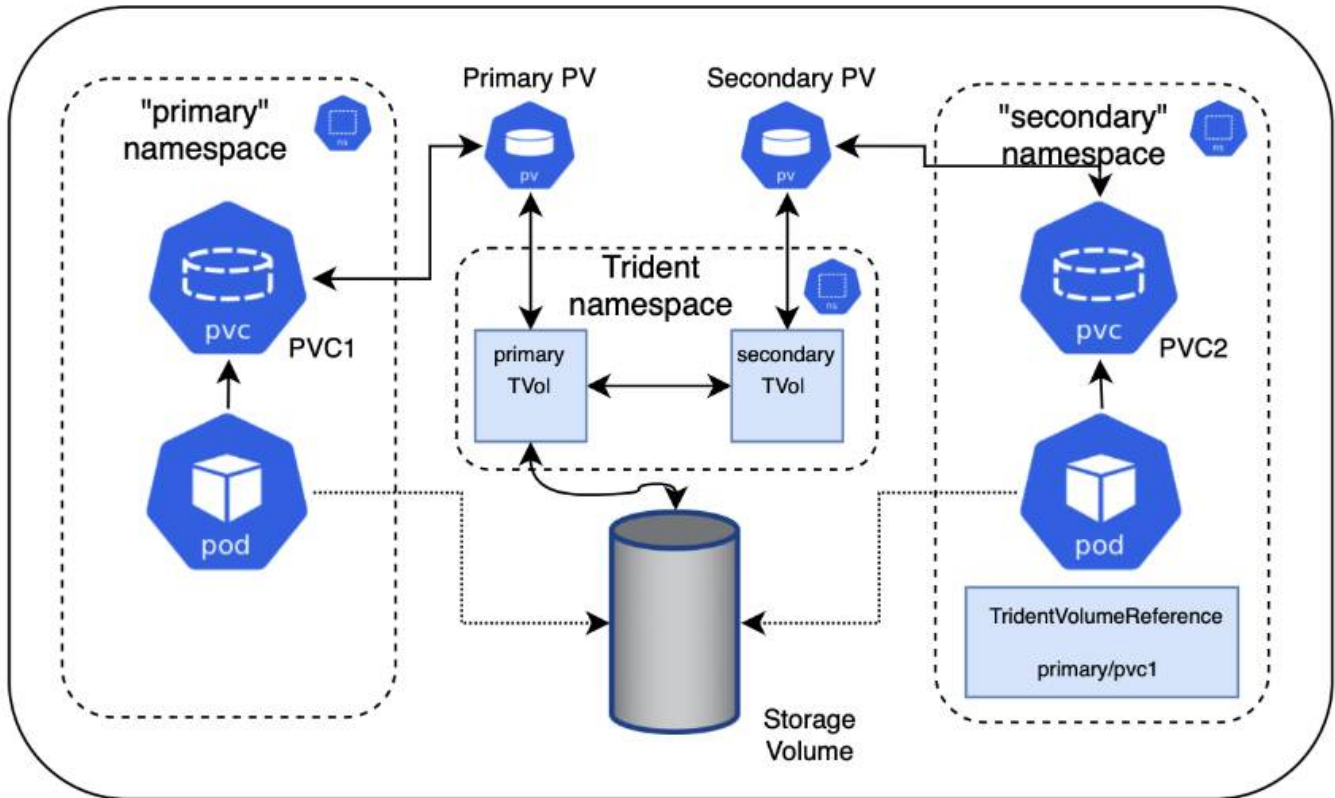
Avec Trident, vous pouvez créer un volume dans un espace de noms principal et le partager dans un ou plusieurs espaces de noms secondaires.

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

Créer 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

Créer 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](https://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 :

- Visitez ["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.

- 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`).
- 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
```

- 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

Veillez 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 flexvols 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
```

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

- Les nœuds du cluster doivent avoir des étiquettes qui permettent 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","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kuber-
netes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Étape 1 : 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

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

JSON

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



`supportedTopologies` sert à fournir une liste de régions et de zones par 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-centrall
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-centrall
    topology.kubernetes.io/zone: us-centrall-a
  - topology.kubernetes.io/region: us-centrall
    topology.kubernetes.io/zone: us-centrall-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-centrall
        topology.kubernetes.io/zone: us-centrall-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-centrall
        topology.kubernetes.io/zone: us-centrall-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:
  name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions:
    - key: topology.kubernetes.io/zone
      values:
        - us-east1-a
        - us-east1-b
    - key: topology.kubernetes.io/region
      values:
        - us-east1
parameters:
  fsType: ext4

```

Dans la définition de 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 :

```

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

```

Pour que Trident puisse créer un volume et le lier 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: voll
        persistentVolumeClaim:
          claimName: pvc-san
    containers:
      - name: sec-ctx-demo
        image: busybox
        command: [ "sh", "-c", "sleep 1h" ]
        volumeMounts:
          - name: voll
            mountPath: /data/demo
        securityContext:
          allowPrivilegeEscalation: false

```

Ce podSpec 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: pvcl-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

```

- 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

```

- 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 .
- 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 TARS CR. Cet exemple crée un CR pour PVC `pvc1` et instantané de volume `pvc1-snapshot`.



Le TARS 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: pvcl
    volumeSnapshotName: pvcl-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: pvcl-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_volumegroupsnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcontents.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"](#)

Gérer et surveiller Trident

Amélioration du Trident

Amélioration du Trident

À compter de la version 24.02, Trident suit un rythme de sortie de quatre mois, proposant trois mises à jour majeures chaque année civile. Chaque nouvelle version s'appuie sur les versions précédentes et propose de nouvelles fonctionnalités, des améliorations de performances, des corrections de bugs et des améliorations générales. Nous vous encourageons à effectuer la mise à niveau au moins une fois par an pour profiter des nouvelles fonctionnalités de Trident.

Considérations avant la mise à niveau

Lors de la mise à niveau vers la dernière version de Trident, tenez compte des points suivants :

- Une seule instance de Trident doit être installée dans tous les espaces de noms d'un cluster Kubernetes donné.
- Trident 23.07 et versions ultérieures nécessitent des instantanés de volume v1 et ne prennent plus en charge les instantanés alpha ou bêta.
- Si vous avez créé un Cloud Volumes Service pour Google Cloud dans le ["type de service CVS"](#), vous devez mettre à jour la configuration du backend pour utiliser le `standardsw` ou `zoneredundantstandardsw` niveau de service lors de la mise à niveau depuis Trident 23.01. Défaut de mise à jour `serviceLevel` Des problèmes au niveau du système dorsal pourraient entraîner des défaillances de volumes. Se référer à ["Exemples de type de service CVS"](#) pour plus de détails.
- Lors de la mise à niveau, il est important que vous fournissiez `parameter.fsType` dans `StorageClasses` utilisé par Trident. Vous pouvez supprimer et recréer `StorageClasses` sans perturber les volumes préexistants.
 - Il s'agit d'une **condition** pour faire respecter ["contextes de sécurité"](#) pour les volumes SAN.
 - Le répertoire <https://github.com/NetApp/trident/tree/master/trident-installer/sample-input> [exemple d'entrée^] contient des exemples, tels que `storage-class-basic.yaml.templ` et lien : `storage-class-bronze-default.yaml`.
 - Pour plus d'informations, veuillez consulter ["Problèmes connus"](#).

Étape 1 : Sélectionnez une version

Les versions de Trident suivent un système basé sur la date. `YY.MM` convention d'appellation, où « AAAA » représente les deux derniers chiffres de l'année et « MM » le mois. Les sorties de Dot suivent un `YY.MM.X` convention, où « X » représente le niveau du patch. Vous sélectionnez la version vers laquelle effectuer la mise à niveau en fonction de la version actuelle.

- Vous pouvez effectuer une mise à niveau directe vers n'importe quelle version cible se trouvant dans un intervalle de quatre versions autour de votre version installée. Par exemple, vous pouvez effectuer une mise à niveau directe de la version 24.06 (ou de toute version 24.06) vers la version 25.06.
- Si vous effectuez une mise à niveau à partir d'une version antérieure à la période de quatre mises à niveau, procédez à une mise à niveau en plusieurs étapes. Utilisez les instructions de mise à niveau pour le ["version antérieure"](#) Vous effectuez une mise à niveau depuis une version antérieure afin de passer à la

version la plus récente compatible avec la fenêtre de quatre versions. Par exemple, si vous utilisez la version 23.07 et que vous souhaitez passer à la version 25.06 :

- a. Première mise à jour du 23.07 au 24.06.
- b. Ensuite, passez de la version 24.06 à la version 25.06.



Lors de la mise à niveau à l'aide de l'opérateur Trident sur OpenShift Container Platform, vous devez effectuer une mise à niveau vers Trident 21.01.1 ou une version ultérieure. L'opérateur Trident publié avec la version 21.01.0 contient un problème connu qui a été corrigé dans la version 21.01.1. Pour plus de détails, veuillez vous référer au ["Détails du problème sur GitHub"](#) .

Étape 2 : Déterminer la méthode d'installation d'origine

Pour déterminer la version que vous avez utilisée pour installer Trident à l'origine :

1. Utiliser `kubectl get pods -n trident` pour examiner les gousses.
 - En l'absence de nacelle opérateur, Trident a été installé via `tridentctl` .
 - S'il existe un module opérateur, Trident a été installé à l'aide de l'opérateur Trident , soit manuellement, soit via Helm.
2. S'il y a une cabine d'opérateur, utilisez-la `kubectl describe torc` pour déterminer si Trident a été installé à l'aide de Helm.
 - Si une étiquette Helm est présente, Trident a été installé à l'aide de Helm.
 - S'il n'y a pas d'étiquette Helm, Trident a été installé manuellement à l'aide de l'opérateur Trident .

Étape 3 : Sélectionnez une méthode de mise à niveau

En règle générale, vous devez effectuer la mise à niveau en utilisant la même méthode que celle utilisée pour l'installation initiale, mais vous pouvez ["passer d'une méthode d'installation à une autre"](#) . Il existe deux options pour mettre à niveau Trident.

- ["Mise à niveau via l'opérateur Trident"](#)



Nous vous suggérons de consulter ["Comprendre le flux de travail de mise à niveau de l'opérateur"](#) avant de procéder à la mise à niveau auprès de l'opérateur.

*

Mise à niveau avec l'opérateur

Comprendre le flux de travail de mise à niveau de l'opérateur

Avant d'utiliser l'opérateur Trident pour mettre à niveau Trident, vous devez comprendre les processus d'arrière-plan qui se produisent pendant la mise à niveau. Cela inclut des modifications apportées au contrôleur Trident , au Pod du contrôleur et aux Pods du nœud, ainsi qu'au DaemonSet du nœud, qui permettent des mises à jour progressives.

Gestion de la mise à niveau de l'opérateur Trident

L'un des nombreux ["avantages de l'utilisation de l'opérateur Trident"](#) L'installation et la mise à niveau de Trident

consistent en la gestion automatique des objets Trident et Kubernetes sans perturber les volumes montés existants. De cette manière, Trident peut prendre en charge les mises à niveau sans aucune interruption de service, ou "*mises à jour continues*". Plus précisément, l'opérateur Trident communique avec le cluster Kubernetes pour :

- Supprimez et recréez le déploiement du contrôleur Trident et le DaemonSet du nœud.
- Remplacez le module de contrôleur Trident et les modules de nœud Trident par les nouvelles versions.
 - Le fait qu'un nœud ne soit pas mis à jour n'empêche pas la mise à jour des autres nœuds.
 - Seuls les nœuds disposant d'un pod de nœud Trident en cours d'exécution peuvent monter des volumes.



Pour plus d'informations sur l'architecture Trident sur le cluster Kubernetes, consultez la documentation. "[Architecture Trident](#)".

Flux de travail de mise à niveau de l'opérateur

Lorsque vous lancez une mise à niveau à l'aide de l'opérateur Trident :

1. L'opérateur * Trident * :
 - a. Détecte la version actuellement installée de Trident (version n).
 - b. Met à jour tous les objets Kubernetes, y compris les CRD, RBAC et Trident SVC.
 - c. Supprime le déploiement du contrôleur Trident pour la version n .
 - d. Crée le déploiement du contrôleur Trident pour la version $n+1$.
2. **Kubernetes** crée un pod de contrôleur Trident pour $n+1$.
3. L'opérateur * Trident * :
 - a. Supprime le DaemonSet de nœud Trident pour n . L'opérateur n'attend pas la fin du Node Pod.
 - b. Crée le Daemonset de nœud Trident pour $n+1$.
4. **Kubernetes** crée des pods de nœuds Trident sur les nœuds n'exécutant pas le pod de nœud Trident n . Cela garantit qu'il n'y a jamais plus d'un Trident Node Pod, quelle que soit sa version, sur un nœud.

Mettez à niveau une installation Trident à l'aide de l'opérateur Trident ou de Helm.

Vous pouvez mettre à niveau Trident à l'aide de l'opérateur Trident , soit manuellement, soit via Helm. Vous pouvez effectuer une mise à niveau d'une installation d'opérateur Trident vers une autre installation d'opérateur Trident ou effectuer une mise à niveau depuis une `tridentctl` installation sur une version opérateur Trident .

Revoir "[Sélectionnez une méthode de mise à niveau](#)" avant de mettre à niveau une installation d'opérateur Trident .

Mise à niveau d'une installation manuelle

Vous pouvez effectuer une mise à niveau d'une installation d'opérateur Trident à l'échelle d'un cluster vers une autre installation d'opérateur Trident à l'échelle d'un cluster. Toutes les versions de Trident utilisent un opérateur à portée de cluster.



Pour mettre à niveau une version de Trident installée à l'aide de l'opérateur à portée d'espace de noms (versions 20.07 à 20.10), suivez les instructions de mise à niveau pour "[votre version installée](#)" de Trident.

À propos de cette tâche

Trident fournit un fichier bundle que vous pouvez utiliser pour installer l'opérateur et créer les objets associés pour votre version de Kubernetes.

- Pour les clusters exécutant Kubernetes 1.24, utilisez "[bundle_pre_1_25.yaml](#)".
- Pour les clusters exécutant Kubernetes 1.25 ou une version ultérieure, utilisez "[bundle_post_1_25.yaml](#)".

Avant de commencer

Assurez-vous d'utiliser un cluster Kubernetes en cours d'exécution. "[une version Kubernetes prise en charge](#)".

Étapes

1. Vérifiez votre version de Trident :

```
./tridentctl -n trident version
```

2. Mettre à jour `operator.yaml`, `tridentorchestrator_cr.yaml`, et `post_1_25_bundle.yaml` avec le registre et les chemins d'accès aux images pour la version vers laquelle vous effectuez la mise à niveau (par exemple 25.06), et le secret correct.
3. Supprimez l'opérateur Trident qui a été utilisé pour installer l'instance Trident actuelle. Par exemple, si vous effectuez une mise à niveau à partir de la version 25.02, exécutez la commande suivante :

```
kubectl delete -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n trident
```

4. Si vous avez personnalisé votre installation initiale en utilisant `TridentOrchestrator` les attributs, vous pouvez les modifier `TridentOrchestrator` objet permettant de modifier les paramètres d'installation. Cela peut inclure des modifications apportées pour spécifier les registres d'images Trident et CSI en miroir pour le mode hors ligne, activer les journaux de débogage ou spécifier les secrets d'extraction d'images.
5. Installez Trident en utilisant le fichier YAML de bundle approprié à votre environnement, où `<bundle.yaml>` est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` en fonction de votre version de Kubernetes. Par exemple, si vous installez Trident 25.06.0, exécutez la commande suivante :

```
kubectl create -f 25.06.0/trident-installer/deploy/<bundle.yaml> -n trident
```

6. Modifiez le torc trident pour inclure l'image 25.06.0.

Mettre à niveau une installation Helm

Vous pouvez mettre à niveau une installation de Trident Helm.



Lors de la mise à niveau d'un cluster Kubernetes de la version 1.24 à la version 1.25 ou ultérieure sur lequel Trident est installé, vous devez mettre à jour le fichier `values.yaml` pour configurer `excludePodSecurityPolicy` à `true` ou ajouter `--set excludePodSecurityPolicy=true` au `helm upgrade` commande avant de pouvoir mettre à niveau le cluster.

Si vous avez déjà mis à niveau votre cluster Kubernetes de la version 1.24 à la version 1.25 sans mettre à niveau le helm Trident, la mise à niveau du helm échoue. Pour que la mise à niveau du poste de pilotage puisse être effectuée, veuillez suivre les étapes préalables suivantes :

1. Installez le plugin `helm-mapkubeapis` depuis <https://github.com/helm/helm-mapkubeapis> .
2. Effectuez un test à blanc pour la version de Trident dans l'espace de noms où Trident est installé. Cette liste répertorie les ressources qui seront nettoyées.

```
helm mapkubeapis --dry-run trident --namespace trident
```

3. Effectuez une exécution complète avec Helm pour effectuer le nettoyage.

```
helm mapkubeapis trident --namespace trident
```

Étapes

1. Si tu "[Trident a été installé à l'aide de Helm.](#)", vous pouvez utiliser `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` mettre à niveau en une seule étape. Si vous n'avez pas ajouté le dépôt Helm ou si vous ne pouvez pas l'utiliser pour effectuer la mise à niveau :
 - a. Téléchargez la dernière version de Trident depuis "[la section Assets sur GitHub](#)".
 - b. Utilisez le `helm upgrade` commande où `trident-operator-25.06.0.tgz` reflète la version vers laquelle vous souhaitez effectuer la mise à niveau.

```
helm upgrade <name> trident-operator-25.06.0.tgz
```



Si vous définissez des options personnalisées lors de l'installation initiale (comme la spécification de registres privés et en miroir pour les images Trident et CSI), ajoutez le `helm upgrade` commande utilisant `--set` afin de garantir que ces options soient incluses dans la commande de mise à niveau, sinon les valeurs seront réinitialisées à leurs valeurs par défaut.

2. Courir `helm list` vérifier que la version du graphique et celle de l'application ont bien été mises à jour. Courir `tridentctl logs` pour examiner les messages de débogage.

Mise à niveau depuis un `tridentctl` installation à l'opérateur Trident

Vous pouvez mettre à jour l'opérateur Trident vers sa dernière version à partir d'un `tridentctl` installation. Les backends et les PVC existants seront automatiquement disponibles.



Avant de changer de méthode d'installation, consultez ["Passer d'une méthode d'installation à une autre"](#) .

Étapes

1. Téléchargez la dernière version de Trident .

```
# Download the release required [25.06.0]
mkdir 25.06.0
cd 25.06.0
wget
https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
installer-25.06.0.tar.gz
tar -xf trident-installer-25.06.0.tar.gz
cd trident-installer
```

2. Créez le tridentorchestrator CRD du manifeste.

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Déployez l'opérateur à portée de cluster dans le même espace de noms.

```
kubectl create -f deploy/<bundle-name.yaml>

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-controller-79df798bdc-m79dc 6/6     Running   0           150d
trident-node-linux-xrst8             2/2     Running   0           150d
trident-operator-5574dbbc68-nthjv    1/1     Running   0           1m30s
```

4. Créer un TridentOrchestrator CR pour l'installation de Trident.

```

cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-79df798bdc-m79dc        6/6     Running   0           1m
trident-csi-xrst8                    2/2     Running   0           1m
trident-operator-5574dbbc68-nthjv   1/1     Running   0           5m41s

```

5. Confirmez que Trident a bien été mis à niveau vers la version prévue.

```

kubectl describe torc trident | grep Message -A 3

Message:          Trident installed
Namespace:        trident
Status:           Installed
Version:          v25.06.0

```

Mise à niveau avec tridentctl

Vous pouvez facilement mettre à niveau une installation Trident existante en utilisant `tridentctl`.

À propos de cette tâche

Désinstaller puis réinstaller Trident équivaut à une mise à jour. Lorsque vous désinstallez Trident, la revendication de volume persistant (PVC) et le volume persistant (PV) utilisés par le déploiement de Trident ne sont pas supprimés. Les volumes persistants (PV) déjà provisionnés resteront disponibles pendant que Trident est hors ligne, et Trident provisionnera des volumes pour tous les volumes persistants (PVC) créés entre-temps après sa remise en ligne.

Avant de commencer

Revoir "[Sélectionnez une méthode de mise à niveau](#)" avant de mettre à niveau en utilisant `tridentctl`.

Étapes

1. Exécutez la commande de désinstallation dans `tridentctl` supprimer toutes les ressources associées à Trident, à l'exception des CRD et des objets associés.

```
./tridentctl uninstall -n <namespace>
```

2. Réinstallez Trident. Se référer à "[Installez Trident à l'aide de tridentctl](#)".



N'interrompez pas le processus de mise à niveau. Assurez-vous que l'installation se déroule jusqu'à son terme.

Gérez Trident à l'aide de tridentctl

Le "[Pack d'installation Trident](#)" comprend le `tridentctl` utilitaire en ligne de commande pour fournir un accès simple à Trident. Les utilisateurs de Kubernetes disposant de privilèges suffisants peuvent l'utiliser pour installer Trident ou gérer l'espace de noms contenant le pod Trident .

Commandes et drapeaux globaux

Vous pouvez courir `tridentctl help` pour obtenir la liste des commandes disponibles pour `tridentctl` ou ajouter le `--help` Utilisez l'option `--flag` avec n'importe quelle commande pour obtenir la liste des options et des indicateurs disponibles pour cette commande spécifique.

```
tridentctl [command] [--optional-flag]
```

Le Trident `tridentctl` Cet utilitaire prend en charge les commandes et les options globales suivantes.

Commandes

create

Ajouter une ressource à Trident.

delete

Supprimer une ou plusieurs ressources de Trident.

get

Obtenez une ou plusieurs ressources de Trident.

help

Aide concernant n'importe quelle commande.

images

Imprimez un tableau des images conteneurs nécessaires à Trident .

import

Importer une ressource existante dans Trident.

install

Installez Trident.

logs

Imprimer les journaux de Trident.

send

Envoyer une ressource depuis Trident.

uninstall

Désinstallez Trident.

update

Modifier une ressource dans Trident.

update backend state

Suspension temporaire des opérations en arrière-plan.

upgrade

Mettre à niveau une ressource dans Trident.

version

Imprimez la version de Trident.

drapeaux du monde

-d, --debug

Sortie de débogage.

-h, --help

Aide pour `tridentctl`.

-k, --kubeconfig string

Précisez le `KUBECONFIG` chemin d'exécution des commandes localement ou d'un cluster Kubernetes à un autre.



Vous pouvez également exporter le `KUBECONFIG` variable permettant de pointer vers un cluster Kubernetes spécifique et de résoudre le problème `tridentctl` commandes à ce cluster.

-n, --namespace string

Espace de noms du déploiement Trident.

-o, --output string

Format de sortie. L'un des formats `json|yaml|name|wide|ps` (par défaut).

-s, --server string

Adresse/port de l'interface REST de Trident.



L'interface REST de Trident peut être configurée pour écouter et servir uniquement à l'adresse `127.0.0.1` (pour IPv4) ou `:::1` (pour IPv6).

Options de commande et drapeaux

créer

Utilisez le `create` commande pour ajouter une ressource à Trident.

```
tridentctl create [option]
```

Options

`backend`: Ajouter un backend à Trident.

supprimer

Utilisez le `delete` commande permettant de supprimer une ou plusieurs ressources de Trident.

```
tridentctl delete [option]
```

Options

`backend`: Supprimez un ou plusieurs backends de stockage de Trident.

`snapshot` : Supprimez un ou plusieurs instantanés de volume de Trident.

`storageclass` : Supprimez une ou plusieurs classes de stockage de Trident.
`volume` : Supprimez un ou plusieurs volumes de stockage de Trident.

obtenir

Utilisez le `get` commande pour obtenir une ou plusieurs ressources de Trident.

```
tridentctl get [option]
```

Options

`backend` : Obtenez un ou plusieurs backends de stockage auprès de Trident.
`snapshot` : Obtenez un ou plusieurs instantanés de Trident.
`storageclass` : Obtenez une ou plusieurs classes de stockage de Trident.
`volume` : Procurez-vous un ou plusieurs volumes chez Trident.

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.

images

Utiliser `images` drapeaux permettant d'imprimer un tableau des images de conteneurs dont Trident a besoin.

```
tridentctl images [flags]
```

drapeaux

`-h, --help` Aide pour les images.
`-v, --k8s-version string` : Version sémantique du cluster Kubernetes.

volume d'importation

Utilisez le `import volume` commande pour importer un volume existant dans Trident.

```
tridentctl import volume <backendName> <volumeName> [flags]
```

Alias

`volume, v`

drapeaux

`-f, --filename string` : Chemin d'accès au fichier PVC YAML ou JSON.
`-h, --help` : Aide pour le volume.
`--no-manage` : Créer uniquement des PV/PVC. Ne présumez pas de la gestion du cycle de vie des volumes.

installer

Utilisez le `install options` pour installer Trident.

```
tridentctl install [flags]
```

drapeaux

`--autosupport-image string`: L'image conteneur pour la télémétrie Autosupport (par défaut « `netapp/trident autosupport:<version-actuelle>` »).

`--autosupport-proxy string`: Adresse/port d'un proxy pour l'envoi des données de télémétrie Autosupport.

`--enable-node-prep` Tentative d'installation des paquets requis sur les nœuds.

`--generate-custom-yaml` Générez des fichiers YAML sans rien installer.

`-h`, `--help`: Aide à l'installation.

`--http-request-timeout`: Remplacer le délai d'expiration de la requête HTTP pour l'API REST du contrôleur Trident (par défaut 1 min 30 s).

`--image-registry string`: Adresse/port d'un registre d'images interne.

`--k8s-timeout duration`: Le délai d'expiration pour toutes les opérations Kubernetes (par défaut 3m0s).

`--kubelet-dir string`: Emplacement hôte de l'état interne de kubelet (par défaut « `/var/lib/kubelet` »).

`--log-format string`: Le format de journalisation Trident (texte, json) (par défaut "texte").

`--node-prep`: Permet à Trident de préparer les nœuds du cluster Kubernetes à gérer les volumes à l'aide du protocole de stockage de données spécifié. **Actuellement, `iscsi` est la seule valeur prise en charge. À partir d'OpenShift 4.19, la version minimale de Trident prise en charge pour cette fonctionnalité est la 25.06.1.**

`--pv string`: Le nom du PV hérité utilisé par Trident, garantit que celui-ci n'existe pas (par défaut « `trident` »).

`--pvc string`: Le nom du PVC hérité utilisé par Trident, garantit que celui-ci n'existe pas (par défaut « `trident` »).

`--silence-autosupport`: Ne pas envoyer automatiquement les bundles d'autosupport à NetApp (valeur par défaut: vrai).

`--silent` Désactiver la plupart des sorties pendant l'installation.

`--trident-image string`: L'image Trident à installer.

`--k8s-api-qps`: La limite de requêtes par seconde (QPS) pour les requêtes API Kubernetes (par défaut 100 ; optionnel).

`--use-custom-yaml` Utilisez les fichiers YAML existants dans le répertoire d'installation.

`--use-ipv6` Utilisez IPv6 pour la communication de Trident.

journaux

Utiliser `logs` options pour imprimer les journaux de Trident.

```
tridentctl logs [flags]
```

drapeaux

`-a`, `--archive`: Créer une archive de support contenant tous les journaux, sauf indication contraire.

`-h`, `--help` Aide pour les journaux.

`-l`, `--log string`: Affichage du journal Trident . L'un des `trident|auto|trident-operator|all` (par défaut "auto").

`--node string`: Nom du nœud Kubernetes à partir duquel collecter les journaux des pods.

`-p`, `--previous` Récupérez les journaux de l'instance de conteneur précédente, si elle existe.

`--sidecars` Récupérez les journaux des conteneurs sidecar.

envoyer

Utilisez le `send` commande pour envoyer une ressource depuis Trident.

```
tridentctl send [option]
```

Options

`autosupport`: Envoyer une archive Autosupport à NetApp.

désinstaller

Utiliser `uninstall` options pour désinstaller Trident.

```
tridentctl uninstall [flags]
```

drapeaux

`-h, --help` Aide à la désinstallation.

`--silent` Désactiver la plupart des sorties lors de la désinstallation.

mise à jour

Utilisez le `update` commande pour modifier une ressource dans Trident.

```
tridentctl update [option]
```

Options

`backend`: Mettre à jour un backend dans Trident.

mettre à jour l'état du backend

Utilisez le `update backend state` commande permettant de suspendre ou de reprendre les opérations en arrière-plan.

```
tridentctl update backend state <backend-name> [flag]
```

Points à considérer

- Si un backend est créé à l'aide d'un `TridentBackendConfig` (tbc), il ne peut pas être mis à jour à l'aide d'un `backend.json` déposer.
- Si le `userState` a été défini dans une date limite, il ne peut pas être modifié à l'aide de `tridentctl update backend state <backend-name> --user-state suspended/normal` commande.
- Pour retrouver la capacité de régler `userState` via `tridentctl` après avoir été configuré via `tbc`, le `userState` Ce champ doit être supprimé du fichier `tbc`. Cela peut être fait en utilisant le `kubectl edit tbc` commande. Après le `userState` Le champ a été supprimé, vous pouvez utiliser le `tridentctl update backend state` commande pour modifier la `userState` d'un backend.
- Utilisez le `tridentctl update backend state` pour changer le `userState`. Vous pouvez également mettre à jour le `userState` en utilisant `TridentBackendConfig` ou `backend.json` fichier ; cela déclenche une réinitialisation complète du système dorsal et peut prendre du temps.

drapeaux

`-h, --help` : Aide pour l'état du backend.

`--user-state` : Définir sur `suspended` interrompre les opérations en arrière-plan. Réglé sur `normal` pour reprendre les opérations en arrière-plan. Lorsqu'il est réglé sur `suspended` :

- `AddVolume` et `Import Volume` sont en pause.
- `CloneVolume`, `ResizeVolume`, `PublishVolume`, `UnPublishVolume`, `CreateSnapshot`,

`GetSnapshot` , `RestoreSnapshot` , `DeleteSnapshot` , `RemoveVolume` , `GetVolumeExternal` , `ReconcileNodeAccess` restent disponibles.

Vous pouvez également mettre à jour l'état du backend en utilisant `userState` champ dans le fichier de configuration du backend `TridentBackendConfig` ou `backend.json` . Pour plus d'informations, veuillez consulter "[Options pour la gestion des backends](#)" et "[Effectuez la gestion du backend avec kubectl](#)" .

Exemple:

JSON

Suivez ces étapes pour mettre à jour le `userState` en utilisant le `backend.json` déposer:

1. Modifier le `backend.json` fichier à inclure le `userState` champ dont la valeur est définie sur « suspendu ».
2. Mettez à jour le backend en utilisant le `tridentctl update backend` la commande et le chemin d'accès à la version mise à jour `backend.json` déposer.

Exemple: `tridentctl update backend -f /<path to backend JSON file>/backend.json -n trident`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "<redacted>",
  "svm": "nas-svm",
  "backendName": "customBackend",
  "username": "<redacted>",
  "password": "<redacted>",
  "userState": "suspended"
}
```

YAML

Vous pouvez modifier le tbc après son application en utilisant `kubectl edit <tbc-name> -n <namespace>` commande. L'exemple suivant met à jour l'état du backend pour le suspendre en utilisant le `userState: suspended` option:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  backendName: customBackend
  storageDriverName: ontap-nas
  managementLIF: <redacted>
  svm: nas-svm
  userState: suspended
  credentials:
    name: backend-tbc-ontap-nas-secret
```

version

Utiliser `version` drapeaux pour imprimer la version de `tridentctl` et le service Trident en fonctionnement.

```
tridentctl version [flags]
```

drapeaux

```
--client`Version client uniquement (aucun serveur requis).  
-h, --help : Aide pour la version.
```

Prise en charge des plugins

Tridentctl prend en charge des plugins similaires à `kubectl`. Tridentctl détecte un plugin si le nom du fichier binaire du plugin suit le schéma « `tridentctl-<plugin>` », et si le binaire est situé dans un dossier répertorié dans la variable d'environnement `PATH`. Tous les plugins détectés sont répertoriés dans la section plugins de l'aide de `tridentctl`. Vous pouvez également limiter la recherche en spécifiant un dossier de plugins dans la variable d'environnement `TRIDENTCTL_PLUGIN_PATH` (Exemple : `TRIDENTCTL_PLUGIN_PATH=~/.tridentctl-plugins/`). Si la variable est utilisée, `tridentctl` effectue la recherche uniquement dans le dossier spécifié.

Monitor Trident

Trident fournit un ensemble de points de terminaison de métriques Prometheus que vous pouvez utiliser pour surveiller les performances de Trident .

Aperçu

Les indicateurs fournis par Trident vous permettent de faire ce qui suit :

- Surveillez l'état et la configuration de Trident. Vous pouvez examiner le bon déroulement des opérations et vérifier si la communication avec les systèmes dorsaux se fait comme prévu.
- Examinez les informations d'utilisation du backend et comprenez combien de volumes sont provisionnés sur un backend et la quantité d'espace consommée, etc.
- Tenir à jour une cartographie des volumes provisionnés sur les serveurs backend disponibles.
- Suivre les performances. Vous pouvez observer combien de temps il faut à Trident pour communiquer avec les serveurs d'arrière-plan et effectuer les opérations.



Par défaut, les métriques de Trident sont exposées sur le port cible 8001 à `/metrics` point final. Ces indicateurs sont **activés par défaut** lorsque Trident est installé.

Ce dont vous aurez besoin

- Un cluster Kubernetes avec Trident installé.
- Une instance de Prometheus. Cela peut être un ["Déploiement Prometheus conteneurisé"](#) ou vous pouvez choisir d'exécuter Prometheus en tant que ["application native"](#) .

Étape 1 : Définir une cible Prometheus

Vous devez définir une cible Prometheus pour collecter les métriques et obtenir des informations sur les backends gérés par Trident , les volumes qu'il crée, etc. Ce ["blog"](#) explique comment utiliser Prometheus et Grafana avec Trident pour récupérer des métriques. Ce blog explique comment exécuter Prometheus en tant

qu'opérateur dans votre cluster Kubernetes et comment créer un ServiceMonitor pour obtenir les métriques Trident .

Étape 2 : Créer un moniteur de service Prometheus

Pour exploiter les métriques Trident , vous devez créer un ServiceMonitor Prometheus qui surveille le `trident-csi` service et écoute sur le `metrics` port. Voici à quoi ressemble un exemple de ServiceMonitor :

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

Cette définition ServiceMonitor récupère les métriques renvoyées par le `trident-csi` le service et recherche spécifiquement le `metrics` point de terminaison du service. Par conséquent, Prometheus est désormais configuré pour comprendre les indicateurs de Trident.

En plus des métriques disponibles directement depuis Trident, kubelet expose de nombreuses autres. `kubelet_volume_*` métriques via son propre point de terminaison de métriques. Kubelet peut fournir des informations sur les volumes qui y sont attachés, ainsi que sur les pods et autres opérations internes qu'il gère. Se référer à "ici" .

Étape 3 : Interroger les métriques Trident avec PromQL

PromQL est idéal pour créer des expressions qui renvoient des données de séries temporelles ou des données tabulaires.

Voici quelques requêtes PromQL que vous pouvez utiliser :

Obtenez des informations sur la santé de Trident

- Pourcentage de réponses HTTP 2XX provenant de Trident

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

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

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

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

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

Obtenez des informations sur l'utilisation de Trident

- **Volume moyen**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Espace total alloué par chaque serveur backend**

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

Obtenez l'utilisation individuelle du volume



Cette option n'est activée que si les métriques kubelet sont également collectées.

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

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *  
100
```

Découvrez la télémétrie de Trident AutoSupport

Par défaut, Trident envoie quotidiennement à NetApp les métriques Prometheus et les informations de base du système dorsal.

- Pour empêcher Trident d'envoyer les métriques Prometheus et les informations de base du système dorsal à NetApp, transmettez le `--silence-autosupport` drapeau pendant l'installation de Trident .

- Trident peut également envoyer les journaux de conteneurs au support NetApp à la demande via `tridentctl send autosupport`. Vous devrez déclencher l'envoi des journaux de Trident. Avant de soumettre les journaux, vous devez accepter les conditions de NetApp. <https://www.netapp.com/company/legal/privacy-policy/>["politique de confidentialité"] .
- Sauf indication contraire, Trident récupère les journaux des dernières 24 heures.
- Vous pouvez spécifier la durée de conservation des journaux avec le `--since` drapeau. Par exemple: `tridentctl send autosupport --since=1h`. Ces informations sont collectées et envoyées via un `trident-autosupport` conteneur installé en parallèle de Trident. Vous pouvez obtenir l'image du conteneur à l'adresse suivante : "Trident AutoSupport" .
- Trident AutoSupport ne recueille ni ne transmet d'informations personnelles identifiables (IPI) ou d'informations personnelles. Il est livré avec un "CLUF" Cela ne s'applique pas à l'image conteneur Trident elle-même. Vous pouvez en apprendre davantage sur l'engagement de NetApp en matière de sécurité et de confiance des données. "ici" .

Voici un exemple de charge utile envoyée par Trident :

```

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

```

- Les messages AutoSupport sont envoyés au point de terminaison AutoSupport de NetApp. Si vous utilisez un registre privé pour stocker des images de conteneurs, vous pouvez utiliser `--image-registry` drapeau.
- Vous pouvez également configurer les URL proxy en générant les fichiers YAML d'installation. Cela peut être réalisé en utilisant `tridentctl install --generate-custom-yaml` pour créer les fichiers YAML et ajouter les `--proxy-url` argument en faveur du `trident-autosupport` conteneur dans `trident-deployment.yaml` .

Désactiver les métriques Trident

Pour **désactiver** le signalement des métriques, vous devez générer des fichiers YAML personnalisés (en utilisant le `--generate-custom-yaml` (drapeau) et modifiez-les pour supprimer le `--metrics` empêcher l'invocation de ce drapeau pour le `trident-main` récipient.

Désinstaller Trident

Vous devez utiliser la même méthode pour désinstaller Trident que celle utilisée pour l'installer.

À propos de cette tâche

- Si vous rencontrez des problèmes de bogues après une mise à niveau, des problèmes de dépendances ou une mise à niveau incomplète ou ayant échoué, vous devez désinstaller Trident et réinstaller la version précédente en suivant les instructions spécifiques. "[version](#)". Il s'agit de la seule méthode recommandée pour revenir à une version antérieure.
- Pour faciliter la mise à niveau et la réinstallation, la désinstallation de Trident ne supprime pas les CRD ni les objets associés créés par Trident. Si vous devez supprimer complètement Trident et toutes ses données, veuillez consulter la documentation. "[Supprimer complètement Trident et CRD](#)".

Avant de commencer

Si vous mettez hors service des clusters Kubernetes, vous devez supprimer toutes les applications qui utilisent des volumes créés par Trident avant la désinstallation. Cela garantit que les PVC sont dépubliées sur les nœuds Kubernetes avant d'être supprimées.

Déterminer la méthode d'installation d'origine

Vous devez utiliser la même méthode pour désinstaller Trident que celle utilisée pour l'installer. Avant de désinstaller, vérifiez quelle version vous avez utilisée pour installer Trident à l'origine.

1. Utiliser `kubectl get pods -n trident` pour examiner les gousses.
 - En l'absence de nacelle opérateur, Trident a été installé via `tridentctl`.
 - S'il existe un module opérateur, Trident a été installé à l'aide de l'opérateur Trident, soit manuellement, soit via Helm.
2. S'il y a une cabine d'opérateur, utilisez-la `kubectl describe tproc trident` pour déterminer si Trident a été installé à l'aide de Helm.
 - Si une étiquette Helm est présente, Trident a été installé à l'aide de Helm.
 - S'il n'y a pas d'étiquette Helm, Trident a été installé manuellement à l'aide de l'opérateur Trident.

Désinstaller une installation d'opérateur Trident

Vous pouvez désinstaller une installation de l'opérateur Trident manuellement ou à l'aide de Helm.

Désinstaller l'installation manuelle

Si vous avez installé Trident à l'aide de l'opérateur, vous pouvez le désinstaller en procédant de l'une des manières suivantes :

1. **Modifier `TridentOrchestrator` CR et définissez l'indicateur de désinstallation :**

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"uninstall":true}}'
```

Quand le `uninstall` L'indicateur est réglé sur `true` L'opérateur Trident désinstalle Trident, mais ne supprime pas `TridentOrchestrator` lui-même. Vous devriez nettoyer le `TridentOrchestrator` et en créer un nouveau si vous souhaitez réinstaller Trident .

2. **Supprimer `TridentOrchestrator`** : En supprimant le `TridentOrchestrator` CR qui a été utilisé pour déployer Trident, vous demandez à l'opérateur de désinstaller Trident. L'opérateur traite le retrait de `TridentOrchestrator` et procède à la suppression du déploiement Trident et du `daemonset`, en supprimant les pods Trident qu'il avait créés dans le cadre de l'installation.

```
kubectl delete -f deploy/<bundle.yaml> -n <namespace>
```

Désinstaller Helm

Si vous avez installé Trident à l'aide de Helm, vous pouvez le désinstaller en utilisant `helm uninstall`.

```
#List the Helm release corresponding to the Trident install.
helm ls -n trident
NAME                NAMESPACE      REVISION      UPDATED
STATUS             CHART           APP VERSION
trident            trident         1             2021-04-20
00:26:42.417764794 +0000 UTC deployed      trident-operator-21.07.1
21.07.1

#Uninstall Helm release to remove Trident
helm uninstall trident -n trident
release "trident" uninstalled
```

Désinstaller un `tridentctl` installation

Utilisez le `uninstall` commande dans `tridentctl` supprimer toutes les ressources associées à Trident, à l'exception des CRD et des objets associés :

```
./tridentctl uninstall -n <namespace>
```

Trident pour Docker

Prérequis pour le déploiement

Vous devez installer et configurer les prérequis de protocole nécessaires sur votre hôte avant de pouvoir déployer Trident.

Vérifiez les exigences

- Vérifiez que votre déploiement répond à toutes les exigences. ["exigences"](#) .
- Vérifiez que vous disposez d'une version compatible de Docker installée. Si votre version de Docker est obsolète, ["installer ou mettre à jour"](#) .

```
docker --version
```

- Vérifiez que les prérequis du protocole sont installés et configurés sur votre hôte.

Outils NFS

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

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.

Outils iSCSI

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

RHEL 8+

1. Installez les paquets système suivants :

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_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
```

Outils NVMe

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
```

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 scsitol
```

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
```

Déployer Trident

Trident pour Docker assure une intégration directe avec l'écosystème Docker pour les plateformes de stockage NetApp . Il prend en charge le provisionnement et la gestion des ressources de stockage de la plateforme de stockage vers les hôtes Docker, avec un cadre permettant d'ajouter des plateformes supplémentaires à l'avenir.

Plusieurs instances de Trident peuvent s'exécuter simultanément sur le même hôte. Cela permet des connexions simultanées à plusieurs systèmes et types de stockage, avec la possibilité de personnaliser le stockage utilisé pour les volumes Docker.

Ce dont vous aurez besoin

Voir le "[conditions préalables au déploiement](#)". Une fois que vous vous êtes assuré que les conditions préalables sont remplies, vous êtes prêt à déployer Trident.

Méthode de plugin géré par Docker (version 1.13/17.03 et ultérieures)

Avant de commencer



Si vous avez utilisé Trident avant Docker 1.13/17.03 dans la méthode traditionnelle du démon, assurez-vous d'arrêter le processus Trident et de redémarrer votre démon Docker avant d'utiliser la méthode du plugin géré.

1. Arrêtez toutes les instances en cours d'exécution :

```
killall /usr/local/bin/netappdvp
killall /usr/local/bin/trident
```

2. Redémarrez Docker.

```
systemctl restart docker
```

3. Assurez-vous d'avoir installé Docker Engine 17.03 (nouvelle version 1.13) ou une version ultérieure.

```
docker --version
```

Si votre version est obsolète, "[installer ou mettre à jour votre installation](#)".

Étapes

1. Créez un fichier de configuration et spécifiez les options comme suit :
 - ° `config`: Le nom de fichier par défaut est `config.json`, vous pouvez toutefois utiliser n'importe quel nom de votre choix en spécifiant le `config` option avec le nom de fichier. Le fichier de configuration doit être situé dans le `/etc/netappdvp` répertoire sur le système hôte.
 - ° `log-level`: Spécifiez le niveau de journalisation (`debug`, `info`, `warn`, `error`, `fatal`). La valeur par défaut est `info`.

- debug: Indiquez si la journalisation de débogage est activée. La valeur par défaut est fausse. Remplace le niveau de journalisation si la valeur est vraie.

- i. Créez un emplacement pour le fichier de configuration :

```
sudo mkdir -p /etc/netappdvp
```

- ii. Créez le fichier de configuration :

```
cat << EOF > /etc/netappdvp/config.json
```

```
{  
  "version": 1,  
  "storageDriverName": "ontap-nas",  
  "managementLIF": "10.0.0.1",  
  "dataLIF": "10.0.0.2",  
  "svm": "svm_nfs",  
  "username": "vsadmin",  
  "password": "password",  
  "aggregate": "aggr1"  
}  
EOF
```

2. Démarrez Trident en utilisant le système de plugins gérés. Remplacer <version> avec la version du plugin (xxx.xx.x) que vous utilisez.

```
docker plugin install --grant-all-permissions --alias netapp  
netapp/trident-plugin:<version> config=myConfigFile.json
```

3. Commencez à utiliser Trident pour consommer du stockage du système configuré.

- a. Créez un volume nommé « firstVolume » :

```
docker volume create -d netapp --name firstVolume
```

- b. Créer un volume par défaut au démarrage du conteneur :

```
docker run --rm -it --volume-driver netapp --volume  
secondVolume:/my_vol alpine ash
```

- c. Supprimer le volume « firstVolume » :

```
docker volume rm firstVolume
```

Méthode traditionnelle (version 1.12 ou antérieure)

Avant de commencer

1. Assurez-vous d'avoir la version 1.10 ou ultérieure de Docker.

```
docker --version
```

Si votre version est obsolète, mettez à jour votre installation.

```
curl -fsSL https://get.docker.com/ | sh
```

Ou, "[Suivez les instructions pour votre distribution](#)".

2. Assurez-vous que NFS et/ou iSCSI sont configurés sur votre système.

Étapes

1. Installez et configurez le plugin NetApp Docker Volume :
 - a. Téléchargez et décompressez l'application :

```
wget  
https://github.com/NetApp/trident/releases/download/v25.06.0/trident-  
installer-25.06.0.tar.gz  
tar xzf trident-installer-25.06.0.tar.gz
```

- b. Déplacez-vous vers un emplacement dans le chemin du bac :

```
sudo mv trident-installer/extras/bin/trident /usr/local/bin/  
sudo chown root:root /usr/local/bin/trident  
sudo chmod 755 /usr/local/bin/trident
```

- c. Créez un emplacement pour le fichier de configuration :

```
sudo mkdir -p /etc/netappdvp
```

- d. Créez le fichier de configuration :

```
cat << EOF > /etc/netappdvp/ontap-nas.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
EOF
```

2. Après avoir placé le fichier binaire et créé le fichier de configuration, démarrez le démon Trident en utilisant le fichier de configuration souhaité.

```
sudo trident --config=/etc/netappdvp/ontap-nas.json
```



Sauf indication contraire, le nom par défaut du pilote de volume est « netapp ».

Une fois le démon démarré, vous pouvez créer et gérer des volumes à l'aide de l'interface de ligne de commande Docker.

3. Créer un volume :

```
docker volume create -d netapp --name trident_1
```

4. Provisionnez un volume Docker lors du démarrage d'un conteneur :

```
docker run --rm -it --volume-driver netapp --volume trident_2:/my_vol
alpine ash
```

5. Supprimer un volume Docker :

```
docker volume rm trident_1
```

```
docker volume rm trident_2
```

Démarrage de Trident au démarrage du système

Un exemple de fichier d'unité pour les systèmes basés sur systemd est disponible à l'adresse suivante :

`contrib/trident.service.example` dans le dépôt Git. Pour utiliser le fichier avec RHEL, procédez comme suit :

1. Copiez le fichier à l'emplacement approprié.

Vous devez utiliser des noms uniques pour les fichiers d'unité si vous avez plusieurs instances en cours d'exécution.

```
cp contrib/trident.service.example
/usr/lib/systemd/system/trident.service
```

2. Modifiez le fichier, changez la description (ligne 2) pour qu'elle corresponde au nom du pilote et le chemin du fichier de configuration (ligne 9) pour qu'il reflète votre environnement.
3. Rechargez `systemd` pour qu'il prenne en compte les modifications :

```
systemctl daemon-reload
```

4. Activez le service.

Ce nom varie en fonction du nom que vous avez donné au fichier dans le `/usr/lib/systemd/system` annuaire.

```
systemctl enable trident
```

5. Démarrer le service.

```
systemctl start trident
```

6. Consultez le statut.

```
systemctl status trident
```



Chaque fois que vous modifiez le fichier d'unité, exécutez le `systemctl daemon-reload` commande pour qu'il prenne connaissance des changements.

Mettez à jour ou désinstallez Trident

Vous pouvez mettre à niveau Trident pour Docker en toute sécurité, sans aucun impact sur les volumes en cours d'utilisation. Durant le processus de mise à niveau, il y aura une brève période pendant laquelle `docker volume` Les commandes adressées au plugin échoueront et les applications ne pourront pas monter de volumes tant que le plugin ne sera pas de nouveau en cours d'exécution. Dans la plupart des cas, cela se prend en

quelques secondes.

Mise à niveau

Suivez les étapes ci-dessous pour mettre à niveau Trident pour Docker.

Étapes

1. Liste des volumes existants :

```
docker volume ls
DRIVER          VOLUME NAME
netapp:latest   my_volume
```

2. Désactiver le plugin :

```
docker plugin disable -f netapp:latest
docker plugin ls
ID                NAME          DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest nDVP - NetApp Docker Volume
Plugin   false
```

3. Mettre à jour le plugin :

```
docker plugin upgrade --skip-remote-check --grant-all-permissions
netapp:latest netapp/trident-plugin:21.07
```



La version 18.01 de Trident remplace le nDVP. Vous devriez effectuer la mise à niveau directement depuis le `netapp/ndvp-plugin` l'image à l' `netapp/trident-plugin` image.

4. Activez le plugin :

```
docker plugin enable netapp:latest
```

5. Vérifiez que le plugin est activé :

```
docker plugin ls
ID                NAME          DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest Trident - NetApp Docker Volume
Plugin   true
```

6. Vérifiez que les volumes sont visibles :

```
docker volume ls
DRIVER          VOLUME NAME
netapp:latest   my_volume
```



Si vous effectuez une mise à niveau depuis une ancienne version de Trident (antérieure à la version 20.10) vers Trident 20.10 ou une version ultérieure, vous pourriez rencontrer une erreur. Pour plus d'informations, veuillez consulter "[Problèmes connus](#)". Si vous rencontrez cette erreur, vous devez d'abord désactiver le plugin, puis le supprimer, et enfin installer la version requise de Trident en passant un paramètre de configuration supplémentaire : `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all -permissions config=config.json`

Désinstaller

Suivez les étapes ci-dessous pour désinstaller Trident pour Docker.

Étapes

1. Supprimez tous les volumes créés par le plugin.
2. Désactiver le plugin :

```
docker plugin disable netapp:latest
docker plugin ls
ID                NAME                DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Supprimez le plugin :

```
docker plugin rm netapp:latest
```

Travailler avec des volumes

Vous pouvez facilement créer, cloner et supprimer des volumes à l'aide de la norme `docker volume` commandes avec le nom du pilote Trident spécifié lorsque nécessaire.

Créer un volume

- Créez un volume avec un pilote utilisant le nom par défaut :

```
docker volume create -d netapp --name firstVolume
```

- Créer un volume avec une instance Trident spécifique :

```
docker volume create -d ntap_bronze --name bronzeVolume
```



Si vous ne spécifiez rien "options", les paramètres par défaut du pilote sont utilisés.

- Remplacez la taille de volume par défaut. Consultez l'exemple suivant pour créer un volume de 20 Gio avec un pilote :

```
docker volume create -d netapp --name my_vol --opt size=20G
```



Les tailles de volume sont exprimées sous forme de chaînes de caractères contenant une valeur entière avec des unités facultatives (exemple : 10G, 20GB, 3TiB). Si aucune unité n'est spécifiée, la valeur par défaut est G. Les unités de taille peuvent être exprimées soit en puissances de 2 (B, KiB, MiB, GiB, TiB), soit en puissances de 10 (B, KB, MB, GB, TB). Les unités abrégées utilisent des puissances de 2 (G = GiB, T = TiB, ...).

Supprimer un volume

- Supprimez le volume comme n'importe quel autre volume Docker :

```
docker volume rm firstVolume
```



Lors de l'utilisation du `solidfire-san` Le pilote, dans l'exemple ci-dessus, supprime et purge le volume.

Suivez les étapes ci-dessous pour mettre à niveau Trident pour Docker.

Cloner un volume

Lors de l'utilisation du `ontap-nas`, `ontap-san`, `solidfire-san`, et `gcp-cvs storage drivers` Trident peut cloner des volumes. Lors de l'utilisation du `ontap-nas-flexgroup` ou `ontap-nas-economy` Le clonage des pilotes n'est pas pris en charge. La création d'un nouveau volume à partir d'un volume existant entraînera la création d'un nouvel instantané.

- Examinez le volume pour énumérer les instantanés :

```
docker volume inspect <volume_name>
```

- Créer un nouveau volume à partir d'un volume existant. Cela entraînera la création d'un nouvel

instantané :

```
docker volume create -d <driver_name> --name <new_name> -o from  
=<source_docker_volume>
```

- Créer un nouveau volume à partir d'un instantané existant sur un volume. Cela ne créera pas de nouvel instantané :

```
docker volume create -d <driver_name> --name <new_name> -o from  
=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

Exemple

```

docker volume inspect firstVolume

[
  {
    "Driver": "ontap-nas",
    "Labels": null,
    "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
    "Name": "firstVolume",
    "Options": {},
    "Scope": "global",
    "Status": {
      "Snapshots": [
        {
          "Created": "2017-02-10T19:05:00Z",
          "Name": "hourly.2017-02-10_1505"
        }
      ]
    }
  }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap

```

Accéder aux volumes créés en externe

Vous pouvez accéder aux périphériques de stockage par blocs créés en externe (ou à leurs clones) par des conteneurs utilisant Trident **uniquement** s'ils ne comportent aucune partition et si leur système de fichiers est pris en charge par Trident (par exemple : un ext4 -formaté /dev/sdc1 ne sera pas accessible via Trident).

Options de volume spécifiques au conducteur

Chaque pilote de stockage possède un ensemble d'options différent, que vous pouvez spécifier lors de la création du volume pour personnaliser le résultat. Vous trouverez ci-dessous les options qui s'appliquent à votre système de stockage configuré.

L'utilisation de ces options lors de la création d'un volume est simple. Indiquez l'option et la valeur en utilisant `-o` opérateur pendant l'opération CLI. Ces valeurs remplacent toutes les valeurs équivalentes du fichier de

configuration JSON.

options de volume ONTAP

Les options de création de volumes pour NFS, iSCSI et FC incluent les suivantes :

Option	Description
<code>size</code>	La taille du volume est par défaut de 1 Gio.
<code>spaceReserve</code>	Le volume peut être mince ou épais ; par défaut, mince. Les valeurs valides sont <code>none</code> (à faible capacité) et <code>volume</code> (provisionné épais).
<code>snapshotPolicy</code>	Cela définira la politique de capture d'écran sur la valeur souhaitée. La valeur par défaut est <code>none</code> , ce qui signifie qu'aucun instantané ne sera automatiquement créé pour le volume. Sauf modification par votre administrateur de stockage, une politique nommée « default » existe sur tous les systèmes ONTAP qui crée et conserve six instantanés horaires, deux instantanés quotidiens et deux instantanés hebdomadaires. Les données conservées dans un instantané peuvent être récupérées en accédant à l'URL <code>.snapshot</code> répertoire dans n'importe quel répertoire du volume.
<code>snapshotReserve</code>	Cela permettra de définir la réserve d'instantanés au pourcentage souhaité. La valeur par défaut est aucune, ce qui signifie ONTAP sélectionnera <code>snapshotReserve</code> (généralement 5 %) si vous avez sélectionné une <code>snapshotPolicy</code> , ou 0 % si aucune <code>snapshotPolicy</code> n'est définie. Vous pouvez définir la valeur par défaut de <code>snapshotReserve</code> dans le fichier de configuration pour tous les backends ONTAP, et vous pouvez l'utiliser comme option de création de volume pour tous les backends ONTAP à l'exception de <code>ontap-nas-economy</code> .
<code>splitOnClone</code>	Lors du clonage d'un volume, ONTAP séparera immédiatement le clone de son parent. La valeur par défaut est <code>false</code> . Dans certains cas d'utilisation du clonage de volumes, il est préférable de séparer immédiatement le clone de son parent dès sa création, car il est peu probable qu'il y ait des possibilités d'optimisation du stockage. Par exemple, le clonage d'une base de données vide peut permettre un gain de temps important mais un gain de stockage minime ; il est donc préférable de scinder immédiatement le clone.

Option	Description
encryption	<p>Activez le chiffrement de volume NetApp (NVE) sur le nouveau volume ; la valeur par défaut est <code>false</code> .</p> <p>Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster.</p> <p>Si NAE est activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NAE.</p> <p>Pour plus d'informations, veuillez consulter : "Comment Trident fonctionne avec NVE et NAE" .</p>
tieringPolicy	Définit la politique de hiérarchisation à utiliser pour le volume. Cela détermine si les données sont déplacées vers le niveau cloud lorsqu'elles deviennent inactives (froides).

Les options supplémentaires suivantes concernent **uniquement** NFS :

Option	Description
unixPermissions	Ceci contrôle les autorisations définies pour le volume lui-même. Par défaut, les autorisations seront définies sur <code>---rwxr-xr-x</code> , ou en notation numérique <code>0755</code> , et <code>root</code> sera le propriétaire. Le format texte ou numérique conviendra.
snapshotDir	Définir ceci à <code>true</code> fera le <code>.snapshot</code> Répertoire visible par les clients accédant au volume. La valeur par défaut est <code>false</code> , ce qui signifie que la visibilité de <code>.snapshot</code> Ce répertoire est désactivé par défaut. Certaines images, par exemple l'image officielle de MySQL, ne fonctionnent pas comme prévu lorsque <code>.snapshot</code> Le répertoire est visible.
exportPolicy	Définit la politique d'exportation à utiliser pour le volume. La valeur par défaut est <code>default</code> .
securityStyle	Définit le style de sécurité à utiliser pour l'accès au volume. La valeur par défaut est <code>unix</code> . Les valeurs valides sont <code>unix</code> et <code>mixed</code> .

Les options supplémentaires suivantes concernent uniquement iSCSI :

Option	Description
fileSystemType	Définit le système de fichiers utilisé pour formater les volumes iSCSI. La valeur par défaut est <code>ext4</code> . Les valeurs valides sont <code>ext3</code> , <code>ext4</code> , et <code>xfs</code> .
spaceAllocation	Définir ceci à <code>false</code> désactivera la fonction d'allocation d'espace du LUN. La valeur par défaut est <code>true</code> , ce qui signifie ONTAP notifie l'hôte lorsque le volume est à court d'espace et que le LUN du volume ne peut plus accepter d'écritures. Cette option permet également à ONTAP de récupérer automatiquement l'espace lorsque votre hôte supprime des données.

Exemples

Voir les exemples ci-dessous :

- Créer un volume de 10 Gio :

```
docker volume create -d netapp --name demo -o size=10G -o encryption=true
```

- Créer un volume de 100 Gio avec des instantanés :

```
docker volume create -d netapp --name demo -o size=100G -o snapshotPolicy=default -o snapshotReserve=10
```

- Créez un volume dont le bit setUID est activé :

```
docker volume create -d netapp --name demo -o unixPermissions=4755
```

La taille minimale du volume est de 20 Mio.

Si la réserve d'instantanés n'est pas spécifiée et que la stratégie d'instantanés est `none` Trident utilise une réserve d'instantanés de 0 %.

- Créer un volume sans stratégie de snapshot et sans réserve de snapshot :

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
```

- Créez un volume sans stratégie de snapshot et avec une réserve de snapshot personnalisée de 10 % :

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none --opt snapshotReserve=10
```

- Créez un volume avec une stratégie de snapshot et une réserve de snapshot personnalisée de 10 % :

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy --opt snapshotReserve=10
```

- Créez un volume avec une stratégie de snapshot et acceptez la réserve de snapshot par défaut d'ONTAP (généralement 5 %) :

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

options de volume du logiciel Element

Les options du logiciel Element exposent la taille et les politiques de qualité de service (QoS) associées au volume. Lors de la création du volume, la politique QoS qui lui est associée est spécifiée à l'aide de `-o type=service_level nomenclature`.

La première étape pour définir un niveau de service QoS avec le pilote Element consiste à créer au moins un type et à spécifier les IOPS minimales, maximales et en rafale associées à un nom dans le fichier de configuration.

Les autres options de création de volumes du logiciel Element incluent les suivantes :

Option	Description
size	La taille du volume, par défaut 1 Gio ou entrée de configuration... "defaults": {"size": "5G"}.
blocksize	Utilisez 512 ou 4096, la valeur par défaut est 512 ou l'entrée de configuration DefaultBlockSize.

Exemple

Voir l'exemple de fichier de configuration suivant avec les définitions QoS :

```

{
  "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
      }
    }
  ]
}

```

Dans la configuration ci-dessus, nous avons trois définitions de politique : Bronze, Argent et Or. Ces noms sont arbitraires.

- Créer un volume Gold de 10 Gio :

```
docker volume create -d solidfire --name sfGold -o type=Gold -o size=10G
```

- Créer un volume Bronze de 100 Gio :

```
docker volume create -d solidfire --name sfBronze -o type=Bronze -o
size=100G
```

Collecter les bûches

Vous pouvez collecter les journaux pour faciliter le dépannage. La méthode utilisée pour collecter les journaux varie en fonction de la manière dont vous exécutez le plugin Docker.

Collecter les journaux pour le dépannage

Étapes

1. Si vous exécutez Trident en utilisant la méthode de plugin géré recommandée (c'est-à-dire en utilisant `docker plugin` (commandes), visualisez-les comme suit :

```
docker plugin ls
```

ID	NAME	DESCRIPTION
4fb97d2b956b	netapp:latest	nDVP - NetApp Docker Volume
ENABLED	Plugin	false

```
journalctl -u docker | grep 4fb97d2b956b
```

Le niveau de journalisation standard devrait vous permettre de diagnostiquer la plupart des problèmes. Si cela ne vous suffit pas, vous pouvez activer la journalisation de débogage.

2. Pour activer la journalisation de débogage, installez le plugin avec la journalisation de débogage activée :

```
docker plugin install netapp/trident-plugin:<version> --alias <alias>  
debug=true
```

Ou activez la journalisation de débogage même si le plugin est déjà installé :

```
docker plugin disable <plugin>
```

```
docker plugin set <plugin> debug=true
```

```
docker plugin enable <plugin>
```

3. Si vous exécutez le binaire lui-même sur l'hôte, les journaux sont disponibles dans le système de fichiers de l'hôte. `/var/log/netappdvp` annuaire. Pour activer la journalisation de débogage, spécifiez `-debug` lorsque vous exécutez le plugin.

Conseils généraux de dépannage

- Le problème le plus fréquent rencontré par les nouveaux utilisateurs est une mauvaise configuration qui empêche l'initialisation du plugin. Dans ce cas, vous verrez probablement un message comme celui-ci lorsque vous tenterez d'installer ou d'activer le plugin :

```
Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock:
connect: no such file or directory
```

Cela signifie que le plugin n'a pas pu démarrer. Heureusement, le plugin a été conçu avec une fonctionnalité de journalisation complète qui devrait vous aider à diagnostiquer la plupart des problèmes que vous êtes susceptible de rencontrer.

- En cas de problème lors du montage d'un panneau photovoltaïque sur un conteneur, assurez-vous que `rpcbind` est installé et fonctionne. Utilisez le gestionnaire de paquets requis pour le système d'exploitation hôte et vérifiez si `rpcbind` est en cours d'exécution. Vous pouvez vérifier l'état du service `rpcbind` en exécutant une `systemctl status rpcbind` ou son équivalent.

Gérer plusieurs instances Trident

Plusieurs instances de Trident sont nécessaires lorsque vous souhaitez disposer simultanément de plusieurs configurations de stockage. La clé pour gérer plusieurs instances est de leur donner des noms différents en utilisant le `--alias` option avec le plugin conteneurisé, ou `--volume-driver` option lors de l'instanciation de Trident sur l'hôte.

Étapes pour le plugin géré par Docker (version 1.13/17.03 ou ultérieure)

1. Lancez la première instance en spécifiant un alias et un fichier de configuration.

```
docker plugin install --grant-all-permissions --alias silver
netapp/trident-plugin:21.07 config=silver.json
```

2. Lancez la deuxième instance en spécifiant un alias et un fichier de configuration différents.

```
docker plugin install --grant-all-permissions --alias gold
netapp/trident-plugin:21.07 config=gold.json
```

3. Créez des volumes en spécifiant l'alias comme nom du pilote.

Par exemple, pour le volume d'or :

```
docker volume create -d gold --name ntapGold
```

Par exemple, pour le volume d'argent :

```
docker volume create -d silver --name ntapSilver
```

Étapes pour la version traditionnelle (1.12 ou antérieure)

1. Lancez le plugin avec une configuration NFS en utilisant un ID de pilote personnalisé :

```
sudo trident --volume-driver=netapp-nas --config=/path/to/config  
-nfs.json
```

2. Lancez le plugin avec une configuration iSCSI utilisant un ID de pilote personnalisé :

```
sudo trident --volume-driver=netapp-san --config=/path/to/config  
-iscsi.json
```

3. Provisionnez des volumes Docker pour chaque instance de pilote :

Par exemple, pour NFS :

```
docker volume create -d netapp-nas --name my_nfs_vol
```

Par exemple, pour iSCSI :

```
docker volume create -d netapp-san --name my_iscsi_vol
```

options de configuration du stockage

Consultez les options de configuration disponibles pour vos configurations Trident .

options de configuration globale

Ces options de configuration s'appliquent à toutes les configurations Trident , quelle que soit la plateforme de stockage utilisée.

Option	Description	Exemple
version	numéro de version du fichier de configuration	1

Option	Description	Exemple
<code>storageDriverName</code>	Nom du pilote de stockage	<code>ontap-nas</code> , <code>ontap-san</code> , <code>ontap-nas-economy</code> , <code>ontap-nas-flexgroup</code> , <code>solidfire-san</code>
<code>storagePrefix</code>	Préfixe optionnel pour les noms de volumes. Défaut: <code>netappdvp_</code> .	<code>staging_</code>
<code>limitVolumeSize</code>	Restriction optionnelle sur les volumes. Valeur par défaut : « » (non appliqué)	10g



Ne pas utiliser `storagePrefix` (y compris la valeur par défaut) pour les backends Element. Par défaut, le `solidfire-san` Le pilote ignorera ce paramètre et n'utilisera pas de préfixe. NetApp recommande d'utiliser soit un ID de locataire spécifique pour le mappage des volumes Docker, soit les données d'attribut renseignées avec la version de Docker, les informations du pilote et le nom brut de Docker dans les cas où un traitement de nommage a pu être utilisé.

Des options par défaut sont disponibles pour éviter d'avoir à les spécifier pour chaque volume que vous créez. Le `size` Cette option est disponible pour tous les types de manettes. Consultez la section relative à la configuration ONTAP pour un exemple de configuration de la taille de volume par défaut.

Option	Description	Exemple
<code>size</code>	Taille par défaut optionnelle pour les nouveaux volumes. Défaut: 1G	10G

Configuration ONTAP

En plus des valeurs de configuration globales ci-dessus, lors de l'utilisation ONTAP, les options de niveau supérieur suivantes sont disponibles.

Option	Description	Exemple
<code>managementLIF</code>	Adresse IP de l'interface logique de gestion ONTAP . Vous pouvez spécifier un nom de domaine pleinement qualifié (FQDN).	10.0.0.1

Option	Description	Exemple
dataLIF	<p>Adresse IP du protocole LIF.</p> <ul style="list-style-type: none"> • Pilotes ONTAP NAS * : 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. • Pilotes SAN ONTAP * : Ne pas spécifier pour iSCSI ou FC. Trident utilise "Carte LUN sélective ONTAP" pour découvrir les interfaces logiques iSCSI ou FC nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si dataLIF est explicitement défini. 	10.0.0.2
svm	Machine virtuelle de stockage à utiliser (obligatoire si l'interface LIF de gestion est une interface LIF de cluster)	svm_nfs
username	Nom d'utilisateur pour se connecter au périphérique de stockage	vsadmin
password	Mot de passe pour se connecter au périphérique de stockage	secret
aggregate	Agrégat pour le provisionnement (facultatif ; s'il est défini, il doit être affecté au SVM). Pour le <code>ontap-nas-flexgroup</code> conducteur, cette option est ignorée. Tous les agrégats affectés au SVM sont utilisés pour provisionner un volume FlexGroup .	aggr1

Option	Description	Exemple
limitAggregateUsage	Facultatif : l'approvisionnement peut échouer si l'utilisation dépasse ce pourcentage.	75%
nfsMountOptions	Contrôle précis des options de montage NFS ; par défaut, « -o nfsvers=3 ». Disponible uniquement pour le ontap-nas et ontap-nas-economy conducteurs. "Consultez ici les informations de configuration de l'hôte NFS" .	-o nfsvers=4
igroupName	Trident crée et gère par nœud igroups comme netappdvp . Cette valeur ne peut être ni modifiée ni omise. Disponible uniquement pour le ontap-san conducteur.	netappdvp
limitVolumeSize	Volume maximal requis.	300g
qtreesPerFlexvol	Nombre maximal de qtrees par FlexVol, doit être compris entre 50 et 300, la valeur par défaut est 200. Pour le ontap-nas-economy pilote, cette option permet de personnaliser le nombre maximal de qtrees par FlexVol.	300
sanType	Prise en charge pour ontap-san conducteur seulement. Utiliser pour sélectionner <code>iscsi</code> pour iSCSI, <code>nvme</code> pour NVMe/TCP ou <code>fc</code> pour SCSI sur Fibre Channel (FC).	<code>`iscsi`</code> si vide
limitVolumePoolSize	Prise en charge pour ontap-san-economy et ontap-san-economy conducteurs seulement. Limite les tailles FlexVol dans les pilotes ONTAP ontap-nas-economy et ontap-SAN-economy.	300g

Des options par défaut sont disponibles pour éviter d'avoir à les spécifier sur chaque volume que vous créez :

Option	Description	Exemple
spaceReserve	Mode de réservation d'espace ; none (à provisionnement limité) ou volume (épais)	none
snapshotPolicy	Stratégie d'instantané à utiliser, la valeur par défaut est none	none
snapshotReserve	Pourcentage de réserve d'instantané, la valeur par défaut est « » pour accepter la valeur par défaut ONTAP	10
splitOnClone	Séparer le clone de son parent lors de sa création, par défaut à false	false
encryption	Active 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 " .	true
unixPermissions	Option NAS pour les volumes NFS provisionnés, par défaut : 777	777
snapshotDir	Option NAS pour accéder au .snapshot annuaire.	« Vrai » pour NFSv4, « Faux » pour NFSv3
exportPolicy	Option NAS pour la stratégie d'exportation NFS à utiliser, par défaut : default	default
securityStyle	Option NAS pour accéder au volume NFS provisionné. NFS prend en charge mixed et unix Styles de sécurité. La valeur par défaut est unix .	unix
fileSystemType	Option SAN permettant de sélectionner le type de système de fichiers, par défaut : ext4	xf
tieringPolicy	Politique de hiérarchisation à utiliser, la valeur par défaut est none .	none

Options de mise à l'échelle

Le `ontap-nas` et `ontap-san` Les pilotes créent un ONTAP FlexVol pour chaque volume Docker. ONTAP prend en charge jusqu'à 1 000 volumes FlexVol par nœud de cluster, avec un maximum de 12 000 volumes FlexVol par cluster. Si vos besoins en volumes Docker respectent cette limite, `ontap-nas` Le pilote est la solution NAS privilégiée en raison des fonctionnalités supplémentaires offertes par FlexVols, telles que les instantanés granulaires de volume Docker et le clonage.

Si vous avez besoin de plus de volumes Docker que ne le permettent les limites de FlexVol , choisissez le `ontap-nas-economy` ou le `ontap-san-economy` conducteur.

Le `ontap-nas-economy` Le pilote crée des volumes Docker sous forme d'arbres Qtree ONTAP au sein d'un pool de volumes FlexVol gérés automatiquement. Les Qtrees offrent une évolutivité bien supérieure, jusqu'à 100 000 par nœud de cluster et 2 400 000 par cluster, au détriment de certaines fonctionnalités. Le `ontap-nas-economy` Le pilote ne prend pas en charge les instantanés granulaires de volume Docker ni le clonage.



Le `ontap-nas-economy` Ce pilote n'est actuellement pas pris en charge dans Docker Swarm, car Docker Swarm n'orchestre pas la création de volumes sur plusieurs nœuds.

Le `ontap-san-economy` Le pilote crée des volumes Docker sous forme de LUN ONTAP au sein d'un pool partagé de volumes FlexVol gérés automatiquement. Ainsi, chaque FlexVol n'est pas limité à un seul LUN et offre une meilleure évolutivité pour les charges de travail SAN. Selon la baie de stockage, ONTAP prend en charge jusqu'à 16 384 LUN par cluster. Étant donné que les volumes sont des LUN sous-jacents, ce pilote prend en charge les instantanés et le clonage granulaires des volumes Docker.

Choisissez le `ontap-nas-flexgroup` pilote permettant d'accroître le parallélisme vers un volume unique pouvant atteindre plusieurs pétaoctets avec des milliards de fichiers. FlexGroups peut notamment être utilisé dans des cas d'usage tels que l'IA/ML/DL, le big data et l'analyse de données, la compilation de logiciels, le streaming, les référentiels de fichiers, etc. Trident utilise tous les agrégats affectés à une SVM lors de la mise en service d'un volume FlexGroup . La prise en charge de FlexGroup dans Trident présente également les considérations suivantes :

- Nécessite ONTAP version 9.2 ou supérieure.
- À l'heure actuelle, FlexGroups ne prend en charge que NFS v3.
- Il est recommandé d'activer les identifiants NFSv3 64 bits pour la SVM.
- La taille minimale recommandée du membre/volume FlexGroup est de 100 Gio.
- Le clonage n'est pas pris en charge pour les volumes FlexGroup .

Pour plus d'informations sur les FlexGroups et les charges de travail compatibles avec les FlexGroups, veuillez consulter la documentation. ["Guide des bonnes pratiques et de mise en œuvre de NetApp FlexGroup"](#)

Pour bénéficier de fonctionnalités avancées et d'une grande capacité dans un même environnement, vous pouvez exécuter plusieurs instances du plugin Docker Volume, dont une utilisant `ontap-nas` et un autre utilisant `ontap-nas-economy` .

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"  
security login create -username <user_name\> -application http -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](#)"

Exemples de fichiers de configuration ONTAP

Exemple NFS pour le pilote `ontap-nas`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "defaults": {
    "size": "10G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Exemple NFS pour le pilote `ontap-nas-flexgroup`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-flexgroup",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "defaults": {
    "size": "100G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Exemple NFS pour le pilote `ontap-nas-economy`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
```

Exemple iSCSI pour le pilote `ontap-san`

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Exemple NFS pour le pilote `ontap-san-economy`

```
{
  "version": 1,
  "storageDriverName": "ontap-san-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi_eco",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Exemple NVMe/TCP pour le pilote `ontap-san`

```
{
  "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 pour le pilote `ontap-san`

```
{
  "version": 1,
  "backendName": "ontap-san-backend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "sanType": "fc",
  "svm": "trident_svm",
  "username": "vsadmin",
  "password": "password",
  "useREST": true
}
```

Configuration du logiciel Element

En plus des valeurs de configuration globales, lors de l'utilisation du logiciel Element (NetApp HCI/ SolidFire), ces options sont disponibles.

Option	Description	Exemple
Endpoint	<code>https://&lt;login&gt;:&lt;password&gt;@&lt;mvip&gt;/json-rpc/&lt;element-version&gt;</code>	https://admin:admin@192.168.160.3/json-rpc/8.0

Option	Description	Exemple
SVIP	Adresse IP et port iSCSI	10.0.0.7:3260
TenantName	Locataire SolidFireF à utiliser (créé s'il n'est pas trouvé)	docker
InitiatorIFace	Spécifiez l'interface lorsque vous limitez le trafic iSCSI à une interface autre que celle par défaut.	default
Types	Spécifications QoS	Voir l'exemple ci-dessous
LegacyNamePrefix	Préfixe pour les installations Trident mises à niveau. Si vous avez utilisé une version de Trident antérieure à la 1.3.2 et que vous effectuez une mise à niveau avec des volumes existants, vous devrez définir cette valeur pour accéder à vos anciens volumes qui étaient mappés via la méthode nom-volume.	netappdvp-

Le `solidfire-san` Le pilote ne prend pas en charge Docker Swarm.

Exemple de fichier de configuration logicielle Element

```

{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
  "SVIP": "10.0.0.7:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "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
      }
    }
  ]
}

```

Problèmes connus et limitations

Trouvez des informations sur les problèmes connus et les limitations liées à l'utilisation de Trident avec Docker.

La mise à niveau du plugin Trident Docker Volume vers la version 20.10 et ultérieures à partir de versions plus anciennes entraîne un échec de mise à niveau avec l'erreur « fichier ou répertoire introuvable ».

Solution de contournement

1. Désactivez le plugin.

```
docker plugin disable -f netapp:latest
```

2. Supprimez le plugin.

```
docker plugin rm -f netapp:latest
```

3. Réinstallez le plugin en fournissant les informations supplémentaires `config` paramètre.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant  
-all-permissions config=config.json
```

Les noms de volumes doivent comporter au minimum 2 caractères.



Il s'agit d'une limitation du client Docker. Le client interprétera un nom d'un seul caractère comme étant un chemin d'accès Windows. "[Voir le bogue 25773](#)".

Docker Swarm présente certains comportements qui empêchent Trident de le prendre en charge avec toutes les combinaisons de stockage et de pilotes.

- Docker Swarm utilise actuellement le nom du volume au lieu de son ID comme identifiant unique.
- Les requêtes de volume sont envoyées simultanément à chaque nœud d'un cluster Swarm.
- Les plugins de volume (y compris Trident) doivent s'exécuter indépendamment sur chaque nœud d'un cluster Swarm. En raison du fonctionnement ONTAP et de la manière dont `ontap-nas` et `ontap-san` Les conducteurs sont les seuls à pouvoir fonctionner dans ces limites.

Les autres pilotes sont sujets à des problèmes tels que des conditions de concurrence qui peuvent entraîner la création d'un grand nombre de volumes pour une seule requête sans « gagnant » clair ; par exemple, Element possède une fonctionnalité qui permet à des volumes d'avoir le même nom mais des identifiants différents.

NetApp a fait part de ses commentaires à l'équipe Docker, mais n'a donné aucune indication quant aux recours futurs possibles.

Lors de la mise en service d'un FlexGroup , ONTAP ne met pas en service un deuxième FlexGroup si ce dernier a un ou plusieurs FlexGroup en commun avec le FlexGroup en cours de mise en service.

Meilleures pratiques et recommandations

Déploiement

Utilisez les recommandations énumérées ici lors du déploiement de Trident.

Déployer dans un espace de noms dédié

"Espaces de noms"Elles assurent une séparation administrative entre les différentes applications et constituent un obstacle au partage des ressources. Par exemple, un PVC d'un espace de noms ne peut pas être consommé depuis un autre. Trident fournit des ressources PV à tous les espaces de noms du cluster Kubernetes et exploite par conséquent un compte de service disposant de privilèges élevés.

De plus, l'accès au module Trident pourrait permettre à un utilisateur d'accéder aux identifiants du système de stockage et à d'autres informations sensibles. Il est important de veiller à ce que les utilisateurs de l'application et les applications de gestion n'aient pas la possibilité d'accéder aux définitions d'objets Trident ni aux pods eux-mêmes.

Utilisez des quotas et des limites de plage pour contrôler la consommation de stockage.

Kubernetes possède deux fonctionnalités qui, combinées, constituent un mécanisme puissant pour limiter la consommation de ressources par les applications. Le **"mécanisme de quotas de stockage"** permet à l'administrateur de mettre en œuvre des limites de consommation de capacité et de nombre d'objets globales et spécifiques à la classe de stockage, sur une base d'espace de noms. De plus, en utilisant un **"limite de portée"** garantit que les demandes de PVC respectent une valeur minimale et maximale avant d'être transmises au fournisseur.

Ces valeurs sont définies pour chaque espace de noms, ce qui signifie que chaque espace de noms doit avoir des valeurs définies qui correspondent à ses besoins en ressources. Consultez cette page pour obtenir des informations sur **"comment tirer parti des quotas"** .

Configuration de stockage

Chaque plateforme de stockage du portefeuille NetApp possède des capacités uniques qui profitent aux applications, conteneurisées ou non.

Présentation de la plateforme

Trident fonctionne avec ONTAP et Element. Il n'existe pas de plateforme mieux adaptée à toutes les applications et à tous les scénarios qu'une autre ; toutefois, les besoins de l'application et de l'équipe administrant l'appareil doivent être pris en compte lors du choix d'une plateforme.

Vous devez suivre les bonnes pratiques de base pour le système d'exploitation hôte avec le protocole que vous utilisez. Vous pouvez également envisager d'intégrer, le cas échéant, les meilleures pratiques d'application aux paramètres backend, de classe de stockage et de PVC afin d'optimiser le stockage pour des applications spécifiques.

ONTAP et Cloud Volumes ONTAP

Découvrez les meilleures pratiques pour configurer ONTAP et Cloud Volumes ONTAP pour Trident.

Les recommandations suivantes sont des lignes directrices pour la configuration ONTAP pour les charges de travail conteneurisées, qui consomment des volumes provisionnés dynamiquement par Trident. Chacune d'entre elles doit être envisagée et évaluée en fonction de sa pertinence dans votre environnement.

Utilisez des SVM dédiés à Trident

Les machines virtuelles de stockage (SVM) fournissent une isolation et une séparation administrative entre les locataires sur un système ONTAP . L'affectation d'une SVM aux applications permet la délégation de privilèges et l'application des meilleures pratiques pour limiter la consommation de ressources.

Plusieurs options sont disponibles pour la gestion du SVM :

- Fournissez l'interface de gestion du cluster dans la configuration du backend, ainsi que les informations d'identification appropriées, et spécifiez le nom du SVM.
- Créez une interface de gestion dédiée pour la SVM en utilisant ONTAP System Manager ou l'interface de ligne de commande (CLI).
- Partagez le rôle de gestion avec une interface de données NFS.

Dans chaque cas, l'interface doit être configurée dans le DNS, et le nom DNS doit être utilisé lors de la configuration de Trident. Cela permet de faciliter certains scénarios de reprise après sinistre, par exemple SVM-DR sans utilisation de la conservation de l'identité du réseau.

Il n'y a pas de préférence entre une LIF de gestion dédiée ou partagée pour la SVM ; cependant, vous devez vous assurer que vos politiques de sécurité réseau sont alignées sur l'approche que vous choisissez. Quoi qu'il en soit, l'interface LIF de gestion doit être accessible via DNS afin de faciliter une flexibilité maximale. "SVM-DR" être utilisé en conjonction avec Trident.

Limiter le nombre de volumes maximum

Les systèmes de stockage ONTAP ont un nombre maximal de volumes, qui varie en fonction de la version du logiciel et de la plateforme matérielle. Se référer à "[Hardware Universe NetApp](#)" pour votre plateforme et votre version ONTAP spécifiques afin de déterminer les limites exactes. Lorsque le nombre de volumes est épuisé, les opérations de provisionnement échouent non seulement pour Trident, mais pour toutes les demandes de stockage.

Trident `ontap-nas` et `ontap-san` Les pilotes provisionnent un FlexVolume pour chaque volume persistant Kubernetes (PV) créé. Le `ontap-nas-economy` Le pilote crée environ un FlexVolume pour 200 PV (configurable entre 50 et 300). Le `ontap-san-economy` Le pilote crée environ un FlexVolume pour 100 PV (configurable entre 50 et 200). Pour empêcher Trident de consommer tous les volumes disponibles sur le système de stockage, vous devez définir une limite sur le SVM. Vous pouvez le faire à partir de la ligne de commande :

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

La valeur de `max-volumes` varie en fonction de plusieurs critères propres à votre environnement :

- Le nombre de volumes existants dans le cluster ONTAP

- Le nombre de volumes que vous prévoyez de provisionner en dehors de Trident pour d'autres applications
- Le nombre de volumes persistants qui devraient être consommés par les applications Kubernetes

Le `max-volumes` La valeur correspond au volume total provisionné sur l'ensemble des nœuds du cluster ONTAP , et non sur un nœud ONTAP individuel. Par conséquent, vous pouvez rencontrer certaines conditions dans lesquelles un nœud de cluster ONTAP peut avoir beaucoup plus ou moins de volumes provisionnés Trident qu'un autre nœud.

Par exemple, un cluster ONTAP à deux nœuds peut héberger un maximum de 2000 volumes FlexVol . Fixer le nombre maximal de volumes à 1250 semble tout à fait raisonnable. Cependant, si seulement "agrégats" Si les agrégats d'un nœud sont affectés à la SVM ou ne peuvent pas être provisionnés (par exemple, en raison de la capacité), alors l'autre nœud devient la cible pour tous les volumes provisionnés Trident . Cela signifie que la limite de volume pourrait être atteinte pour ce nœud avant le `max-volumes` La valeur est atteinte, ce qui a un impact sur Trident et sur les autres opérations de volume utilisant ce nœud. **Vous pouvez éviter cette situation en veillant à ce que les agrégats de chaque nœud du cluster soient affectés en nombre égal au SVM utilisé par Trident .**

Cloner un volume

NetApp Trident prend en charge le clonage de volumes lors de l'utilisation de `ontap-nas` , `ontap-san` , `solidfire-san` , et `gcp-cvs` pilotes de stockage. Lors de l'utilisation du `ontap-nas-flexgroup` ou `ontap-nas-economy` Le clonage des pilotes n'est pas pris en charge. La création d'un nouveau volume à partir d'un volume existant entraînera la création d'un nouvel instantané.



Évitez de cloner un PVC associé à une StorageClass différente. Effectuez les opérations de clonage au sein de la même StorageClass afin de garantir la compatibilité et d'éviter tout comportement inattendu.

Limiter la taille maximale des volumes créés par Trident

Pour configurer la taille maximale des volumes pouvant être créés par Trident, utilisez le `limitVolumeSize` paramètre dans votre `backend.json` définition.

En plus de contrôler la taille du volume au niveau de la baie de stockage, vous devez également exploiter les fonctionnalités de Kubernetes.

Limiter la taille maximale des FlexVols créés par Trident

Pour configurer la taille maximale des FlexVols utilisés comme pools pour les pilotes `ontap-san-economy` et `ontap-nas-economy`, utilisez le `limitVolumePoolSize` paramètre dans votre `backend.json` définition.

Configurez Trident pour utiliser CHAP bidirectionnel

Vous pouvez spécifier les noms d'utilisateur et les mots de passe de l'initiateur et de la cible CHAP dans votre définition `backend` et demander à Trident d'activer CHAP sur la SVM. En utilisant le `useCHAP` Dans votre configuration `backend`, Trident authentifie les connexions iSCSI pour les backends ONTAP avec CHAP.

Créer et utiliser une politique QoS SVM

L'utilisation d'une politique QoS ONTAP , appliquée au SVM, limite le nombre d'IOPS consommables par les volumes provisionnés Trident . Cela contribue à "prévenir l'intimidation" ou un conteneur incontrôlé susceptible d'affecter les charges de travail en dehors du SVM Trident .

Vous pouvez créer une politique QoS pour la SVM en quelques étapes. Pour obtenir les informations les plus précises, veuillez consulter la documentation correspondant à votre version d' ONTAP . L'exemple ci-dessous crée une politique QoS qui limite le nombre total d'IOPS disponibles pour la SVM à 5000.

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

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

De plus, si votre version d' ONTAP le prend en charge, vous pouvez envisager d'utiliser un minimum de QoS pour garantir un certain débit aux charges de travail conteneurisées. La QoS adaptative n'est pas compatible avec une politique au niveau SVM.

Le nombre d'IOPS dédiés aux charges de travail conteneurisées dépend de nombreux facteurs. Cela comprend notamment :

- Autres charges de travail utilisant la baie de stockage. Si d'autres charges de travail, non liées au déploiement Kubernetes, utilisent les ressources de stockage, il convient de veiller à ce que ces charges de travail ne soient pas accidentellement affectées négativement.
- Charges de travail attendues exécutées dans des conteneurs. Si des charges de travail ayant des exigences élevées en matière d'IOPS s'exécutent dans des conteneurs, une politique de QoS faible se traduira par une mauvaise expérience.

Il est important de rappeler qu'une politique QoS attribuée au niveau de la SVM a pour conséquence que tous les volumes provisionnés sur la SVM partagent le même pool d'IOPS. Si une ou quelques applications conteneurisées ont des exigences élevées en matière d'IOPS, elles pourraient devenir un frein pour les autres charges de travail conteneurisées. Dans ce cas, vous pourriez envisager d'utiliser une automatisation externe pour attribuer des politiques QoS par volume.



Vous ne devez attribuer le groupe de stratégie QoS au SVM que si votre version ONTAP est antérieure à 9.8.

Créer des groupes de stratégies QoS pour Trident

La qualité de service (QoS) garantit que les performances des charges de travail critiques ne sont pas dégradées par des charges de travail concurrentes. Les groupes de politiques QoS ONTAP offrent des options QoS pour les volumes et permettent aux utilisateurs de définir le plafond de débit pour une ou plusieurs charges de travail. Pour plus d'informations sur la QoS, consultez "[Garantir le débit grâce à la QoS](#)". Vous pouvez spécifier des groupes de stratégies QoS dans le système dorsal ou dans un pool de stockage, et ils sont appliqués à chaque volume créé dans ce pool ou ce système dorsal.

ONTAP propose deux types de groupes de politiques QoS : traditionnels et adaptatifs. Les groupes de politiques traditionnels offrent un débit maximal (ou minimal, dans les versions ultérieures) fixe en IOPS. La QoS adaptative ajuste automatiquement le débit à la taille de la charge de travail, en maintenant le ratio IOPS/TB/GB à mesure que la taille de la charge de travail change. Cela représente un avantage considérable lorsque vous gérez des centaines, voire des milliers, de charges de travail dans un déploiement de grande envergure.

Tenez compte des éléments suivants lors de la création de groupes de stratégies QoS :

- Vous devriez paramétrer le `qosPolicy` clé dans le `defaults` bloc de la configuration du backend. Voir l'exemple de configuration backend suivant :

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

- Vous devez appliquer les groupes de stratégies par volume, afin que chaque volume bénéficie du débit total spécifié par le groupe de stratégies. Les groupes de politiques partagées ne sont pas pris en charge.

Pour plus d'informations sur les groupes de politiques QoS, consultez ["Référence des commandes ONTAP"](#) .

Limiter l'accès aux ressources de stockage aux membres du cluster Kubernetes

Limiter l'accès aux volumes NFS, aux LUN iSCSI et aux LUN FC créés par Trident est un élément essentiel de la sécurité de votre déploiement Kubernetes. Cela empêche les hôtes qui ne font pas partie du cluster Kubernetes d'accéder aux volumes et de modifier potentiellement les données de manière inattendue.

Il est important de comprendre que les espaces de noms constituent la limite logique des ressources dans Kubernetes. On part du principe que les ressources appartenant au même espace de noms peuvent être partagées ; cependant, et c'est important, il n'existe aucune capacité inter-espaces de noms. Cela signifie que même si les PV sont des objets globaux, lorsqu'ils sont liés à un PVC, ils ne sont accessibles que par les pods qui se trouvent dans le même espace de noms. **Il est essentiel de veiller à ce que les espaces de noms soient utilisés pour assurer la séparation lorsque cela est approprié.**

Pour la plupart des organisations, la principale préoccupation en matière de sécurité des données dans un contexte Kubernetes est qu'un processus dans un conteneur puisse accéder à un stockage monté sur l'hôte, mais qui n'est pas destiné au conteneur. ["Espaces de noms"](#) sont conçues pour empêcher ce type de compromission. Il existe cependant une exception : les conteneurs privilégiés.

Un conteneur privilégié est un conteneur exécuté avec des autorisations au niveau de l'hôte nettement supérieures à la normale. Ces options ne sont pas désactivées par défaut ; assurez-vous donc de désactiver

cette fonctionnalité en utilisant "[politiques de sécurité des pods](#)".

Pour les volumes pour lesquels l'accès est souhaité à la fois depuis Kubernetes et des hôtes externes, le stockage doit être géré de manière traditionnelle, avec le PV introduit par l'administrateur et non géré par Trident. Cela garantit que le volume de stockage n'est détruit que lorsque les hôtes Kubernetes et externes se sont déconnectés et n'utilisent plus le volume. De plus, une politique d'exportation personnalisée peut être appliquée, permettant l'accès depuis les nœuds du cluster Kubernetes et les serveurs cibles situés en dehors du cluster Kubernetes.

Pour les déploiements comportant des nœuds d'infrastructure dédiés (par exemple, OpenShift) ou d'autres nœuds incapables de planifier des applications utilisateur, des politiques d'exportation distinctes doivent être utilisées pour limiter davantage l'accès aux ressources de stockage. Cela inclut la création d'une politique d'exportation pour les services déployés sur ces nœuds d'infrastructure (par exemple, les services de métriques et de journalisation OpenShift) et les applications standard déployées sur des nœuds non infrastructurels.

Utilisez une politique d'exportation dédiée

Vous devez vous assurer qu'une politique d'exportation existe pour chaque backend, n'autorisant l'accès qu'aux nœuds présents dans le cluster Kubernetes. Trident peut créer et gérer automatiquement des politiques d'exportation. De cette façon, Trident limite l'accès aux volumes qu'il provisionne aux nœuds du cluster Kubernetes et simplifie l'ajout/la suppression de nœuds.

Vous pouvez également créer manuellement une politique d'exportation et la remplir avec une ou plusieurs règles d'exportation qui traitent chaque demande d'accès au nœud :

- Utilisez le `vserver export-policy create` Commande CLI ONTAP pour créer la politique d'exportation.
- Ajoutez des règles à la politique d'exportation en utilisant `vserver export-policy rule create` Commande CLI ONTAP .

L'exécution de ces commandes vous permet de restreindre l'accès aux données aux nœuds Kubernetes qui les utilisent.

Désactiver `showmount` pour l'application SVM

Le `showmount` Cette fonctionnalité permet à un client NFS d'interroger le SVM pour obtenir la liste des exportations NFS disponibles. Un pod déployé sur le cluster Kubernetes peut émettre `showmount -e` commande contre et recevoir une liste des points de montage disponibles, y compris ceux auxquels il n'a pas accès. Bien que cela ne constitue pas en soi une faille de sécurité, cela fournit des informations inutiles susceptibles d'aider un utilisateur non autorisé à se connecter à une exportation NFS.

Vous devriez désactiver `showmount` en utilisant la commande CLI ONTAP au niveau SVM :

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

Bonnes pratiques SolidFire

Découvrez les meilleures pratiques pour configurer le stockage SolidFire pour Trident.

Créer un compte Solidfire

Chaque compte SolidFire représente un propriétaire de volume unique et reçoit son propre ensemble d'identifiants CHAP (Challenge-Handshake Authentication Protocol). Vous pouvez accéder aux volumes attribués à un compte soit en utilisant le nom du compte et les informations d'identification CHAP correspondantes, soit via un groupe d'accès aux volumes. Un compte peut se voir attribuer jusqu'à deux mille volumes, mais un volume ne peut appartenir qu'à un seul compte.

Créer une politique QoS

Utilisez les politiques de qualité de service (QoS) SolidFire si vous souhaitez créer et enregistrer un paramètre de qualité de service standardisé qui peut être appliqué à de nombreux volumes.

Vous pouvez définir les paramètres QoS volume par volume. Les performances de chaque volume peuvent être assurées en définissant trois paramètres configurables qui définissent la QoS : IOPS min, IOPS max et IOPS en rafale.

Voici les valeurs IOPS minimales, maximales et en rafale possibles pour une taille de bloc de 4 Ko.

Paramètre IOPS	Définition	Valeur minimale	Valeur par défaut	Valeur maximale (4 Ko)
IOPS minimales	Le niveau de performance garanti pour un volume donné.	50	50	15000
IOPS max	Les performances ne dépasseront pas cette limite.	50	15000	200 000
IOPS en rafale	Nombre maximal d'IOPS autorisé dans un scénario de rafale courte.	50	15000	200 000



Bien que les valeurs Max IOPS et Burst IOPS puissent être fixées à 200 000, les performances maximales réelles d'un volume sont limitées par l'utilisation du cluster et les performances de chaque nœud.

La taille des blocs et la bande passante ont une influence directe sur le nombre d'IOPS. À mesure que la taille des blocs augmente, le système accroît sa bande passante au niveau nécessaire pour traiter ces blocs plus volumineux. À mesure que la bande passante augmente, le nombre d'IOPS que le système est capable d'atteindre diminue. Se référer à "[Qualité de service SolidFire](#)" pour plus d'informations sur la QoS et les performances.

Authentification SolidFire

Element prend en charge deux méthodes d'authentification : CHAP et les groupes d'accès aux volumes (VAG). CHAP utilise le protocole CHAP pour authentifier l'hôte auprès du serveur. Les groupes d'accès aux volumes contrôlent l'accès aux volumes qu'ils provisionnent. NetApp recommande l'utilisation de CHAP pour l'authentification car il est plus simple et ne présente aucune limite d'évolutivité.



Trident, avec son provisionneur CSI amélioré, prend en charge l'utilisation de l'authentification CHAP. Les VAG ne doivent être utilisés qu'en mode de fonctionnement traditionnel, hors CSI.

L'authentification CHAP (vérification que l'initiateur est l'utilisateur du volume prévu) n'est prise en charge qu'avec le contrôle d'accès basé sur les comptes. Si vous utilisez CHAP pour l'authentification, deux options sont disponibles : CHAP unidirectionnel et CHAP bidirectionnel. Le protocole CHAP unidirectionnel authentifie l'accès au volume en utilisant le nom de compte SolidFire et le secret de l'initiateur. L'option CHAP bidirectionnelle offre la méthode d'authentification du volume la plus sûre, car le volume authentifie l'hôte via le nom de compte et le secret de l'initiateur, puis l'hôte authentifie le volume via le nom de compte et le secret cible.

Toutefois, si CHAP ne peut pas être activé et que des VAG sont nécessaires, créez le groupe d'accès et ajoutez les initiateurs hôtes et les volumes au groupe d'accès. Chaque IQN que vous ajoutez à un groupe d'accès peut accéder à chaque volume du groupe avec ou sans authentification CHAP. Si l'initiateur iSCSI est configuré pour utiliser l'authentification CHAP, un contrôle d'accès basé sur les comptes est utilisé. Si l'initiateur iSCSI n'est pas configuré pour utiliser l'authentification CHAP, alors le contrôle d'accès du groupe d'accès aux volumes est utilisé.

Où trouver plus d'informations ?

Vous trouverez ci-dessous une liste de quelques documents présentant les meilleures pratiques. Rechercher le "[Bibliothèque NetApp](#)" pour les versions les plus récentes.

- ONTAP*
- "[Guide des meilleures pratiques et de mise en œuvre du NFS](#)"
- "[Administration SAN](#)"(pour iSCSI)
- "[Configuration iSCSI Express pour RHEL](#)"

Logiciel Element

- "[Configuration de SolidFire pour Linux](#)"
- NetApp HCI*
- "[Prérequis pour le déploiement de NetApp HCI](#)"
- "[Accédez au moteur de déploiement NetApp](#)"

Informations sur les meilleures pratiques d'application

- "[Meilleures pratiques pour MySQL sur ONTAP](#)"
- "[Bonnes pratiques pour MySQL sur SolidFire](#)"
- "[NetApp SolidFire et Cassandra](#)"
- "[Meilleures pratiques Oracle sur SolidFire](#)"
- "[Bonnes pratiques PostgreSQL sur SolidFire](#)"

Toutes les applications ne disposent pas de directives spécifiques ; il est important de collaborer avec votre équipe NetApp et d'utiliser les "[Bibliothèque NetApp](#)" pour trouver la documentation la plus récente.

Intégrer Trident

Pour intégrer Trident, les éléments de conception et d'architecture suivants doivent être intégrés : sélection et déploiement des pilotes, conception de la classe de stockage, conception du pool virtuel, impacts de la revendication de volume persistant (PVC) sur l'approvisionnement du stockage, opérations sur les volumes et déploiement des services OpenShift à l'aide de Trident.

Sélection et déploiement des conducteurs

Sélectionnez et déployez un pilote backend pour votre système de stockage.

Pilotes backend ONTAP

Les pilotes backend ONTAP se différencient par le protocole utilisé et par la manière dont les volumes sont provisionnés sur le système de stockage. Par conséquent, réfléchissez bien avant de choisir le pilote à déployer.

À un niveau supérieur, si votre application comporte des composants nécessitant un stockage partagé (plusieurs pods accédant au même PVC), les pilotes basés sur NAS seraient le choix par défaut, tandis que les pilotes iSCSI basés sur des blocs répondraient aux besoins d'un stockage non partagé. Choisissez le protocole en fonction des exigences de l'application et du niveau de confort des équipes de stockage et d'infrastructure. D'une manière générale, il y a peu de différence entre eux pour la plupart des applications, la décision repose donc souvent sur la nécessité ou non d'un stockage partagé (où plusieurs pods auront besoin d'un accès simultané).

Les pilotes backend ONTAP disponibles sont :

- ``ontap-nas`` Chaque PV provisionné est un FlexVolume ONTAP complet.
- ``ontap-nas-economy`` Chaque PV provisionné est un qtree, avec un nombre configurable de qtrees par FlexVolume (la valeur par défaut est de 200).
- ``ontap-nas-flexgroup`` Chaque PV est provisionné en tant que FlexGroup ONTAP complet, et tous les agrégats affectés à une SVM sont utilisés.
- ``ontap-san`` Chaque PV provisionné est un LUN au sein de son propre FlexVolume.
- ``ontap-san-economy`` Chaque PV provisionné est un LUN, avec un nombre configurable de LUN par FlexVolume (100 par défaut).

Le choix entre les trois pilotes NAS a des répercussions sur les fonctionnalités mises à la disposition de l'application.

Notez que, dans les tableaux ci-dessous, toutes les fonctionnalités ne sont pas exposées via Trident. Certaines doivent être appliquées par l'administrateur de stockage après la mise en service si cette fonctionnalité est souhaitée. Les notes de bas de page en exposant permettent de distinguer les fonctionnalités par fonction et par pilote.

Pilotes ONTAP NAS	Snapshots	Clones	Politiques d'exportation dynamiques	Multi-attache	Qualité de service	Redimensionner	Réplication
ontap-nas	Oui	Oui	Oui, note de bas de page : 5[]	Oui	Oui, note de bas de page : 1[]	Oui	Oui, note de bas de page : 1[]
ontap-nas-economy	NO [3]	NO [3]	Oui, note de bas de page : 5[]	Oui	NO [3]	Oui	NO [3]
ontap-nas-flexgroup	Oui, note de bas de page : 1[]	NON	Oui, note de bas de page : 5[]	Oui	Oui, note de bas de page : 1[]	Oui	Oui, note de bas de page : 1[]

Trident propose 2 pilotes SAN pour ONTAP, dont les capacités sont présentées ci-dessous.

Pilotes SAN ONTAP	Snapshots	Clones	Multi-attache	CHAP bidirectionnel	Qualité de service	Redimensionner	Réplication
ontap-san	Oui	Oui	Oui, note de bas de page : 4[]	Oui	Oui, note de bas de page : 1[]	Oui	Oui, note de bas de page : 1[]
ontap-san-economy	Oui	Oui	Oui, note de bas de page : 4[]	Oui	NO [3]	Oui	NO [3]

Notes de bas de page pour les tableaux ci-dessus : Oui¹ : Non géré par Trident ; Oui² : Géré par Trident, mais pas au niveau des volumes persistants ; Non³ : Non géré par Trident et pas au niveau des volumes persistants ; Oui⁴ : Pris en charge pour les volumes à blocs bruts ; Oui⁵ : Pris en charge par Trident.

Les fonctionnalités qui ne sont pas granulaires au niveau du PV sont appliquées à l'ensemble du FlexVolume et tous les PV (c'est-à-dire les qtrees ou les LUN dans les FlexVols partagés) partageront un calendrier commun.

Comme on peut le constater dans les tableaux ci-dessus, une grande partie des fonctionnalités entre les ontap-nas et ontap-nas-economy c'est la même chose. Cependant, parce que le ontap-nas-economy Le pilote limite la capacité de contrôler la planification au niveau de chaque PV, ce qui peut affecter en particulier votre planification de reprise après sinistre et de sauvegarde. Pour les équipes de développement qui souhaitent exploiter la fonctionnalité de clonage PVC sur le stockage ONTAP , cela n'est possible qu'en utilisant le ontap-nas , ontap-san ou ontap-san-economy conducteurs.



Le solidfire-san Le pilote est également capable de cloner des PVC.

Pilotes backend Cloud Volumes ONTAP

Cloud Volumes ONTAP offre un contrôle des données ainsi que des fonctionnalités de stockage de classe entreprise pour divers cas d'utilisation, notamment les partages de fichiers et le stockage au niveau bloc prenant en charge les protocoles NAS et SAN (NFS, SMB / CIFS et iSCSI). Les pilotes compatibles pour Cloud Volume ONTAP sont `ontap-nas`, `ontap-nas-economy`, `ontap-san` et `ontap-san-economy`. Ces instructions s'appliquent à Cloud Volume ONTAP pour Azure et à Cloud Volume ONTAP pour GCP.

Pilotes backend Amazon FSx pour ONTAP

Amazon FSx for NetApp 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 de nombreuses fonctionnalités du système de fichiers ONTAP et des API d'administration. Les pilotes compatibles pour Cloud Volume ONTAP sont `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `ontap-san` et `ontap-san-economy`.

Pilotes de backend NetApp HCI/ SolidFire

Le `solidfire-san` Ce pilote, utilisé avec les plateformes NetApp HCI/ SolidFire, aide l'administrateur à configurer un backend Element pour Trident en fonction des limites de QoS. Si vous souhaitez concevoir votre backend pour définir les limites QoS spécifiques sur les volumes provisionnés par Trident, utilisez le `type` paramètre dans le fichier backend. L'administrateur peut également limiter la taille du volume pouvant être créé sur le stockage à l'aide de `limitVolumeSize` paramètre. Actuellement, les fonctionnalités de stockage Element telles que le redimensionnement et la réplication de volumes ne sont pas prises en charge via le `solidfire-san` conducteur. Ces opérations doivent être effectuées manuellement via l'interface utilisateur web d'Element Software.

Pilote SolidFire	Snapshots	Clones	Multi-attache	TYPE	Qualité de service	Redimensionner	Réplication
<code>solidfire-san</code>	Oui	Oui	Oui, note de bas de page : 2[]	Oui	Oui	Oui	Oui, note de bas de page : 1[]

Note de bas de page : Oui¹ : Non géré par Trident ² : Pris en charge pour les volumes de blocs bruts

Pilotes de backend Azure NetApp Files

Trident utilise le `azure-netapp-files` le conducteur doit gérer le "Azure NetApp Files" service.

Vous trouverez plus d'informations sur ce pilote et sa configuration dans "[Configuration du backend Trident pour Azure NetApp Files](#)".

Pilote de Azure NetApp Files	Snapshots	Clones	Multi-attache	Qualité de service	Développer	Réplication
<code>azure-netapp-files</code>	Oui	Oui	Oui	Oui	Oui	Oui, note de bas de page : 1[]

Note de bas de page : Oui, note de bas de page 1 : Non géré par Trident

Cloud Volumes Service sur le pilote backend Google Cloud

Trident utilise le `gcp-cvs` Pilote permettant de se connecter au Cloud Volumes Service sur Google Cloud.

Le `gcp-cvs` Le pilote utilise des pools virtuels pour abstraire le backend et permettre à Trident de déterminer l'emplacement des volumes. L'administrateur définit les pools virtuels dans le `backend.json` fichiers. Les classes de stockage utilisent des sélecteurs pour identifier les pools virtuels par étiquette.

- Si des pools virtuels sont définis dans le backend, Trident tentera de créer un volume dans les pools de stockage Google Cloud auquel ces pools virtuels seront limités.
- Si les pools virtuels ne sont pas définis dans le backend, Trident sélectionnera un pool de stockage Google Cloud parmi les pools de stockage disponibles dans la région.

Pour configurer le backend Google Cloud sur Trident, vous devez spécifier `projectNumber`, `apiRegion`, et `apiKey` dans le fichier backend. Vous trouverez le numéro de projet dans la console Google Cloud. La clé API est extraite du fichier de clé privée du compte de service que vous avez créé lors de la configuration de l'accès API pour Cloud Volumes Service sur Google Cloud.

Pour plus d'informations sur les types de services et les niveaux de service de Cloud Volumes Service sur Google Cloud, veuillez consulter la documentation. "[Découvrez la prise en charge de CVS pour GCP par Trident](#)".

Pilote Cloud Volumes Service pour Google Cloud	Snapshots	Clones	Multi-attache	Qualité de service	Développer	Réplication
<code>gcp-cvs</code>	Oui	Oui	Oui	Oui	Oui	Disponible uniquement sur le type de service CVS-Performance.



Notes de réplication

- La réplication n'est pas gérée par Trident.
- Le clone sera créé dans le même pool de stockage que le volume source.

Conception de classe de stockage

Il est nécessaire de configurer et d'appliquer individuellement les classes de stockage pour créer un objet de classe de stockage Kubernetes. Cette section explique comment concevoir une classe de stockage pour votre application.

Utilisation spécifique du backend

Le filtrage peut être utilisé au sein d'un objet de classe de stockage spécifique pour déterminer quel pool de stockage ou ensemble de pools doit être utilisé avec cette classe de stockage spécifique. Trois ensembles de filtres peuvent être définis dans la classe de stockage : `storagePools`, `additionalStoragePools` et/ou `excludeStoragePools`.

Le `storagePools` Ce paramètre permet de limiter le stockage à l'ensemble des pools correspondant aux

attributs spécifiés. Le `additionalStoragePools` Ce paramètre permet d'étendre l'ensemble des pools que Trident utilise pour le provisionnement, en plus de l'ensemble des pools sélectionnés par les attributs et `storagePools` paramètres. Vous pouvez utiliser l'un ou l'autre paramètre seul ou les deux ensemble pour vous assurer que l'ensemble approprié de pools de stockage est sélectionné.

Le `excludeStoragePools` Ce paramètre permet d'exclure spécifiquement l'ensemble de pools listés qui correspondent aux attributs.

Émuler les politiques QoS

Si vous souhaitez concevoir des classes de stockage pour émuler des politiques de qualité de service, créez une classe de stockage avec le `media` attribut comme `hdd` ou `ssd`. Basé sur le `media` L'attribut mentionné dans la classe de stockage permettra à Trident de sélectionner le backend approprié qui sert `hdd` ou `ssd` regroupe les données pour correspondre à l'attribut média, puis dirige la mise à disposition des volumes vers l'agrégat spécifique. Nous pouvons donc créer une classe de stockage PREMIUM qui aurait `media` attribut défini comme `ssd` qui pourrait être classée comme la politique QoS PREMIUM. Nous pouvons créer une autre classe de stockage STANDARD dont l'attribut média serait défini sur `hdd`, ce qui pourrait être classé comme la politique QoS STANDARD. Nous pourrions également utiliser l'attribut « IOPS » dans la classe de stockage pour rediriger le provisionnement vers un dispositif Element qui peut être défini comme une politique QoS.

Utiliser le backend en fonction de fonctionnalités spécifiques

Les classes de stockage peuvent être conçues pour diriger le provisionnement des volumes sur un backend spécifique où des fonctionnalités telles que le provisionnement fin et épais, les instantanés, les clones et le chiffrement sont activées. Pour spécifier le stockage à utiliser, créez des classes de stockage qui définissent le système de stockage approprié avec la fonctionnalité requise activée.

Piscines virtuelles

Des pools virtuelles sont disponibles pour tous les backends Trident. Vous pouvez définir des pools virtuels pour n'importe quel backend, en utilisant n'importe quel pilote fourni par Trident.

Les pools virtuels permettent à un administrateur de créer un niveau d'abstraction sur les backends, qui peut être référencé via les classes de stockage, pour une plus grande flexibilité et un placement efficace des volumes sur les backends. Différents systèmes d'arrière-plan peuvent être définis avec la même classe de service. De plus, plusieurs pools de stockage peuvent être créés sur le même système dorsal, mais avec des caractéristiques différentes. Lorsqu'une classe de stockage est configurée avec un sélecteur comportant des étiquettes spécifiques, Trident choisit un backend correspondant à toutes les étiquettes du sélecteur pour placer le volume. Si les étiquettes du sélecteur de classe de stockage correspondent à plusieurs pools de stockage, Trident en choisira un pour provisionner le volume.

Conception de piscine virtuelle

Lors de la création d'un backend, vous pouvez généralement spécifier un ensemble de paramètres. Il était impossible pour l'administrateur de créer un autre backend avec les mêmes identifiants de stockage et un ensemble de paramètres différent. L'introduction des pools virtuels a résolu ce problème. Un pool virtuel est une abstraction de niveau introduite entre le backend et la classe de stockage Kubernetes. L'administrateur peut ainsi définir des paramètres ainsi que des étiquettes référencées via les classes de stockage Kubernetes comme sélecteur, indépendamment du backend. Les pools virtuels peuvent être définis pour tous les backends NetApp pris en charge par Trident. Cela inclut SolidFire/ NetApp HCI, ONTAP, Cloud Volumes Service sur GCP, ainsi qu'Azure Azure NetApp Files.



Lors de la définition de pools virtuels, il est recommandé de ne pas tenter de réorganiser l'ordre des pools virtuels existants dans une définition de backend. Il est également conseillé de ne pas modifier les attributs d'un pool virtuel existant et de définir plutôt un nouveau pool virtuel.

Émulation de différents niveaux de service/QoS

Il est possible de concevoir des pools virtuels pour émuler des classes de services. En utilisant l'implémentation de pool virtuel pour Cloud Volume Service pour Azure NetApp Files, examinons comment configurer différentes classes de service. Configurez le backend Azure NetApp Files avec plusieurs étiquettes, représentant différents niveaux de performance. Ensemble `servicellevel` ajouter les aspects au niveau de performance approprié et ajouter les autres aspects requis sous chaque étiquette. Créez maintenant différentes classes de stockage Kubernetes qui correspondront à différents pools virtuels. En utilisant le `parameters.selector` Dans ce champ, chaque StorageClass indique quels pools virtuels peuvent être utilisés pour héberger un volume.

Attribuer un ensemble spécifique d'aspects

Il est possible de concevoir plusieurs pools virtuelles présentant des caractéristiques spécifiques à partir d'un seul système de stockage dorsal. Pour ce faire, configurez le backend avec plusieurs étiquettes et définissez les aspects requis sous chaque étiquette. Créez maintenant différentes classes de stockage Kubernetes en utilisant `parameters.selector` champ qui correspondrait à différents pools virtuels. Les volumes provisionnés sur le système dorsal auront les caractéristiques définies dans le pool virtuel choisi.

Caractéristiques du PVC qui affectent la capacité de stockage

Certains paramètres autres que la classe de stockage demandée peuvent affecter le processus de décision de provisionnement Trident lors de la création d'un PVC.

Mode d'accès

Lors d'une demande de stockage via une PVC, l'un des champs obligatoires est le mode d'accès. Le mode souhaité peut affecter le serveur dorsal sélectionné pour héberger la requête de stockage.

Trident tentera de faire correspondre le protocole de stockage utilisé avec la méthode d'accès spécifiée selon la matrice suivante. Ceci est indépendant de la plateforme de stockage sous-jacente.

	Lire/Écrire une seule fois	Lecture seule de plusieurs	Lire/Écrire/Nombreux
iSCSI	Oui	Oui	Oui (bloc cru)
NFS	Oui	Oui	Oui

Une demande de volume persistant ReadWriteMany soumise à un déploiement Trident sans backend NFS configuré n'entraînera la création d'aucun volume. Pour cette raison, le demandeur doit utiliser le mode d'accès approprié à son application.

Opérations de volume

Modifier les volumes persistants

Les volumes persistants sont, à deux exceptions près, des objets immuables dans Kubernetes. Une fois créée, la politique de récupération et la taille peuvent être modifiées. Cependant, cela n'empêche pas certains aspects du volume d'être modifiés en dehors de Kubernetes. Cela peut s'avérer utile pour personnaliser le volume en fonction d'applications spécifiques, pour éviter toute consommation accidentelle de capacité, ou

simplement pour déplacer le volume vers un autre contrôleur de stockage pour quelque raison que ce soit.



Les provisionneurs intégrés à Kubernetes ne prennent pas en charge les opérations de redimensionnement de volume pour les PV NFS, iSCSI ou FC pour le moment. Trident prend en charge l'extension des volumes NFS, iSCSI et FC.

Les détails de connexion du PV ne peuvent pas être modifiés après sa création.

Créer des instantanés de volume à la demande

Trident prend en charge la création d'instantanés de volumes à la demande et la création de PVC à partir d'instantanés à l'aide du framework CSI. Les snapshots offrent une méthode pratique pour conserver des copies ponctuelles des données et ont un cycle de vie indépendant du PV source dans Kubernetes. Ces instantanés peuvent être utilisés pour cloner des PVC.

Créer des volumes à partir d'instantanés

Trident prend également en charge la création de PersistentVolumes à partir d'instantanés de volumes. Pour ce faire, il suffit de créer un PersistentVolumeClaim et de mentionner le `datasource` comme instantané requis à partir duquel le volume doit être créé. Trident gèrera ce PVC en créant un volume contenant les données présentes sur l'instantané. Grâce à cette fonctionnalité, il est possible de dupliquer des données entre régions, de créer des environnements de test, de remplacer intégralement un volume de production endommagé ou corrompu, ou de récupérer des fichiers et répertoires spécifiques et de les transférer vers un autre volume connecté.

Déplacer les volumes dans le cluster

Les administrateurs de stockage ont la possibilité de déplacer des volumes entre les agrégats et les contrôleurs du cluster ONTAP sans perturber le consommateur de stockage. Cette opération n'affecte ni Trident ni le cluster Kubernetes, tant que l'agrégat de destination est un agrégat auquel le SVM utilisé par Trident a accès. Il est important de noter que si l'agrégat a été récemment ajouté au SVM, le backend devra être actualisé en le réajoutant à Trident. Cela déclenchera une nouvelle inventaire du SVM par Trident afin que le nouvel agrégat soit reconnu.

Cependant, le déplacement de volumes entre les systèmes backend n'est pas pris en charge automatiquement par Trident. Cela inclut entre les SVM du même cluster, entre les clusters ou sur une plateforme de stockage différente (même si ce système de stockage est connecté à Trident).

Si un volume est copié vers un autre emplacement, la fonction d'importation de volumes peut être utilisée pour importer les volumes actuels dans Trident.

Augmenter les volumes

Trident prend en charge le redimensionnement des volumes persistants NFS, iSCSI et FC. Cela permet aux utilisateurs de redimensionner leurs volumes directement via la couche Kubernetes. L'extension de volume est possible pour toutes les principales plateformes de stockage NetApp, y compris ONTAP, SolidFire/ NetApp HCI et les backends Cloud Volumes Service. Pour permettre une éventuelle extension ultérieure, définissez `allowVolumeExpansion` à `true` dans votre StorageClass associée au volume. Lorsque le volume persistant doit être redimensionné, modifiez le `spec.resources.requests.storage` annotation dans la revendication de volume persistant à la taille de volume requise. Trident se chargera automatiquement du redimensionnement du volume sur le cluster de stockage.

Importer un volume existant dans Kubernetes

L'importation de volumes permet d'importer un volume de stockage existant dans un environnement Kubernetes. Ceci est actuellement pris en charge par le `ontap-nas`, `ontap-nas-flexgroup`, `solidfire-san`, `azure-netapp-files`, et `gcp-cvs` conducteurs. Cette fonctionnalité est utile lors du portage d'une application existante vers Kubernetes ou lors de scénarios de reprise après sinistre.

Lors de l'utilisation d'ONTAP et `solidfire-san` conducteurs, utilisez la commande `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` pour importer un volume existant dans Kubernetes pour qu'il soit géré par Trident. Le fichier YAML ou JSON du PVC utilisé dans la commande d'importation de volume pointe vers une classe de stockage qui identifie Trident comme le provisionneur. Lors de l'utilisation d'un système dorsal NetApp HCI/ SolidFire, assurez-vous que les noms de volumes sont uniques. Si les noms de volumes sont dupliqués, clonez le volume sous un nom unique afin que la fonction d'importation de volumes puisse les distinguer.

Si le `azure-netapp-files` ou `gcp-cvs` Le pilote est utilisé, utilisez la commande `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` pour importer le volume dans Kubernetes afin qu'il soit géré par Trident. Cela garantit une référence de volume unique.

Lorsque la commande ci-dessus est exécutée, Trident trouvera le volume sur le serveur et lira sa taille. Il ajoutera automatiquement (et écrasera si nécessaire) la taille du volume configurée du PVC. Trident crée ensuite le nouveau PV et Kubernetes lie le PVC au PV.

Si un conteneur était déployé de manière à nécessiter le PVC importé spécifique, il resterait en attente jusqu'à ce que la paire PVC/PV soit liée via le processus d'importation de volume. Une fois la paire PVC/PV assemblée, le conteneur devrait remonter, à condition qu'il n'y ait pas d'autres problèmes.

Service d'enregistrement

Le déploiement et la gestion du stockage pour le registre ont été documentés sur "netapp.io" dans le "[blog](#)".

Service de journalisation

Comme les autres services OpenShift, le service de journalisation est déployé à l'aide d'Ansible avec des paramètres de configuration fournis par le fichier d'inventaire, également appelé `hosts`, fourni au `playbook`. Deux méthodes d'installation seront abordées : le déploiement de la journalisation lors de l'installation initiale d'OpenShift et le déploiement de la journalisation après l'installation d'OpenShift.



À partir de la version 3.9 de Red Hat OpenShift, la documentation officielle déconseille l'utilisation de NFS pour le service de journalisation en raison de problèmes de corruption de données. Cela repose sur les tests effectués par Red Hat sur ses produits. Le serveur NFS ONTAP ne présente pas ces problèmes et peut facilement prendre en charge un déploiement de journalisation. En définitive, le choix du protocole pour le service de journalisation vous appartient ; sachez simplement que les deux fonctionnent parfaitement avec les plateformes NetApp et qu'il n'y a aucune raison d'éviter NFS si c'est votre préférence.

Si vous choisissez d'utiliser NFS avec le service de journalisation, vous devrez définir la variable Ansible `openshift_enable_unsupported_configurations` à `true` pour éviter que l'installation ne tombe en panne.

Commencer

Le service de journalisation peut, en option, être déployé à la fois pour les applications et pour les opérations de base du cluster OpenShift lui-même. Si vous choisissez de déployer la journalisation des opérations, en

spécifiant la variable `openshift_logging_use_ops` comme `true` Deux instances du service seront créées. Les variables qui contrôlent l'instance de journalisation pour les opérations contiennent le terme « ops », contrairement à l'instance pour les applications.

Il est important de configurer les variables Ansible en fonction de la méthode de déploiement afin de garantir que le stockage approprié soit utilisé par les services sous-jacents. Examinons les options pour chacune des méthodes de déploiement.



Les tableaux ci-dessous contiennent uniquement les variables pertinentes pour la configuration du stockage en ce qui concerne le service de journalisation. Vous pouvez trouver d'autres options dans "[Documentation de journalisation de Red Hat OpenShift](#)" qui doivent être examinés, configurés et utilisés en fonction de votre déploiement.

Les variables du tableau ci-dessous permettront au playbook Ansible de créer un PV et un PVC pour le service de journalisation en utilisant les informations fournies. Cette méthode est nettement moins flexible que l'utilisation du playbook d'installation des composants après l'installation d'OpenShift ; cependant, si vous disposez de volumes existants, elle reste une option.

Variable	Détails
<code>openshift_logging_storage_kind</code>	Réglé sur <code>nfs</code> pour que le programme d'installation crée un volume persistant NFS pour le service de journalisation.
<code>openshift_logging_storage_host</code>	Le nom d'hôte ou l'adresse IP de l'hôte NFS. Cette valeur doit être définie sur le dataLIF de votre machine virtuelle.
<code>openshift_logging_storage_nfs_directory</code>	Le chemin de montage pour l'exportation NFS. Par exemple, si le volume est joint comme <code>/openshift_logging</code> Vous utiliseriez ce chemin pour cette variable.
<code>openshift_logging_storage_volume_name</code>	Le nom, par exemple <code>pv_ose_logs</code> , du PV à créer.
<code>openshift_logging_storage_volume_size</code>	La taille de l'exportation NFS, par exemple <code>100Gi</code> .

Si votre cluster OpenShift est déjà en cours d'exécution, et que Trident a donc été déployé et configuré, le programme d'installation peut utiliser le provisionnement dynamique pour créer les volumes. Les variables suivantes devront être configurées.

Variable	Détails
<code>openshift_logging_es_pvc_dynamic</code>	Définissez cette option sur « <code>true</code> » pour utiliser des volumes provisionnés dynamiquement.
<code>openshift_logging_es_pvc_storage_class_name</code>	Le nom de la classe de stockage qui sera utilisée dans le PVC.
<code>openshift_logging_es_pvc_size</code>	Le volume demandé dans le PVC.
<code>openshift_logging_es_pvc_prefix</code>	Un préfixe pour les PVC utilisés par le service d'exploitation forestière.
<code>openshift_logging_es_ops_pvc_dynamic</code>	Réglé sur <code>true</code> utiliser des volumes provisionnés dynamiquement pour l'instance de journalisation des opérations.

Variable	Détails
<code>openshift_logging_es_ops_pvc_storage_class_name</code>	Le nom de la classe de stockage pour l'instance de journalisation des opérations.
<code>openshift_logging_es_ops_pvc_size</code>	La taille de la requête de volume pour l'instance d'opérations.
<code>openshift_logging_es_ops_pvc_prefix</code>	Un préfixe pour les PVC d'instance d'opérations.

Déployez la pile de journalisation

Si vous déployez la journalisation dans le cadre du processus d'installation initial d'OpenShift, il vous suffit de suivre la procédure de déploiement standard. Ansible configurera et déploiera les services et objets OpenShift nécessaires afin que le service soit disponible dès que l'exécution d'Ansible sera terminée.

Toutefois, si vous effectuez un déploiement après l'installation initiale, Ansible devra utiliser le playbook du composant. Ce processus peut légèrement varier selon les versions d'OpenShift ; assurez-vous donc de lire et de suivre les instructions. "[Documentation de Red Hat OpenShift Container Platform 3.11](#)" pour votre version.

Service de métriques

Le service de métriques fournit à l'administrateur des informations précieuses concernant l'état, l'utilisation des ressources et la disponibilité du cluster OpenShift. Elle est également nécessaire pour la fonctionnalité de mise à l'échelle automatique des pods, et de nombreuses organisations utilisent les données du service de métriques pour leurs applications de refacturation et/ou de présentation.

Comme pour le service de journalisation et pour OpenShift dans son ensemble, Ansible est utilisé pour déployer le service de métriques. De même que le service de journalisation, le service de métriques peut être déployé lors de la configuration initiale du cluster ou après sa mise en service en utilisant la méthode d'installation de composants. Les tableaux suivants contiennent les variables importantes lors de la configuration du stockage persistant pour le service de métriques.



Les tableaux ci-dessous ne contiennent que les variables pertinentes pour la configuration du stockage en ce qui concerne le service de métriques. De nombreuses autres options sont décrites dans la documentation et doivent être examinées, configurées et utilisées en fonction de votre déploiement.

Variable	Détails
<code>openshift_metrics_storage_kind</code>	Réglé sur <code>nfs</code> pour que le programme d'installation crée un volume persistant NFS pour le service de journalisation.
<code>openshift_metrics_storage_host</code>	Le nom d'hôte ou l'adresse IP de l'hôte NFS. Cette valeur doit être définie sur dataLIF pour votre SVM.
<code>openshift_metrics_storage_nfs_directory</code>	Le chemin de montage pour l'exportation NFS. Par exemple, si le volume est joint comme <code>/openshift_metrics</code> Vous utiliseriez ce chemin pour cette variable.
<code>openshift_metrics_storage_volume_name</code>	Le nom, par exemple <code>pv_ose_metrics</code> , du PV à créer.
<code>openshift_metrics_storage_volume_size</code>	La taille de l'exportation NFS, par exemple <code>100Gi</code> .

Si votre cluster OpenShift est déjà en cours d'exécution, et que Trident a donc été déployé et configuré, le programme d'installation peut utiliser le provisionnement dynamique pour créer les volumes. Les variables suivantes devront être configurées.

Variable	Détails
<code>openshift_metrics_cassandra_pvc_prefix</code>	Un préfixe à utiliser pour les PVC métriques.
<code>openshift_metrics_cassandra_pvc_size</code>	La taille des volumes à demander.
<code>openshift_metrics_cassandra_storage_type</code>	Le type de stockage à utiliser pour les métriques doit être défini sur dynamique pour qu'Ansible puisse créer des PVC avec la classe de stockage appropriée.
<code>openshift_metrics_cassandra_pvc_storage_class_name</code>	Le nom de la classe de stockage à utiliser.

Déployer le service de métriques

Une fois les variables Ansible appropriées définies dans votre fichier `hosts/inventory`, déployez le service à l'aide d'Ansible. Si vous effectuez le déploiement lors de l'installation d'OpenShift, le PV sera créé et utilisé automatiquement. Si vous déployez à l'aide des playbooks de composants, après l'installation d'OpenShift, Ansible crée les PVC nécessaires et, une fois que Trident a provisionné le stockage pour ceux-ci, déploie le service.

Les variables ci-dessus, ainsi que le processus de déploiement, peuvent changer avec chaque version d'OpenShift. Assurez-vous de consulter et de suivre ["Guide de déploiement OpenShift de Red Hat"](#) pour votre version afin qu'elle soit configurée pour votre environnement.

Protection des données et reprise après sinistre

Découvrez les options de protection et de récupération pour Trident et les volumes créés avec Trident. Vous devez disposer d'une stratégie de protection et de récupération des données pour chaque application ayant une exigence de persistance.

Réplication et récupération du Trident

Vous pouvez créer une sauvegarde pour restaurer Trident en cas de sinistre.

Réplication du Trident

Trident utilise les CRD Kubernetes pour stocker et gérer son propre état et le cluster Kubernetes etcd pour stocker ses métadonnées.

Étapes

1. Sauvegardez le cluster Kubernetes etcd à l'aide de ["Kubernetes : Sauvegarde d'un cluster etcd"](#) .
2. Placez les artefacts de sauvegarde sur un FlexVol volume



NetApp recommande de protéger le SVM sur lequel réside le FlexVol par une relation SnapMirror avec un autre SVM.

Récupération de Trident

En utilisant les CRD Kubernetes et l'instantané etcd du cluster Kubernetes, vous pouvez récupérer Trident.

Étapes

1. Depuis la SVM de destination, montez le volume contenant les fichiers de données et les certificats etcd de Kubernetes sur l'hôte qui sera configuré comme nœud maître.
2. Copiez tous les certificats requis relatifs au cluster Kubernetes sous `/etc/kubernetes/pki` et les fichiers membres etcd sous `/var/lib/etcd`.
3. Restaurez le cluster Kubernetes à partir de la sauvegarde etcd en utilisant "[Kubernetes : Restauration d'un cluster etcd](#)".
4. Courir `kubect1 get crd` vérifier que toutes les ressources personnalisées Trident sont opérationnelles et récupérer les objets Trident pour vérifier que toutes les données sont disponibles.

Réplication et récupération SVM

Trident ne permet pas de configurer les relations de réplication ; toutefois, l'administrateur de stockage peut utiliser "[ONTAP SnapMirror](#)" pour répliquer une SVM.

En cas de sinistre, vous pouvez activer le SVM de destination SnapMirror pour commencer à diffuser des données. Vous pourrez revenir au système principal une fois les systèmes restaurés.

À propos de cette tâche

Tenez compte des points suivants lors de l'utilisation de la fonctionnalité de réplication SVM de SnapMirror :

- Vous devez créer un backend distinct pour chaque SVM avec SVM-DR activé.
- Configurez les classes de stockage pour sélectionner les backends répliqués uniquement lorsque cela est nécessaire afin d'éviter que des volumes qui n'ont pas besoin d'être répliqués soient provisionnés sur les backends qui prennent en charge SVM-DR.
- Les administrateurs d'applications doivent prendre en compte les coûts et la complexité supplémentaires liés à la réplication et examiner attentivement leur plan de reprise d'activité avant d'entamer ce processus.

Réplication SVM

Vous pouvez utiliser "[ONTAP: Réplication SnapMirror SVM](#)" pour créer la relation de réplication SVM.

SnapMirror vous permet de définir des options pour contrôler ce qui doit être répliqué. Vous devrez savoir quelles options vous avez sélectionnées lors de l'exécution. [Récupération SVM à l'aide de Trident](#).

- `"-identité-préserver vrai"` réplique l'intégralité de la configuration SVM.
- `"-discard-configs réseau"` Exclut les LIF et les paramètres réseau associés.
- `"-identité-préserver faux"` Réplique uniquement les volumes et la configuration de sécurité.

Récupération SVM à l'aide de Trident

Trident ne détecte pas automatiquement les défaillances des SVM. En cas de sinistre, l'administrateur peut lancer manuellement le basculement Trident vers la nouvelle SVM.

Étapes

1. Annulez les transferts SnapMirror planifiés et en cours, interrompez la relation de réplication, arrêtez le SVM source puis activez le SVM de destination SnapMirror.

2. Si vous avez spécifié `-identity-preserve false` ou `-discard-config network` Lors de la configuration de votre réplication SVM, mettez à jour le `managementLIF` et `dataLIF` dans le fichier de définition du backend Trident .
3. Confirmer `storagePrefix` est présent dans le fichier de définition du backend Trident . Ce paramètre ne peut pas être modifié. Omission `storagePrefix` provoquera l'échec de la mise à jour du serveur.
4. Mettez à jour tous les serveurs d'arrière-plan requis pour refléter le nouveau nom de la SVM de destination en utilisant :

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

5. Si vous avez spécifié `-identity-preserve false` ou `discard-config network` Vous devez redémarrer tous les pods de l'application.



Si vous avez spécifié `-identity-preserve true`, tous les volumes provisionnés par Trident commencent à diffuser des données lorsque le SVM de destination est activé.

Réplication et récupération de volume

Trident ne peut pas configurer les relations de réplication SnapMirror ; toutefois, l'administrateur de stockage peut utiliser "[Réplication et récupération ONTAP SnapMirror](#)" pour répliquer les volumes créés par Trident.

Vous pouvez ensuite importer les volumes récupérés dans Trident en utilisant "[importation de volume tridentctl](#)".



L'importation n'est pas prise en charge sur `ontap-nas-economy`, `ontap-san-economy`, ou `ontap-flexgroup-economy` conducteurs.

Protection des données instantanées

Vous pouvez protéger et restaurer vos données à l'aide de :

- Un contrôleur de snapshots externe et des CRD pour créer des snapshots de volumes persistants (PV) Kubernetes.

"[Instantanés de volume](#)"

- Les snapshots ONTAP permettent de restaurer l'intégralité du contenu d'un volume ou de récupérer des fichiers ou des LUN individuels.

"[Instantanés ONTAP](#)"

Sécurité

Sécurité

Suivez les recommandations ci-dessous pour garantir la sécurité de votre installation

Trident .

Exécutez Trident dans son propre espace de noms

Il est important d'empêcher les applications, les administrateurs d'applications, les utilisateurs et les applications de gestion d'accéder aux définitions d'objets Trident ou aux pods afin de garantir un stockage fiable et de bloquer toute activité malveillante potentielle.

Pour séparer les autres applications et utilisateurs de Trident, installez toujours Trident dans son propre espace de noms Kubernetes.(`trident`). Placer Trident dans son propre espace de noms garantit que seul le personnel administratif de Kubernetes a accès au pod Trident et aux artefacts (tels que les secrets backend et CHAP, le cas échéant) stockés dans les objets CRD de l'espace de noms. Vous devez vous assurer que seuls les administrateurs ont accès à l'espace de noms Trident et donc à l'accès à `tridentctl` application.

Utilisez l'authentification CHAP avec les backends SAN ONTAP.

Trident prend en charge l'authentification basée sur CHAP pour les charges de travail ONTAP SAN (en utilisant le `ontap-san` et `ontap-san-economy` conducteurs). NetApp recommande l'utilisation du protocole CHAP bidirectionnel avec Trident pour l'authentification entre un hôte et le système de stockage dorsal.

Pour les backends ONTAP utilisant les pilotes de stockage SAN, Trident peut configurer un protocole CHAP bidirectionnel et gérer les noms d'utilisateur et les secrets CHAP via `tridentctl` . Se référer à "[Préparez-vous à configurer le backend avec les pilotes SAN ONTAP](#)" pour comprendre comment Trident configure CHAP sur les backends ONTAP .

Utilisez l'authentification CHAP avec les backends NetApp HCI et SolidFire.

NetApp recommande le déploiement du protocole CHAP bidirectionnel pour garantir l'authentification entre un hôte et les serveurs dorsaux NetApp HCI et SolidFire . Trident utilise un objet secret qui inclut deux mots de passe CHAP par locataire. Lors de son installation, Trident gère les secrets CHAP et les stocke dans un `tridentvolume` Objet CR pour le PV respectif. Lorsque vous créez un PV, Trident utilise les secrets CHAP pour initier une session iSCSI et communiquer avec le système NetApp HCI et SolidFire via CHAP.



Les volumes créés par Trident ne sont associés à aucun groupe d'accès aux volumes.

Utilisez Trident avec NVE et NAE

NetApp ONTAP assure le chiffrement des données au repos afin de protéger les données sensibles en cas de vol, de retour ou de réutilisation d'un disque. Pour plus de détails, reportez-vous à "[Présentation de la configuration du chiffrement de volume NetApp](#)" .

- Si NAE est activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NAE.
 - Vous pouvez définir l'indicateur de chiffrement NVE sur "" pour créer des volumes compatibles NAE.
- Si NAE n'est pas activé sur le système dorsal, tout volume provisionné dans Trident sera compatible NVE, sauf si l'indicateur de chiffrement NVE est défini sur `false` (la valeur par défaut) dans la configuration du backend.

Les volumes créés dans Trident sur un backend compatible NAE doivent être chiffrés NVE ou NAE.



- Vous pouvez définir l'indicateur de chiffrement NVE sur `true` dans la configuration du backend Trident pour remplacer le chiffrement NAE et utiliser une clé de chiffrement spécifique pour chaque volume.
- Définir l'indicateur de chiffrement NVE sur `false` sur un backend compatible NAE crée un volume compatible NAE. Vous ne pouvez pas désactiver le chiffrement NAE en définissant l'indicateur de chiffrement NVE sur `false`.

- Vous pouvez créer manuellement un volume NVE dans Trident en définissant explicitement l'indicateur de chiffrement NVE sur `true`.

Pour plus d'informations sur les options de configuration du backend, consultez :

- ["Options de configuration SAN ONTAP"](#)
- ["Options de configuration ONTAP NAS"](#)

Configuration unifiée des clés Linux (LUKS)

Vous pouvez activer Linux Unified Key Setup (LUKS) pour chiffrer les volumes ONTAP SAN et ONTAP SAN ECONOMY sur Trident. Trident prend en charge la rotation des phrases de passe et l'extension de volume pour les volumes chiffrés LUKS.

Dans Trident, les volumes chiffrés LUKS utilisent le chiffrement et le mode `aes-xts-plain64`, comme recommandé par ["NIST"](#).



Le chiffrement LUKS n'est pas pris 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"](#).

Avant de commencer

- Les nœuds de travail doivent avoir `cryptsetup 2.1` ou supérieur (mais inférieur à 3.0) installé. Pour plus d'informations, consultez ["Gitlab : cryptsetup"](#).
- Pour des raisons de performance, NetApp recommande que les nœuds de travail prennent en charge les nouvelles instructions de la norme de chiffrement avancé (AES-NI). Pour vérifier la prise en charge d'AES-NI, exécutez la commande suivante :

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

Si aucune réponse n'est reçue, votre processeur ne prend pas en charge AES-NI. Pour plus d'informations sur AES-NI, consultez : ["Intel : Instructions de la norme de chiffrement avancée \(AES-NI\)"](#).

Activer le chiffrement LUKS

Vous pouvez activer le chiffrement par volume, côté hôte, à l'aide de Linux Unified Key Setup (LUKS) pour les volumes ONTAP SAN et ONTAP SAN ECONOMY.

Étapes

1. Définissez les attributs de chiffrement LUKS dans la configuration du backend. Pour plus d'informations sur les options de configuration du backend pour ONTAP SAN, veuillez consulter la documentation "[Options de configuration SAN ONTAP](#)".

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

2. Utilisez `parameters.selector` définir les pools de stockage à l'aide du chiffrement LUKS. Par exemple:

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

3. Créez un secret contenant la phrase de passe LUKS. Par exemple:

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

Limites

Les volumes chiffrés LUKS ne peuvent pas bénéficier de la déduplication et de la compression ONTAP .

Configuration du backend pour l'importation des volumes LUKS

Pour importer un volume LUKS, vous devez configurer `luksEncryption` à `true` en arrière-plan. Le `luksEncryption` Cette option indique à Trident si le volume est conforme à la norme LUKS. (`true`) ou non conforme à LUKS (`false`) comme le montre l'exemple suivant.

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

Configuration PVC pour l'importation de volumes LUKS

Pour importer dynamiquement des volumes LUKS, définissez l'annotation `trident.netapp.io/luksEncryption` à `true` et inclure une classe de stockage compatible LUKS dans le PVC comme indiqué dans cet exemple.

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

Rotation d'une phrase de passe LUKS

Vous pouvez modifier la phrase de passe LUKS et confirmer la modification.



N'oubliez pas une phrase de passe tant que vous n'avez pas vérifié qu'elle n'est plus référencée par aucun volume, instantané ou secret. Si la phrase de passe de référence est perdue, vous risquez de ne pas pouvoir monter le volume et les données resteront chiffrées et inaccessibles.

À propos de cette tâche

La rotation de la phrase de passe LUKS se produit lorsqu'un pod qui monte le volume est créé après la spécification d'une nouvelle phrase de passe LUKS. Lors de la création d'un nouveau pod, Trident compare la phrase de passe LUKS du volume à la phrase de passe active du secret.

- Si la phrase de passe figurant sur le volume ne correspond pas à la phrase de passe active dans le secret, une rotation a lieu.
- Si la phrase de passe figurant sur le volume correspond à la phrase de passe active dans le secret, le paramètre `previous-luks-passphrase` Ce paramètre est ignoré.

Étapes

1. Ajoutez le paramètre `node-publish-secret-name` et `node-publish-secret-namespace` Paramètres de StorageClass. Par exemple:

```

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

```

2. Identifier les phrases de passe existantes sur le volume ou l'instantané.

Volume

```

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

...luksPassphraseNames: ["A"]

```

Instantané

```

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

...luksPassphraseNames: ["A"]

```

3. Mettez à jour le secret LUKS du volume pour spécifier les nouvelles et anciennes phrases de passe. Assurer `previous-luke-passphrase-name` et `previous-luks-passphrase` correspondre à la phrase de passe précédente.

```

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

```

4. Créez un nouveau pod en montant le volume. Ceci est nécessaire pour lancer la rotation.

5. Vérifiez que la phrase de passe a été modifiée.

Volume

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

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

Instantané

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

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

Résultats

La phrase de passe a été renouvelée lorsque seule la nouvelle phrase de passe a été renvoyée sur le volume et l'instantané.



Si deux phrases de passe sont renvoyées, par exemple `luksPassphraseNames: ["B", "A"]` La rotation est incomplète. Vous pouvez déclencher une nouvelle capsule pour tenter de terminer la rotation.

Activer l'expansion du volume

Vous pouvez activer l'extension de volume sur un volume chiffré LUKS.

Étapes

1. Activer `CSINodeExpandSecret` fonctionnalité gate (bêta 1.25+). Se référer à ["Kubernetes 1.25 : Utilisation des secrets pour l'extension des volumes CSI pilotée par les nœuds"](#) pour plus de détails.
2. Ajoutez le `node-expand-secret-name` et `node-expand-secret-namespace` Paramètres de `StorageClass`. Par exemple:

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

Résultats

Lorsque vous lancez l'extension du stockage en ligne, le kubelet transmet les informations d'identification appropriées au pilote.

Chiffrement Kerberos en transit

En utilisant le chiffrement Kerberos en transit, vous pouvez améliorer la sécurité d'accès aux données en activant le chiffrement du trafic entre votre cluster géré et le système de stockage dorsal.

Trident prend en charge le chiffrement Kerberos pour ONTAP en tant que système de stockage dorsal :

- * ONTAP sur site* - Trident prend en charge le chiffrement Kerberos sur les connexions NFSv3 et NFSv4 depuis Red Hat OpenShift et les clusters Kubernetes en amont vers les volumes ONTAP sur site.

Vous pouvez créer, supprimer, redimensionner, prendre un instantané, cloner, cloner en lecture seule et importer des volumes utilisant le chiffrement NFS.

Configurer le chiffrement Kerberos en transit avec les volumes ONTAP sur site

Vous pouvez activer le chiffrement Kerberos sur le trafic de stockage entre votre cluster géré et un système de stockage ONTAP sur site.



Le chiffrement Kerberos pour le trafic NFS avec des systèmes de stockage ONTAP sur site n'est pris en charge qu'avec l'interface suivante : `ontap-nas` pilote de stockage.

Avant de commencer

- Assurez-vous d'avoir accès à `tridentctl` utilitaire.
- Assurez-vous de disposer d'un accès administrateur au système de stockage ONTAP .
- Assurez-vous de connaître le nom du ou des volumes que vous partagerez depuis le système de stockage ONTAP .
- Assurez-vous d'avoir préparé la machine virtuelle de stockage ONTAP pour prendre en charge le chiffrement Kerberos pour les volumes NFS. Se référer à "[Activer Kerberos sur une LIF de données](#)" pour les instructions.
- Assurez-vous que tous les volumes NFSv4 que vous utilisez avec le chiffrement Kerberos sont correctement configurés. Reportez-vous à la section Configuration du domaine NetApp NFSv4 (page 13) du manuel. "[Guide des améliorations et des bonnes pratiques NetApp NFSv4](#)" .

Ajouter ou modifier les politiques d'exportation ONTAP

Vous devez ajouter des règles aux politiques d'exportation ONTAP existantes ou créer de nouvelles politiques d'exportation prenant en charge le chiffrement Kerberos pour le volume racine de la machine virtuelle de stockage ONTAP ainsi que pour tous les volumes ONTAP partagés avec le cluster Kubernetes en amont. Les règles de politique d'exportation que vous ajoutez, ou les nouvelles politiques d'exportation que vous créez, doivent prendre en charge les protocoles d'accès et les autorisations d'accès suivants :

Protocoles d'accès

Configurez la politique d'exportation avec les protocoles d'accès NFS, NFSv3 et NFSv4.

Détails d'accès

Vous pouvez configurer l'une des trois versions différentes du chiffrement Kerberos, en fonction des besoins de votre volume :

- **Kerberos 5** - (authentification et chiffrement)
- **Kerberos 5i** - (authentification et chiffrement avec protection de l'identité)
- **Kerberos 5p** - (authentification et chiffrement avec protection de l'identité et de la vie privée)

Configurez la règle de stratégie d'exportation ONTAP avec les autorisations d'accès appropriées. Par exemple, si les clusters doivent monter les volumes NFS avec un mélange de chiffrement Kerberos 5i et Kerberos 5p, utilisez les paramètres d'accès suivants :

Type	Accès en lecture seule	Accès en lecture/écriture	accès superutilisateur
UNIX	Activé	Activé	Activé
Kerberos 5i	Activé	Activé	Activé
Kerberos 5p	Activé	Activé	Activé

Consultez la documentation suivante pour savoir comment créer des politiques d'exportation ONTAP et des règles de politique d'exportation :

- ["Créer une politique d'exportation"](#)
- ["Ajouter une règle à une politique d'exportation"](#)

Créer un backend de stockage

Vous pouvez créer une configuration de stockage Trident qui inclut la capacité de chiffrement Kerberos.

À propos de cette tâche

Lorsque vous créez un fichier de configuration de stockage dorsal qui configure le chiffrement Kerberos, vous pouvez spécifier l'une des trois versions différentes de chiffrement Kerberos à l'aide de `spec.nfsMountOptions` paramètre:

- `spec.nfsMountOptions: sec=krb5`(authentification et chiffrement)
- `spec.nfsMountOptions: sec=krb5i`(authentification et chiffrement avec protection de l'identité)
- `spec.nfsMountOptions: sec=krb5p`(authentification et chiffrement avec protection de l'identité et de la vie privée)

Spécifiez un seul niveau Kerberos. Si vous spécifiez plusieurs niveaux de chiffrement Kerberos dans la liste des paramètres, seule la première option sera utilisée.

Étapes

1. Sur le cluster géré, créez un fichier de configuration de stockage en utilisant l'exemple suivant. Remplacez les valeurs entre crochets `<>` par les informations provenant de votre environnement :

```

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

```

2. Utilisez le fichier de configuration que vous avez créé à l'étape précédente pour créer le backend :

```
tridentctl create backend -f <backend-configuration-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.

Créer une classe de stockage

Vous pouvez créer une classe de stockage pour provisionner des volumes avec chiffrement Kerberos.

À propos de cette tâche

Lorsque vous créez un objet de classe de stockage, vous pouvez spécifier l'une des trois versions différentes du chiffrement Kerberos à l'aide de `mountOptions` paramètre:

- `mountOptions: sec=krb5`(authentification et chiffrement)
- `mountOptions: sec=krb5i`(authentification et chiffrement avec protection de l'identité)
- `mountOptions: sec=krb5p`(authentification et chiffrement avec protection de l'identité et de la vie privée)

Spécifiez un seul niveau Kerberos. Si vous spécifiez plusieurs niveaux de chiffrement Kerberos dans la liste des paramètres, seule la première option sera utilisée. Si le niveau de chiffrement que vous avez spécifié dans la configuration du backend de stockage est différent de celui que vous avez spécifié dans l'objet de classe de stockage, c'est l'objet de classe de stockage qui prévaut.

Étapes

1. Créez un objet `StorageClass` Kubernetes en utilisant l'exemple suivant :

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

2. Créez la classe de stockage :

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

3. Assurez-vous que la classe de stockage a été créée :

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

Vous devriez obtenir un résultat similaire à celui-ci :

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

Volumes de provision

Une fois que vous avez créé un système de stockage et une classe de stockage, vous pouvez maintenant provisionner un volume. Pour les instructions, reportez-vous à ["Provisionnez un volume"](#) .

Configurer le chiffrement Kerberos en transit avec les volumes Azure NetApp Files

Vous pouvez activer le chiffrement Kerberos sur le trafic de stockage entre votre cluster géré et un seul backend de stockage Azure NetApp Files ou un pool virtuel de backends de stockage Azure NetApp Files .

Avant de commencer

- Assurez-vous d'avoir activé Trident sur le cluster Red Hat OpenShift géré.
- Assurez-vous d'avoir accès à `tridentctl` utilitaire.
- Assurez-vous d'avoir préparé le système de stockage Azure NetApp Files pour le chiffrement Kerberos en prenant en compte les exigences et en suivant les instructions. ["Documentation Azure NetApp Files"](#) .
- Assurez-vous que tous les volumes NFSv4 que vous utilisez avec le chiffrement Kerberos sont correctement configurés. Reportez-vous à la section Configuration du domaine NetApp NFSv4 (page 13) du manuel. ["Guide des améliorations et des bonnes pratiques NetApp NFSv4"](#) .

Créer un backend de stockage

Vous pouvez créer une configuration de stockage backend Azure NetApp Files incluant la fonctionnalité de chiffrement Kerberos.

À propos de cette tâche

Lorsque vous créez un fichier de configuration de stockage qui configure le chiffrement Kerberos, vous pouvez le définir de sorte qu'il soit appliqué à l'un des deux niveaux suivants :

- Le **niveau du backend de stockage** utilisant le `spec.kerberos` champ
- Le **niveau de piscine virtuelle** utilisant le `spec.storage.kerberos` champ

Lorsque vous définissez la configuration au niveau du pool virtuel, le pool est sélectionné à l'aide de l'étiquette dans la classe de stockage.

À chaque niveau, vous pouvez spécifier l'une des trois versions différentes du chiffrement Kerberos :

- `kerberos: sec=krb5`(authentification et chiffrement)
- `kerberos: sec=krb5i`(authentification et chiffrement avec protection de l'identité)
- `kerberos: sec=krb5p`(authentification et chiffrement avec protection de l'identité et de la vie privée)

Étapes

1. Sur le cluster géré, créez un fichier de configuration de backend de stockage en utilisant l'un des exemples suivants, selon l'endroit où vous devez définir le backend de stockage (niveau du backend de stockage ou niveau du pool virtuel). Remplacez les valeurs entre crochets `<>` par les informations provenant de votre environnement :

Exemple de niveau backend de stockage

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

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

Exemple de niveau de piscine virtuelle

```

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

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

```

2. Utilisez le fichier de configuration que vous avez créé à l'étape précédente pour créer le backend :

```
tridentctl create backend -f <backend-configuration-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.

Créer une classe de stockage

Vous pouvez créer une classe de stockage pour provisionner des volumes avec chiffrement Kerberos.

Étapes

1. Créez un objet StorageClass Kubernetes en utilisant l'exemple suivant :

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

2. Créez la classe de stockage :

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

3. Assurez-vous que la classe de stockage a été créée :

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

Vous devriez obtenir un résultat similaire à celui-ci :

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

Volumes de provision

Une fois que vous avez créé un système de stockage et une classe de stockage, vous pouvez maintenant provisionner un volume. Pour les instructions, reportez-vous à "[Provisionnez un volume](#)".

Protégez vos applications avec Trident Protect

Découvrez Trident Protect

NetApp Trident Protect offre des fonctionnalités avancées de gestion des données d'application qui améliorent la fonctionnalité et la disponibilité des applications Kubernetes avec état, prises en charge par les systèmes de stockage NetApp ONTAP et le provisionneur de stockage NetApp Trident CSI. Trident Protect simplifie la gestion, la protection et le déplacement des charges de travail conteneurisées entre les clouds publics et les environnements sur site. Il offre également des capacités d'automatisation via son API et sa CLI.

Vous pouvez protéger les applications avec Trident Protect en créant des ressources personnalisées (CR) ou en utilisant l'interface de ligne de commande (CLI) de Trident Protect.

Quelle est la prochaine étape ?

Vous pouvez vous renseigner sur les exigences de Trident Protect avant de l'installer :

- ["Exigences de Trident Protect"](#)

Installer Trident Protect

Exigences de Trident Protect

Commencez par vérifier l'état de préparation de votre environnement opérationnel, de vos clusters d'applications, de vos applications et de vos licences. Assurez-vous que votre environnement répond à ces exigences pour déployer et utiliser Trident Protect.

Compatibilité avec les clusters Kubernetes de Trident Protect

Trident Protect est compatible avec une large gamme d'offres Kubernetes entièrement gérées et autogérées, notamment :

- Amazon Elastic Kubernetes Service (EKS)
- Moteur Google Kubernetes (GKE)
- Service Kubernetes Microsoft Azure (AKS)
- Red Hat OpenShift
- SUSE Rancher
- Portefeuille VMware Tanzu
- Kubernetes en amont



- Les sauvegardes Trident Protect sont prises en charge uniquement sur les nœuds de calcul Linux. Les nœuds de calcul Windows ne sont pas pris en charge pour les opérations de sauvegarde.
- Assurez-vous que le cluster sur lequel vous installez Trident Protect est configuré avec un contrôleur de snapshots en cours d'exécution et les CRD associés. Pour installer un contrôleur de snapshots, reportez-vous à la documentation. "[ces instructions](#)".

Compatibilité du système de stockage Trident Protect

Trident Protect prend en charge les systèmes de stockage suivants :

- Amazon FSx for NetApp ONTAP
- Cloud Volumes ONTAP
- baies de stockage ONTAP
- Google Cloud NetApp Volumes
- Azure NetApp Files

Assurez-vous que votre système de stockage réponde aux exigences suivantes :

- Assurez-vous que le stockage NetApp connecté au cluster utilise Trident 24.02 ou une version plus récente (Trident 24.10 est recommandé).
- Assurez-vous de disposer d'un système de stockage NetApp ONTAP .
- Assurez-vous d'avoir configuré un compartiment de stockage d'objets pour stocker les sauvegardes.
- Créez les espaces de noms d'application que vous prévoyez d'utiliser pour les applications ou les opérations de gestion des données d'application. Trident Protect ne crée pas ces espaces de noms pour vous ; si vous spécifiez un espace de noms inexistant dans une ressource personnalisée, l'opération échouera.

Exigences pour les volumes de l'économie NAS

Trident Protect prend en charge les opérations de sauvegarde et de restauration sur les volumes NAS économiques. Les snapshots, les clones et la réplication SnapMirror vers des volumes nas-economy ne sont actuellement pas pris en charge. Vous devez activer un répertoire de snapshots pour chaque volume nas-economy que vous prévoyez d'utiliser avec Trident Protect.



Certaines applications ne sont pas compatibles avec les volumes qui utilisent un répertoire de snapshots. Pour ces applications, vous devez masquer le répertoire des instantanés en exécutant la commande suivante sur le système de stockage ONTAP :

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

Vous pouvez activer le répertoire de snapshots en exécutant la commande suivante pour chaque volume nas-economy, en remplaçant <volume-UUID> avec l'UUID du volume que vous souhaitez modifier :

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level  
=true -n trident
```



Vous pouvez activer par défaut les répertoires de snapshots pour les nouveaux volumes en configurant l'option de configuration du backend Trident `snapshotDir` à `true`. Les volumes existants ne sont pas affectés.

Protection des données avec les machines virtuelles KubeVirt

Trident Protect 24.10 et 24.10.1 et versions ultérieures ont un comportement différent lorsque vous protégez des applications exécutées sur des machines virtuelles KubeVirt. Pour les deux versions, vous pouvez activer ou désactiver le gel et le dégel du système de fichiers pendant les opérations de protection des données.



Lors des opérations de restauration, tout `VirtualMachineSnapshots` Les éléments créés pour une machine virtuelle (VM) ne sont pas restaurés.

Trident Protect 24.10

Trident Protect 24.10 ne garantit pas automatiquement un état cohérent pour les systèmes de fichiers de machines virtuelles KubeVirt lors des opérations de protection des données. Si vous souhaitez protéger les données de votre machine virtuelle KubeVirt à l'aide de Trident Protect 24.10, vous devez activer manuellement la fonctionnalité de gel/dégel des systèmes de fichiers avant l'opération de protection des données. Cela garantit que les systèmes de fichiers sont dans un état cohérent.

Vous pouvez configurer Trident Protect 24.10 pour gérer le gel et le dégel du système de fichiers de la machine virtuelle lors des opérations de protection des données. "[configuration de la virtualisation](#)" puis en utilisant la commande suivante :

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=true -n trident-protect
```

Trident Protect 24.10.1 et versions ultérieures

À partir de Trident Protect 24.10.1, Trident Protect gèle et débloque automatiquement les systèmes de fichiers KubeVirt lors des opérations de protection des données. Vous pouvez désactiver ce comportement automatique à l'aide de la commande suivante :

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

Exigences pour la réplication SnapMirror

La réplication NetApp SnapMirror est disponible pour une utilisation avec Trident Protect pour les solutions ONTAP suivantes :

- Clusters NetApp FAS, AFF et ASA sur site
- ONTAP Select NetApp ONTAP
- NetApp Cloud Volumes ONTAP
- Amazon FSx for NetApp ONTAP

Configuration requise pour la réplication SnapMirror dans un cluster ONTAP

Assurez-vous que votre cluster ONTAP répond aux exigences suivantes si vous prévoyez d'utiliser la réplication SnapMirror :

- *** NetApp Trident*** : NetApp Trident doit exister à la fois sur les clusters Kubernetes source et de destination qui utilisent ONTAP comme backend. Trident Protect prend en charge la réplication avec la technologie NetApp SnapMirror utilisant des classes de stockage reposant sur 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)
- **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.

Considérations relatives au peering pour la réplication SnapMirror

Si vous prévoyez d'utiliser le peering de stockage backend, assurez-vous que votre environnement réponde aux exigences suivantes :

- **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.

- *** NetApp Trident et SVM *** : Les SVM distants appariés doivent être disponibles pour NetApp Trident sur le cluster de destination.
- **Systèmes de stockage backend gérés** : Vous devez ajouter et gérer des systèmes de stockage backend ONTAP dans Trident Protect pour créer une relation de réplication.

Configuration Trident / ONTAP pour la réplication SnapMirror

Trident Protect exige que vous configuriez au moins un système de stockage dorsal prenant en charge la réplication pour les clusters source et de destination. Si les clusters source et de destination sont identiques, l'application de destination doit utiliser un système de stockage différent de celui de l'application source pour une résilience optimale.

Configuration requise pour la réplication SnapMirror dans un cluster Kubernetes

Assurez-vous que vos clusters Kubernetes répondent aux exigences suivantes :

- **Accessibilité à AppVault** : Les clusters source et de destination doivent avoir un accès réseau pour lire et

écrire dans AppVault pour la réplication des objets d'application.

- **Connectivité réseau** : Configurez les règles de pare-feu, les autorisations de compartiment et les listes d'adresses IP autorisées pour permettre la communication entre les deux clusters et AppVault via les réseaux WAN.



De nombreux environnements d'entreprise mettent en œuvre des politiques de pare-feu strictes sur les connexions WAN. Vérifiez ces exigences réseau auprès de votre équipe d'infrastructure avant de configurer la réplication.

Installez et configurez Trident Protect.

Si votre environnement répond aux exigences de Trident Protect, vous pouvez suivre ces étapes pour installer Trident Protect sur votre cluster. Vous pouvez obtenir Trident Protect auprès de NetApp ou l'installer à partir de votre propre registre privé. L'installation à partir d'un registre privé est utile si votre cluster ne peut pas accéder à Internet.

Installer Trident Protect

Installez Trident Protect de NetApp

Étapes

1. Ajouter le dépôt Trident Helm :

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

2. Utilisez Helm pour installer Trident Protect. Remplacer `<name-of-cluster>` avec un nom de cluster, qui sera attribué au cluster et utilisé pour identifier les sauvegardes et les instantanés du cluster :

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --version 100.2506.0 --create
--namespace --namespace trident-protect
```

Installez Trident Protect à partir d'un registre privé

Vous pouvez installer Trident Protect à partir d'un registre d'images privé si votre cluster Kubernetes ne peut pas accéder à Internet. Dans ces exemples, remplacez les valeurs entre crochets par les informations provenant de votre environnement :

Étapes

1. Téléchargez les images suivantes sur votre machine locale, mettez à jour les étiquettes, puis envoyez-les vers votre registre privé :

```
netapp/controller:25.06.0
netapp/restic:25.06.0
netapp/kopia:25.06.0
netapp/trident-autosupport:25.06.0
netapp/exehook:25.06.0
netapp/resourcebackup:25.06.0
netapp/resourcerestore:25.06.0
netapp/resourcedelete:25.06.0
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

Par exemple:

```
docker pull netapp/controller:25.06.0
```

```
docker tag netapp/controller:25.06.0 <private-registry-
url>/controller:25.06.0
```

```
docker push <private-registry-url>/controller:25.06.0
```

2. Créez l'espace de noms système Trident Protect :

```
kubectl create ns trident-protect
```

3. Connectez-vous au registre :

```
helm registry login <private-registry-url> -u <account-id> -p <api-token>
```

4. Créez un secret d'extraction à utiliser pour l'authentification du registre privé :

```
kubectl create secret docker-registry regcred --docker-username=<registry-username> --docker-password=<api-token> -n trident-protect --docker-server=<private-registry-url>
```

5. Ajouter le dépôt Trident Helm :

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

6. Créez un fichier nommé `protectValues.yaml` . Assurez-vous qu'il contienne les paramètres Trident Protect suivants :

```

---
image:
  registry: <private-registry-url>
imagePullSecrets:
  - name: regcred
controller:
  image:
    registry: <private-registry-url>
rbacProxy:
  image:
    registry: <private-registry-url>
crCleanup:
  imagePullSecrets:
    - name: regcred
webhooksCleanup:
  imagePullSecrets:
    - name: regcred

```

7. Utilisez Helm pour installer Trident Protect. Remplacer `<name_of_cluster>` avec un nom de cluster, qui sera attribué au cluster et utilisé pour identifier les sauvegardes et les instantanés du cluster :

```

helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2506.0 --create
--namespace --namespace trident-protect -f protectValues.yaml

```

Installez le plugin CLI Trident Protect

Vous pouvez utiliser le plugin en ligne de commande Trident Protect, qui est une extension de Trident. `tridentctl` utilitaire, pour créer et interagir avec les ressources personnalisées (CR) de Trident Protect.

Installez le plugin CLI Trident Protect

Avant d'utiliser l'utilitaire en ligne de commande, vous devez l'installer sur la machine que vous utilisez pour accéder à votre cluster. Suivez ces étapes, selon que votre machine utilise un processeur x64 ou ARM .

Télécharger le plugin pour les processeurs Linux AMD64

Étapes

1. Téléchargez le plugin CLI Trident Protect :

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-amd64
```

Télécharger le plugin pour les processeurs Linux ARM64

Étapes

1. Téléchargez le plugin CLI Trident Protect :

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-arm64
```

Télécharger le plugin pour les processeurs Mac AMD64

Étapes

1. Téléchargez le plugin CLI Trident Protect :

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-amd64
```

Télécharger le plugin pour les processeurs Mac ARM64

Étapes

1. Téléchargez le plugin CLI Trident Protect :

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-arm64
```

1. Activer les permissions d'exécution pour le fichier binaire du plugin :

```
chmod +x tridentctl-protect
```

2. Copiez le fichier binaire du plugin à un emplacement défini dans votre variable PATH. Par exemple, /usr/bin ou /usr/local/bin (Vous pourriez avoir besoin de privilèges élevés) :

```
cp ./tridentctl-protect /usr/local/bin/
```

3. Vous pouvez également copier le fichier binaire du plugin dans un emplacement de votre répertoire personnel. Dans ce cas, il est recommandé de s'assurer que l'emplacement fait partie de votre variable PATH :

```
cp ./tridentctl-protect ~/bin/
```



Copier le plugin dans un emplacement figurant dans votre variable PATH vous permet de l'utiliser en tapant : `tridentctl-protect` ou `tridentctl protect` de n'importe quel endroit.

Consultez l'aide du plugin Trident CLI

Vous pouvez utiliser les fonctionnalités d'aide intégrées au plugin pour obtenir une aide détaillée sur ses capacités :

Étapes

1. Utilisez la fonction d'aide pour consulter les instructions d'utilisation :

```
tridentctl-protect help
```

Activer la saisie semi-automatique des commandes

Une fois le plugin CLI Trident Protect installé, vous pouvez activer la saisie semi-automatique pour certaines commandes.

Activer la saisie semi-automatique pour le shell Bash

Étapes

1. Télécharger le script de complétion :

```
curl -L -O https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-completion.bash
```

2. Créez un nouveau répertoire dans votre répertoire personnel pour y placer le script :

```
mkdir -p ~/.bash/completions
```

3. Déplacez le script téléchargé vers le ~/.bash/completions annuaire:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Ajoutez la ligne suivante à la ~/.bashrc fichier dans votre répertoire personnel :

```
source ~/.bash/completions/tridentctl-completion.bash
```

Activer la saisie semi-automatique pour le shell Z

Étapes

1. Télécharger le script de complétion :

```
curl -L -O https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-completion.zsh
```

2. Créez un nouveau répertoire dans votre répertoire personnel pour y placer le script :

```
mkdir -p ~/.zsh/completions
```

3. Déplacez le script téléchargé vers le ~/.zsh/completions annuaire:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Ajoutez la ligne suivante à la ~/.zprofile fichier dans votre répertoire personnel :

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

Résultat

Lors de votre prochaine connexion à l'interpréteur de commandes, vous pourrez utiliser la saisie semi-automatique des commandes grâce au plugin `tridentctl-protect`.

Personnaliser l'installation de Trident Protect

Vous pouvez personnaliser la configuration par défaut de Trident Protect pour répondre aux exigences spécifiques de votre environnement.

Spécifiez les limites de ressources du conteneur Trident Protect

Vous pouvez utiliser un fichier de configuration pour spécifier les limites de ressources des conteneurs Trident Protect après l'installation de Trident Protect. La définition de limites de ressources vous permet de contrôler la quantité de ressources du cluster consommées par les opérations de Trident Protect.

Étapes

1. Créez un fichier nommé `resourceLimits.yaml`.
2. Renseignez le fichier avec les options de limite de ressources pour les conteneurs Trident Protect en fonction des besoins de votre environnement.

Le fichier de configuration d'exemple suivant présente les paramètres disponibles et contient les valeurs par défaut pour chaque limite de ressources :

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
      memory: ""
```

```

    ephemeralStorage: ""
  requests:
    cpu: ""
    memory: ""
    ephemeralStorage: ""
  kopiaVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""

```

3. Appliquez les valeurs de `resourceLimits.yaml` déposer:

```

helm upgrade trident-protect -n trident-protect netapp-trident-protect/trident-protect -f resourceLimits.yaml --reuse-values

```

Personnaliser les contraintes de contexte de sécurité

Vous pouvez utiliser un fichier de configuration pour modifier les contraintes de contexte de sécurité OpenShift (SCC) pour les conteneurs Trident Protect après avoir installé Trident Protect. Ces contraintes définissent les restrictions de sécurité des pods dans un cluster Red Hat OpenShift.

Étapes

1. Créez un fichier nommé `sccconfig.yaml`.
2. Ajoutez l'option SCC au fichier et modifiez les paramètres en fonction des besoins de votre environnement.

L'exemple suivant illustre les valeurs par défaut des paramètres de l'option SCC :

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

Ce tableau décrit les paramètres de l'option SCC :

Paramètre	Description	Défaut
créer	Détermine si une ressource SCC peut être créée. Une ressource SCC ne sera créée que si <code>scc.create</code> est réglé sur <code>true</code> et le processus d'installation de Helm identifie un environnement OpenShift. Si vous n'utilisez pas OpenShift, ou si <code>scc.create</code> est réglé sur <code>false</code> Aucune ressource SCC ne sera créée.	true
nom	Spécifie le nom du SCC.	trident-protect-emploi
priorité	Définit la priorité du SCC. Les SCC ayant des valeurs de priorité plus élevées sont évaluées avant celles ayant des valeurs plus faibles.	1

3. Appliquez les valeurs de `sccconfig.yaml` déposer:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect -f
sccconfig.yaml --reuse-values
```

Cela remplacera les valeurs par défaut par celles spécifiées dans le `sccconfig.yaml` déposer.

Configurer les paramètres supplémentaires du graphique de barre de Trident Protect

Vous pouvez personnaliser les paramètres AutoSupport et le filtrage des espaces de noms pour répondre à vos besoins spécifiques. Le tableau suivant décrit les paramètres de configuration disponibles :

Paramètre	Type	Description
autoSupport.proxy	chaîne	Configure une URL proxy pour les connexions NetApp AutoSupport . Utilisez ceci pour acheminer les téléchargements de bundles de support via un serveur proxy. Exemple: http://my.proxy.url .

Paramètre	Type	Description
autoSupport.insecure	booléen	Ignore la vérification TLS pour les connexions proxy AutoSupport lorsque cette option est activée. <code>true</code> . À utiliser uniquement pour les connexions proxy non sécurisées. (défaut: <code>false</code>)
autoSupport.activé	booléen	Active ou désactive les téléchargements quotidiens des modules Trident Protect AutoSupport . Lorsqu'il est réglé sur <code>false</code> Les téléchargements quotidiens programmés sont désactivés, mais vous pouvez toujours générer manuellement des ensembles de support. (défaut: <code>true</code>)
restaurerSkipNamespaceAnnotations	chaîne	Liste séparée par des virgules d'annotations d'espace de noms à exclure des opérations de sauvegarde et de restauration. Vous permet de filtrer les espaces de noms en fonction des annotations.
restaurerSkipNamespaceLabels	chaîne	Liste séparée par des virgules d'étiquettes d'espace de noms à exclure des opérations de sauvegarde et de restauration. Vous permet de filtrer les espaces de noms en fonction des étiquettes.

Vous pouvez configurer ces options à l'aide d'un fichier de configuration YAML ou d'indicateurs de ligne de commande :

Utiliser le fichier YAML

Étapes

1. Créez un fichier de configuration et nommez-le `values.yaml`.
2. Dans le fichier que vous avez créé, ajoutez les options de configuration que vous souhaitez personnaliser.

```
autoSupport:
  enabled: false
  proxy: http://my.proxy.url
  insecure: true
restoreSkipNamespaceAnnotations: "annotation1,annotation2"
restoreSkipNamespaceLabels: "label1,label2"
```

3. Après avoir rempli le `values.yaml` Fichier contenant les valeurs correctes, appliquez le fichier de configuration :

```
helm upgrade trident-protect -n trident-protect netapp-trident-protect/trident-protect -f values.yaml --reuse-values
```

Utiliser l'indicateur CLI

Étapes

1. Utilisez la commande suivante avec le `--set` indicateur permettant de spécifier des paramètres individuels :

```
helm upgrade trident-protect -n trident-protect netapp-trident-protect/trident-protect \
  --set autoSupport.enabled=false \
  --set autoSupport.proxy=http://my.proxy.url \
  --set restoreSkipNamespaceAnnotations="annotation1,annotation2" \
  --set restoreSkipNamespaceLabels="label1,label2" \
  --reuse-values
```

Limiter les pods Trident Protect à des nœuds spécifiques

Vous pouvez utiliser la contrainte de sélection de nœuds `nodeSelector` de Kubernetes pour contrôler quels nœuds sont éligibles pour exécuter des pods Trident Protect, en fonction des étiquettes de nœud. Par défaut, Trident Protect est limité aux nœuds exécutant Linux. Vous pouvez personnaliser davantage ces contraintes en fonction de vos besoins.

Étapes

1. Créez un fichier nommé `nodeSelectorConfig.yaml`.

2. Ajoutez l'option `nodeSelector` au fichier et modifiez ce dernier pour ajouter ou modifier les étiquettes des nœuds afin de les adapter aux besoins de votre environnement. Par exemple, le fichier suivant contient la restriction par défaut du système d'exploitation, mais cible également une région et un nom d'application spécifiques :

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. Appliquez les valeurs de `nodeSelectorConfig.yaml` déposer:

```
helm upgrade trident-protect -n trident-protect netapp-trident-protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

Cela remplace les restrictions par défaut par celles que vous avez spécifiées dans le `nodeSelectorConfig.yaml` déposer.

Gérer Trident Protect

Gérer l'autorisation et le contrôle d'accès de Trident Protect

Trident Protect utilise le modèle Kubernetes de contrôle d'accès basé sur les rôles (RBAC). Par défaut, Trident Protect fournit un seul espace de noms système et son compte de service par défaut associé. Si votre organisation compte de nombreux utilisateurs ou des besoins de sécurité spécifiques, vous pouvez utiliser les fonctionnalités RBAC de Trident Protect pour obtenir un contrôle plus précis sur l'accès aux ressources et aux espaces de noms.

L'administrateur du cluster a toujours accès aux ressources par défaut `trident-protect` espace de noms, et peut également accéder aux ressources de tous les autres espaces de noms. Pour contrôler l'accès aux ressources et aux applications, vous devez créer des espaces de noms supplémentaires et y ajouter des ressources et des applications.

Notez qu'aucun utilisateur ne peut créer de demandes de changement (CR) de gestion des données d'application par défaut. `trident-protect` espace de noms. Vous devez créer des CR de gestion des données d'application dans un espace de noms d'application (il est recommandé de créer les CR de gestion des données d'application dans le même espace de noms que leur application associée).

Seuls les administrateurs devraient avoir accès aux objets de ressources personnalisés privilégiés de Trident Protect, notamment :



- **AppVault** : Nécessite des données d'identification de compartiment
- **AutoSupportBundle** : Collecte les métriques, les journaux et autres données sensibles de Trident Protect
- **AutoSupportBundleSchedule** : Gère les planifications de collecte des journaux

Il est recommandé d'utiliser le contrôle d'accès basé sur les rôles (RBAC) pour limiter l'accès aux objets privilégiés aux seuls administrateurs.

Pour plus d'informations sur la manière dont le RBAC régule l'accès aux ressources et aux espaces de noms, consultez la documentation. "[Documentation RBAC Kubernetes](#)".

Pour plus d'informations sur les comptes de service, veuillez consulter le "[Documentation des comptes de service Kubernetes](#)".

Exemple : Gérer l'accès pour deux groupes d'utilisateurs

Par exemple, une organisation possède un administrateur de cluster, un groupe d'utilisateurs ingénieurs et un groupe d'utilisateurs marketing. L'administrateur du cluster effectuerait les tâches suivantes pour créer un environnement où le groupe d'ingénierie et le groupe marketing n'auraient accès qu'aux ressources attribuées à leurs espaces de noms respectifs.

Étape 1 : Créer un espace de noms pour contenir les ressources de chaque groupe

La création d'un espace de noms vous permet de séparer logiquement les ressources et de mieux contrôler qui y a accès.

Étapes

1. Créez un espace de noms pour le groupe d'ingénierie :

```
kubectl create ns engineering-ns
```

2. Créez un espace de noms pour le groupe marketing :

```
kubectl create ns marketing-ns
```

Étape 2 : Créez de nouveaux comptes de service pour interagir avec les ressources de chaque espace de noms

Chaque nouvel espace de noms que vous créez est fourni avec un compte de service par défaut, mais vous devriez créer un compte de service pour chaque groupe d'utilisateurs afin de pouvoir répartir davantage les privilèges entre les groupes à l'avenir si nécessaire.

Étapes

1. Créer un compte de service pour le groupe d'ingénierie :

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Créer un compte de service pour le groupe marketing :

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Étape 3 : Créez un secret pour chaque nouveau compte de service

Un secret de compte de service est utilisé pour s'authentifier auprès du compte de service et peut être facilement supprimé et recréé en cas de compromission.

Étapes

1. Créez un secret pour le compte du service d'ingénierie :

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
type: kubernetes.io/service-account-token
```

2. Créez un secret pour le compte du service marketing :

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
type: kubernetes.io/service-account-token
```

Étape 4 : Créez un objet RoleBinding pour lier l'objet ClusterRole à chaque nouveau compte de service.

Un objet ClusterRole par défaut est créé lors de l'installation de Trident Protect. Vous pouvez lier ce ClusterRole au compte de service en créant et en appliquant un objet RoleBinding.

Étapes

1. Associez le rôle ClusterRole au compte de service d'ingénierie :

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

2. Associez le rôle ClusterRole au compte de service marketing :

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

Étape 5 : Tester les autorisations

Vérifiez que les autorisations sont correctes.

Étapes

1. Confirmer que les utilisateurs ingénieurs peuvent accéder aux ressources d'ingénierie :

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user  
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Confirmer que les utilisateurs ingénieurs n'ont pas accès aux ressources marketing :

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user  
get applications.protect.trident.netapp.io -n marketing-ns
```

Étape 6 : Accorder l'accès aux objets AppVault

Pour effectuer des tâches de gestion des données telles que les sauvegardes et les instantanés, l'administrateur du cluster doit accorder l'accès aux objets AppVault aux utilisateurs individuels.

Étapes

1. Créez et appliquez un fichier YAML combinant AppVault et secret, qui accorde à un utilisateur l'accès à un AppVault. Par exemple, la demande de changement suivante accorde à l'utilisateur l'accès à un AppVault eng-user :

```

apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident Protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3

```

2. Créez et appliquez un rôle CR pour permettre aux administrateurs de cluster d'accorder l'accès à des ressources spécifiques dans un espace de noms. Par exemple:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Créez et appliquez une ressource personnalisée RoleBinding pour lier les autorisations à l'utilisateur eng-user. Par exemple:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Vérifiez que les autorisations sont correctes.

- a. Tentative de récupération des informations des objets AppVault pour tous les espaces de noms :

```
kubectl get appvaults -n trident-protect
--as=system:serviceaccount:engineering-ns:eng-user
```

Vous devriez obtenir un résultat similaire à celui-ci :

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is forbidden: User "system:serviceaccount:engineering-ns:eng-user" cannot list resource "appvaults" in API group "protect.trident.netapp.io" in the namespace "trident-protect"
```

- b. Vérifiez si l'utilisateur peut accéder aux informations AppVault auxquelles il est désormais autorisé à accéder :

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n trident-protect
```

Vous devriez obtenir un résultat similaire à celui-ci :

```
yes
```

Résultat

Les utilisateurs auxquels vous avez accordé des autorisations AppVault doivent pouvoir utiliser les objets AppVault autorisés pour les opérations de gestion des données d'application, et ne doivent pas pouvoir accéder à des ressources en dehors des espaces de noms attribués ni créer de nouvelles ressources auxquelles ils n'ont pas accès.

Surveiller les ressources de Trident Protect

Vous pouvez utiliser les outils open source kube-state-metrics, Prometheus et Alertmanager pour surveiller l'état des ressources protégées par Trident Protect.

Le service kube-state-metrics génère des métriques à partir des communications de l'API Kubernetes. Son utilisation conjointe avec Trident Protect permet d'obtenir des informations utiles sur l'état des ressources de votre environnement.

Prometheus est une boîte à outils capable d'ingérer les données générées par kube-state-metrics et de les présenter sous forme d'informations facilement lisibles sur ces objets. Ensemble, kube-state-metrics et Prometheus vous permettent de surveiller l'état et la santé des ressources que vous gérez avec Trident Protect.

Alertmanager est un service qui ingère les alertes envoyées par des outils tels que Prometheus et les achemine vers des destinations que vous configurez.



Les configurations et les conseils inclus dans ces étapes ne sont que des exemples ; vous devez les personnaliser en fonction de votre environnement. Veuillez vous référer à la documentation officielle suivante pour obtenir des instructions et une assistance spécifiques :

- ["documentation kube-state-metrics"](#)
- ["Documentation Prometheus"](#)
- ["Documentation d'Alertmanager"](#)

Étape 1 : Installer les outils de surveillance

Pour activer la surveillance des ressources dans Trident Protect, vous devez installer et configurer kube-state-metrics, Prometheus et Alertmanager.

Installer kube-state-metrics

Vous pouvez installer kube-state-metrics à l'aide de Helm.

Étapes

1. Ajoutez le graphique Helm kube-state-metrics. Par exemple:

```
helm repo add prometheus-community https://prometheus-  
community.github.io/helm-charts  
helm repo update
```

2. Appliquez le CRD Prometheus ServiceMonitor au cluster :

```
kubectl apply -f https://raw.githubusercontent.com/prometheus-  
operator/prometheus-operator/main/example/prometheus-operator-  
crd/monitoring.coreos.com_servicemonitors.yaml
```

3. Créez un fichier de configuration pour le graphique Helm (par exemple, `metrics-config.yaml`). Vous pouvez personnaliser la configuration d'exemple suivante pour l'adapter à votre environnement :

metrics-config.yaml : configuration du graphique Helm kube-state-metrics

```
---
extraArgs:
  # Collect only custom metrics
  - --custom-resource-state-only=true

customResourceState:
  enabled: true
  config:
    kind: CustomResourceStateMetrics
    spec:
      resources:
        - groupVersionKind:
            group: protect.trident.netapp.io
            kind: "Backup"
            version: "v1"
          labelsFromPath:
            backup_uid: [metadata, uid]
            backup_name: [metadata, name]
            creation_time: [metadata, creationTimestamp]
          metrics:
            - name: backup_info
              help: "Exposes details about the Backup state"
              each:
                type: Info
                info:
                  labelsFromPath:
                    appVaultReference: ["spec", "appVaultRef"]
                    appReference: ["spec", "applicationRef"]
rbac:
  extraRules:
    - apiGroups: ["protect.trident.netapp.io"]
      resources: ["backups"]
      verbs: ["list", "watch"]

# Collect metrics from all namespaces
namespaces: ""

# Ensure that the metrics are collected by Prometheus
prometheus:
  monitor:
    enabled: true
```

4. Installez kube-state-metrics en déployant le graphique Helm. Par exemple:

```
helm install custom-resource -f metrics-config.yaml prometheus-
community/kube-state-metrics --version 5.21.0
```

5. Configurez kube-state-metrics pour générer des métriques pour les ressources personnalisées utilisées par Trident Protect en suivant les instructions du guide. "[Documentation sur la ressource personnalisée kube-state-metrics](#)".

Installer Prometheus

Vous pouvez installer Prometheus en suivant les instructions du "[Documentation Prometheus](#)".

Installer Alertmanager

Vous pouvez installer Alertmanager en suivant les instructions du "[Documentation d'Alertmanager](#)".

Étape 2 : Configurer les outils de surveillance pour qu'ils fonctionnent ensemble

Après avoir installé les outils de surveillance, vous devez les configurer pour qu'ils fonctionnent ensemble.

Étapes

1. Intégrer kube-state-metrics avec Prometheus. Modifier le fichier de configuration Prometheus(`prometheus.yaml`) et ajoutez les informations du service kube-state-metrics. Par exemple:

prometheus.yaml : intégration du service kube-state-metrics avec Prometheus

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
  namespace: trident-protect
data:
  prometheus.yaml: |
    global:
      scrape_interval: 15s
    scrape_configs:
      - job_name: 'kube-state-metrics'
        static_configs:
          - targets: ['kube-state-metrics.trident-protect.svc:8080']
```

2. Configurez Prometheus pour acheminer les alertes vers Alertmanager. Modifier le fichier de configuration Prometheus(`prometheus.yaml`) et ajoutez la section suivante :

prometheus.yaml : Envoyer des alertes à Alertmanager

```
alerting:
  alertmanagers:
    - static_configs:
      - targets:
          - alertmanager.trident-protect.svc:9093
```

Résultat

Prometheus peut désormais collecter des métriques à partir de kube-state-metrics et envoyer des alertes à Alertmanager. Vous êtes maintenant prêt à configurer les conditions qui déclenchent une alerte et les destinations de ces alertes.

Étape 3 : Configurer les alertes et leurs destinations

Une fois les outils configurés pour fonctionner ensemble, vous devez configurer le type d'informations qui déclenchent les alertes et l'endroit où ces alertes doivent être envoyées.

Exemple d'alerte : échec de la sauvegarde

L'exemple suivant définit une alerte critique qui se déclenche lorsque l'état de la ressource personnalisée de sauvegarde est défini sur `Error` pendant 5 secondes ou plus. Vous pouvez personnaliser cet exemple pour qu'il corresponde à votre environnement et inclure cet extrait YAML dans votre fichier `prometheus.yaml` fichier de configuration :

rules.yaml : Définir une alerte Prometheus pour les sauvegardes ayant échoué

```
rules.yaml: |
  groups:
    - name: fail-backup
      rules:
        - alert: BackupFailed
          expr: kube_customresource_backup_info{status="Error"}
          for: 5s
          labels:
            severity: critical
          annotations:
            summary: "Backup failed"
            description: "A backup has failed."
```

Configurez Alertmanager pour envoyer des alertes vers d'autres canaux.

Vous pouvez configurer Alertmanager pour envoyer des notifications vers d'autres canaux, tels que le courrier électronique, PagerDuty, Microsoft Teams ou d'autres services de notification, en spécifiant la configuration correspondante dans le `alertmanager.yaml` déposer.

L'exemple suivant configure Alertmanager pour envoyer des notifications à un canal Slack. Pour adapter cet exemple à votre environnement, remplacez la valeur de `api_url` clé avec l'URL du webhook Slack utilisée dans votre environnement :

alertmanager.yaml : Envoyer les alertes à un canal Slack

```
data:
  alertmanager.yaml: |
    global:
      resolve_timeout: 5m
    route:
      receiver: 'slack-notifications'
    receivers:
      - name: 'slack-notifications'
        slack_configs:
          - api_url: '<your-slack-webhook-url>'
            channel: '#failed-backups-channel'
            send_resolved: false
```

Générer un ensemble de support Trident Protect

Trident Protect permet aux administrateurs de générer des ensembles contenant des informations utiles au support NetApp , notamment des journaux, des métriques et des informations de topologie sur les clusters et les applications gérés. Si vous êtes connecté à Internet, vous pouvez télécharger des bundles de support sur le site de support NetApp (NSS) à l'aide d'un fichier de ressources personnalisé (CR).

Créez un ensemble de support à l'aide d'une demande de changement.

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-support-bundle.yaml`).
2. Configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.triggerType** : (*Obligatoire*) Détermine si le bundle de support est généré immédiatement ou planifié. La génération des paquets est programmée à minuit UTC. Valeurs possibles :
 - Programmé
 - Manuel
 - **spec.uploadEnabled** : (*Optionnel*) Contrôle si le bundle de support doit être téléchargé sur le site de support NetApp après sa génération. Si non spécifié, la valeur par défaut est `false`. Valeurs possibles :
 - `true`
 - faux (par défaut)
 - **spec.dataWindowStart** : (*Optionnel*) Une chaîne de date au format RFC 3339 qui spécifie la date et l'heure auxquelles la fenêtre de données incluses dans le bundle de support doit commencer. Si aucune valeur n'est spécifiée, la date par défaut est celle d'il y a 24 heures. La date la plus ancienne que vous pouvez spécifier remonte à 7 jours.

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Après avoir rempli le `trident-protect-support-bundle.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-support-bundle.yaml -n trident-protect
```

Créez un ensemble de support à l'aide de l'interface de ligne de commande (CLI).

Étapes

1. Créez le paquet de support en remplaçant les valeurs entre crochets par les informations provenant

de votre environnement. Le `trigger-type` détermine si le lot est créé immédiatement ou si le délai de création est dicté par la planification, et peut être `Manual` ou `Scheduled`. Le paramètre par défaut est `Manual`.

Par exemple:

```
tridentctl-protect create autosupportbundle <my-bundle-name>
--trigger-type <trigger-type> -n trident-protect
```

Surveiller et récupérer le pack de support

Après avoir créé un bundle de support à l'aide de l'une ou l'autre méthode, vous pouvez surveiller sa progression de génération et le récupérer sur votre système local.

Étapes

1. Attendez le `status.generationState` atteindre `Completed` État. Vous pouvez surveiller la progression de la génération avec la commande suivante :

```
kubectl get autosupportbundle trident-protect-support-bundle -n trident-protect
```

2. Récupérez le pack de support sur votre système local. Obtenez la commande de copie à partir du bundle `AutoSupport` terminé :

```
kubectl describe autosupportbundle trident-protect-support-bundle -n trident-protect
```

Trouvez le `kubectl cp` Copiez la commande affichée dans le résultat et exécutez-la en remplaçant l'argument de destination par votre répertoire local préféré.

Amélioration de Trident Protect

Vous pouvez mettre à jour Trident Protect vers la dernière version pour bénéficier des nouvelles fonctionnalités ou des correctifs de bugs.



Lors de la mise à niveau depuis la version 24.10, les instantanés exécutés pendant la mise à niveau peuvent échouer. Cette défaillance n'empêche pas la création de futurs instantanés, qu'ils soient manuels ou planifiés. Si la création d'un instantané échoue lors de la mise à niveau, vous pouvez en créer manuellement un nouveau pour garantir la protection de votre application.

Pour éviter d'éventuelles défaillances, vous pouvez désactiver toutes les planifications de capture d'instantanés avant la mise à niveau et les réactiver après. Cependant, cela a pour conséquence de manquer toutes les captures d'écran planifiées pendant la période de mise à niveau.

Pour mettre à niveau Trident Protect, procédez comme suit.

Étapes

1. Mettre à jour le dépôt Trident Helm :

```
helm repo update
```

2. Mettez à niveau les CRD Trident Protect :



Cette étape est nécessaire si vous effectuez une mise à niveau depuis une version antérieure à la 25.06, car les CRD sont désormais inclus dans le tableau Trident Protect Helm.

- a. Exécutez cette commande pour transférer la gestion des CRD depuis `trident-protect-crds` à `trident-protect` :

```
kubectl get crd | grep protect.trident.netapp.io | awk '{print $1}' |  
xargs -I {} kubectl patch crd {} --type merge -p '{"metadata":  
{"annotations":{"meta.helm.sh/release-name": "trident-protect"}}}'
```

- b. Exécutez cette commande pour supprimer le secret Helm pour le `trident-protect-crds` graphique:



Ne désinstallez pas le `trident-protect-crds` Utilisez Helm pour créer un graphique, car cela pourrait supprimer vos CRD et toutes les données associées.

```
kubectl delete secret -n trident-protect -l name=trident-protect-  
crds,owner=helm
```

3. Mise à niveau de Trident Protect :

```
helm upgrade trident-protect netapp-trident-protect/trident-protect  
--version 100.2506.0 --namespace trident-protect
```

Gérer et protéger les applications

Utilisez les objets Trident Protect AppVault pour gérer les compartiments.

La ressource personnalisée (CR) du compartiment pour Trident Protect est connue sous le nom d'AppVault. Les objets AppVault sont la représentation déclarative du flux de travail Kubernetes d'un bucket de stockage. Un AppVault CR contient les configurations nécessaires pour qu'un bucket soit utilisé dans les opérations de protection, telles que les sauvegardes, les snapshots, les opérations de restauration et la réplication SnapMirror .

Seuls les administrateurs peuvent créer des AppVaults.

Vous devez créer une CR AppVault manuellement ou depuis la ligne de commande lorsque vous effectuez des opérations de protection des données sur une application. La CR AppVault est spécifique à votre environnement et vous pouvez vous inspirer des exemples de cette page pour créer des CR AppVault.



Assurez-vous que le fichier CR d'AppVault se trouve sur le cluster où Trident Protect est installé. Si le CR AppVault n'existe pas ou si vous ne pouvez pas y accéder, la ligne de commande affiche une erreur.

Configurer l'authentification et les mots de passe AppVault

Avant de créer un AppVault CR, assurez-vous que l'AppVault et le dispositif de transfert de données que vous choisissez peuvent s'authentifier auprès du fournisseur et de toutes les ressources associées.

Mots de passe du référentiel de déplacement de données

Lorsque vous créez des objets AppVault à l'aide de CR ou du plugin CLI Trident Protect, vous pouvez spécifier un secret Kubernetes avec des mots de passe personnalisés pour le chiffrement Restic et Kopia. Si vous ne spécifiez pas de mot de passe secret, Trident Protect utilise un mot de passe par défaut.

- Lors de la création manuelle de demandes de changement AppVault, utilisez le champ **spec.dataMoverPasswordSecretRef** pour spécifier le secret.
- Lors de la création d'objets AppVault à l'aide de l'interface de ligne de commande Trident Protect, utilisez le `--data-mover-password-secret-ref` argument permettant de préciser le secret.

Créer un mot de passe secret pour le référentiel de déplacement de données

Utilisez les exemples suivants pour créer le mot de passe secret. Lorsque vous créez des objets AppVault, vous pouvez indiquer à Trident Protect d'utiliser ce secret pour s'authentifier auprès du référentiel de transfert de données.



- Selon le logiciel de transfert de données que vous utilisez, il vous suffit d'indiquer le mot de passe correspondant. Par exemple, si vous utilisez Restic et que vous ne prévoyez pas d'utiliser Kopia à l'avenir, vous pouvez inclure uniquement le mot de passe Restic lors de la création du secret.
- Conservez le mot de passe en lieu sûr. Vous en aurez besoin pour restaurer les données sur le même cluster ou sur un autre. Si le cluster ou le `trident-protect` L'espace de noms a été supprimé ; vous ne pourrez pas restaurer vos sauvegardes ou vos instantanés sans le mot de passe.

Utilisez un CR

```
---
apiVersion: v1
data:
  KOPIA_PASSWORD: <base64-encoded-password>
  RESTIC_PASSWORD: <base64-encoded-password>
kind: Secret
metadata:
  name: my-optional-data-mover-secret
  namespace: trident-protect
type: Opaque
```

Utiliser la CLI

```
kubectl create secret generic my-optional-data-mover-secret \
--from-literal=KOPIA_PASSWORD=<plain-text-password> \
--from-literal=RESTIC_PASSWORD=<plain-text-password> \
-n trident-protect
```

Autorisations IAM de stockage compatibles S3

Lorsque vous accédez à un stockage compatible S3 tel qu'Amazon S3, Generic S3, "StorageGrid S3", ou "ONTAP S3" Lors de l'utilisation de Trident Protect, vous devez vous assurer que les informations d'identification de l'utilisateur que vous fournissez disposent des autorisations nécessaires pour accéder au compartiment. Voici un exemple de politique accordant les autorisations minimales requises pour l'accès avec Trident Protect. Vous pouvez appliquer cette politique à l'utilisateur qui gère les politiques de compartiment compatibles S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

Pour plus d'informations sur les politiques Amazon S3, reportez-vous aux exemples dans la documentation. ["Documentation Amazon S3"](#) .

Identité du pod EKS pour l'authentification Amazon S3 (AWS)

Trident Protect prend en charge EKS Pod Identity pour les opérations de déplacement de données Kopia. Cette fonctionnalité permet un accès sécurisé aux buckets S3 sans stocker les informations d'identification AWS dans les secrets Kubernetes.

Configuration requise pour l'identification du pod EKS avec Trident Protect

Avant d'utiliser EKS Pod Identity avec Trident Protect, assurez-vous des points suivants :

- L'identité de pod est activée sur votre cluster EKS.
- Vous avez créé un rôle IAM avec les autorisations de compartiment S3 nécessaires. Pour en savoir plus, consultez ["Autorisations IAM de stockage compatibles S3"](#).
- Le rôle IAM est associé aux comptes de service Trident Protect suivants :
 - `<trident-protect>-controller-manager`
 - `<trident-protect>-resource-backup`
 - `<trident-protect>-resource-restore`
 - `<trident-protect>-resource-delete`

Pour obtenir des instructions détaillées sur l'activation de Pod Identity et l'association des rôles IAM aux comptes de service, veuillez consulter la documentation. ["Documentation sur l'identité des pods AWS EKS"](#) .

Configuration d'AppVault Lorsque vous utilisez EKS Pod Identity, configurez votre ressource personnalisée AppVault avec le `useIAM: true` drapeau au lieu d'identifiants explicites :

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: eks-protect-vault
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-aws
      endpoint: s3.example.com
      useIAM: true
```

Exemples de génération de clés AppVault pour les fournisseurs de cloud

Lors de la définition d'un AppVault CR, vous devez inclure des informations d'identification pour accéder aux ressources hébergées par le fournisseur, sauf si vous utilisez l'authentification IAM. La manière dont vous générez les clés pour les informations d'identification varie selon le fournisseur. Voici des exemples de génération de clés en ligne de commande pour plusieurs fournisseurs. Vous pouvez utiliser les exemples suivants pour créer des clés pour les informations d'identification de chaque fournisseur de cloud.

Google Cloud

```
kubectl create secret generic <secret-name> \  
--from-file=credentials=<mycreds-file.json> \  
-n trident-protect
```

Amazon S3 (AWS)

```
kubectl create secret generic <secret-name> \  
--from-literal=accessKeyID=<objectstorage-accesskey> \  
--from-literal=secretAccessKey=<amazon-s3-trident-protect-src-bucket  
-secret> \  
-n trident-protect
```

Microsoft Azure

```
kubectl create secret generic <secret-name> \  
--from-literal=accountKey=<secret-name> \  
-n trident-protect
```

S3 générique

```
kubectl create secret generic <secret-name> \  
--from-literal=accessKeyID=<objectstorage-accesskey> \  
--from-literal=secretAccessKey=<generic-s3-trident-protect-src-bucket  
-secret> \  
-n trident-protect
```

ONTAP S3

```
kubectl create secret generic <secret-name> \  
--from-literal=accessKeyID=<objectstorage-accesskey> \  
--from-literal=secretAccessKey=<ontap-s3-trident-protect-src-bucket  
-secret> \  
-n trident-protect
```

StorageGrid S3

```
kubectl create secret generic <secret-name> \  
--from-literal=accessKeyID=<objectstorage-accesskey> \  
--from-literal=secretAccessKey=<storagegrid-s3-trident-protect-src  
-bucket-secret> \  
-n trident-protect
```

Exemples de création d'AppVault

Vous trouverez ci-dessous des exemples de définitions AppVault pour chaque fournisseur.

Exemples de CR AppVault

Vous pouvez utiliser les exemples de CR suivants pour créer des objets AppVault pour chaque fournisseur de cloud.



- Vous pouvez éventuellement spécifier un secret Kubernetes contenant des mots de passe personnalisés pour le chiffrement des référentiels Restic et Kopia. Se référer à [Mots de passe du référentiel de déplacement de données](#) pour plus d'informations.
- Pour les objets Amazon S3 (AWS) AppVault, vous pouvez éventuellement spécifier un jeton de session, ce qui est utile si vous utilisez l'authentification unique (SSO). Ce jeton est créé lorsque vous générez des clés pour le fournisseur dans [Exemples de génération de clés AppVault pour les fournisseurs de cloud](#).
- Pour les objets S3 AppVault, vous pouvez éventuellement spécifier une URL de proxy de sortie pour le trafic S3 sortant à l'aide de `spec.providerConfig.S3.proxyURL` clé.

Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GCP
  providerConfig:
    gcp:
      bucketName: trident-protect-src-bucket
      projectID: project-id
  providerCredentials:
    credentials:
      valueFromSecret:
        key: credentials
        name: gcp-trident-protect-src-bucket-secret
```

Amazon S3 (AWS)

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
    sessionToken:
      valueFromSecret:
        key: sessionToken
        name: s3-secret
```



Pour les environnements EKS utilisant Pod Identity avec Kopia Data Mover, vous pouvez supprimer le `providerCredentials` section et ajouter `useIAM: true` sous le `s3` plutôt une configuration.

Microsoft Azure

```

apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret

```

S3 générique

```

apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret

```

ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

StorageGrid S3

```

apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret

```

Exemples de création d'AppVault à l'aide de l'interface de ligne de commande Trident Protect

Vous pouvez utiliser les exemples de commandes CLI suivants pour créer des demandes de changement AppVault pour chaque fournisseur.



- Vous pouvez éventuellement spécifier un secret Kubernetes contenant des mots de passe personnalisés pour le chiffrement des référentiels Restic et Kopia. Se référer à [Mots de passe du référentiel de déplacement de données](#) pour plus d'informations.
- Pour les objets S3 AppVault, vous pouvez éventuellement spécifier une URL de proxy de sortie pour le trafic S3 sortant à l'aide de `--proxy-url <ip_address:port>` argument.

Google Cloud

```
tridentctl-protect create vault GCP <vault-name> \  
--bucket <mybucket> \  
--project <my-gcp-project> \  
--secret <secret-name>/credentials \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> \  
--bucket <bucket-name> \  
--secret <secret-name> \  
--endpoint <s3-endpoint> \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> \  
--account <account-name> \  
--bucket <bucket-name> \  
--secret <secret-name> \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

S3 générique

```
tridentctl-protect create vault GenericS3 <vault-name> \  
--bucket <bucket-name> \  
--secret <secret-name> \  
--endpoint <s3-endpoint> \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> \  
--bucket <bucket-name> \  
--secret <secret-name> \  
--endpoint <s3-endpoint> \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

StorageGrid S3

```
tridentctl-protect create vault StorageGridS3 <vault-name> \  
--bucket <bucket-name> \  
--secret <secret-name> \  
--endpoint <s3-endpoint> \  
--data-mover-password-secret-ref <my-optional-data-mover-secret> \  
-n trident-protect
```

Consultez les informations d'AppVault

Vous pouvez utiliser le plugin CLI Trident Protect pour consulter les informations relatives aux objets AppVault que vous avez créés sur le cluster.

Étapes

1. Afficher le contenu d'un objet AppVault :

```
tridentctl-protect get appvaultcontent gcp-vault \  
--show-resources all \  
-n trident-protect
```

Exemple de résultat :

```

+-----+-----+-----+-----+
+-----+
| CLUSTER | APP | TYPE | NAME |
TIMESTAMP |
+-----+-----+-----+-----+
+-----+
|          | mysql | snapshot | mysnap | 2024-
08-09 21:02:11 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815180300 | 2024-
08-15 18:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815200300 | 2024-
08-15 20:03:06 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815180300 | 2024-
08-15 18:04:25 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:30 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815200300 | 2024-
08-15 20:04:21 (UTC) |
| production1 | mysql | backup | mybackup5 | 2024-
08-09 22:25:13 (UTC) |
|          | mysql | backup | mybackup | 2024-
08-09 21:02:52 (UTC) |
+-----+-----+-----+-----+
+-----+

```

2. Pour afficher le chemin AppVaultPath de chaque ressource, vous pouvez également utiliser l'indicateur. `--show-paths`.

Le nom du cluster dans la première colonne du tableau n'est disponible que si un nom de cluster a été spécifié dans l'installation Helm de Trident Protect. Par exemple: `--set clusterName=production1`.

Supprimer un AppVault

Vous pouvez supprimer un objet AppVault à tout moment.



Ne retirez pas le `finalizers` Saisissez la clé dans la ressource personnalisée AppVault avant de supprimer l'objet AppVault. Si vous procédez ainsi, cela peut entraîner la présence de données résiduelles dans le compartiment AppVault et de ressources orphelines dans le cluster.

Avant de commencer

Assurez-vous d'avoir supprimé toutes les ressources de configuration (CR) de snapshot et de sauvegarde utilisées par l'AppVault que vous souhaitez supprimer.

Supprimer un AppVault à l'aide de l'interface de ligne de commande Kubernetes

1. Supprimez l'objet AppVault, en le remplaçant `appvault-name` avec le nom de l'objet AppVault à supprimer :

```
kubectl delete appvault <appvault-name> \  
-n trident-protect
```

Supprimer un AppVault à l'aide de l'interface de ligne de commande Trident Protect

1. Supprimez l'objet AppVault, en le remplaçant `appvault-name` avec le nom de l'objet AppVault à supprimer :

```
tridentctl-protect delete appvault <appvault-name> \  
-n trident-protect
```

Définissez une application de gestion avec Trident Protect

Vous pouvez définir une application que vous souhaitez gérer avec Trident Protect en créant une demande de changement d'application et une demande de changement AppVault associée.

Créer une demande de changement AppVault

Vous devez créer une ressource personnalisée AppVault qui sera utilisée lors des opérations de protection des données sur l'application, et cette ressource personnalisée AppVault doit résider sur le cluster où Trident Protect est installé. La demande de changement (CR) AppVault est spécifique à votre environnement ; pour des exemples de CR AppVault, reportez-vous à "[Ressources personnalisées AppVault](#)."

Définir une application

Vous devez définir chaque application que vous souhaitez gérer avec Trident Protect. Vous pouvez définir une application de gestion soit en créant manuellement une demande de changement d'application, soit en utilisant l'interface de ligne de commande (CLI) de Trident Protect.

Ajouter une application utilisant un CR

Étapes

1. Créez le fichier CR de l'application de destination :
 - a. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `maria-app.yaml`).
 - b. Configurez les attributs suivants :
 - **metadata.name:** (*Obligatoire*) Le nom de la ressource personnalisée de l'application. Notez le nom que vous choisissez, car d'autres fichiers CR nécessaires aux opérations de protection font référence à cette valeur.
 - **spec.includedNamespaces:** (*Obligatoire*) Utilisez le sélecteur d'espace de noms et d'étiquette pour spécifier les espaces de noms et les ressources utilisés par l'application. L'espace de noms de l'application doit figurer dans cette liste. Le sélecteur d'étiquettes est facultatif et peut être utilisé pour filtrer les ressources au sein de chaque espace de noms spécifié.
 - **spec.includedClusterScopedResources:** (*Optionnel*) Utilisez cet attribut pour spécifier les ressources à portée de cluster à inclure dans la définition de l'application. Cet attribut vous permet de sélectionner ces ressources en fonction de leur groupe, de leur version, de leur type et de leurs étiquettes.
 - **groupVersionKind:** (*Obligatoire*) Spécifie le groupe d'API, la version et le type de la ressource à portée de cluster.
 - **labelSelector:** (*Optionnel*) Filtre les ressources à portée de cluster en fonction de leurs étiquettes.
 - **metadata.annotations.protect.trident.netapp.io/skip-vm-freeze:** (*Optional*) Cette annotation s'applique uniquement aux applications définies à partir de machines virtuelles, comme dans les environnements KubeVirt, où les gels du système de fichiers se produisent avant les instantanés. Indiquez si cette application peut écrire sur le système de fichiers lors d'un instantané. Si cette option est activée (`true`), l'application ignore le paramètre global et peut écrire sur le système de fichiers pendant la création d'un instantané. Si cette option est désactivée, l'application ignore le paramètre global et le système de fichiers est figé lors de la création d'un instantané. Si cette annotation est spécifiée mais que l'application ne comporte aucune machine virtuelle dans sa définition, elle est ignorée. Sauf indication contraire, la demande suit la procédure "[Paramètre de gel global Trident Protect](#)".

Si vous devez appliquer cette annotation après la création d'une application, vous pouvez utiliser la commande suivante :

```
kubectl annotate application -n <application CR namespace> <application CR name> protect.trident.netapp.io/skip-vm-freeze="true"
```

+

Exemple de YAML :

+

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  annotations:
    protect.trident.netapp.io/skip-vm-freeze: "false"
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: namespace-1
      labelSelector:
        matchLabels:
          app: example-app
    - namespace: namespace-2
      labelSelector:
        matchLabels:
          app: another-example-app
  includedClusterScopedResources:
    - groupVersionKind:
        group: rbac.authorization.k8s.io
        kind: ClusterRole
        version: v1
      labelSelector:
        matchLabels:
          mylabel: test
```

1. (*Facultatif*) Ajouter un filtrage qui inclut ou exclut les ressources marquées avec des étiquettes particulières :

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `Include` ou `Exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.
 - **resourceMatchers[].group**: (*Optionnel*) Groupe de la ressource à filtrer.
 - **resourceMatchers[].kind**: (*Optionnel*) Type de ressource à filtrer.
 - **resourceMatchers[].version**: (*Optionnel*) Version de la ressource à filtrer.
 - **resourceMatchers[].names**: (*Optionnel*) Noms dans le champ `metadata.name` de Kubernetes de la ressource à filtrer.

- **resourceMatchers[].namespaces**: (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors** : (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .



Lorsque les deux `resourceFilter` et `labelSelector` sont utilisés, `resourceFilter` court d'abord, puis ensuite `labelSelector` est appliquée aux ressources résultantes.

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
  resourceMatchers:
    - group: my-resource-group-1
      kind: my-resource-kind-1
      version: my-resource-version-1
      names: ["my-resource-names"]
      namespaces: ["my-resource-namespaces"]
      labelSelectors: ["trident.netapp.io/os=linux"]
    - group: my-resource-group-2
      kind: my-resource-kind-2
      version: my-resource-version-2
      names: ["my-resource-names"]
      namespaces: ["my-resource-namespaces"]
      labelSelectors: ["trident.netapp.io/os=linux"]
```

2. Après avoir créé la demande de changement (CR) d'application correspondant à votre environnement, appliquez-la. Par exemple:

```
kubectl apply -f maria-app.yaml
```

Étapes

1. Créez et appliquez la définition de l'application en utilisant l'un des exemples suivants, en remplaçant les valeurs entre crochets par les informations de votre environnement. Vous pouvez inclure des espaces de noms et des ressources dans la définition de l'application en utilisant des listes séparées par des virgules avec les arguments indiqués dans les exemples.

Vous pouvez, si vous le souhaitez, utiliser une annotation lors de la création d'une application pour spécifier si celle-ci peut écrire sur le système de fichiers pendant la prise d'un instantané. Ceci ne s'applique qu'aux applications définies à partir de machines virtuelles, comme dans les environnements KubeVirt, où des blocages du système de fichiers se produisent avant la création des instantanés. Si vous définissez l'annotation sur `true` L'application ignore le paramètre global et peut écrire sur le système de fichiers pendant la création d'un instantané. Si vous le configurez à `false`

L'application ignore alors le paramètre global et le système de fichiers est figé lors de la prise d'un instantané. Si vous utilisez l'annotation mais que l'application ne comporte aucune machine virtuelle dans sa définition, l'annotation est ignorée. Si vous n'utilisez pas l'annotation, l'application suit le comportement attendu. "[Paramètre de gel global Trident Protect](#)".

Pour spécifier l'annotation lorsque vous utilisez l'interface de ligne de commande pour créer une application, vous pouvez utiliser la `--annotation` drapeau.

- Créez l'application et utilisez le paramètre global pour le comportement de gel du système de fichiers :

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
```

- Créez l'application et configurez les paramètres locaux de l'application pour le comportement de gel du système de fichiers :

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
--annotation protect.trident.netapp.io/skip-vm-freeze
=<"true" | "false">
```

Vous pouvez utiliser `--resource-filter-include` et `--resource-filter-exclude` indicateurs permettant d'inclure ou d'exclure des ressources en fonction de `resourceSelectionCriteria` tels que le groupe, le type, la version, les étiquettes, les noms et les espaces de noms, comme illustré dans l'exemple suivant :

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
--resource-filter-include
' [{"Group": "apps", "Kind": "Deployment", "Version": "v1", "Names": ["my-
deployment"], "Namespaces": ["my-
namespace"], "LabelSelectors": ["app=my-app"]} ] '
```

Protégez les applications à l'aide de Trident Protect

Vous pouvez protéger toutes les applications gérées par Trident Protect en prenant des instantanés et des sauvegardes à l'aide d'une politique de protection automatisée ou de manière ponctuelle.



Vous pouvez configurer Trident Protect pour geler et dégeler les systèmes de fichiers pendant les opérations de protection des données. ["Apprenez-en davantage sur la configuration du gel du système de fichiers avec Trident Protect."](#)

Créer un instantané à la demande

Vous pouvez créer un instantané à la demande à tout moment.



Les ressources à portée de cluster sont incluses dans une sauvegarde, un instantané ou un clone si elles sont explicitement référencées dans la définition de l'application ou si elles font référence à l'un des espaces de noms de l'application.

Créez un instantané à l'aide d'une CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-snapshot-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.applicationRef** : Le nom Kubernetes de l'application à capturer.
 - **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où le contenu de l'instantané (métadonnées) doit être stocké.
 - **spec.reclaimPolicy**: (*Optionnel*) Définit ce qui arrive à l'AppArchive d'un instantané lorsque le CR de l'instantané est supprimé. Cela signifie que même lorsqu'il est réglé sur `Retain`, le cliché sera supprimé. Options valides :
 - `Retain`(défaut)
 - `Delete`

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Snapshot
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  reclaimPolicy: Delete
```

3. Après avoir rempli le `trident-protect-snapshot-cr.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-snapshot-cr.yaml
```

Créez un instantané à l'aide de l'interface de ligne de commande (CLI).

Étapes

1. Créez l'instantané en remplaçant les valeurs entre crochets par les informations provenant de votre environnement. Par exemple:

```
tridentctl-protect create snapshot <my_snapshot_name> --appvault
<my_appvault_name> --app <name_of_app_to_snapshot> -n
<application_namespace>
```

Créer une sauvegarde à la demande

Vous pouvez sauvegarder une application à tout moment.



Les ressources à portée de cluster sont incluses dans une sauvegarde, un instantané ou un clone si elles sont explicitement référencées dans la définition de l'application ou si elles font référence à l'un des espaces de noms de l'application.

Avant de commencer

Assurez-vous que la durée de validité du jeton de session AWS est suffisante pour toute opération de sauvegarde S3 de longue durée. Si le jeton expire pendant l'opération de sauvegarde, l'opération peut échouer.

- Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
- Se référer à "[Documentation AWS IAM](#)" pour plus d'informations sur les identifiants avec les ressources AWS.

Créez une sauvegarde à l'aide d'une CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-backup-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.applicationRef**: (*Obligatoire*) Le nom Kubernetes de l'application à sauvegarder.
 - **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où le contenu de la sauvegarde doit être stocké.
 - **spec.dataMover**: (*Optionnel*) Une chaîne de caractères indiquant l'outil de sauvegarde à utiliser pour l'opération de sauvegarde. Valeurs possibles (sensible à la casse) :
 - Restic
 - Kopia(défaut)
 - **spec.reclaimPolicy**: (*Optional*) Définit ce qui arrive à une sauvegarde lorsqu'elle est libérée de sa revendication. Valeurs possibles :
 - Delete
 - Retain(défaut)
 - **spec.snapshotRef**: (*Optionnel*): Nom du snapshot à utiliser comme source de la sauvegarde. Si aucune information n'est fournie, un instantané temporaire sera créé et sauvegardé.
 - **metadata.annotations.protect.trident.netapp.io/full-backup** : (*Optionnel*) Cette annotation est utilisée pour spécifier si une sauvegarde doit être non incrémentale. Par défaut, toutes les sauvegardes sont incrémentielles. Toutefois, si cette annotation est définie sur `true`, la sauvegarde devient non incrémentale. Si aucune option n'est spécifiée, la sauvegarde suit le paramètre de sauvegarde incrémentielle par défaut. Il est recommandé d'effectuer périodiquement une sauvegarde complète, puis d'effectuer des sauvegardes incrémentales entre les sauvegardes complètes afin de minimiser les risques liés aux restaurations.

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup: "true"
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Après avoir rempli le `trident-protect-backup-cr.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-backup-cr.yaml
```

Créez une sauvegarde à l'aide de l'interface de ligne de commande (CLI).

Étapes

1. Créez la sauvegarde en remplaçant les valeurs entre crochets par les informations provenant de votre environnement. Par exemple:

```
tridentctl-protect create backup <my_backup_name> --appvault <my-vault-name> --app <name_of_app_to_back_up> --data-mover <Kopia_or_Restic> -n <application_namespace>
```

Vous pouvez utiliser, si vous le souhaitez, le `--full-backup` indicateur permettant de spécifier si une sauvegarde doit être non incrémentale. Par défaut, toutes les sauvegardes sont incrémentielles. Lorsque cette option est utilisée, la sauvegarde devient non incrémentale. Il est recommandé d'effectuer périodiquement une sauvegarde complète, puis d'effectuer des sauvegardes incrémentales entre les sauvegardes complètes afin de minimiser les risques liés aux restaurations.

Élaborer un calendrier de protection des données

Une politique de protection protège une application en créant des instantanés, des sauvegardes ou les deux selon un calendrier défini. Vous pouvez choisir de créer des instantanés et des sauvegardes toutes les heures, tous les jours, toutes les semaines et tous les mois, et vous pouvez spécifier le nombre de copies à conserver. Vous pouvez planifier une sauvegarde complète non incrémentielle en utilisant l'annotation `full-backup-rule`. Par défaut, toutes les sauvegardes sont incrémentielles. L'exécution périodique d'une sauvegarde complète, ainsi que de sauvegardes incrémentielles entre les deux, permet de réduire le risque associé aux restaurations.



- Vous pouvez créer des planifications pour les instantanés uniquement en configurant `backupRetention` à zéro et `snapshotRetention` à une valeur supérieure à zéro. Paramètre `snapshotRetention` La valeur zéro signifie que les sauvegardes planifiées continueront à créer des instantanés, mais ceux-ci seront temporaires et supprimés immédiatement une fois la sauvegarde terminée.
- Les ressources à portée de cluster sont incluses dans une sauvegarde, un instantané ou un clone si elles sont explicitement référencées dans la définition de l'application ou si elles font référence à l'un des espaces de noms de l'application.

Créez un planning à l'aide d'une CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-schedule-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.dataMover**: (*Optionnel*) Une chaîne de caractères indiquant l'outil de sauvegarde à utiliser pour l'opération de sauvegarde. Valeurs possibles (sensible à la casse) :
 - Restic
 - Kopia(défaut)
 - **spec.applicationRef** : Nom Kubernetes de l'application à sauvegarder.
 - **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où le contenu de la sauvegarde doit être stocké.
 - **spec.backupRetention**: Le nombre de sauvegardes à conserver. Zéro indique qu'aucune sauvegarde ne doit être créée (instantanés uniquement).
 - **spec.snapshotRetention** : Le nombre d'instantanés à conserver. La valeur zéro indique qu'aucun instantané ne doit être créé.
 - **specgranularity** : La fréquence à laquelle la planification doit s'exécuter. Valeurs possibles, ainsi que les champs associés obligatoires :
 - Hourly(nécessite que vous précisiez `spec.minute`)
 - Daily(nécessite que vous précisiez `spec.minute` et `spec.hour`)
 - Weekly(nécessite que vous précisiez `spec.minute`, `spec.hour` , et `spec.dayOfWeek`)
 - Monthly(nécessite que vous précisiez `spec.minute`, `spec.hour` , et `spec.dayOfMonth`)
 - Custom
 - **spec.dayOfMonth**: (*Facultatif*) Le jour du mois (1 - 31) auquel la planification doit s'exécuter. Ce champ est obligatoire si la granularité est définie sur `Monthly` . La valeur doit être fournie sous forme de chaîne.
 - **spec.dayOfWeek**: (*Facultatif*) Le jour de la semaine (0 - 7) pendant lequel la planification doit s'exécuter. Les valeurs de 0 ou 7 indiquent dimanche. Ce champ est obligatoire si la granularité est définie sur `Weekly` . La valeur doit être fournie sous forme de chaîne.
 - **spec.hour**: (*Facultatif*) L'heure de la journée (0 - 23) à laquelle le programme doit s'exécuter. Ce champ est obligatoire si la granularité est définie sur `Daily` , `Weekly` , ou `Monthly` . La valeur doit être fournie sous forme de chaîne.
 - **spec.minute**: (*Facultatif*) La minute de l'heure (0 - 59) à laquelle la planification doit s'exécuter. Ce champ est obligatoire si la granularité est définie sur `Hourly` , `Daily` , `Weekly` , ou `Monthly` . La valeur doit être fournie sous forme de chaîne.
 - **metadata.annotations.protect.trident.netapp.io/full-backup-rule**: (*Optionnel*) Cette annotation est utilisée pour spécifier la règle de planification de la sauvegarde complète. Vous pouvez le paramétrer pour `always` pour une sauvegarde complète et permanente ou personnalisez-la en fonction de vos besoins. Par exemple, si vous choisissez une granularité quotidienne, vous

pouvez spécifier les jours de la semaine où la sauvegarde complète doit avoir lieu.

Exemple de YAML pour la planification de sauvegarde et de snapshot :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup-rule: "Monday,Thursday"
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: Daily
  hour: "0"
  minute: "0"
```

Exemple de YAML pour la planification des instantanés uniquement :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-snapshot-schedule
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "0"
  snapshotRetention: "15"
  granularity: Daily
  hour: "2"
  minute: "0"
```

3. Après avoir rempli le `trident-protect-schedule-cr.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

Créez une planification à l'aide de l'interface de ligne de commande (CLI).

Étapes

1. Créez le calendrier de protection en remplaçant les valeurs entre parenthèses par les informations provenant de votre environnement. Par exemple:



Vous pouvez utiliser `tridentctl-protect create schedule --help` pour afficher des informations d'aide détaillées sur cette commande.

```
tridentctl-protect create schedule <my_schedule_name> --appvault
<my_appvault_name> --app <name_of_app_to_snapshot> --backup
--retention <how_many_backups_to_retain> --data-mover
<Kopia_or_Restic> --day-of-month <day_of_month_to_run_schedule>
--day-of-week <day_of_month_to_run_schedule> --granularity
<frequency_to_run> --hour <hour_of_day_to_run> --minute
<minute_of_hour_to_run> --recurrence-rule <recurrence> --snapshot
--retention <how_many_snapshots_to_retain> -n <application_namespace>
--full-backup-rule <string>
```

Vous pouvez paramétrer le `--full-backup-rule` drapeau à `always` pour une sauvegarde complète et permanente ou personnalisez-la en fonction de vos besoins. Par exemple, si vous choisissez une granularité quotidienne, vous pouvez spécifier les jours de la semaine où la sauvegarde complète doit avoir lieu. Par exemple, utilisez `--full-backup-rule "Monday, Thursday"` programmer une sauvegarde complète les lundis et jeudis.

Pour les planifications ne prenant en compte que les instantanés, définissez `--backup-retention 0` et spécifiez une valeur supérieure à 0 pour `--snapshot-retention`.

Supprimer un instantané

Supprimez les instantanés planifiés ou à la demande dont vous n'avez plus besoin.

Étapes

1. Supprimez la ressource personnalisée (CR) associée à l'instantané :

```
kubectl delete snapshot <snapshot_name> -n my-app-namespace
```

Supprimer une sauvegarde

Supprimez les sauvegardes planifiées ou à la demande dont vous n'avez plus besoin.



Assurez-vous que la politique de réclamation est configurée pour `Delete` supprimer toutes les données de sauvegarde du stockage d'objets. Le paramètre par défaut de la politique est `Retain` pour éviter toute perte accidentelle de données. Si la politique n'est pas modifiée à `Delete` Les données de sauvegarde resteront stockées dans l'objet et devront être supprimées manuellement.

Étapes

1. Supprimez la CR de sauvegarde associée à la sauvegarde :

```
kubectl delete backup <backup_name> -n my-app-namespace
```

Vérifier l'état d'une opération de sauvegarde

Vous pouvez utiliser la ligne de commande pour vérifier l'état d'une opération de sauvegarde : en cours, terminée ou ayant échoué.

Étapes

1. Utilisez la commande suivante pour récupérer l'état de l'opération de sauvegarde, en remplaçant les valeurs entre parenthèses par les informations provenant de votre environnement :

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath  
='{.status}'
```

Activer la sauvegarde et la restauration des opérations azure-netapp-files (ANF)

Si vous avez installé Trident Protect, vous pouvez activer une fonctionnalité de sauvegarde et de restauration économe en espace pour les backends de stockage qui utilisent la classe de stockage azure-netapp-files et qui ont été créés avant Trident 24.06. Cette fonctionnalité est compatible avec les volumes NFSv4 et ne consomme pas d'espace supplémentaire du pool de capacité.

Avant de commencer

Assurez-vous de ce qui suit :

- Vous avez installé Trident Protect.
- Vous avez défini une application dans Trident Protect. Cette application offrira des fonctionnalités de protection limitées jusqu'à ce que vous ayez terminé cette procédure.
- Tu as `azure-netapp-files` sélectionné comme classe de stockage par défaut pour votre système de stockage dorsal.

Développez pour afficher les étapes de configuration

1. Procédez comme suit dans Trident si le volume ANF a été créé avant la mise à niveau vers Trident 24.10 :

a. Activez le répertoire de snapshots pour chaque PV basé sur azure-netapp-files et associé à l'application :

```
tridentctl update volume <pv name> --snapshot-dir=true -n trident
```

b. Vérifiez que le répertoire des instantanés a été activé pour chaque PV associé :

```
tridentctl get volume <pv name> -n trident -o yaml | grep  
snapshotDir
```

Réponse:

```
snapshotDirectory: "true"
```

+

Lorsque le répertoire de snapshots n'est pas activé, le système choisit la fonctionnalité de sauvegarde standard, qui consomme temporairement de l'espace dans le pool de capacité pendant le processus de sauvegarde. Dans ce cas, assurez-vous de disposer d'un espace suffisant dans le pool de capacité pour créer un volume temporaire de la taille du volume sauvegardé.

Résultat

L'application est prête pour la sauvegarde et la restauration à l'aide de Trident Protect. Chaque PVC peut également être utilisé par d'autres applications pour les sauvegardes et les restaurations.

Restaurer les applications

Restaurez les applications à l'aide de Trident Protect

Vous pouvez utiliser Trident Protect pour restaurer votre application à partir d'un instantané ou d'une sauvegarde. La restauration à partir d'un instantané existant sera plus rapide lors de la restauration de l'application sur le même cluster.



- Lorsque vous restaurez une application, tous les points d'exécution configurés pour celle-ci sont restaurés avec elle. Si un point d'entrée d'exécution post-restauration est présent, il s'exécute automatiquement dans le cadre de l'opération de restauration.
- La restauration à partir d'une sauvegarde vers un espace de noms différent ou vers l'espace de noms d'origine est prise en charge pour les volumes qtree. Cependant, la restauration à partir d'un instantané vers un espace de noms différent ou vers l'espace de noms d'origine n'est pas prise en charge pour les volumes qtree.
- Vous pouvez utiliser des paramètres avancés pour personnaliser les opérations de restauration. Pour en savoir plus, consultez "[Utilisez les paramètres de restauration avancés de Trident Protect](#)".

Restaurer à partir d'une sauvegarde vers un espace de noms différent

Lorsque vous restaurez une sauvegarde dans un espace de noms différent à l'aide d'une ressource personnalisée BackupRestore, Trident Protect restaure l'application dans un nouvel espace de noms et crée une ressource personnalisée d'application pour l'application restaurée. Pour protéger l'application restaurée, créez des sauvegardes ou des instantanés à la demande, ou établissez un calendrier de protection.



La restauration d'une sauvegarde dans un espace de noms différent contenant des ressources existantes ne modifiera pas les ressources qui partagent les mêmes noms que celles de la sauvegarde. Pour restaurer toutes les ressources de la sauvegarde, supprimez et recréez l'espace de noms cible, ou restaurez la sauvegarde dans un nouvel espace de noms.

Avant de commencer

Assurez-vous que la durée d'expiration du jeton de session AWS est suffisante pour toute opération de restauration S3 de longue durée. Si le jeton expire pendant l'opération de restauration, celle-ci peut échouer.

- Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
- Se référer à "[Documentation AWS IAM](#)" pour plus d'informations sur les identifiants avec les ressources AWS.



Lorsque vous restaurez des sauvegardes à l'aide de Kopia comme outil de transfert de données, vous pouvez éventuellement spécifier des annotations dans le CR ou utiliser l'interface de ligne de commande pour contrôler le comportement du stockage temporaire utilisé par Kopia. Se référer à "[Documentation Kopia](#)" pour plus d'informations sur les options que vous pouvez configurer. Utilisez le `tridentctl-protect create --help` commande pour plus d'informations sur la spécification des annotations avec l'interface de ligne de commande Trident Protect.

Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-backup-restore-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.appArchivePath** : Chemin d'accès dans AppVault où sont stockés les contenus de sauvegarde. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
```

- **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où sont stockés les contenus de sauvegarde.
- **spec.namespaceMapping** : Le mappage de l'espace de noms source de l'opération de restauration vers l'espace de noms de destination. Remplacer `my-source-namespace` et `my-destination-namespace` avec des informations provenant de votre environnement.

```
---
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appArchivePath: my-backup-path
  appVaultRef: appvault-name
  namespaceMapping: [{"source": "my-source-namespace",
"destination": "my-destination-namespace"}]
```

3. (*Facultatif*) Si vous devez sélectionner uniquement certaines ressources de l'application à restaurer, ajoutez un filtrage qui inclut ou exclut les ressources marquées par des étiquettes particulières :



Trident Protect sélectionne automatiquement certaines ressources en raison de leur relation avec les ressources que vous sélectionnez. Par exemple, si vous sélectionnez une ressource de revendication de volume persistant et qu'elle possède un pod associé, Trident Protect restaurera également le pod associé.

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `Include` ou `Exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs

à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.

- **resourceMatchers[].group**: (*Optionnel*) Groupe de la ressource à filtrer.
- **resourceMatchers[].kind**: (*Optionnel*) Type de ressource à filtrer.
- **resourceMatchers[].version**: (*Optionnel*) Version de la ressource à filtrer.
- **resourceMatchers[].names**: (*Optionnel*) Noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].namespaces**: (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors** : (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Après avoir rempli le trident-protect-backup-restore-cr.yaml fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utiliser la CLI

Étapes

1. Restaurez la sauvegarde dans un espace de noms différent, en remplaçant les valeurs entre crochets par les informations de votre environnement. Le namespace-mapping L'argument utilise des espaces de noms séparés par deux-points pour associer les espaces de noms source aux espaces de noms de destination corrects au format source1:dest1, source2:dest2 . Par exemple:

```
tridentctl-protect create backuprestore <my_restore_name> \  
--backup <backup_namespace>/<backup_to_restore> \  
--namespace-mapping <source_to_destination_namespace_mapping> \  
-n <application_namespace>
```

Restaurer à partir d'une sauvegarde vers l'espace de noms d'origine

Vous pouvez restaurer une sauvegarde dans l'espace de noms d'origine à tout moment.

Avant de commencer

Assurez-vous que la durée d'expiration du jeton de session AWS est suffisante pour toute opération de restauration S3 de longue durée. Si le jeton expire pendant l'opération de restauration, celle-ci peut échouer.

- Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
- Se référer à "[Documentation AWS IAM](#)" pour plus d'informations sur les identifiants avec les ressources AWS.



Lorsque vous restaurez des sauvegardes à l'aide de Kopia comme outil de transfert de données, vous pouvez éventuellement spécifier des annotations dans le CR ou utiliser l'interface de ligne de commande pour contrôler le comportement du stockage temporaire utilisé par Kopia. Se référer à "[Documentation Kopia](#)" pour plus d'informations sur les options que vous pouvez configurer. Utilisez le `tridentctl-protect create --help` commande pour plus d'informations sur la spécification des annotations avec l'interface de ligne de commande Trident Protect.

Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-backup-ipr-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :

- **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
- **spec.appArchivePath** : Chemin d'accès dans AppVault où sont stockés les contenus de sauvegarde. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où sont stockés les contenus de sauvegarde.

Par exemple:

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: BackupInplaceRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appArchivePath: my-backup-path  
  appVaultRef: appvault-name
```

3. (*Facultatif*) Si vous devez sélectionner uniquement certaines ressources de l'application à restaurer, ajoutez un filtrage qui inclut ou exclut les ressources marquées par des étiquettes particulières :



Trident Protect sélectionne automatiquement certaines ressources en raison de leur relation avec les ressources que vous sélectionnez. Par exemple, si vous sélectionnez une ressource de revendication de volume persistant et qu'elle possède un pod associé, Trident Protect restaurera également le pod associé.

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `Include` ou `Exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.
 - **resourceMatchers[].group**: (*Optionnel*) Groupe de la ressource à filtrer.
 - **resourceMatchers[].kind**: (*Optionnel*) Type de ressource à filtrer.

- **resourceMatchers[].version:** (*Optionnel*) Version de la ressource à filtrer.
- **resourceMatchers[].names:** (*Optionnel*) Noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].namespaces:** (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors :** (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Après avoir rempli le trident-protect-backup-ipr-cr.yaml fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

Utiliser la CLI

Étapes

1. Restaurez la sauvegarde dans l'espace de noms d'origine, en remplaçant les valeurs entre crochets par les informations de votre environnement. Le backup L'argument utilise un espace de noms et un nom de sauvegarde au format <namespace>/<name> . Par exemple:

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

Restaurer à partir d'une sauvegarde vers un cluster différent

Vous pouvez restaurer une sauvegarde sur un autre cluster en cas de problème avec le cluster d'origine.



Lorsque vous restaurez des sauvegardes à l'aide de Kopia comme outil de transfert de données, vous pouvez éventuellement spécifier des annotations dans le CR ou utiliser l'interface de ligne de commande pour contrôler le comportement du stockage temporaire utilisé par Kopia. Se référer à "[Documentation Kopia](#)" pour plus d'informations sur les options que vous pouvez configurer. Utilisez le `tridentctl-protect create --help` commande pour plus d'informations sur la spécification des annotations avec l'interface de ligne de commande Trident Protect.

Avant de commencer

Assurez-vous que les conditions préalables suivantes sont remplies :

- Le cluster de destination possède Trident Protect installé.
- Le cluster de destination a accès au chemin du compartiment du même AppVault que le cluster source, où la sauvegarde est stockée.
- Assurez-vous que votre environnement local peut se connecter au compartiment de stockage d'objets défini dans la ressource personnalisée AppVault lors de l'exécution. `tridentctl-protect get appvaultcontent` commande. Si des restrictions réseau empêchent l'accès, exécutez plutôt l'interface de ligne de commande Trident Protect depuis un pod sur le cluster de destination.
- Assurez-vous que la durée de validité du jeton de session AWS est suffisante pour toute opération de restauration de longue durée. Si le jeton expire pendant l'opération de restauration, celle-ci peut échouer.
 - Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
 - Se référer à "[Documentation AWS](#)" pour plus d'informations sur les identifiants avec les ressources AWS.

Étapes

1. Vérifiez la disponibilité de la ressource personnalisée AppVault sur le cluster de destination à l'aide du plugin CLI Trident Protect :

```
tridentctl-protect get appvault --context <destination_cluster_name>
```



Assurez-vous que l'espace de noms destiné à la restauration de l'application existe sur le cluster de destination.

2. Consultez le contenu de la sauvegarde de l'AppVault disponible sur le cluster de destination :

```
tridentctl-protect get appvaultcontent <appvault_name> \  
--show-resources backup \  
--show-paths \  
--context <destination_cluster_name>
```

L'exécution de cette commande affiche les sauvegardes disponibles dans AppVault, y compris leurs clusters d'origine, les noms des applications correspondantes, les horodatages et les chemins d'accès aux

archives.

Exemple de résultat :

```
+-----+-----+-----+-----+
+-----+-----+
|  CLUSTER  |  APP   |  TYPE  |  NAME          |  TIMESTAMP
|  PATH     |        |        |                |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| production1 | wordpress | backup | wordpress-bkup-1 | 2024-10-30
08:37:40 (UTC) | backuppath1 |
| production1 | wordpress | backup | wordpress-bkup-2 | 2024-10-30
08:37:40 (UTC) | backuppath2 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

3. Restaurez l'application sur le cluster de destination en utilisant le nom AppVault et le chemin d'accès à l'archive :

Utilisez un CR

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-backup-restore-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.appVaultRef** : (*Obligatoire*) Le nom de l'AppVault où sont stockés les contenus de sauvegarde.
 - **spec.appArchivePath** : Chemin d'accès dans AppVault où sont stockés les contenus de sauvegarde. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
```



Si BackupRestore CR n'est pas disponible, vous pouvez utiliser la commande mentionnée à l'étape 2 pour afficher le contenu de la sauvegarde.

- **spec.namespaceMapping** : Le mappage de l'espace de noms source de l'opération de restauration vers l'espace de noms de destination. Remplacer `my-source-namespace` et `my-destination-namespace` avec des informations provenant de votre environnement.

Par exemple:

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  namespaceMapping: [{"source": "my-source-namespace", "
  destination": "my-destination-namespace"}]
```

3. Après avoir rempli le `trident-protect-backup-restore-cr.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utiliser la CLI

1. Utilisez la commande suivante pour restaurer l'application, en remplaçant les valeurs entre crochets par les informations provenant de votre environnement. L'argument `namespace-mapping` utilise des

espaces de noms séparés par deux-points pour mapper les espaces de noms sources aux espaces de noms de destination corrects au format `source1:dest1,source2:dest2`. Par exemple:

```
tridentctl-protect create backuprestore <restore_name> \  
--namespace-mapping <source_to_destination_namespace_mapping> \  
--appvault <appvault_name> \  
--path <backup_path> \  
--context <destination_cluster_name> \  
-n <application_namespace>
```

Restaurer à partir d'un instantané vers un espace de noms différent

Vous pouvez restaurer des données à partir d'un instantané à l'aide d'un fichier de ressources personnalisé (CR), soit dans un espace de noms différent, soit dans l'espace de noms source d'origine. Lorsque vous restaurez un instantané dans un espace de noms différent à l'aide d'une ressource personnalisée `SnapshotRestore`, Trident Protect restaure l'application dans un nouvel espace de noms et crée une ressource personnalisée d'application pour l'application restaurée. Pour protéger l'application restaurée, créez des sauvegardes ou des instantanés à la demande, ou établissez un calendrier de protection.



`SnapshotRestore` prend en charge `spec.storageClassMapping` cet attribut est valable uniquement lorsque les classes de stockage source et de destination utilisent le même système de stockage dorsal. Si vous tentez de restaurer un `StorageClass` Si le système de stockage utilisé est différent, l'opération de restauration échouera.

Avant de commencer

Assurez-vous que la durée d'expiration du jeton de session AWS est suffisante pour toute opération de restauration S3 de longue durée. Si le jeton expire pendant l'opération de restauration, celle-ci peut échouer.

- Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
- Se référer à "[Documentation AWS IAM](#)" pour plus d'informations sur les identifiants avec les ressources AWS.

Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-snapshot-restore-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.appVaultRef** : (*Obligatoire*) Le nom de l'AppVault où le contenu de l'instantané est stocké.
 - **spec.appArchivePath** : Le chemin d'accès dans AppVault où le contenu des instantanés est stocké. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **spec.namespaceMapping** : Le mappage de l'espace de noms source de l'opération de restauration vers l'espace de noms de destination. Remplacer `my-source-namespace` et `my-destination-namespace` avec des informations provenant de votre environnement.

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: SnapshotRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appVaultRef: appvault-name  
  appArchivePath: my-snapshot-path  
  namespaceMapping: [{"source": "my-source-namespace",  
"destination": "my-destination-namespace"}]
```

3. (*Facultatif*) Si vous devez sélectionner uniquement certaines ressources de l'application à restaurer, ajoutez un filtrage qui inclut ou exclut les ressources marquées par des étiquettes particulières :



Trident Protect sélectionne automatiquement certaines ressources en raison de leur relation avec les ressources que vous sélectionnez. Par exemple, si vous sélectionnez une ressource de revendication de volume persistant et qu'elle possède un pod associé, Trident Protect restaurera également le pod associé.

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `Include` ou `Exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.

- **resourceMatchers[].group**: (*Optionnel*) Groupe de la ressource à filtrer.
- **resourceMatchers[].kind**: (*Optionnel*) Type de ressource à filtrer.
- **resourceMatchers[].version**: (*Optionnel*) Version de la ressource à filtrer.
- **resourceMatchers[].names**: (*Optionnel*) Noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].namespaces**: (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors** : (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Après avoir rempli le trident-protect-snapshot-restore-cr.yaml fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Utiliser la CLI

Étapes

1. Restaurez l'instantané dans un espace de noms différent, en remplaçant les valeurs entre crochets par les informations de votre environnement.
 - Le snapshot L'argument utilise un espace de noms et un nom d'instantané au format <namespace>/<name> .
 - Le namespace-mapping L'argument utilise des espaces de noms séparés par deux-points pour associer les espaces de noms source aux espaces de noms de destination corrects au format

```
source1:dest1,source2:dest2 .
```

Par exemple:

```
tridentctl-protect create snapshotrestore <my_restore_name> \  
--snapshot <namespace/snapshot_to_restore> \  
--namespace-mapping <source_to_destination_namespace_mapping> \  
-n <application_namespace>
```

Restaurer à partir d'un instantané vers l'espace de noms d'origine

Vous pouvez restaurer un instantané dans l'espace de noms d'origine à tout moment.



Si votre application utilise plusieurs espaces de noms et que ces espaces de noms contiennent des PVC portant le même nom, les opérations de restauration d'instantané (sur place et vers un nouvel espace de noms) ne fonctionneront pas correctement. Tous les volumes restaurés auront les mêmes données au lieu des données correctes pour chaque espace de noms. Utilisez la restauration de sauvegarde plutôt que la restauration d'instantané, ou effectuez une mise à niveau vers la version 26.02 ou ultérieure qui corrige ce problème.

Avant de commencer

Assurez-vous que la durée d'expiration du jeton de session AWS est suffisante pour toute opération de restauration S3 de longue durée. Si le jeton expire pendant l'opération de restauration, celle-ci peut échouer.

- Se référer à "[Documentation AWS API](#)" pour plus d'informations sur la vérification de l'expiration du jeton de session actuel.
- Se référer à "[Documentation AWS IAM](#)" pour plus d'informations sur les identifiants avec les ressources AWS.

Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-snapshot-ipr-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.appVaultRef** : (*Obligatoire*) Le nom de l'AppVault où le contenu de l'instantané est stocké.
 - **spec.appArchivePath** : Le chemin d'accès dans AppVault où le contenu des instantanés est stocké. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: SnapshotInplaceRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appVaultRef: appvault-name  
  appArchivePath: my-snapshot-path
```

3. (*Facultatif*) Si vous devez sélectionner uniquement certaines ressources de l'application à restaurer, ajoutez un filtrage qui inclut ou exclut les ressources marquées par des étiquettes particulières :



Trident Protect sélectionne automatiquement certaines ressources en raison de leur relation avec les ressources que vous sélectionnez. Par exemple, si vous sélectionnez une ressource de revendication de volume persistant et qu'elle possède un pod associé, Trident Protect restaurera également le pod associé.

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `Include` ou `Exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.
 - **resourceMatchers[].group** : (*Optionnel*) Groupe de la ressource à filtrer.
 - **resourceMatchers[].kind** : (*Optionnel*) Type de ressource à filtrer.
 - **resourceMatchers[].version** : (*Optionnel*) Version de la ressource à filtrer.
 - **resourceMatchers[].names** : (*Optionnel*) Noms dans le champ `metadata.name` de Kubernetes de la ressource à filtrer.

- **resourceMatchers[].namespaces**: (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors** : (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Après avoir rempli le trident-protect-snapshot-ipr-cr.yaml fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

Utiliser la CLI

Étapes

1. Restaurez l'instantané dans l'espace de noms d'origine, en remplaçant les valeurs entre crochets par les informations de votre environnement. Par exemple:

```
tridentctl-protect create snapshotinplacerestore <my_restore_name> \
--snapshot <snapshot_to_restore> \
-n <application_namespace>
```

Vérifier l'état d'une opération de restauration

Vous pouvez utiliser la ligne de commande pour vérifier l'état d'une opération de restauration : en cours, terminée ou ayant échoué.

Étapes

1. Utilisez la commande suivante pour récupérer l'état de l'opération de restauration, en remplaçant les valeurs entre parenthèses par les informations provenant de votre environnement :

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o  
jsonpath='{.status}'
```

Utilisez les paramètres de restauration avancés de Trident Protect

Vous pouvez personnaliser les opérations de restauration à l'aide de paramètres avancés tels que les annotations, les paramètres d'espace de noms et les options de stockage pour répondre à vos besoins spécifiques.

Annotations et étiquettes d'espace de noms lors des opérations de restauration et de basculement

Lors des opérations de restauration et de basculement, les étiquettes et annotations de l'espace de noms de destination sont mises en correspondance avec celles de l'espace de noms source. Les étiquettes ou annotations de l'espace de noms source qui n'existent pas dans l'espace de noms de destination sont ajoutées, et toutes les étiquettes ou annotations existantes sont écrasées pour correspondre à la valeur de l'espace de noms source. Les étiquettes ou annotations qui existent uniquement dans l'espace de noms de destination restent inchangées.



Si vous utilisez Red Hat OpenShift, il est important de noter le rôle essentiel des annotations d'espace de noms dans les environnements OpenShift. Les annotations d'espace de noms garantissent que les pods restaurés adhèrent aux autorisations et aux configurations de sécurité appropriées définies par les contraintes de contexte de sécurité (SCC) OpenShift et peuvent accéder aux volumes sans problèmes d'autorisation. Pour plus d'informations, veuillez consulter le ["Documentation sur les contraintes de contexte de sécurité d'OpenShift"](#).

Vous pouvez empêcher l'écrasement de certaines annotations dans l'espace de noms de destination en définissant la variable d'environnement Kubernetes. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` avant d'effectuer l'opération de restauration ou de basculement. Par exemple:

```
helm upgrade trident-protect --set  
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key  
_to_skip_2> --reuse-values
```



Lors d'une opération de restauration ou de basculement, toutes les annotations et étiquettes d'espace de noms spécifiées dans `restoreSkipNamespaceAnnotations` et `restoreSkipNamespaceLabels` sont exclues de l'opération de restauration ou de basculement. Assurez-vous que ces paramètres sont configurés lors de l'installation initiale de Helm. Pour en savoir plus, consultez ["Configurer les options de filtrage AutoSupport et d'espace de noms"](#).

Si vous avez installé l'application source à l'aide de Helm avec le `--create-namespace` Le drapeau, un traitement spécial est accordé au `name` Légende. Lors du processus de restauration ou de basculement, Trident Protect copie cette étiquette dans l'espace de noms de destination, mais met à jour la valeur avec la valeur de l'espace de noms de destination si la valeur de la source correspond à l'espace de noms source. Si

cette valeur ne correspond pas à l'espace de noms source, elle est copiée dans l'espace de noms de destination sans modification.

Exemple

L'exemple suivant présente un espace de noms source et un espace de noms de destination, chacun avec des annotations et des étiquettes différentes. Vous pouvez voir l'état de l'espace de noms de destination avant et après l'opération, et comment les annotations et les étiquettes sont combinées ou écrasées dans l'espace de noms de destination.

Avant l'opération de restauration ou de basculement

Le tableau suivant illustre l'état des espaces de noms source et de destination de l'exemple avant l'opération de restauration ou de basculement :

Espace de noms	Annotations	Étiquettes
Espace de noms ns-1 (source)	<ul style="list-style-type: none">• annotation.one/key: "updatedvalue"• annotation.two/key: "true"	<ul style="list-style-type: none">• environnement=production• conformité=hippaa• nom=ns-1
Espace de noms ns-2 (destination)	<ul style="list-style-type: none">• annotation.one/key: "true"• annotation.three/key: "false"	<ul style="list-style-type: none">• rôle=base de données

Après l'opération de restauration

Le tableau suivant illustre l'état de l'espace de noms de destination d'exemple après l'opération de restauration ou de basculement. Des touches ont été ajoutées, d'autres ont été écrasées, et le `name` L'étiquette a été mise à jour pour correspondre à l'espace de noms de destination :

Espace de noms	Annotations	Étiquettes
Espace de noms ns-2 (destination)	<ul style="list-style-type: none">• annotation.one/key: "updatedvalue"• annotation.two/key: "true"• annotation.three/key: "false"	<ul style="list-style-type: none">• nom=ns-2• conformité=hippaa• environnement=production• rôle=base de données

Champs pris en charge

Cette section décrit les champs supplémentaires disponibles pour les opérations de restauration.

Mappage des classes de stockage

Le `spec.storageClassMapping` L'attribut définit un mappage entre une classe de stockage présente dans l'application source et une nouvelle classe de stockage sur le cluster cible. Vous pouvez l'utiliser lors de la migration d'applications entre des clusters avec différentes classes de stockage ou lors du changement du backend de stockage pour les opérations BackupRestore.

Exemple:

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

Annotations prises en charge

Cette section répertorie les annotations prises en charge pour configurer différents comportements du système. Si aucune annotation n'est explicitement définie par l'utilisateur, le système utilisera la valeur par défaut.

Annotation	Type	Description	Valeur par défaut
protect.trident.netapp.io/data-mover-timeout-sec	chaîne	Le temps maximal (en secondes) pendant lequel l'opération de déplacement de données peut être bloquée.	"300"
protect.trident.netapp.io/kopia-content-cache-size-limit-mb	chaîne	La limite de taille maximale (en mégaoctets) pour le cache de contenu Kopia.	"1000"

Répliquez les applications à l'aide de NetApp SnapMirror et Trident Protect.

Avec Trident Protect, vous pouvez utiliser les capacités de réplication asynchrone de la technologie NetApp SnapMirror pour répliquer les données et les modifications d'applications d'un système de stockage à un autre, sur le même cluster ou entre différents clusters.

Annotations et étiquettes d'espace de noms lors des opérations de restauration et de basculement

Lors des opérations de restauration et de basculement, les étiquettes et annotations de l'espace de noms de destination sont mises en correspondance avec celles de l'espace de noms source. Les étiquettes ou annotations de l'espace de noms source qui n'existent pas dans l'espace de noms de destination sont ajoutées, et toutes les étiquettes ou annotations existantes sont écrasées pour correspondre à la valeur de l'espace de noms source. Les étiquettes ou annotations qui existent uniquement dans l'espace de noms de destination restent inchangées.



Si vous utilisez Red Hat OpenShift, il est important de noter le rôle essentiel des annotations d'espace de noms dans les environnements OpenShift. Les annotations d'espace de noms garantissent que les pods restaurés adhèrent aux autorisations et aux configurations de sécurité appropriées définies par les contraintes de contexte de sécurité (SCC) OpenShift et peuvent accéder aux volumes sans problèmes d'autorisation. Pour plus d'informations, veuillez consulter le ["Documentation sur les contraintes de contexte de sécurité d'OpenShift"](#).

Vous pouvez empêcher l'écrasement de certaines annotations dans l'espace de noms de destination en définissant la variable d'environnement Kubernetes. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` avant

d'effectuer l'opération de restauration ou de basculement. Par exemple:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```



Lors d'une opération de restauration ou de basculement, toutes les annotations et étiquettes d'espace de noms spécifiées dans `restoreSkipNamespaceAnnotations` et `restoreSkipNamespaceLabels` sont exclues de l'opération de restauration ou de basculement. Assurez-vous que ces paramètres sont configurés lors de l'installation initiale de Helm. Pour en savoir plus, consultez ["Configurer les options de filtrage AutoSupport et d'espace de noms"](#).

Si vous avez installé l'application source à l'aide de Helm avec le `--create-namespace` Le drapeau, un traitement spécial est accordé au `name` Légende. Lors du processus de restauration ou de basculement, Trident Protect copie cette étiquette dans l'espace de noms de destination, mais met à jour la valeur avec la valeur de l'espace de noms de destination si la valeur de la source correspond à l'espace de noms source. Si cette valeur ne correspond pas à l'espace de noms source, elle est copiée dans l'espace de noms de destination sans modification.

Exemple

L'exemple suivant présente un espace de noms source et un espace de noms de destination, chacun avec des annotations et des étiquettes différentes. Vous pouvez voir l'état de l'espace de noms de destination avant et après l'opération, et comment les annotations et les étiquettes sont combinées ou écrasées dans l'espace de noms de destination.

Avant l'opération de restauration ou de basculement

Le tableau suivant illustre l'état des espaces de noms source et de destination de l'exemple avant l'opération de restauration ou de basculement :

Espace de noms	Annotations	Étiquettes
Espace de noms ns-1 (source)	<ul style="list-style-type: none">• <code>annotation.one/key: "updatedvalue"</code>• <code>annotation.two/key: "true"</code>	<ul style="list-style-type: none">• <code>environnement=production</code>• <code>conformité=hippaa</code>• <code>nom=ns-1</code>
Espace de noms ns-2 (destination)	<ul style="list-style-type: none">• <code>annotation.one/key: "true"</code>• <code>annotation.three/key: "false"</code>	<ul style="list-style-type: none">• <code>rôle=base de données</code>

Après l'opération de restauration

Le tableau suivant illustre l'état de l'espace de noms de destination d'exemple après l'opération de restauration ou de basculement. Des touches ont été ajoutées, d'autres ont été écrasées, et le `name` L'étiquette a été mise à jour pour correspondre à l'espace de noms de destination :

Espace de noms	Annotations	Étiquettes
Espace de noms ns-2 (destination)	<ul style="list-style-type: none"> • annotation.one/key: "updatedvalue" • annotation.two/key: "true" • annotation.three/key: "false" 	<ul style="list-style-type: none"> • nom=ns-2 • conformité=hippaa • environnement=production • rôle=base de données



Vous pouvez configurer Trident Protect pour geler et dégeler les systèmes de fichiers pendant les opérations de protection des données. ["Apprenez-en davantage sur la configuration du gel du système de fichiers avec Trident Protect."](#)

Points d'exécution lors des opérations de basculement et d'inversion

Lorsque vous utilisez la relation AppMirror pour protéger votre application, il existe des comportements spécifiques liés aux points d'exécution dont vous devez tenir compte lors des opérations de basculement et de restauration.

- Lors d'un basculement, les points d'entrée d'exécution sont automatiquement copiés du cluster source vers le cluster de destination. Vous n'avez pas besoin de les recréer manuellement. Après le basculement, des points d'entrée d'exécution sont présents dans l'application et s'exécuteront lors de toute action pertinente.
- Lors d'une opération inverse ou d'une resynchronisation inverse, tous les points d'entrée d'exécution existants sur l'application sont supprimés. Lorsque l'application source devient l'application de destination, ces points d'ancrage d'exécution ne sont plus valides et sont supprimés pour empêcher leur exécution.

Pour en savoir plus sur les hooks d'exécution, consultez ["Gérer les hooks d'exécution de Trident Protect"](#).

Établir une relation de réplication

La mise en place d'une relation de réplication implique les éléments suivants :

- Choisir la fréquence à laquelle Trident Protect doit prendre un instantané de l'application (qui inclut les ressources Kubernetes de l'application ainsi que les instantanés de volume pour chacun des volumes de l'application).
- Choix du calendrier de réplication (incluant les ressources Kubernetes ainsi que les données de volume persistant)
- Définir l'heure de la prise de vue

Étapes

1. Sur le cluster source, créez un AppVault pour l'application source. Selon votre fournisseur de stockage, modifiez un exemple dans ["Ressources personnalisées AppVault"](#) pour s'adapter à votre environnement :

Créez un AppVault à l'aide d'une demande de changement.

- a. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-appvault-primary-source.yaml`).
- b. Configurez les attributs suivants :
 - **metadata.name:** (*Obligatoire*) Le nom de la ressource personnalisée AppVault. Notez bien le nom que vous choisissez, car d'autres fichiers CR nécessaires à une relation de réplication font référence à cette valeur.
 - **spec.providerConfig:** (*Obligatoire*) Stocke la configuration nécessaire pour accéder à AppVault à l'aide du fournisseur spécifié. Choisissez un nom de compartiment et toutes autres informations nécessaires pour votre fournisseur. Notez les valeurs que vous choisissez, car d'autres fichiers CR nécessaires à une relation de réplication font référence à ces valeurs. Se référer à "[Ressources personnalisées AppVault](#)" pour des exemples de demandes de changement AppVault avec d'autres fournisseurs.
 - **spec.providerCredentials:** (*Obligatoire*) Stocke les références à toutes les informations d'identification requises pour accéder à AppVault à l'aide du fournisseur spécifié.
 - **spec.providerCredentials.valueFromSecret:** (*Obligatoire*) Indique que la valeur d'identification doit provenir d'un secret.
 - **clé:** (*Obligatoire*) La clé valide du secret à sélectionner.
 - **nom :** (*Obligatoire*) Nom du secret contenant la valeur de ce champ. Doit se trouver dans le même espace de noms.
 - **spec.providerCredentials.secretAccessKey:** (*Obligatoire*) La clé d'accès utilisée pour accéder au fournisseur. Le **nom** doit correspondre à **spec.providerCredentials.valueFromSecret.name**.
 - **spec.providerType:** (*Obligatoire*) Détermine ce qui fournit la sauvegarde ; par exemple, NetApp ONTAP S3, S3 générique, Google Cloud ou Microsoft Azure. Valeurs possibles :
 - `aws`
 - `azuré`
 - `GCP`
 - `générique-s3`
 - `ontap-s3`
 - `storagegrid-s3`
- c. Après avoir rempli le `trident-protect-appvault-primary-source.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n trident-protect
```

Créez un AppVault à l'aide de l'interface de ligne de commande (CLI).

- a. Créez l'AppVault en remplaçant les valeurs entre crochets par les informations provenant de votre environnement :

```
tridentctl-protect create vault Azure <vault-name> --account  
<account-name> --bucket <bucket-name> --secret <secret-name> -n  
trident-protect
```

2. Sur le cluster source, créez l'application source CR :

Créez l'application source à l'aide d'une CR

a. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-app-source.yaml`).

b. Configurez les attributs suivants :

- **metadata.name:** (*Obligatoire*) Le nom de la ressource personnalisée de l'application. Notez bien le nom que vous choisissez, car d'autres fichiers CR nécessaires à une relation de réplication font référence à cette valeur.
- **spec.includedNamespaces:** (*Obligatoire*) Un tableau d'espaces de noms et d'étiquettes associées. Utilisez les noms d'espaces de noms et, éventuellement, restreignez la portée des espaces de noms à l'aide d'étiquettes pour spécifier les ressources qui existent dans les espaces de noms listés ici. L'espace de noms de l'application doit faire partie de ce tableau.

Exemple de fichier YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
      labelSelector: {}
```

c. Après avoir rempli le `trident-protect-app-source.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-namespace
```

Créez l'application source à l'aide de l'interface de ligne de commande (CLI).

a. Créez l'application source. Par exemple:

```
tridentctl-protect create app <my-app-name> --namespaces
<namespaces-to-be-included> -n <my-app-namespace>
```

3. Vous pouvez également, si vous le souhaitez, prendre un instantané de l'application source sur le cluster source. Cet instantané sert de base à l'application sur le cluster de destination. Si vous ignorez cette étape, vous devrez attendre la prochaine capture d'écran planifiée pour obtenir une version récente. Pour créer un instantané à la demande, reportez-vous à ["Créer un instantané à la demande"](#).

4. Sur le cluster source, créez la ressource personnalisée (CR) de planification de réplication :

En plus du calendrier fourni ci-dessous, il est recommandé de créer un calendrier de capture instantanée quotidienne distinct avec une période de conservation de 7 jours afin de maintenir une capture instantanée commune entre les clusters ONTAP appariés. Cela garantit la disponibilité des instantanés pendant une durée maximale de 7 jours, mais cette période de conservation peut être personnalisée en fonction des besoins de l'utilisateur.



En cas de basculement, le système peut utiliser ces instantanés pendant une durée maximale de 7 jours pour les opérations inverses. Cette approche rend le processus inverse plus rapide et plus efficace car seules les modifications apportées depuis le dernier instantané seront transférées, et non la totalité des données.

Si un calendrier existant pour la demande répond déjà aux exigences de conservation souhaitées, aucun calendrier supplémentaire n'est requis.

Créez la planification de réplication à l'aide d'une ressource personnalisée (CR).

a. Créez une planification de réplication pour l'application source :

- i. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-schedule.yaml`).
- ii. Configurez les attributs suivants :
 - **metadata.name**: (*Obligatoire*) Le nom de la ressource personnalisée de planification.
 - **spec.appVaultRef** : (*Obligatoire*) Cette valeur doit correspondre au champ `metadata.name` de l'AppVault pour l'application source.
 - **spec.applicationRef**: (*Obligatoire*) Cette valeur doit correspondre au champ `metadata.name` de la ressource personnalisée (CR) de l'application source.
 - **spec.backupRetention**: (*Obligatoire*) Ce champ est obligatoire et sa valeur doit être définie sur 0.
 - **spec.enabled** : Doit être défini sur `true`.
 - **specgranularity** : Doit être défini sur `Custom`.
 - **spec.recurrenceRule** : Définissez une date de début en temps UTC et un intervalle de récurrence.
 - **spec.snapshotRetention** : Doit être défini sur 2.

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  name: appmirror-schedule
  namespace: my-app-namespace
spec:
  appVaultRef: my-appvault-name
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: Custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

- i. Après avoir rempli le `trident-protect-schedule.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-schedule.yaml -n my-app-namespace
```

Créez la planification de réplication à l'aide de l'interface de ligne de commande (CLI).

- a. Créez la planification de réplication en remplaçant les valeurs entre crochets par les informations provenant de votre environnement :

```
tridentctl-protect create schedule --name appmirror-schedule --app <my_app_name> --appvault <my_app_vault> --granularity Custom --recurrence-rule <rule> --snapshot-retention <snapshot_retention_count> -n <my_app_namespace>
```

Exemple:

```
tridentctl-protect create schedule --name appmirror-schedule --app <my_app_name> --appvault <my_app_vault> --granularity Custom --recurrence-rule "DTSTART:20220101T000200Z\nRRULE:FREQ=MINUTELY;INTERVAL=5" --snapshot-retention 2 -n <my_app_namespace>
```

5. Sur le cluster de destination, créez une demande de changement (CR) AppVault d'application source identique à celle que vous avez appliquée sur le cluster source et nommez-la (par exemple, `trident-protect-appvault-primary-destination.yaml`).
6. Appliquer le CR :

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n trident-protect
```

7. Créez une ressource personnalisée AppVault de destination pour l'application de destination sur le cluster de destination. Selon votre fournisseur de stockage, modifiez un exemple dans "[Ressources personnalisées AppVault](#)" pour s'adapter à votre environnement :

- a. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-appvault-secondary-destination.yaml`).
- b. Configurez les attributs suivants :
 - **metadata.name:** (*Obligatoire*) Le nom de la ressource personnalisée AppVault. Notez bien le nom que vous choisissez, car d'autres fichiers CR nécessaires à une relation de réplication font référence à cette valeur.
 - **spec.providerConfig:** (*Obligatoire*) Stocke la configuration nécessaire pour accéder à AppVault à l'aide du fournisseur spécifié. Choisissez un `bucketName` et toute autre information nécessaire à votre fournisseur. Notez les valeurs que vous choisissez, car d'autres fichiers CR nécessaires à une relation de réplication font référence à ces valeurs. Se référer à "[Ressources personnalisées](#)"

AppVault" pour des exemples de demandes de changement AppVault avec d'autres fournisseurs.

- **spec.providerCredentials:** (*Obligatoire*) Stocke les références à toutes les informations d'identification requises pour accéder à AppVault à l'aide du fournisseur spécifié.
 - **spec.providerCredentials.valueFromSecret:** (*Obligatoire*) Indique que la valeur d'identification doit provenir d'un secret.
 - **clé:** (*Obligatoire*) La clé valide du secret à sélectionner.
 - **nom :** (*Obligatoire*) Nom du secret contenant la valeur de ce champ. Doit se trouver dans le même espace de noms.
 - **spec.providerCredentials.secretAccessKey:** (*Obligatoire*) La clé d'accès utilisée pour accéder au fournisseur. Le **nom** doit correspondre à **spec.providerCredentials.valueFromSecret.name**.
- **spec.providerType:** (*Obligatoire*) Détermine ce qui fournit la sauvegarde ; par exemple, NetApp ONTAP S3, S3 générique, Google Cloud ou Microsoft Azure. Valeurs possibles :
 - aws
 - azuré
 - GCP
 - générique-s3
 - ontap-s3
 - storagegrid-s3

c. Après avoir rempli le `trident-protect-appvault-secondary-destination.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n trident-protect
```

8. Sur le cluster de destination, créez un fichier CR AppMirrorRelationship :

Créez une relation AppMirror à l'aide d'un CR

- a. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-relationship.yaml`).
- b. Configurez les attributs suivants :
 - **metadata.name:** (Obligatoire) Le nom de la ressource personnalisée AppMirrorRelationship.
 - **spec.destinationAppVaultRef :** (*Obligatoire*) Cette valeur doit correspondre au nom de l'AppVault pour l'application de destination sur le cluster de destination.
 - **spec.namespaceMapping:** (*Obligatoire*) Les espaces de noms de destination et source doivent correspondre à l'espace de noms de l'application défini dans la CR de l'application respective.
 - **spec.sourceAppVaultRef :** (*Obligatoire*) Cette valeur doit correspondre au nom de l'AppVault pour l'application source.
 - **spec.sourceApplicationName :** (*Obligatoire*) Cette valeur doit correspondre au nom de l'application source que vous avez définie dans la ressource personnalisée de l'application source.
 - **spec.sourceApplicationUID :** (Obligatoire) Cette valeur doit correspondre à l'UID de l'application source que vous avez définie dans la CR de l'application source.
 - **spec.storageClassName:** (*Optionnel*) Choisissez le nom d'une classe de stockage valide sur le cluster. La classe de stockage doit être liée à une machine virtuelle de stockage ONTAP qui est appariée avec l'environnement source. Si la classe de stockage n'est pas spécifiée, la classe de stockage par défaut du cluster sera utilisée.
 - **spec.recurrenceRule :** Définissez une date de début en temps UTC et un intervalle de récurrence.

Exemple de YAML :

```

---
apiVersion: protect.trident.netapp.io/v1
kind: AppMirrorRelationship
metadata:
  name: amr-16061e80-1b05-4e80-9d26-d326dc1953d8
  namespace: my-app-namespace
spec:
  desiredState: Established
  destinationAppVaultRef: generic-s3-trident-protect-dst-bucket-
8fe0b902-f369-4317-93d1-ad7f2edc02b5
  namespaceMapping:
    - destination: my-app-namespace
      source: my-app-namespace
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  sourceAppVaultRef: generic-s3-trident-protect-src-bucket-
b643cc50-0429-4ad5-971f-ac4a83621922
  sourceApplicationName: my-app-name
  sourceApplicationUID: 7498d32c-328e-4ddd-9029-122540866aeb
  storageClassName: sc-vsims-2

```

- c. Après avoir rempli le `trident-protect-relationship.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

Créez une relation AppMirror à l'aide de l'interface de ligne de commande (CLI).

- a. Créez et appliquez l'objet AppMirrorRelationship en remplaçant les valeurs entre crochets par les informations provenant de votre environnement :

```

tridentctl-protect create appmirrorrelationship
<name_of_appmirrorrelationship> --destination-app-vault
<my_vault_name> --source-app-vault <my_vault_name> --recurrence
-rule <rule> --namespace-mapping <ns_mapping> --source-app-id
<source_app_UID> --source-app <my_source_app_name> --storage
-class <storage_class_name> -n <application_namespace>

```

Exemple:

```
tridentctl-protect create appmirrorrelationship my-amr
--destination-app-vault appvault2 --source-app-vault appvault1
--recurrence-rule
"DTSTART:20220101T000200Z\nRRULE:FREQ=MINUTELY;INTERVAL=5"
--source-app my-app --namespace-mapping "my-source-ns1:my-dest-
ns1,my-source-ns2:my-dest-ns2" --source-app-id 373f24c1-5769-
404c-93c3-5538af6ccc36 --storage-class my-storage-class -n my-
dest-ns1
```

9. (Facultatif) Sur le cluster de destination, vérifiez l'état et le statut de la relation de réplication :

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Basculement vers le cluster de destination

Avec Trident Protect, vous pouvez basculer les applications répliquées vers un cluster de destination. Cette procédure interrompt la relation de réplication et met l'application en ligne sur le cluster de destination. Trident Protect n'arrête pas l'application sur le cluster source si elle était opérationnelle.

Étapes

1. Sur le cluster de destination, modifiez le fichier CR AppMirrorRelationship (par exemple, `trident-protect-relationship.yaml`) et modifiez la valeur de **spec.desiredState** en `Promoted`.
2. Enregistrez le fichier CR.
3. Appliquez le CR :

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (Facultatif) Créez les plans de protection nécessaires sur l'application basculée.
5. (Facultatif) Vérifiez l'état et le statut de la relation de réplication :

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Resynchroniser une relation de réplication ayant échoué

L'opération de resynchronisation rétablit la relation de réplication. Après une opération de resynchronisation, l'application source d'origine devient l'application en cours d'exécution, et toutes les modifications apportées à l'application en cours d'exécution sur le cluster de destination sont annulées.

Le processus interrompt l'application sur le cluster de destination avant de rétablir la réplication.



Toutes les données écrites dans l'application de destination pendant le basculement seront perdues.

Étapes

1. Facultatif : sur le cluster source, créez un instantané de l'application source. Cela permet de garantir que les dernières modifications provenant du cluster source sont prises en compte.
2. Sur le cluster de destination, modifiez le fichier CR AppMirrorRelationship (par exemple, `trident-protect-relationship.yaml`) et modifiez la valeur de `spec.desiredState` en `Established`.
3. Enregistrez le fichier CR.
4. Appliquer le CR :

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Si vous avez créé des plans de protection sur le cluster de destination pour protéger l'application basculée, supprimez-les. Toute planification restante entraîne des échecs de snapshots de volume.

Resynchronisation inverse d'une relation de réplication ayant échoué

Lors d'une resynchronisation inverse d'une relation de réplication ayant basculé, l'application de destination devient l'application source et la source devient la destination. Les modifications apportées à l'application de destination pendant le basculement sont conservées.

Étapes

1. Sur le cluster de destination d'origine, supprimez la ressource personnalisée AppMirrorRelationship. Cela a pour conséquence que la destination devienne la source. S'il reste des plans de protection sur le nouveau cluster de destination, supprimez-les.
2. Établissez une relation de réplication en appliquant les fichiers CR que vous avez initialement utilisés pour établir la relation aux clusters opposés.
3. Assurez-vous que la nouvelle destination (cluster source d'origine) est configurée avec les deux CR AppVault.
4. Configurez une relation de réplication sur le cluster opposé, en configurant les valeurs pour la direction inverse.

sens de réplication de l'application inverse

Lorsque vous inversez le sens de la réplication, Trident Protect déplace l'application vers le système de stockage de destination tout en continuant à répliquer vers le système de stockage source d'origine. Trident Protect arrête l'application source et réplique les données vers la destination avant de basculer vers l'application de destination.

Dans ce cas, vous inversez la source et la destination.

Étapes

1. Sur le cluster source, créez un instantané d'arrêt :

Créez un instantané d'arrêt à l'aide d'une CR

- a. Désactivez les calendriers de stratégie de protection pour l'application source.
- b. Créer un fichier CR ShutdownSnapshot :
 - i. Créez le fichier de ressource personnalisée (CR) et nommez-le (par exemple, `trident-protect-shutdownsnapshot.yaml`).
 - ii. Configurez les attributs suivants :
 - **metadata.name**: (*Obligatoire*) Le nom de la ressource personnalisée.
 - **spec.AppVaultRef**: (*Obligatoire*) Cette valeur doit correspondre au champ `metadata.name` de l'AppVault pour l'application source.
 - **spec.ApplicationRef**: (*Obligatoire*) Cette valeur doit correspondre au champ `metadata.name` du fichier CR de l'application source.

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ShutdownSnapshot
metadata:
  name: replication-shutdown-snapshot-afc4c564-e700-4b72-86c3-
c08a5dbe844e
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
```

- c. Après avoir rempli le `trident-protect-shutdownsnapshot.yaml` fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-
namespace
```

Créez un instantané d'arrêt à l'aide de l'interface de ligne de commande (CLI).

- a. Créez un instantané d'arrêt en remplaçant les valeurs entre crochets par les informations de votre environnement. Par exemple:

```
tridentctl-protect create shutdownsnapshot <my_shutdown_snapshot>
--appvault <my_vault> --app <app_to_snapshot> -n
<application_namespace>
```

2. Sur le cluster source, une fois la capture instantanée de l'arrêt terminée, obtenez l'état de cette capture :

```
kubectl get shutdownsnapshot -n my-app-namespace  
<shutdown_snapshot_name> -o yaml
```

3. Sur le cluster source, recherchez la valeur de **shutdownsnapshot.status.appArchivePath** à l'aide de la commande suivante et notez la dernière partie du chemin d'accès au fichier (également appelée nom de base ; il s'agit de tout ce qui suit la dernière barre oblique) :

```
k get shutdownsnapshot -n my-app-namespace <shutdown_snapshot_name> -o  
jsonpath='{.status.appArchivePath}'
```

4. Effectuez un basculement du nouveau cluster de destination vers le nouveau cluster source, avec la modification suivante :



À l'étape 2 de la procédure de basculement, incluez le `spec.promotedSnapshot` champ dans le fichier CR AppMirrorRelationship, et définissez sa valeur sur le nom de base que vous avez enregistré à l'étape 3 ci-dessus.

5. Effectuez les étapes de resynchronisation inverses dans [Resynchronisation inverse d'une relation de réplication ayant échoué](#) .

6. Activez les plans de protection sur le nouveau cluster source.

Résultat

Les actions suivantes se produisent en raison de la réplication inverse :

- Une capture instantanée des ressources Kubernetes de l'application source d'origine est effectuée.
- Les pods de l'application source d'origine sont arrêtés en douceur en supprimant les ressources Kubernetes de l'application (en laissant les PVC et les PV en place).
- Une fois les pods arrêtés, des instantanés des volumes de l'application sont pris et répliqués.
- Les relations SnapMirror sont rompues, ce qui rend les volumes de destination prêts pour la lecture/écriture.
- Les ressources Kubernetes de l'application sont restaurées à partir de l'instantané antérieur à l'arrêt, en utilisant les données de volume répliquées après l'arrêt de l'application source d'origine.
- La réplication est rétablie dans le sens inverse.

Rétablir les applications sur le cluster source d'origine

Avec Trident Protect, vous pouvez effectuer un « retour en arrière » après une opération de basculement en utilisant la séquence d'opérations suivante. Dans ce flux de travail visant à rétablir le sens de réplication d'origine, Trident Protect réplique (resynchronise) toutes les modifications apportées à l'application vers l'application source d'origine avant d'inverser le sens de réplication.

Ce processus débute à partir d'une relation ayant effectué un basculement vers une destination et comprend les étapes suivantes :

- Commencez par un état de basculement.
- Inverser la resynchronisation de la relation de réplication.



N'effectuez pas d'opération de resynchronisation normale, car cela supprimerait les données écrites sur le cluster de destination pendant la procédure de basculement.

- Inverser le sens de la réplication.

Étapes

1. Effectuez le [Resynchronisation inverse d'une relation de réplication ayant échoué](#) mesures.
2. Effectuez le [sens de réplication de l'application inverse](#) mesures.

Supprimer une relation de réplication

Vous pouvez supprimer une relation de réplication à tout moment. Lorsque vous supprimez la relation de réplication d'applications, cela crée deux applications distinctes sans aucune relation entre elles.

Étapes

1. Sur le cluster de destination actuel, supprimez la ressource personnalisée AppMirrorRelationship :

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

Migrer les applications à l'aide de Trident Protect

Vous pouvez migrer vos applications entre clusters ou vers différentes classes de stockage en restaurant les données de sauvegarde.



Lors de la migration d'une application, tous les points d'exécution configurés pour celle-ci sont migrés avec elle. Si un point d'entrée d'exécution post-restauration est présent, il s'exécute automatiquement dans le cadre de l'opération de restauration.

opérations de sauvegarde et de restauration

Pour effectuer des opérations de sauvegarde et de restauration dans les scénarios suivants, vous pouvez automatiser des tâches spécifiques de sauvegarde et de restauration.

Cloner sur le même cluster

Pour cloner une application sur le même cluster, créez un instantané ou une sauvegarde et restaurez les données sur le même cluster.

Étapes

1. Effectuez l'une des opérations suivantes :
 - a. "[Créer un instantané](#)".
 - b. "[Créer une sauvegarde](#)".
2. Sur le même cluster, effectuez l'une des opérations suivantes, selon que vous ayez créé un instantané ou une sauvegarde :

- a. "Restaurez vos données à partir de l'instantané."
- b. "Restaurez vos données à partir de la sauvegarde".

Cloner sur un cluster différent

Pour cloner une application sur un cluster différent (effectuer un clonage inter-clusters), créez une sauvegarde sur le cluster source, puis restaurez la sauvegarde sur un cluster différent. Assurez-vous que Trident Protect est installé sur le cluster de destination.



Vous pouvez répliquer une application entre différents clusters en utilisant "[Réplication SnapMirror](#)".

Étapes

1. "Créer une sauvegarde".
2. Assurez-vous que la ressource personnalisée AppVault pour le compartiment de stockage d'objets contenant la sauvegarde a été configurée sur le cluster de destination.
3. Sur le cluster de destination, "[restaurez vos données à partir de la sauvegarde](#)".

Migrer des applications d'une classe de stockage à une autre classe de stockage

Vous pouvez migrer des applications d'une classe de stockage vers une autre classe de stockage en restaurant une sauvegarde vers la classe de stockage de destination.

Par exemple (en excluant les secrets de la restauration CR) :

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: "${snapshotRestoreCRName}"
spec:
  appArchivePath: "${snapshotArchivePath}"
  appVaultRef: "${appVaultCRName}"
  namespaceMapping:
    - destination: "${destinationNamespace}"
      source: "${sourceNamespace}"
  storageClassMapping:
    - destination: "${destinationStorageClass}"
      source: "${sourceStorageClass}"
  resourceFilter:
    resourceMatchers:
      kind: Secret
      version: v1
    resourceSelectionCriteria: exclude
```

Restaurez l'instantané à l'aide d'une CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-snapshot-restore-cr.yaml`.
2. Dans le fichier que vous avez créé, configurez les attributs suivants :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.appArchivePath** : Le chemin d'accès dans AppVault où le contenu des instantanés est stocké. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
```

- **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où le contenu de l'instantané est stocké.
- **spec.namespaceMapping** : Le mappage de l'espace de noms source de l'opération de restauration vers l'espace de noms de destination. Remplacer `my-source-namespace` et `my-destination-namespace` avec des informations provenant de votre environnement.

```
---
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: my-cr-name
  namespace: trident-protect
spec:
  appArchivePath: my-snapshot-path
  appVaultRef: appvault-name
  namespaceMapping: [{"source": "my-source-namespace",
"destination": "my-destination-namespace"}]
```

3. Si vous souhaitez sélectionner uniquement certaines ressources de l'application à restaurer, vous pouvez ajouter un filtrage qui inclut ou exclut les ressources marquées d'étiquettes particulières :

- **resourceFilter.resourceSelectionCriteria** : (Requis pour le filtrage) Utiliser `include` or `exclude` inclure ou exclure une ressource définie dans `resourceMatchers`. Ajoutez les paramètres `resourceMatchers` suivants pour définir les ressources à inclure ou à exclure :
 - **resourceFilter.resourceMatchers** : Un tableau d'objets `resourceMatcher`. Si vous définissez plusieurs éléments dans ce tableau, ils correspondent selon une opération OU, et les champs à l'intérieur de chaque élément (groupe, type, version) correspondent selon une opération ET.
 - **resourceMatchers[].group**: (*Optionnel*) Groupe de la ressource à filtrer.
 - **resourceMatchers[].kind**: (*Optionnel*) Type de ressource à filtrer.
 - **resourceMatchers[].version**: (*Optionnel*) Version de la ressource à filtrer.
 - **resourceMatchers[].names**: (*Optionnel*) Noms dans le champ `metadata.name` de

Kubernetes de la ressource à filtrer.

- **resourceMatchers[].namespaces**: (*Optionnel*) Espaces de noms dans le champ metadata.name de Kubernetes de la ressource à filtrer.
- **resourceMatchers[].labelSelectors** : (*Facultatif*) Chaîne de sélection d'étiquette dans le champ metadata.name de la ressource Kubernetes, telle que définie dans le ["Documentation Kubernetes"](#) . Par exemple: "trident.netapp.io/os=linux" .

Par exemple:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Après avoir rempli le trident-protect-snapshot-restore-cr.yaml fichier contenant les valeurs correctes, appliquer le CR :

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Restaurez l'instantané à l'aide de l'interface de ligne de commande (CLI).

Étapes

1. Restaurez l'instantané dans un espace de noms différent, en remplaçant les valeurs entre crochets par les informations de votre environnement.
 - Le snapshot L'argument utilise un espace de noms et un nom d'instantané au format <namespace>/<name> .
 - Le namespace-mapping L'argument utilise des espaces de noms séparés par deux-points pour associer les espaces de noms source aux espaces de noms de destination corrects au format source1:dest1, source2:dest2 .

Par exemple:

```
tridentctl-protect create snapshotrestore <my_restore_name>
--snapshot <namespace/snapshot_to_restore> --namespace-mapping
<source_to_destination_namespace_mapping>
```

Gérer les hooks d'exécution de Trident Protect

Un hook d'exécution est une action personnalisée que vous pouvez configurer pour qu'elle s'exécute conjointement avec une opération de protection des données d'une application gérée. Par exemple, si vous disposez d'une application de base de données, vous pouvez utiliser un hook d'exécution pour suspendre toutes les transactions de base de données avant un instantané et reprendre les transactions une fois l'instantané terminé. Cela garantit des instantanés cohérents avec les applications.

Types de hooks d'exécution

Trident Protect prend en charge les types de hooks d'exécution suivants, en fonction du moment où ils peuvent être exécutés :

- Pré-instantané
- Post-instantané
- Pré-sauvegarde
- Post-sauvegarde
- Post-restauration
- Après le basculement

Ordre d'exécution

Lorsqu'une opération de protection des données est exécutée, les événements de hook d'exécution se produisent dans l'ordre suivant :

1. Tous les hooks d'exécution de pré-opération personnalisés applicables sont exécutés sur les conteneurs appropriés. Vous pouvez créer et exécuter autant de hooks de pré-opération personnalisés que vous le souhaitez, mais l'ordre d'exécution de ces hooks avant l'opération n'est ni garanti ni configurable.
2. Des blocages du système de fichiers se produisent, le cas échéant. ["Apprenez-en davantage sur la configuration du gel du système de fichiers avec Trident Protect."](#)
3. L'opération de protection des données est effectuée.
4. Les systèmes de fichiers gelés sont dégelés, le cas échéant.
5. Tous les hooks d'exécution post-opération personnalisés applicables sont exécutés sur les conteneurs appropriés. Vous pouvez créer et exécuter autant de hooks post-opération personnalisés que vous le souhaitez, mais l'ordre d'exécution de ces hooks après l'opération n'est ni garanti ni configurable.

Si vous créez plusieurs hooks d'exécution du même type (par exemple, pré-snapshot), l'ordre d'exécution de ces hooks n'est pas garanti. Cependant, l'ordre d'exécution des hooks de différents types est garanti. Par exemple, voici l'ordre d'exécution d'une configuration qui possède tous les différents types de hooks :

1. Hooks pré-instantanés exécutés
2. Hooks post-instantanés exécutés
3. Hooks de pré-sauvegarde exécutés
4. Hooks post-sauvegarde exécutés



L'exemple de commande précédent ne s'applique que lorsque vous exécutez une sauvegarde qui n'utilise pas d'instantané existant.



Vous devez toujours tester vos scripts d'exécution avant de les activer dans un environnement de production. Vous pouvez utiliser la commande « `kubectl exec` » pour tester facilement les scripts. Après avoir activé les hooks d'exécution dans un environnement de production, testez les snapshots et les sauvegardes résultants pour vous assurer qu'ils sont cohérents. Vous pouvez le faire en clonant l'application dans un espace de noms temporaire, en restaurant l'instantané ou la sauvegarde, puis en testant l'application.



Si un hook d'exécution pré-snapshot ajoute, modifie ou supprime des ressources Kubernetes, ces modifications sont incluses dans le snapshot ou la sauvegarde et dans toute opération de restauration ultérieure.

Remarques importantes sur les hooks d'exécution personnalisés

Tenez compte des éléments suivants lors de la planification des hooks d'exécution pour vos applications.

- Un hook d'exécution doit utiliser un script pour effectuer des actions. De nombreux hooks d'exécution peuvent référencer le même script.
- Trident Protect exige que les scripts utilisés par les hooks d'exécution soient écrits au format de scripts shell exécutables.
- La taille du script est limitée à 96 Ko.
- Trident Protect utilise les paramètres d'exécution et tous les critères correspondants pour déterminer quels hooks sont applicables à une opération de snapshot, de sauvegarde ou de restauration.



Étant donné que les hooks d'exécution réduisent ou désactivent souvent complètement les fonctionnalités de l'application sur laquelle ils s'exécutent, vous devez toujours essayer de minimiser le temps d'exécution de vos hooks d'exécution personnalisés. Si vous démarrez une opération de sauvegarde ou de snapshot avec des hooks d'exécution associés, mais que vous l'annulez ensuite, les hooks sont toujours autorisés à s'exécuter si l'opération de sauvegarde ou de snapshot a déjà commencé. Cela signifie que la logique utilisée dans un hook d'exécution post-sauvegarde ne peut pas supposer que la sauvegarde a été terminée.

Filtres de crochet d'exécution

Lorsque vous ajoutez ou modifiez un hook d'exécution pour une application, vous pouvez ajouter des filtres au hook d'exécution pour gérer les conteneurs auxquels le hook correspondra. Les filtres sont utiles pour les applications qui utilisent la même image de conteneur sur tous les conteneurs, mais peuvent utiliser chaque image à des fins différentes (comme Elasticsearch). Les filtres vous permettent de créer des scénarios dans lesquels les hooks d'exécution s'exécutent sur certains conteneurs identiques, mais pas nécessairement sur tous. Si vous créez plusieurs filtres pour un seul hook d'exécution, ils sont combinés avec un opérateur AND logique. Vous pouvez avoir jusqu'à 10 filtres actifs par hook d'exécution.

Chaque filtre que vous ajoutez à un hook d'exécution utilise une expression régulière pour faire correspondre les conteneurs de votre cluster. Lorsqu'un hook correspond à un conteneur, le hook exécutera son script associé sur ce conteneur. Les expressions régulières pour les filtres utilisent la syntaxe d'expression régulière 2 (RE2), qui ne prend pas en charge la création d'un filtre excluant les conteneurs de la liste des correspondances. Pour plus d'informations sur la syntaxe prise en charge par Trident Protect pour les expressions régulières dans les filtres de hook d'exécution, consultez "[Prise en charge de la syntaxe des expressions régulières 2 \(RE2\)](#)".



Si vous ajoutez un filtre d'espace de noms à un hook d'exécution qui s'exécute après une opération de restauration ou de clonage et que la source et la destination de restauration ou de clonage se trouvent dans des espaces de noms différents, le filtre d'espace de noms est appliqué uniquement à l'espace de noms de destination.

Exemples de crochets d'exécution

Visitez le "[Projet GitHub NetApp Verda](#)" pour télécharger de véritables hooks d'exécution pour des applications populaires telles qu'Apache Cassandra et Elasticsearch. Vous pouvez également voir des exemples et obtenir des idées pour structurer vos propres hooks d'exécution personnalisés.

Créer un point d'accroche d'exécution

Vous pouvez créer un hook d'exécution personnalisé pour une application en utilisant . Vous devez disposer des autorisations de propriétaire, d'administrateur ou de membre pour créer des points d'exécution.

Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-hook.yaml`.
2. Configurez les attributs suivants pour qu'ils correspondent à votre environnement Trident Protect et à la configuration de votre cluster :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.applicationRef**: (*Obligatoire*) Le nom Kubernetes de l'application pour laquelle exécuter le hook d'exécution.
 - **spec.stage**: (*Obligatoire*) Une chaîne de caractères indiquant à quelle étape de l'action le hook d'exécution doit s'exécuter. Valeurs possibles :
 - Pré
 - Poste
 - **spec.action**: (*Obligatoire*) Une chaîne indiquant l'action que le hook d'exécution entreprendra, en supposant que tous les filtres de hook d'exécution spécifiés correspondent. Valeurs possibles :
 - Instantané
 - Sauvegarde
 - Restaurer
 - Basculement
 - **spec.enabled**: (*Optionnel*) Indique si ce point d'accroche d'exécution est activé ou désactivé. Si aucune valeur n'est spécifiée, la valeur par défaut est « vrai ».
 - **spec.hookSource**: (*Obligatoire*) Une chaîne contenant le script de hook encodé en base64.
 - **spec.timeout**: (*Optionnel*) Un nombre définissant la durée en minutes pendant laquelle le hook d'exécution est autorisé à s'exécuter. La valeur minimale est de 1 minute, et la valeur par défaut est de 25 minutes si elle n'est pas spécifiée.
 - **spec.arguments**: (*Optionnel*) Une liste YAML d'arguments que vous pouvez spécifier pour le hook d'exécution.
 - **spec.matchingCriteria**: (*Optionnel*) Une liste facultative de paires clé-valeur de critères, chaque paire constituant un filtre de crochet d'exécution. Vous pouvez ajouter jusqu'à 10 filtres par point d'exécution.
 - **spec.matchingCriteria.type**: (*Optionnel*) Une chaîne identifiant le type de filtre de crochet d'exécution. Valeurs possibles :
 - Image conteneur
 - Nom du conteneur
 - Nom du pod
 - Étiquette de podcast
 - Nom de l'espace de noms
 - **spec.matchingCriteria.value**: (*Optionnel*) Une chaîne de caractères ou une expression régulière identifiant la valeur du filtre du point d'exécution.

Exemple de YAML :

```
apiVersion: protect.trident.netapp.io/v1
kind: ExecHook
metadata:
  name: example-hook-cr
  namespace: my-app-namespace
  annotations:
    astra.netapp.io/astra-control-hook-source-id:
/account/test/hookSource/id
spec:
  applicationRef: my-app-name
  stage: Pre
  action: Snapshot
  enabled: true
  hookSource: IyEvYmluL2Jhc2gKZWNobyAiZXhhbXBsZSBzY3JpcHQiCg==
  timeout: 10
  arguments:
    - FirstExampleArg
    - SecondExampleArg
  matchingCriteria:
    - type: containerName
      value: mysql
    - type: containerImage
      value: bitnami/mysql
    - type: podName
      value: mysql
    - type: namespaceName
      value: mysql-a
    - type: podLabel
      value: app.kubernetes.io/component=primary
    - type: podLabel
      value: helm.sh/chart=mysql-10.1.0
    - type: podLabel
      value: deployment-type=production
```

3. Après avoir rempli le fichier CR avec les valeurs correctes, appliquez le CR :

```
kubectl apply -f trident-protect-hook.yaml
```

Utiliser la CLI

Étapes

1. Créez le point d'exécution en remplaçant les valeurs entre crochets par les informations provenant de votre environnement. Par exemple:

```
tridentctl-protect create exehook <my_exec_hook_name> --action  
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>  
--source-file <script-file> -n <application_namespace>
```

Exécuter manuellement un hook d'exécution

Vous pouvez exécuter manuellement un hook d'exécution à des fins de test ou si vous devez le réexécuter manuellement après un échec. Vous devez disposer des autorisations de propriétaire, d'administrateur ou de membre pour exécuter manuellement les hooks d'exécution.

L'exécution manuelle d'un hook se compose de deux étapes de base :

1. Créez une sauvegarde des ressources, qui collecte les ressources et en crée une copie, déterminant ainsi où le hook s'exécutera.
2. Exécutez le script d'exécution sur la sauvegarde

Étape 1 : Créer une sauvegarde des ressources



Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-resource-backup.yaml`.
2. Configurez les attributs suivants pour qu'ils correspondent à votre environnement Trident Protect et à la configuration de votre cluster :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.applicationRef**: (*Obligatoire*) Le nom Kubernetes de l'application pour laquelle créer la sauvegarde de ressource.
 - **spec.appVaultRef**: (*Obligatoire*) Le nom de l'AppVault où sont stockés les contenus de sauvegarde.
 - **spec.appArchivePath** : Chemin d'accès dans AppVault où sont stockés les contenus de sauvegarde. Vous pouvez utiliser la commande suivante pour trouver ce chemin :

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
```

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ResourceBackup
metadata:
  name: example-resource-backup
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
```

3. Après avoir rempli le fichier CR avec les valeurs correctes, appliquez le CR :

```
kubectl apply -f trident-protect-resource-backup.yaml
```

Utiliser la CLI

Étapes

1. Créez la sauvegarde en remplaçant les valeurs entre crochets par les informations provenant de votre environnement. Par exemple:

```
tridentctl protect create resourcebackup <my_backup_name> --app  
<my_app_name> --appvault <my_appvault_name> -n  
<my_app_namespace> --app-archive-path <app_archive_path>
```

2. Consultez l'état de la sauvegarde. Vous pouvez utiliser cette commande d'exemple à plusieurs reprises jusqu'à ce que l'opération soit terminée :

```
tridentctl protect get resourcebackup -n <my_app_namespace>  
<my_backup_name>
```

3. Vérifiez que la sauvegarde a réussi :

```
kubectl describe resourcebackup <my_backup_name>
```

Étape 2 : Exécuter le hook d'exécution



Utilisez un CR

Étapes

1. Créez le fichier de ressource personnalisée (CR) et nommez-le. `trident-protect-hook-run.yaml`.
2. Configurez les attributs suivants pour qu'ils correspondent à votre environnement Trident Protect et à la configuration de votre cluster :
 - **metadata.name** : (*Obligatoire*) Le nom de cette ressource personnalisée ; choisissez un nom unique et pertinent pour votre environnement.
 - **spec.applicationRef** : (*Obligatoire*) Assurez-vous que cette valeur corresponde au nom de l'application de la ressource de sauvegarde que vous avez créée à l'étape 1.
 - **spec.appVaultRef** : (*Obligatoire*) Assurez-vous que cette valeur correspond à l'appVaultRef de la ressource CR ResourceBackup que vous avez créée à l'étape 1.
 - **spec.appArchivePath** : Assurez-vous que cette valeur correspond à appArchivePath de la ressource personnalisée ResourceBackup que vous avez créée à l'étape 1.

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **spec.action**: (*Obligatoire*) Une chaîne indiquant l'action que le hook d'exécution entreprendra, en supposant que tous les filtres de hook d'exécution spécifiés correspondent. Valeurs possibles :
 - Instantané
 - Sauvegarde
 - Restaurer
 - Basculement
- **spec.stage**: (*Obligatoire*) Une chaîne de caractères indiquant à quelle étape de l'action le hook d'exécution doit s'exécuter. Cette opération d'accrochage ne permettra pas d'accrocher les hameçons à aucune autre étape. Valeurs possibles :
 - Pré
 - Poste

Exemple de YAML :

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ExecHooksRun
metadata:
  name: example-hook-run
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
  stage: Post
  action: Failover
```

3. Après avoir rempli le fichier CR avec les valeurs correctes, appliquez le CR :

```
kubectl apply -f trident-protect-hook-run.yaml
```

Utiliser la CLI

Étapes

1. Créer la requête d'exécution manuelle du hook :

```
tridentctl protect create exehooksruntime <my_exec_hook_run_name>
-n <my_app_namespace> --action snapshot --stage <pre_or_post>
--app <my_app_name> --appvault <my_appvault_name> --path
<my_backup_name>
```

2. Vérifiez l'état d'exécution du hook. Vous pouvez exécuter cette commande à plusieurs reprises jusqu'à ce que l'opération soit terminée :

```
tridentctl protect get exehooksruntime -n <my_app_namespace>
<my_exec_hook_run_name>
```

3. Décrivez l'objet exehooksruntime pour voir les détails et l'état finaux :

```
kubectl -n <my_app_namespace> describe exehooksruntime
<my_exec_hook_run_name>
```

Désinstallez Trident Protect

Vous devrez peut-être supprimer des composants de Trident Protect si vous passez d'une version d'essai à une version complète du produit.

Pour retirer Trident Protect, procédez comme suit.

Étapes

1. Supprimez les fichiers CR de Trident Protect :



Cette étape n'est pas requise pour la version 25.06 et les versions ultérieures.

```
helm uninstall -n trident-protect trident-protect-crds
```

2. Supprimer Trident Protect :

```
helm uninstall -n trident-protect trident-protect
```

3. Supprimez l'espace de noms Trident Protect :

```
kubectl delete ns trident-protect
```

Blogs de Trident et Trident Protect

Vous trouverez ici d'excellents blogs sur NetApp Trident et Trident Protect :

Blogs de Trident

- 9 mai 2025 : "[Configuration automatique du backend Trident pour FSx for ONTAP avec l'extension Amazon EKS](#)"
- 19 août 2025 : "[Amélioration de la cohérence des données : instantanés de groupes de volumes dans la virtualisation OpenShift avec Trident](#)"
- 15 avril 2025 : "[Protocole NetApp Trident avec Google Cloud NetApp Volumes pour SMB](#)"
- 14 avril 2025 : "[Exploiter le protocole Fibre Channel avec Trident 25.02 pour le stockage persistant sur Kubernetes](#)"
- 14 avril 2025 : "[Exploiter la puissance des systèmes NetApp ASA r2 pour le stockage par blocs Kubernetes](#)"
- 31 mars 2025 : "[Simplification de l'installation de Trident sur Red Hat OpenShift avec le nouvel opérateur certifié](#)"
- 27 mars 2025 : "[Provisionnement de Trident pour PME avec Google Cloud NetApp Volumes](#)"
- 5 mars 2025 : "[Débloquez l'intégration transparente du stockage iSCSI : Guide FSxN sur les clusters ROSA pour AWS](#)"
- 27 février 2025 : "[Déploiement de l'identité cloud avec Trident, GKE et Google Cloud NetApp Volumes](#)"
- 12 décembre 2024 : "[Introduction de la prise en charge de Fibre Channel dans Trident](#)"
- 26 novembre 2024 : "[Trident 25.01 : Amélioration de l'expérience de stockage Kubernetes avec de nouvelles fonctionnalités et améliorations](#)"
- 11 novembre 2024 : "[NetApp Trident avec Google Cloud NetApp Volumes](#)"
- 29 octobre 2024 : "[Amazon FSx for NetApp ONTAP avec Red Hat OpenShift Service sur AWS \(ROSA\) à l'aide de Trident](#)"
- 29 octobre 2024 : "[Migration en direct de machines virtuelles avec OpenShift Virtualization sur ROSA et Amazon FSx for NetApp ONTAP](#)"
- 8 juillet 2024 : "[Utilisation de NVMe/TCP pour consommer le stockage ONTAP pour vos applications conteneurisées modernes sur Amazon EKS](#)"
- 1er juillet 2024 : "[Stockage Kubernetes transparent avec Google Cloud NetApp Volumes Flex et Astra Trident](#)"
- 11 juin 2024 : "[ONTAP comme stockage backend pour le registre d'images intégré dans OpenShift](#)"

Blogs de Trident Protect

- 16 mai 2025 : "[Automatisation du basculement du registre pour la reprise après sinistre avec les hooks post-restauration de Trident Protect](#)"
- 16 mai 2025 : "[Reprise après sinistre de la virtualisation OpenShift avec NetApp Trident Protect](#)"
- 13 mai 2025 : "[Migration de classes de stockage avec sauvegarde et restauration Trident Protect](#)"
- 9 mai 2025 : "[Redimensionnez les applications Kubernetes avec les hooks post-restauration de Trident Protect](#)"

- 3 avril 2025 : "Trident Protect Power Up : réplication Kubernetes pour la protection et la reprise après sinistre"
- 13 mars 2025 : "Opérations de sauvegarde et de restauration cohérentes en cas de panne pour les machines virtuelles OpenShift Virtualization"
- 11 mars 2025 : "Étendre les modèles GitOps à la protection des données applicatives avec NetApp Trident"
- 3 mars 2025 : "Trident 25.02 : Une expérience Red Hat OpenShift améliorée grâce à de nouvelles fonctionnalités exceptionnelles"
- 15 janvier 2025 : "Présentation de Trident Protect, le contrôle d'accès basé sur les rôles"
- 11 novembre 2024 : "Présentation de tridentctl protect : la puissante interface de ligne de commande pour Trident Protect"
- 11 novembre 2024 : "Gestion des données pilotée par Kubernetes : la nouvelle ère avec Trident Protect"

Connaissances et soutien

Foire aux questions

Trouvez les réponses aux questions fréquemment posées concernant l'installation, la configuration, la mise à niveau et le dépannage de Trident.

Questions générales

À quelle fréquence Trident est-il mis à jour ?

À compter de la version 24.02, Trident sort tous les quatre mois : février, juin et octobre.

Trident prend-il en charge toutes les fonctionnalités publiées dans une version particulière de Kubernetes ?

Trident ne prend généralement pas en charge les fonctionnalités alpha dans Kubernetes. Trident pourrait prendre en charge des fonctionnalités bêta dans les deux versions de Trident qui suivront la version bêta de Kubernetes.

Trident dépend-il d'autres produits NetApp pour fonctionner ?

Trident ne dépend d'aucun autre produit logiciel NetApp et fonctionne comme une application autonome. Toutefois, vous devez disposer d'un périphérique de stockage dorsal NetApp .

Comment puis-je obtenir les détails complets de la configuration Trident ?

Utilisez le `tridentctl get` commande permettant d'obtenir plus d'informations sur votre configuration Trident .

Puis-je obtenir des statistiques sur la manière dont le stockage est provisionné par Trident?

Oui. Points de terminaison Prometheus pouvant être utilisés pour recueillir des informations sur le fonctionnement de Trident , telles que le nombre de serveurs backend gérés, le nombre de volumes provisionnés, les octets consommés, etc. Vous pouvez également utiliser "[Cloud Insights](#)" pour le suivi et l'analyse.

L'expérience utilisateur est-elle modifiée lors de l'utilisation de Trident en tant que fournisseur CSI ?

Non. Il n'y a aucun changement en ce qui concerne l'expérience utilisateur et les fonctionnalités. Le nom du fournisseur utilisé est `csi.trident.netapp.io` . Cette méthode d'installation de Trident est recommandée si vous souhaitez utiliser toutes les nouvelles fonctionnalités offertes par les versions actuelles et futures.

Installer et utiliser Trident sur un cluster Kubernetes

Trident prend-il en charge l'installation hors ligne à partir d'un registre privé ?

Oui, Trident peut être installé hors ligne. Se référer à "[Découvrez l'installation de Trident](#)" .

Puis-je installer Trident à distance ?

Oui. Trident 18.10 et versions ultérieures prennent en charge l'installation à distance depuis n'importe quelle machine disposant de `kubectl` accès au cluster. Après `kubectl` L'accès est vérifié (par exemple, initier une `kubectl get nodes` (commande depuis la machine distante pour vérifier), suivez les instructions d'installation.

Puis-je configurer la haute disponibilité avec Trident?

Trident est installé en tant que déploiement Kubernetes (ReplicaSet) avec une seule instance, et intègre donc la haute disponibilité. Il est déconseillé d'augmenter le nombre de réplicas dans le déploiement. Si le nœud sur lequel Trident est installé est perdu ou si le pod est inaccessible, Kubernetes redéploie automatiquement le pod sur un nœud sain de votre cluster. Le Trident ne concerne que le plan de contrôle ; les nacelles actuellement installées ne sont donc pas affectées en cas de redéploiement du Trident .

Trident a-t-il besoin d'accéder à l'espace de noms kube-system ?

Trident lit les données du serveur d'API Kubernetes pour déterminer quand les applications demandent de nouveaux PVC ; il a donc besoin d'accéder à kube-system.

Quels sont les rôles et les privilèges utilisés par Trident?

Le programme d'installation Trident crée un Kubernetes ClusterRole, qui dispose d'un accès spécifique aux ressources PersistentVolume, PersistentVolumeClaim, StorageClass et Secret du cluster Kubernetes. Se référer à "[Personnaliser l'installation de tridentctl](#)" .

Puis-je générer localement les fichiers manifestes exacts utilisés par Trident pour l'installation ?

Vous pouvez générer et modifier localement, si nécessaire, les fichiers manifestes exacts utilisés par Trident pour l'installation. Se référer à "[Personnaliser l'installation de tridentctl](#)" .

Puis-je partager le même SVM backend ONTAP pour deux instances Trident distinctes pour deux clusters Kubernetes distincts ?

Bien que cela ne soit pas conseillé, vous pouvez utiliser le même SVM backend pour deux instances Trident . Spécifiez un nom de volume unique pour chaque instance lors de l'installation et/ou spécifiez un nom de volume unique `StoragePrefix` paramètre dans le `setup/backend.json` déposer. Ceci afin de garantir que le même FlexVol volume ne soit pas utilisé dans les deux cas.

Est-il possible d'installer Trident sous ContainerLinux (anciennement CoreOS) ?

Trident est simplement un pod Kubernetes et peut être installé partout où Kubernetes est exécuté.

Puis-je utiliser Trident avec NetApp Cloud Volumes ONTAP?

Oui, Trident est compatible avec AWS, Google Cloud et Azure.

Trident est-il compatible avec Cloud Volumes Services ?

Oui, Trident prend en charge le service Azure NetApp Files dans Azure ainsi que le Cloud Volumes Service dans GCP.

Dépannage et assistance

NetApp prend-il en charge Trident?

Bien que Trident soit un logiciel libre et gratuit, NetApp le prend entièrement en charge à condition que votre système dorsal NetApp soit compatible.

Comment puis-je ouvrir un dossier d'assistance ?

Pour ouvrir une demande de prise en charge, procédez comme suit :

1. Contactez votre responsable de compte d'assistance et obtenez de l'aide pour créer un ticket.
2. Ouvrez un ticket d'assistance en contactant "[Assistance NetApp](#)".

Comment générer un bundle de journal de support ?

Vous pouvez créer un ensemble de support en exécutant `tridentctl logs -a`. En plus des journaux capturés dans le bundle, capturez le journal kubelet pour diagnostiquer les problèmes de montage côté Kubernetes. Les instructions pour obtenir le journal kubelet varient en fonction de la manière dont Kubernetes est installé.

Que dois-je faire si je dois soumettre une demande pour une nouvelle fonctionnalité ?

Créer un problème sur "[Trident Github](#)" et mentionnez **RFE** dans l'objet et la description du problème.

Où dois-je signaler un défaut ?

Créer un problème sur "[Trident Github](#)". Veillez à inclure toutes les informations et tous les journaux nécessaires relatifs au problème.

Que se passe-t-il si j'ai une question rapide sur Trident sur laquelle j'ai besoin d'éclaircissements ? Existe-t-il une communauté ou un forum ?

Si vous avez des questions, des problèmes ou des demandes, contactez-nous via notre Trident "[Chaîne Discord](#)" ou GitHub.

Le mot de passe de mon système de stockage a changé et Trident ne fonctionne plus. Comment puis-je le récupérer ?

Mettez à jour le mot de passe du backend avec `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Remplacer `myBackend` dans l'exemple avec le nom de votre backend, et `</path/to_new_backend.json` avec le chemin vers le bon `backend.json` déposer.

Trident ne trouve pas mon nœud Kubernetes. Comment puis-je résoudre ce problème ?

Deux scénarios sont susceptibles d'expliquer pourquoi Trident ne trouve pas de nœud Kubernetes. Cela peut être dû à un problème de réseau au sein de Kubernetes ou à un problème DNS. Le daemonset de nœud Trident qui s'exécute sur chaque nœud Kubernetes doit pouvoir communiquer avec le contrôleur Trident pour enregistrer le nœud auprès de Trident. Si des modifications du réseau sont survenues après l'installation de Trident, vous ne rencontrerez ce problème qu'avec les nouveaux nœuds Kubernetes ajoutés au cluster.

Si la capsule Trident est détruite, vais-je perdre les données ?

Aucune donnée ne sera perdue en cas de destruction de la capsule Trident . Les métadonnées Trident sont stockées dans des objets CRD. Tous les PV provisionnés par Trident fonctionneront normalement.

Amélioration du Trident

Puis-je passer directement d'une version antérieure à une version plus récente (en sautant quelques versions) ?

NetApp prend en charge la mise à niveau de Trident d'une version majeure à la version majeure suivante. Vous pouvez passer de la version 18.xx à la version 19.xx, de la version 19.xx à la version 20.xx, et ainsi de suite. Il est conseillé de tester la mise à niveau en laboratoire avant le déploiement en production.

Est-il possible de revenir à une version antérieure de Trident ?

Si vous avez besoin d'un correctif pour des bogues observés après une mise à niveau, des problèmes de dépendances ou une mise à niveau incomplète ou ayant échoué, vous devriez ["désinstaller Trident"](#) et réinstallez la version précédente en suivant les instructions spécifiques à cette version. Il s'agit de la seule méthode recommandée pour revenir à une version antérieure.

Gérer les backends et les volumes

Dois-je définir à la fois les LIF de gestion et de données dans un fichier de définition backend ONTAP ?

La LIF de gestion est obligatoire. DataLIF varie :

- SAN ONTAP : 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. Se référer à ["Options et exemples de configuration SAN ONTAP"](#) pour plus de détails.
- ONTAP NAS : 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`. Se référer à ["Options et exemples de configuration ONTAP NAS"](#) pour plus de détails

Trident peut-il configurer CHAP pour les systèmes backend ONTAP ?

Oui. Trident prend en charge le protocole CHAP bidirectionnel pour les systèmes backend ONTAP . Cela nécessite de paramétrer `useCHAP=true` dans votre configuration backend.

Comment gérer les politiques d'exportation avec Trident?

Trident peut créer et gérer dynamiquement des politiques d'exportation à partir de la version 20.04. Cela permet à l'administrateur de stockage de fournir un ou plusieurs blocs CIDR dans sa configuration backend et de faire en sorte que Trident ajoute les adresses IP des nœuds qui se trouvent dans ces plages à une politique d'exportation qu'il crée. De cette manière, Trident gère automatiquement l'ajout et la suppression de règles pour les nœuds dont les adresses IP se situent dans les CIDR spécifiés.

Les adresses IPv6 peuvent-elles être utilisées pour les LIF de gestion et de données ?

Trident prend en charge la définition d'adresses IPv6 pour :

- `managementLIF`` et ``dataLIF` pour les serveurs NAS ONTAP .
- `managementLIF`` pour les backends SAN ONTAP . Vous ne pouvez pas spécifier ``dataLIF` sur un système dorsal SAN ONTAP .

Trident doit être installé à l'aide du drapeau `--use-ipv6` (pour `tridentctl` installation), `IPv6` (pour opérateur Trident), ou `tridentTPv6` (pour l'installation de Helm) pour qu'il fonctionne sur IPv6.

Est-il possible de mettre à jour le LIF de gestion côté serveur ?

Oui, il est possible de mettre à jour le LIF de gestion backend à l'aide de `tridentctl update backend` commande.

Est-il possible de mettre à jour le DataLIF côté serveur ?

Vous pouvez mettre à jour le DataLIF sur `ontap-nas` et `ontap-nas-economy` seulement.

Est-il possible de créer plusieurs backends dans Trident pour Kubernetes ?

Trident peut prendre en charge simultanément plusieurs serveurs dorsaux, avec le même pilote ou des pilotes différents.

Comment Trident stocke-t-il les identifiants du serveur ?

Trident stocke les identifiants du backend sous forme de secrets Kubernetes.

Comment Trident sélectionne-t-il un serveur dorsal spécifique ?

Si les attributs du backend ne peuvent pas être utilisés pour sélectionner automatiquement les pools appropriés pour une classe, `storagePools` et `additionalStoragePools`. Des paramètres sont utilisés pour sélectionner un ensemble spécifique de pools.

Comment puis-je m'assurer que Trident ne sera pas provisionné à partir d'un serveur dorsal spécifique ?

Le `excludeStoragePools` Ce paramètre sert à filtrer l'ensemble des pools que Trident utilise pour le provisionnement et supprimera tous les pools correspondants.

S'il existe plusieurs serveurs dorsaux du même type, comment Trident sélectionne-t-il celui à utiliser ?

S'il existe plusieurs serveurs backend configurés du même type, Trident sélectionne le serveur backend approprié en fonction des paramètres présents dans `StorageClass` et `PersistentVolumeClaim`. Par exemple, s'il existe plusieurs backends de pilotes `ontap-nas`, Trident tente de faire correspondre les paramètres dans le `StorageClass` et `PersistentVolumeClaim` combinés et adaptés à un backend capable de répondre aux exigences énumérées dans `StorageClass` et `PersistentVolumeClaim`. S'il existe plusieurs serveurs backend correspondant à la requête, Trident en sélectionne un au hasard.

Trident prend-il en charge le protocole CHAP bidirectionnel avec Element/ SolidFire?

Oui.

Comment Trident déploie-t-il des Qtrees sur un volume ONTAP ? Combien de Qtrees peuvent être déployés sur un seul volume ?

Le `ontap-nas-economy` Le pilote crée jusqu'à 200 Qtrees dans le même FlexVol volume (configurable entre 50 et 300), 100 000 Qtrees par nœud de cluster et 2,4 millions par cluster. Lorsque vous entrez dans un nouveau `PersistentVolumeClaim` qui est pris en charge par le pilote économique, ce dernier vérifie s'il existe déjà un FlexVol volume capable de prendre en charge le nouveau Qtree. Si aucun FlexVol volume ne peut prendre en charge l'arbre Qtree, un nouveau FlexVol volume est créé.

Comment puis-je configurer les permissions Unix pour les volumes provisionnés sur un NAS ONTAP ?

Vous pouvez définir les permissions Unix sur le volume provisionné par Trident en définissant un paramètre dans le fichier de définition du backend.

Comment puis-je configurer un ensemble explicite d'options de montage NFS ONTAP lors du provisionnement d'un volume ?

Par défaut, Trident ne configure aucune option de montage avec Kubernetes. Pour spécifier les options de montage dans la classe de stockage Kubernetes, suivez l'exemple fourni ["ici"](#) .

Comment puis-je configurer les volumes provisionnés selon une politique d'exportation spécifique ?

Pour autoriser les hôtes appropriés à accéder à un volume, utilisez le `exportPolicy` Paramètre configuré dans le fichier de définition du backend.

Comment configurer le chiffrement des volumes via Trident avec ONTAP?

Vous pouvez configurer le chiffrement du volume provisionné par Trident en utilisant le paramètre de chiffrement dans le fichier de définition du backend. Pour plus d'informations, veuillez consulter : "[Comment Trident fonctionne avec NVE et NAE](#)"

Quelle est la meilleure façon de mettre en œuvre la QoS pour ONTAP via Trident?

Utiliser `StorageClasses` implémenter la QoS pour ONTAP.

Comment puis-je spécifier un provisionnement fin ou épais via Trident?

Les pilotes ONTAP prennent en charge le provisionnement fin ou épais. Les pilotes ONTAP utilisent par défaut le provisionnement fin. Si le provisionnement épais est souhaité, vous devez configurer soit le fichier de définition du backend, soit le `StorageClass` . Si les deux sont configurés, `StorageClass` a priorité. Configurez les éléments suivants pour ONTAP:

1. Sur `StorageClass` , définissez le `provisioningType` Attribut : épais.
2. Dans le fichier de définition du backend, activez les volumes épais en configurant `backend spaceReserve parameter` en volume.

Comment puis-je m'assurer que les volumes utilisés ne sont pas supprimés même si je supprime accidentellement le PVC ?

La protection PVC est automatiquement activée sur Kubernetes à partir de la version 1.10.

Puis-je cultiver des PVC NFS créées par Trident?

Oui. Vous pouvez dilater un PVC créé par Trident. Notez que l'extension automatique des volumes est une fonctionnalité ONTAP qui ne s'applique pas à Trident.

Puis-je importer un volume alors qu'il est en mode SnapMirror Data Protection (DP) ou en mode hors ligne ?

L'importation du volume échoue si le volume externe est en mode DP ou hors ligne. Vous recevez le message d'erreur suivant :

```
Error: could not import volume: volume import failed to get size of
volume: volume <name> was not found (400 Bad Request) command terminated
with exit code 1.
Make sure to remove the DP mode or put the volume online before importing
the volume.
```

Comment le quota de ressources est-il traduit dans un cluster NetApp ?

Le système de quotas de ressources de stockage Kubernetes devrait fonctionner tant que le stockage NetApp dispose de capacités. Lorsque le stockage NetApp ne peut pas respecter les paramètres de quota Kubernetes en raison d'un manque de capacité, Trident tente de provisionner mais génère une erreur.

Puis-je créer des instantanés de volume avec Trident?

Oui. La création d'instantanés de volumes à la demande et de volumes persistants à partir d'instantanés est prise en charge par Trident. Pour créer des PV à partir d'instantanés, assurez-vous que les `VolumeSnapshotDataSource` La fonctionnalité de contrôle d'accès a été activée.

Quels sont les pilotes qui prennent en charge les instantanés de volume Trident ?

À compter d'aujourd'hui, la prise en charge des instantanés à la demande est disponible pour notre `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, et `azure-netapp-files` Pilotes backend.

Comment puis-je effectuer une sauvegarde instantanée d'un volume provisionné par Trident avec ONTAP?

Ceci est disponible sur `ontap-nas`, `ontap-san`, et `ontap-nas-flexgroup` conducteurs. Vous pouvez également spécifier un `snapshotPolicy` pour le `ontap-san-economy` pilote au niveau FlexVol .

Ceci est également disponible sur le `ontap-nas-economy` pilotes, mais au niveau de granularité du FlexVol volume et non au niveau de granularité de l'arbre `qtree`. Pour activer la possibilité de créer des instantanés des volumes provisionnés par Trident, définissez l'option du paramètre backend. `snapshotPolicy` à la politique de snapshot souhaitée telle que définie sur le backend ONTAP . Trident ne connaît pas les instantanés pris par le contrôleur de stockage.

Puis-je définir un pourcentage de réserve d'instantané pour un volume provisionné via Trident?

Oui, vous pouvez réserver un pourcentage spécifique d'espace disque pour stocker les copies d'instantanés via Trident en configurant le `snapshotReserve` attribut dans le fichier de définition du backend. Si vous avez configuré `snapshotPolicy` et `snapshotReserve` Dans le fichier de définition du backend, le pourcentage

de réserve de snapshots est défini en fonction de `snapshotReserve` pourcentage mentionné dans le fichier backend. Si le `snapshotReserve` Le pourcentage n'est pas mentionné ; par défaut, ONTAP prend un pourcentage de réserve de snapshot de 5. Si le `snapshotPolicy` L'option est définie sur aucune, le pourcentage de réserve d'instantané est défini sur 0.

Puis-je accéder directement au répertoire des instantanés de volume et copier des fichiers ?

Oui, vous pouvez accéder au répertoire des instantanés sur le volume provisionné par Trident en configurant le `snapshotDir` paramètre dans le fichier de définition du backend.

Puis-je configurer SnapMirror pour les volumes via Trident?

Actuellement, SnapMirror doit être configuré en externe à l'aide de l'interface de ligne de commande ONTAP ou du OnCommand System Manager.

Comment restaurer des volumes persistants à partir d'un instantané ONTAP spécifique ?

Pour restaurer un volume à partir d'un instantané ONTAP , procédez comme suit :

1. Mettez en veille le pod d'application qui utilise le volume persistant.
2. Revenez à l'instantané requis via l'interface de ligne de commande ONTAP ou OnCommand System Manager.
3. Redémarrez le pod d'application.

Trident peut-il provisionner des volumes sur des SVM ayant un miroir de partage de charge configuré ?

Des miroirs à répartition de charge peuvent être créés pour les volumes racine des SVM qui diffusent des données via NFS. ONTAP met automatiquement à jour les miroirs de partage de charge pour les volumes créés par Trident. Cela peut entraîner des retards dans le montage des volumes. Lorsque plusieurs volumes sont créés à l'aide de Trident, le provisionnement d'un volume dépend de la mise à jour par ONTAP du miroir de partage de charge.

Comment puis-je séparer l'utilisation des classes de stockage pour chaque client/locataire ?

Kubernetes n'autorise pas les classes de stockage dans les espaces de noms. Cependant, vous pouvez utiliser Kubernetes pour limiter l'utilisation d'une classe de stockage spécifique par espace de noms en utilisant des quotas de ressources de stockage, qui sont par espace de noms. Pour refuser à un espace de noms spécifique l'accès à un stockage spécifique, définissez le quota de ressources à 0 pour cette classe de stockage.

Dépannage

Utilisez les indications fournies ici pour résoudre les problèmes que vous pourriez rencontrer lors de l'installation et de l'utilisation de Trident.



Pour obtenir de l'aide concernant Trident, créez un dossier de support en utilisant `tridentctl logs -a -n trident` et envoyez-le au support NetApp .

Dépannage général

- Si la capsule Trident ne remonte pas correctement (par exemple, si la capsule Trident est bloquée dans le

ContainerCreating phase avec moins de deux conteneurs prêts), en cours d'exécution `kubectl -n trident describe deployment trident` et `kubectl -n trident describe pod trident--**` peut fournir des informations complémentaires. Obtention des journaux kubelet (par exemple, via `journalctl -xeu kubelet`) peut également être utile.

- Si les journaux de Trident ne contiennent pas suffisamment d'informations, vous pouvez essayer d'activer le mode débogage pour Trident en passant le `-d` Attribut au paramètre d'installation en fonction de votre option d'installation.

Vérifiez ensuite que le mode débogage est activé. `./tridentctl logs -n trident` et à la recherche de `level=debug msg` dans le journal.

Installé avec l'opérateur

```
kubectl patch torc trident -n <namespace> --type=merge -p
'{"spec":{"debug":true}}'
```

Cela redémarrera tous les pods Trident, ce qui peut prendre plusieurs secondes. Vous pouvez le vérifier en observant la colonne « ÂGE » dans le résultat de `kubectl get pod -n trident`.

Pour Trident 20.07 et 20.10, utilisez `tprov` au lieu de `torc`.

Installé avec Helm

```
helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
tridentDebug=true`
```

Installé avec tridentctl

```
./tridentctl uninstall -n trident
./tridentctl install -d -n trident
```

- Vous pouvez également obtenir des journaux de débogage pour chaque backend en incluant `debugTraceFlags` dans votre définition de backend. Par exemple, inclure `debugTraceFlags: {"api":true, "method":true,}` pour obtenir les appels d'API et les parcours de méthodes dans les journaux Trident. Les backends existants peuvent avoir `debugTraceFlags` configuré avec un `tridentctl backend update`.
- Lors de l'utilisation de Red Hat Enterprise Linux CoreOS (RHCOS), assurez-vous que `iscsid` est activé sur les nœuds de travail et démarré par défaut. Cela peut se faire en utilisant les configurations de machines OpenShift ou en modifiant les modèles d'allumage.
- Un problème courant que vous pourriez rencontrer lors de l'utilisation de Trident avec ["Azure NetApp Files"](#) Cela se produit lorsque les secrets du locataire et du client proviennent d'un enregistrement d'application avec des autorisations insuffisantes. Pour obtenir la liste complète des exigences du Trident, veuillez consulter le site web suivant : ["Azure NetApp Files"](#) configuration.
- En cas de problème lors du montage d'un panneau photovoltaïque sur un conteneur, assurez-vous que `rpcbind` est installé et fonctionne. Utilisez le gestionnaire de paquets requis pour le système d'exploitation hôte et vérifiez si `rpcbind` est en cours d'exécution. Vous pouvez vérifier l'état du `rpcbind` service en exécutant un `systemctl status rpcbind` ou son équivalent.

- Si un serveur Trident signale qu'il est dans le `failed` Bien que cet état ait fonctionné auparavant, il est probablement dû à la modification des identifiants SVM/administrateur associés au serveur. Mise à jour des informations du backend à l'aide de `tridentctl update backend` ou faire rebondir la capsule Trident résoudra ce problème.
- Si vous rencontrez des problèmes d'autorisation lors de l'installation de Trident avec Docker comme environnement d'exécution de conteneurs, essayez d'installer Trident avec... `--in cluster=false` drapeau. Cela n'utilisera pas de pod d'installation et évitera les problèmes d'autorisation rencontrés en raison de `trident-installer` utilisateur.
- Utilisez le `uninstall` parameter `<Uninstalling Trident>` pour le nettoyage après un échec. Par défaut, le script ne supprime pas les CRD créées par Trident, ce qui permet de désinstaller et de réinstaller en toute sécurité même dans un déploiement en cours d'exécution.
- Si vous souhaitez revenir à une version antérieure de Trident, exécutez d'abord la commande suivante : `tridentctl uninstall` commande pour supprimer Trident. Téléchargez le fichier souhaité "[Version Trident](#)" et installez en utilisant le `tridentctl install` commande.
- Après une installation réussie, si un tuyau en PVC est coincé dans le `Pending` phase, course `kubectl describe pvc` peut fournir des informations supplémentaires sur les raisons pour lesquelles Trident n'a pas pu provisionner un PV pour ce PVC.

Déploiement Trident infructueux avec l'opérateur

Si vous déployez Trident à l'aide de l'opérateur, l'état de `TridentOrchestrator` changements de `Installing` à `Installed`. Si vous observez le `Failed` Si l'état est incorrect et que l'opérateur ne parvient pas à se rétablir par lui-même, vous devez consulter les journaux de l'opérateur en exécutant la commande suivante :

```
tridentctl logs -l trident-operator
```

L'analyse des journaux du conteneur `trident-operator` peut permettre de localiser l'origine du problème. Par exemple, un tel problème pourrait être l'impossibilité de récupérer les images de conteneurs requises à partir des registres en amont dans un environnement isolé du réseau.

Pour comprendre pourquoi l'installation de Trident a échoué, il convient de consulter le `TridentOrchestrator` statut.

```

kubect1 describe torc trident-2
Name:          trident-2
Namespace:
Labels:        <none>
Annotations:   <none>
API Version:   trident.netapp.io/v1
Kind:          TridentOrchestrator
...
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Image Pull Secrets:          <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:          Trident is bound to another CR 'trident'
  Namespace:        trident-2
  Status:           Error
  Version:
Events:
  Type      Reason  Age                From                Message
  ----      -
Warning    Error    16s (x2 over 16s)  trident-operator.netapp.io  Trident
is bound to another CR 'trident'

```

Cette erreur indique qu'il existe déjà un `TridentOrchestrator` qui a été utilisé pour installer Trident. Étant donné que chaque cluster Kubernetes ne peut avoir qu'une seule instance de Trident, l'opérateur garantit qu'à tout moment, il n'existe qu'une seule instance active. `TridentOrchestrator` qu'elle peut créer.

De plus, l'observation de l'état des capsules Trident peut souvent indiquer si quelque chose ne va pas.

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

NAME	READY	STATUS	RESTARTS
trident-csi-4p5kq	1/2	ImagePullBackOff	0
trident-csi-6f45bfd8b6-vfrkw	4/5	ImagePullBackOff	0
trident-csi-9q5xc	1/2	ImagePullBackOff	0
trident-csi-9v95z	1/2	ImagePullBackOff	0
trident-operator-766f7b8658-ldzsv	1/1	Running	0

On constate clairement que les pods ne peuvent pas s'initialiser complètement car une ou plusieurs images de conteneur n'ont pas été récupérées.

Pour résoudre ce problème, vous devriez modifier le `TridentOrchestrator` CR. Vous pouvez également supprimer `TridentOrchestrator` et en créer une nouvelle avec la définition modifiée et précise.

Déploiement Trident infructueux utilisant `tridentctl`

Pour tenter de comprendre ce qui s'est mal passé, vous pouvez relancer le programme d'installation en utilisant `-d` argument, qui activera le mode débogage et vous aidera à comprendre quel est le problème :

```
./tridentctl install -n trident -d
```

Après avoir résolu le problème, vous pouvez nettoyer l'installation comme suit, puis exécuter le programme. `tridentctl install` commande à nouveau :

```
./tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Removed Trident user from security context constraint.
INFO Trident uninstallation succeeded.
```

Supprimer complètement Trident et CRD

Vous pouvez supprimer complètement Trident ainsi que toutes les CRD créées et les ressources personnalisées associées.



Cela ne peut pas être annulé. Ne faites pas cela à moins de vouloir une installation entièrement neuve de Trident. Pour désinstaller Trident sans supprimer les CRD, reportez-vous à "[Désinstaller Trident](#)".

Opérateur Trident

Pour désinstaller Trident et supprimer complètement les CRD à l'aide de l'opérateur Trident :

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

Barre

Pour désinstaller Trident et supprimer complètement les CRD à l'aide de Helm :

```
kubectl patch torc trident --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

`tridentctl`

Pour supprimer complètement les CRD après la désinstallation de Trident, utilisez `tridentctl`

```
tridentctl obliviate crd
```

Échec du déstockage des nœuds NVMe avec les espaces de noms de blocs bruts RWX sous Kubernetes 1.26

Si vous utilisez Kubernetes 1.26, la suppression des nœuds peut échouer lors de l'utilisation de NVMe/TCP avec des espaces de noms de blocs bruts RWX. Les scénarios suivants proposent des solutions de contournement à cette défaillance. Vous pouvez également mettre à niveau Kubernetes vers la version 1.27.

Suppression de l'espace de noms et du pod

Prenons l'exemple d'un espace de noms géré par Trident (volume persistant NVMe) attaché à un pod. Si vous supprimez l'espace de noms directement depuis le backend ONTAP, le processus de désinstallation se bloque après votre tentative de suppression du pod. Ce scénario n'a aucun impact sur le cluster Kubernetes ni sur les autres fonctions.

Solution de contournement

Démontez le volume persistant (correspondant à cet espace de noms) du nœud respectif et supprimez-le.

LIF de données bloquées

If you block (or bring down) all the dataLIFs of the NVMe Trident backend, the unstaging process gets stuck when you attempt to delete the pod. In this scenario, you cannot run any NVMe CLI commands on the Kubernetes node.

.Solution de contournement

Mettez en service le système de fichiers dataLIFS pour rétablir toutes les fonctionnalités.

Suppression du mappage d'espace de noms

If you remove the `hostNQN` of the worker node from the corresponding subsystem, the unstaging process gets stuck when you attempt to delete the pod. In this scenario, you cannot run any NVMe CLI commands on the Kubernetes node.

.Solution de contournement

Ajoutez le `hostNQN` retour au sous-système.

Les clients NFSv4.2 signalent un « argument non valide » après la mise à niveau ONTAP alors qu'ils s'attendent à ce que « v4.2-xattrs » soit activé

Après la mise à niveau ONTAP, les clients NFSv4.2 peuvent signaler des erreurs « argument non valide » lors de la tentative de montage des exportations NFSv4.2. Ce problème se produit lorsque le `v4.2-xattrs` l'option n'est pas activée sur le SVM. .Solution de contournement Activer le `v4.2-xattrs` option sur le SVM ou mise à niveau vers ONTAP 9.12.1 ou version ultérieure, où cette option est activée par défaut.

Support

NetApp propose une assistance pour Trident de différentes manières. De nombreuses options d'assistance gratuite sont disponibles 24h/24 et 7j/7, telles que des articles de base de connaissances (KB) et un canal Discord.

Soutien Trident

Trident propose trois niveaux de support en fonction de votre version. Se référer à "[Prise en charge des définitions par les versions logicielles NetApp](#)".

Support complet

Trident offre un support complet pendant douze mois à compter de la date de sortie.

Soutien limité

Trident offre un support limité pour les mois 13 à 24 à compter de la date de sortie.

Autonomie

La documentation Trident est disponible pour les mois 25 à 36 à compter de la date de sortie.

Version	Support complet	Soutien limité	Autonomie
---------	-----------------	----------------	-----------

"25,06"	Juin 2026	Juin 2027	Juin 2028
"25,02"	Février 2026	Février 2027	Février 2028
"24,10"	Octobre 2025	Octobre 2026	Octobre 2027
"24,06"	Juin 2025	Juin 2026	Juin 2027
"24,02"	Février 2025	Février 2026	Février 2027
"23,10"	—	Octobre 2025	Octobre 2026
"23,07"	—	Juillet 2025	Juillet 2026
"23,04"	—	Avril 2025	Avril 2026
"23,01"	—	—	Janvier 2026
"22,10"	—	—	Octobre 2025

Autonomie

Pour une liste complète des articles de dépannage, consultez ["Base de connaissances NetApp \(connexion requise\)"](#) .

soutien communautaire

Il existe une communauté publique dynamique d'utilisateurs de conteneurs (y compris des développeurs Trident) sur notre plateforme. ["Chaîne Discord"](#) . C'est un excellent endroit pour poser des questions générales sur le projet et discuter de sujets connexes avec des pairs partageant les mêmes idées.

Assistance technique NetApp

Pour obtenir de l'aide concernant Trident, créez un dossier de support en utilisant `tridentctl logs -a -n trident` et l'envoyer à [NetApp Support <Getting Help>](#) .

Pour plus d'informations

- ["Ressources Trident"](#)
- ["Hub Kubernetes"](#)

Référence

Ports Trident

Découvrez plus d'informations sur les ports utilisés par Trident pour la communication.

Ports Trident

Trident utilise les ports suivants pour la communication au sein de Kubernetes :

Port	But
8443	HTTPS de canal de retour
8001	point de terminaison des métriques Prometheus
8000	Serveur REST Trident
17546	Port de sonde de disponibilité/d'état de fonctionnement utilisé par les pods du démon Trident



Le port de la sonde de disponibilité/d'état peut être modifié lors de l'installation à l'aide de `--probe-port` drapeau. Il est important de vérifier que ce port n'est pas utilisé par un autre processus sur les nœuds de travail.

API REST Trident

Alors que ["commandes et options de tridentctl"](#) sont le moyen le plus simple d'interagir avec l'API REST de Trident , mais vous pouvez utiliser directement le point de terminaison REST si vous préférez.

Quand utiliser l'API REST

L'API REST est utile pour les installations avancées qui utilisent Trident comme binaire autonome dans des déploiements non-Kubernetes.

Pour une meilleure sécurité, le Trident REST API par défaut, l'exécution dans un pod est limitée à l'hôte local. Pour modifier ce comportement, vous devez configurer les paramètres de Trident. `-address` argument dans sa configuration de pod.

Utilisation de l'API REST

Pour des exemples d'appels à ces API, transmettez le débogage(`-d`) drapeau. Pour plus d'informations, veuillez consulter ["Gérez Trident à l'aide de tridentctl"](#) .

L'API fonctionne comme suit :

OBTENIR

GET <trident-address>/trident/v1/<object-type>

Liste tous les objets de ce type.

GET <trident-address>/trident/v1/<object-type>/<object-name>

Obtient les détails de l'objet nommé.

POSTE

POST <trident-address>/trident/v1/<object-type>

Crée un objet du type spécifié.

- Nécessite une configuration JSON pour que l'objet soit créé. Pour connaître les spécifications de chaque type d'objet, veuillez vous référer à "[Gérez Trident à l'aide de tridentctl](#)".
- Si l'objet existe déjà, le comportement varie : les backends mettent à jour l'objet existant, tandis que tous les autres types d'objets échoueront lors de l'opération.

SUPPRIMER

DELETE <trident-address>/trident/v1/<object-type>/<object-name>

Supprime la ressource nommée.



Les volumes associés aux serveurs backend ou aux classes de stockage continueront d'exister ; ceux-ci doivent être supprimés séparément. Pour plus d'informations, veuillez consulter "[Gérez Trident à l'aide de tridentctl](#)".

Options de ligne de commande

Trident propose plusieurs options de ligne de commande pour l'orchestrateur Trident . Vous pouvez utiliser ces options pour modifier votre déploiement.

Enregistrement

-debug

Active la sortie de débogage.

-loglevel <level>

Définit le niveau de journalisation (débogage, info, avertissement, erreur, fatal). Valeur par défaut : info.

Kubernetes

-k8s_pod

Utilisez cette option ou **-k8s_api_server** pour activer la prise en charge de Kubernetes. Ce paramétrage permet à Trident d'utiliser les informations d'identification du compte de service Kubernetes du pod conteneur pour contacter le serveur API. Cela ne fonctionne que lorsque Trident est exécuté en tant que pod dans un cluster Kubernetes avec des comptes de service activés.

-k8s_api_server <insecure-address:insecure-port>

Utilisez cette option ou **-k8s_pod** pour activer la prise en charge de Kubernetes. Lorsque cela est spécifié, Trident se connecte au serveur d'API Kubernetes en utilisant l'adresse et le port non sécurisés fournis. Cela

permet de déployer Trident en dehors d'un pod ; cependant, cela ne prend en charge que les connexions non sécurisées au serveur API. Pour une connexion sécurisée, déployez Trident dans un pod avec le `-k8s_pod` option.

Docker

-volume_driver <name>

Nom du pilote utilisé lors de l'enregistrement du plugin Docker. Valeur par défaut `netapp` .

-driver_port <port-number>

Écoutez sur ce port plutôt que sur un socket de domaine UNIX.

-config <file>

Obligatoire ; vous devez spécifier ce chemin d'accès à un fichier de configuration backend.

REPOS

-address <ip-or-host>

Spécifie l'adresse sur laquelle le serveur REST de Trident doit écouter. Par défaut, `localhost`. Lorsqu'elle est exécutée en local et dans un pod Kubernetes, l'interface REST n'est pas directement accessible depuis l'extérieur du pod. Utiliser `-address ""` rendre l'interface REST accessible depuis l'adresse IP du pod.



L'interface REST de Trident peut être configurée pour écouter et servir uniquement à l'adresse `127.0.0.1` (pour IPv4) ou `:::1` (pour IPv6).

-port <port-number>

Spécifie le port sur lequel le serveur REST de Trident doit écouter. Par défaut, `8000`.

-rest

Active l'interface REST. Par défaut, la valeur est « vrai ».

Objets Kubernetes et Trident

Vous pouvez interagir avec Kubernetes et Trident à l'aide d'API REST en lisant et en écrivant des objets de ressources. Plusieurs objets de ressources déterminent la relation entre Kubernetes et Trident, Trident et le stockage, et Kubernetes et le stockage. Certains de ces objets sont gérés via Kubernetes et les autres via Trident.

Comment les objets interagissent-ils entre eux ?

La manière la plus simple de comprendre ces objets, leur utilité et leurs interactions consiste peut-être à suivre une simple requête de stockage provenant d'un utilisateur Kubernetes :

1. Un utilisateur crée un `PersistentVolumeClaim` demander un nouveau `PersistentVolume` d'une taille particulière à partir d'un Kubernetes `StorageClass` qui avait été configurée précédemment par l'administrateur.
2. Kubernetes `StorageClass` identifie Trident comme son fournisseur et inclut des paramètres qui indiquent à Trident comment provisionner un volume pour la classe demandée.

3. Trident examine sa propre `StorageClass` portant le même nom qui identifie la correspondance `Backends` et `StoragePools` qu'il peut utiliser pour provisionner des volumes pour la classe.
4. Trident provisionne le stockage sur un backend correspondant et crée deux objets : un `PersistentVolume` dans Kubernetes qui indique à Kubernetes comment trouver, monter et traiter le volume, et un volume dans Trident qui conserve la relation entre les `PersistentVolume` et le stockage proprement dit.
5. Kubernetes lie les `PersistentVolumeClaim` au nouveau `PersistentVolume` . Des capsules qui comprennent les `PersistentVolumeClaim` Montez ce `PersistentVolume` sur n'importe quel hôte sur lequel il s'exécute.
6. Un utilisateur crée un `VolumeSnapshot` d'un PVC existant, en utilisant un `VolumeSnapshotClass` cela désigne Trident.
7. Trident identifie le volume associé au PVC et crée un instantané de ce volume sur son système backend. Cela crée également un `VolumeSnapshotContent` qui indique à Kubernetes comment identifier l'instantané.
8. Un utilisateur peut créer un `PersistentVolumeClaim` en utilisant `VolumeSnapshot` comme source.
9. Trident identifie l'instantané requis et effectue la même série d'étapes que pour la création d'un `PersistentVolume` et un `Volume` .



Pour en savoir plus sur les objets Kubernetes, nous vous recommandons vivement de lire... "[Volumes persistants](#)" section de la documentation Kubernetes.

Kubernetes `PersistentVolumeClaim` objets

Un Kubernetes `PersistentVolumeClaim` L'objet représente une demande de stockage effectuée par un utilisateur d'un cluster Kubernetes.

En plus de la spécification standard, Trident permet aux utilisateurs de spécifier les annotations spécifiques au volume suivantes s'ils souhaitent remplacer les valeurs par défaut définies dans la configuration du backend :

Annotation	Option de volume	Pilotes pris en charge
<code>trident.netapp.io/fileSystem</code>	système de fichiers	ontap-san, solidfire-san, ontap-san-économie
<code>trident.netapp.io/cloneFromPVC</code>	<code>cloneSourceVolume</code>	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, gcp-cvs, ontap-san-economy
<code>trident.netapp.io/splitOnClone</code>	<code>splitOnClone</code>	ontap-nas, ontap-san
<code>trident.netapp.io/protocol</code>	protocole	n'importe lequel
<code>trident.netapp.io/exportPolicy</code>	politique d'exportation	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup
<code>trident.netapp.io/snapshotPolicy</code>	instantanéPolitique	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san
<code>trident.netapp.io/snapshotReserve</code>	<code>snapshotReserve</code>	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
<code>trident.netapp.io/snapshotDirectory</code>	répertoire snapshot	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup

Annotation	Option de volume	Pilotes pris en charge
trident.netapp.io/unixPermissions	Autorisations Unix	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup
trident.netapp.io/blockSize	taille du bloc	solidefire-san

Si le PV créé a le `Delete` En vertu de la politique de récupération, Trident supprime à la fois le PV et le volume de sauvegarde lorsque le PV est libéré (c'est-à-dire lorsque l'utilisateur supprime le PVC). En cas d'échec de la suppression, Trident marque le PV comme tel et réessaie périodiquement l'opération jusqu'à ce qu'elle réussisse ou que le PV soit supprimé manuellement. Si le PV utilise le `Retain` Trident ignore cette politique et suppose que l'administrateur la supprimera de Kubernetes et du système dorsal, permettant ainsi de sauvegarder ou d'inspecter le volume avant sa suppression. Notez que la suppression du PV n'entraîne pas la suppression du volume de sauvegarde par Trident . Vous devriez le supprimer à l'aide de l'API REST(`tridentctl`).

Trident prend en charge la création d'instantanés de volume à l'aide de la spécification CSI : vous pouvez créer un instantané de volume et l'utiliser comme source de données pour cloner des PVC existants. De cette manière, des copies ponctuelles des PV peuvent être exposées à Kubernetes sous forme d'instantanés. Ces instantanés peuvent ensuite être utilisés pour créer de nouveaux volumes persistants. Jetez un oeil à `On-Demand Volume Snapshots` pour voir comment cela fonctionnerait.

Trident fournit également `cloneFromPVC` et `splitOnClone` annotations pour la création de clones. Vous pouvez utiliser ces annotations pour cloner un PVC sans avoir à utiliser l'implémentation CSI.

Voici un exemple : si un utilisateur possède déjà un PVC appelé `mysql` L'utilisateur peut créer un nouveau PVC appelé `mysqlclone` en utilisant l'annotation, par exemple `trident.netapp.io/cloneFromPVC:mysql` . Avec cet ensemble d'annotations, Trident clone le volume correspondant au PVC `mysql`, au lieu de provisionner un volume à partir de zéro.

Considérez les points suivants :

- NetApp recommande de cloner un volume inactif.
- Un PVC et son clone doivent se trouver dans le même espace de noms Kubernetes et avoir la même classe de stockage.
- Avec le `ontap-nas` et `ontap-san` Pour les conducteurs, il pourrait être souhaitable de définir l'annotation `trident.netapp.io/splitOnClone` en conjonction avec `trident.netapp.io/cloneFromPVC` . Avec `trident.netapp.io/splitOnClone` défini à `true` Trident sépare le volume cloné du volume parent et découple ainsi complètement le cycle de vie du volume cloné de celui de son parent, au prix d'une perte d'efficacité de stockage. Ne pas paramétrer `trident.netapp.io/splitOnClone` ou en le paramétrant `false` Il en résulte une réduction de la consommation d'espace sur le système dorsal, au prix de la création de dépendances entre les volumes parent et clone, de sorte que le volume parent ne peut être supprimé que si le clone est supprimé au préalable. Un scénario où la division du clone est judicieuse est le clonage d'un volume de base de données vide, où l'on s'attend à ce que le volume et son clone divergent considérablement et ne bénéficient pas des gains d'efficacité de stockage offerts par ONTAP.

Le `sample-input` Ce répertoire contient des exemples de définitions PVC à utiliser avec Trident. Se référer à pour une description complète des paramètres et réglages associés aux volumes Trident .

Kubernetes `PersistentVolume` objets

Un Kubernetes `PersistentVolume` L'objet représente un élément de stockage mis à la disposition du cluster

Kubernetes. Son cycle de vie est indépendant du pod qui l'utilise.



Trident crée `PersistentVolume` il crée des objets et les enregistre automatiquement auprès du cluster Kubernetes en fonction des volumes qu'il provisionne. Vous n'êtes pas censé les gérer vous-même.

Lorsque vous créez un PVC qui fait référence à un Trident-based `StorageClass` Trident provisionne un nouveau volume en utilisant la classe de stockage correspondante et enregistre un nouveau PV pour ce volume. Lors de la configuration du volume provisionné et du PV correspondant, Trident suit les règles suivantes :

- Trident génère un nom de PV pour Kubernetes et un nom interne qu'il utilise pour provisionner le stockage. Dans les deux cas, cela garantit que les noms sont uniques dans leur portée.
- Le volume correspond au mieux à la taille demandée dans le PVC, bien qu'il puisse être arrondi à la quantité allouable la plus proche, selon la plateforme.

Kubernetes `StorageClass` objets

Kubernetes `StorageClass` Les objets sont spécifiés par leur nom dans `PersistentVolumeClaims` provisionner un espace de stockage avec un ensemble de propriétés. La classe de stockage elle-même identifie le fournisseur à utiliser et définit cet ensemble de propriétés en des termes que le fournisseur comprend.

Il s'agit de l'un des deux objets de base que l'administrateur doit créer et gérer. L'autre est l'objet backend Trident .

Un Kubernetes `StorageClass` Un objet utilisant Trident ressemble à ceci :

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters: <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Ces paramètres sont spécifiques à Trident et indiquent à Trident comment provisionner les volumes pour cette classe.

Les paramètres de la classe de stockage sont :

Attribut	Type	Obligatoire	Description
attributs	carte[chaîne]chaîne	Non	Voir la section attributs ci-dessous

Attribut	Type	Obligatoire	Description
Piscines de stockage	map[chaîne]StringList	Non	Carte des noms de backend vers des listes de pools de stockage au sein
Bassins de stockage supplémentaires	map[chaîne]StringList	Non	Carte des noms de backend vers des listes de pools de stockage au sein
exclure les pools de stockage	map[chaîne]StringList	Non	Carte des noms de backend vers des listes de pools de stockage au sein

Les attributs de stockage et leurs valeurs possibles peuvent être classés en attributs de sélection du pool de stockage et en attributs Kubernetes.

attributs de sélection du pool de stockage

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

Attribut	Type	Valeurs	Offre	Demande	Soutenu par
clones	booléen	vrai, faux	Pool prend en charge les volumes de clonage	Volume avec clones activés	ontap-nas, ontap-san, solidfire-san, gcp-cvs
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

Dans la plupart des cas, les valeurs demandées influencent directement le provisionnement ; par exemple, demander un provisionnement épais entraîne la création d'un volume provisionné de manière épaisse. Cependant, un pool de stockage Element utilise ses valeurs IOPS minimales et maximales offertes pour définir les valeurs QoS, plutôt que la valeur demandée. Dans ce cas, la valeur demandée sert uniquement à sélectionner le pool de stockage.

Idéalement, vous pouvez utiliser `attributes` seul pour modéliser les qualités du stockage dont vous avez besoin pour satisfaire les besoins d'une classe particulière. Trident découvre et sélectionne automatiquement les pools de stockage qui correspondent à *tous* les `attributes` que vous spécifiez.

Si vous vous trouvez dans l'incapacité d'utiliser `attributes` Pour sélectionner automatiquement les bons pools pour une classe, vous pouvez utiliser le `storagePools` et `additionalStoragePools` des paramètres permettant d'affiner davantage les pools, voire de sélectionner un ensemble spécifique de pools.

Vous pouvez utiliser le `storagePools` paramètre permettant de restreindre davantage l'ensemble des pools correspondant à une spécification donnée `attributes` . Autrement dit, Trident utilise l'intersection des bassins identifiés par le `attributes` et `storagePools` paramètres de provisionnement. Vous pouvez utiliser chaque paramètre seul ou les deux ensemble.

Vous pouvez utiliser le `additionalStoragePools` paramètre permettant d'étendre l'ensemble des pools que Trident utilise pour le provisionnement, indépendamment des pools sélectionnés par le `attributes` et `storagePools` paramètres.

Vous pouvez utiliser le `excludeStoragePools` Paramètre permettant de filtrer l'ensemble des pools que Trident utilise pour le provisionnement. L'utilisation de ce paramètre supprime tous les pools correspondants.

Dans le `storagePools` et `additionalStoragePools` en paramètres, chaque entrée prend la forme `<backend>:<storagePoolList>` , où `<storagePoolList>` est une liste de pools de stockage séparés par des virgules pour le backend spécifié. Par exemple, une valeur pour `additionalStoragePools` pourrait ressembler à `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze` . Ces listes acceptent les valeurs regex pour le backend et les valeurs de la liste. Vous pouvez utiliser `tridentctl get backend` pour obtenir la liste des serveurs backend et de leurs pools.

attributs Kubernetes

Ces attributs n'ont aucun impact sur la sélection des pools de stockage/backends par Trident lors du provisionnement dynamique. Ces attributs fournissent simplement des paramètres pris en charge par les volumes persistants Kubernetes. Les nœuds de travail sont responsables des opérations de création du système de fichiers et peuvent nécessiter des utilitaires de système de fichiers, tels que xfsprogs.

Attribut	Type	Valeurs	Description	Conducteurs pertinents	Version Kubernetes
fsType	chaîne	ext4, ext3, xfs	Le type de système de fichiers pour les volumes de blocs	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy	Tous
autoriser l'expansion du volume	booléen	vrai, faux	Activer ou désactiver la prise en charge de l'augmentation de la taille du PVC	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy, solidfire-san, gcp-cvs, azure-netapp-files	1,11+
mode de liaison du volume	chaîne	Immédiat, attendez le premier consommateur	Choisissez quand la liaison de volume et le provisionnement dynamique ont lieu.	Tous	1,19 - 1,26

- Le `fsType` Ce paramètre permet de contrôler le type de système de fichiers souhaité pour les LUN SAN. De plus, Kubernetes utilise également la présence de `fsType` dans une classe de stockage pour indiquer l'existence d'un système de fichiers. La propriété des volumes peut être contrôlée à l'aide de `fsGroup` contexte de sécurité d'un pod uniquement si `fsType` est réglé. Se référer à "[Kubernetes : Configurer un contexte de sécurité pour un pod ou un conteneur](#)" pour un aperçu de la configuration de la propriété des volumes à l'aide de `fsGroup` contexte. Kubernetes appliquera le `fsGroup` valeur uniquement si :



- `fsType` est défini dans la classe de stockage.
- Le mode d'accès au PVC est RWO.

Pour les pilotes de stockage NFS, un système de fichiers existe déjà dans le cadre de l'exportation NFS. Afin d'utiliser `fsGroup` la classe de stockage doit encore spécifier un `fsType` Vous pouvez le paramétrer pour `nfs` ou toute valeur non nulle.

- Se référer à "[Augmenter les volumes](#)" pour plus de détails sur l'expansion du volume.
- Le package d'installation de Trident fournit plusieurs exemples de définitions de classes de stockage à utiliser avec Trident `.sample-input/storage-class-*.yaml` . La suppression d'une classe de stockage Kubernetes entraîne également la suppression de la classe de stockage Trident correspondante.

Kubernetes VolumeSnapshotClass objets

Kubernetes `VolumeSnapshotClass` les objets sont analogues à `StorageClasses` . Ils permettent de définir plusieurs classes de stockage et sont référencés par les instantanés de volume pour associer l'instantané à la classe d'instantané requise. Chaque instantané de volume est associé à une seule classe d'instantané de volume.

UN `VolumeSnapshotClass` La création d'instantanés doit être définie par un administrateur. Une classe d'instantané de volume est créée avec la définition suivante :

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

Le `driver` indique à Kubernetes que les demandes de snapshots de volume du `csi-snapclass` Les classes sont gérées par Trident. Le `deletionPolicy` Spécifie l'action à entreprendre lorsqu'un instantané doit être supprimé. Quand `deletionPolicy` est réglé sur `Delete` , les objets de snapshot de volume ainsi que le snapshot sous-jacent sur le cluster de stockage sont supprimés lorsqu'un snapshot est supprimé. Vous pouvez aussi le paramétrer sur `Retain` signifie que `VolumeSnapshotContent` et l'instantané physique sont conservés.

Kubernetes VolumeSnapshot objets

Un Kubernetes `VolumeSnapshot` L'objet est une requête visant à créer un instantané d'un volume. De même qu'un PVC représente une demande faite par un utilisateur pour un volume, un instantané de volume est une

demande faite par un utilisateur pour créer un instantané d'un PVC existant.

Lorsqu'une demande de snapshot de volume est reçue, Trident gère automatiquement la création du snapshot pour le volume sur le serveur et l'expose en créant un identifiant unique.

VolumeSnapshotContent objet. Vous pouvez créer des instantanés à partir de PVC existants et utiliser ces instantanés comme source de données lors de la création de nouveaux PVC.



Le cycle de vie d'un VolumeSnapshot est indépendant du PVC source : un snapshot persiste même après la suppression du PVC source. Lors de la suppression d'un PVC auquel sont associés des instantanés, Trident marque le volume de support de ce PVC dans un état **Suppression en cours**, mais ne le supprime pas complètement. Le volume est supprimé lorsque tous les instantanés associés sont supprimés.

Kubernetes VolumeSnapshotContent objets

Un Kubernetes VolumeSnapshotContent Cet objet représente un instantané pris à partir d'un volume déjà provisionné. Il est analogue à un PersistentVolume et désigne un instantané provisionné sur le cluster de stockage. Similaire à PersistentVolumeClaim et PersistentVolume objets, lors de la création d'un instantané, les VolumeSnapshotContent l'objet maintient une correspondance un-à-un avec le VolumeSnapshot objet, qui avait demandé la création de l'instantané.

Le VolumeSnapshotContent L'objet contient des détails qui identifient de manière unique la capture d'écran, tels que le snapshotHandle . Ce snapshotHandle est une combinaison unique du nom du PV et du nom du VolumeSnapshotContent objet.

Lorsqu'une demande de snapshot arrive, Trident crée le snapshot sur le serveur. Une fois l'instantané créé, Trident configure un VolumeSnapshotContent objet et expose ainsi l'instantané à l'API Kubernetes.



En général, vous n'avez pas besoin de gérer le VolumeSnapshotContent objet. Une exception à cette règle est lorsque vous souhaitez ["importer un instantané de volume"](#) créé en dehors de Trident.

Kubernetes VolumeGroupSnapshotClass objets

Kubernetes VolumeGroupSnapshotClass les objets sont analogues à VolumeSnapshotClass . Ils permettent de définir plusieurs classes de stockage et sont référencés par les instantanés de groupes de volumes pour associer l'instantané à la classe d'instantané requise. Chaque instantané de groupe de volumes est associé à une seule classe d'instantané de groupe de volumes.

UN VolumeGroupSnapshotClass Un administrateur doit définir un groupe d'instantanés. Une classe d'instantané de groupe de volumes est créée avec la définition suivante :

```
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
```

Le `driver` indique à Kubernetes que les demandes de snapshots de groupes de volumes du `csi-group-snap-class` Les classes sont gérées par Trident. Le `deletionPolicy` Spécifie l'action à entreprendre lorsqu'un instantané de groupe doit être supprimé. Quand `deletionPolicy` est réglé sur `Delete`, les objets d'instantané du groupe de volumes ainsi que l'instantané sous-jacent sur le cluster de stockage sont supprimés lorsqu'un instantané est supprimé. Vous pouvez aussi le paramétrer sur `Retain` signifie que `VolumeGroupSnapshotContent` et l'instantané physique sont conservés.

Kubernetes VolumeGroupSnapshot objets

Un Kubernetes `VolumeGroupSnapshot` L'objet est une requête visant à créer un instantané de plusieurs volumes. De même qu'un PVC représente une demande faite par un utilisateur pour un volume, un instantané de groupe de volumes est une demande faite par un utilisateur pour créer un instantané d'un PVC existant.

Lorsqu'une demande de snapshot de groupe de volumes est reçue, Trident gère automatiquement la création du snapshot de groupe pour les volumes sur le serveur et expose le snapshot en créant un identifiant unique. `VolumeGroupSnapshotContent` objet. Vous pouvez créer des instantanés à partir de PVC existants et utiliser ces instantanés comme source de données lors de la création de nouveaux PVC.



Le cycle de vie d'un `VolumeGroupSnapshot` est indépendant du PVC source : un snapshot persiste même après la suppression du PVC source. Lors de la suppression d'un PVC auquel sont associés des instantanés, Trident marque le volume de support de ce PVC dans un état **Suppression en cours**, mais ne le supprime pas complètement. L'instantané du groupe de volumes est supprimé lorsque tous les instantanés associés sont supprimés.

Kubernetes VolumeGroupSnapshotContent objets

Un Kubernetes `VolumeGroupSnapshotContent` Cet objet représente un instantané de groupe pris à partir d'un volume déjà provisionné. Il est analogue à un `PersistentVolume` et désigne un instantané provisionné sur le cluster de stockage. Similaire à `PersistentVolumeClaim` et `PersistentVolume` objets, lors de la création d'un instantané, les `VolumeSnapshotContent` l'objet maintient une correspondance un-à-un avec le `VolumeSnapshot` objet, qui avait demandé la création de l'instantané.

Le `VolumeGroupSnapshotContent` L'objet contient des détails qui identifient le groupe d'instantanés, tels que le `volumeGroupSnapshotHandle` et les descripteurs `Snapshot` de volume individuels existants sur le système de stockage.

Lorsqu'une demande de snapshot arrive, Trident crée le snapshot du groupe de volumes sur le backend. Une fois l'instantané du groupe de volumes créé, Trident configure un `VolumeGroupSnapshotContent` objet et expose ainsi l'instantané à l'API Kubernetes.

Kubernetes CustomResourceDefinition objets

Les ressources personnalisées Kubernetes sont des points de terminaison de l'API Kubernetes définis par l'administrateur et utilisés pour regrouper des objets similaires. Kubernetes prend en charge la création de ressources personnalisées pour stocker une collection d'objets. Vous pouvez obtenir ces définitions de ressources en exécutant `kubectl get crds`.

Les définitions de ressources personnalisées (CRD) et leurs métadonnées d'objet associées sont stockées par Kubernetes dans son magasin de métadonnées. Cela élimine le besoin d'un magasin séparé pour Trident.

Trident utilise CustomResourceDefinition objets permettant de préserver l'identité des objets Trident, tels que les backends Trident, les classes de stockage Trident et les volumes Trident. Ces objets sont gérés par Trident. De plus, le cadre d'instantanés de volume CSI introduit certaines CRD nécessaires à la définition des instantanés de volume.

Les CRD sont une construction Kubernetes. Les objets des ressources définies ci-dessus sont créés par Trident. Par exemple, lorsqu'un backend est créé à l'aide de `tridentctl`, un correspondant `tridentbackends` L'objet CRD est créé pour être utilisé par Kubernetes.

Voici quelques points à retenir concernant les CRD de Trident :

- Lors de l'installation de Trident, un ensemble de CRD est créé et peut être utilisé comme n'importe quel autre type de ressource.
- Lors de la désinstallation de Trident à l'aide de `tridentctl uninstall` La commande supprime les pods Trident, mais ne nettoie pas les CRD créés. Se référer à "[Désinstaller Trident](#)" pour comprendre comment Trident peut être entièrement supprimé et reconfiguré à partir de zéro.

Trident StorageClass objets

Trident crée des classes de stockage adaptées à Kubernetes StorageClass objets qui spécifient `csi.trident.netapp.io` dans leur domaine d'approvisionnement. Le nom de la classe de stockage correspond à celui de Kubernetes StorageClass objet qu'il représente.



Avec Kubernetes, ces objets sont créés automatiquement lorsqu'un Kubernetes StorageClass qui utilise Trident comme fournisseur est enregistré.

Les classes de stockage comprennent un ensemble d'exigences relatives aux volumes. Trident compare ces exigences aux attributs présents dans chaque pool de stockage ; s'ils correspondent, ce pool de stockage est une cible valide pour le provisionnement de volumes utilisant cette classe de stockage.

Vous pouvez créer des configurations de classes de stockage pour définir directement les classes de stockage à l'aide de l'API REST. Cependant, pour les déploiements Kubernetes, nous nous attendons à ce qu'ils soient créés lors de l'enregistrement de nouveaux Kubernetes StorageClass objets.

Objets backend Trident

Les backends représentent les fournisseurs de stockage sur lesquels Trident provisionne les volumes ; une seule instance de Trident peut gérer un nombre quelconque de backends.



Il s'agit de l'un des deux types d'objets que vous créez et gérez vous-même. L'autre est Kubernetes StorageClass objet.

Pour plus d'informations sur la construction de ces objets, veuillez consulter "[configuration des backends](#)".

Trident StoragePool objets

Les pools de stockage représentent les emplacements distincts disponibles pour le provisionnement sur chaque backend. Pour ONTAP, cela correspond à des agrégats dans les SVM. Pour NetApp HCI/ SolidFire, cela correspond à des bandes QoS spécifiées par l'administrateur. Pour Cloud Volumes Service, cela correspond aux régions des fournisseurs de cloud. Chaque pool de stockage possède un ensemble d'attributs de stockage distincts, qui définissent ses caractéristiques de performance et de protection des données.

Contrairement aux autres objets présentés ici, les candidats au pool de stockage sont toujours détectés et gérés automatiquement.

Trident Volume objets

Les volumes constituent l'unité de base du provisionnement, comprenant des points de terminaison backend, tels que les partages NFS, et les LUN iSCSI et FC. Dans Kubernetes, ceux-ci correspondent directement à `PersistentVolumes`. Lorsque vous créez un volume, assurez-vous qu'il possède une classe de stockage, qui détermine où ce volume peut être provisionné, ainsi qu'une taille.



- Dans Kubernetes, ces objets sont gérés automatiquement. Vous pouvez les consulter pour voir ce que Trident a fourni.
- Lors de la suppression d'un PV avec des instantanés associés, le volume Trident correspondant est mis à jour à l'état **Suppression en cours**. Pour supprimer le volume Trident, vous devez supprimer les instantanés de ce volume.

Une configuration de volume définit les propriétés que doit posséder un volume provisionné.

Attribut	Type	Obligatoire	Description
version	chaîne	Non	Version de l'API Trident (« 1 »)
nom	chaîne	Oui	Nom du volume à créer
classe de stockage	chaîne	Oui	Classe de stockage à utiliser lors du provisionnement du volume
taille	chaîne	Oui	Taille du volume à provisionner en octets
protocole	chaîne	Non	Type de protocole à utiliser : « fichier » ou « bloc »
nom interne	chaîne	Non	Nom de l'objet sur le système de stockage ; généré par Trident
cloneSourceVolume	chaîne	Non	ontap (nas, san) et solidfire-* : Nom du volume à cloner à partir de

Attribut	Type	Obligatoire	Description
splitOnClone	chaîne	Non	ontap (nas, san) : Séparer le clone de son parent
instantanéPolitique	chaîne	Non	ontap-* : Stratégie d'instantané à utiliser
snapshotReserve	chaîne	Non	ontap-* : Pourcentage du volume réservé aux instantanés
politique d'exportation	chaîne	Non	ontap-nas* : Politique d'exportation à utiliser
répertoire snapshot	booléen	Non	ontap-nas* : Indique si le répertoire des instantanés est visible
Autorisations Unix	chaîne	Non	ontap-nas* : Permissions UNIX initiales
taille du bloc	chaîne	Non	solidfire-*: Taille du bloc/secteur
système de fichiers	chaîne	Non	Type de système de fichiers

Trident génère `internalName` lors de la création du volume. Cela se compose de deux étapes. Premièrement, il ajoute le préfixe de stockage (soit le préfixe par défaut). `trident` ou le préfixe dans la configuration du backend) au nom du volume, ce qui donne un nom de la forme `<prefix>-<volume-name>`. Il procède ensuite à la désinfection du nom, en remplaçant les caractères non autorisés par le système. Pour les backends ONTAP, il remplace les tirets par des traits de soulignement (ainsi, le nom interne devient `<prefix>_<volume-name>`). Pour les backends Element, il remplace les underscores par des tirets.

Vous pouvez utiliser des configurations de volume pour provisionner directement des volumes via l'API REST, mais dans les déploiements Kubernetes, nous prévoyons que la plupart des utilisateurs utiliseront la configuration standard de Kubernetes. `PersistentVolumeClaim` méthode. Trident crée automatiquement cet objet volume dans le cadre du processus de provisionnement.

Trident Snapshot objets

Les snapshots sont une copie à un instant précis des volumes, qui peuvent être utilisés pour provisionner de nouveaux volumes ou restaurer un état. Dans Kubernetes, ceux-ci correspondent directement à `VolumeSnapshotContent` objets. Chaque instantané est associé à un volume, qui est la source des données de cet instantané.

Chaque Snapshot L'objet comprend les propriétés énumérées ci-dessous :

Attribut	Type	Obligatoire	Description
version	Chaîne	Oui	Version de l'API Trident (« 1 »)
nom	Chaîne	Oui	Nom de l'objet instantané Trident

Attribut	Type	Obligatoire	Description
nom interne	Chaîne	Oui	Nom de l'objet snapshot Trident sur le système de stockage
nom_du_volume	Chaîne	Oui	Nom du volume persistant pour lequel l'instantané est créé
volumeInternalName	Chaîne	Oui	Nom de l'objet volume Trident associé sur le système de stockage



Dans Kubernetes, ces objets sont gérés automatiquement. Vous pouvez les consulter pour voir ce que Trident a fourni.

Lorsqu'un Kubernetes `VolumeSnapshot` Lorsqu'une requête d'objet est créée, Trident fonctionne en créant un objet instantané sur le système de stockage sous-jacent. Le `internalName` Cet objet instantané est généré en combinant le préfixe `snapshot-` avec le `UID` de la `VolumeSnapshot` objet (par exemple, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` et `volumeInternalName` sont remplies en obtenant les détails du volume de support.

Trident `ResourceQuota` objet

Le démon Trident dévore un `system-node-critical` Classe de priorité – la classe de priorité la plus élevée disponible dans Kubernetes – pour garantir que Trident puisse identifier et nettoyer les volumes lors de l'arrêt progressif des nœuds et permettre aux pods du daemonset Trident de préempter les charges de travail de priorité inférieure dans les clusters où la pression sur les ressources est élevée.

Pour ce faire, Trident utilise un `ResourceQuota` objet pour garantir qu'une classe de priorité « critique pour le nœud système » sur le daemonset Trident est satisfaite. Avant le déploiement et la création du `DaemonSet`, Trident recherche le `ResourceQuota` objet et, s'il n'est pas découvert, l'applique.

Si vous avez besoin de plus de contrôle sur le quota de ressources par défaut et la classe de priorité, vous pouvez générer un `custom.yaml` ou configurer le `ResourceQuota` objet utilisant le graphique Helm.

Voici un exemple d'objet `ResourceQuota` priorisant le daemonset Trident .

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
    matchExpressions:
      - operator: In
        scopeName: PriorityClass
        values:
          - system-node-critical
```

Pour plus d'informations sur les quotas de ressources, veuillez consulter "[Kubernetes : Quotas de ressources](#)".

Nettoyage ResourceQuota si l'installation échoue

Dans les rares cas où l'installation échoue après le ResourceQuota L'objet est créé, première tentative "[désinstallation](#)" puis réinstaller.

Si cela ne fonctionne pas, retirez manuellement le ResourceQuota objet.

Retirer ResourceQuota

Si vous préférez gérer vous-même l'allocation de vos ressources, vous pouvez retirer le Trident ResourceQuota objet utilisant la commande :

```
kubectl delete quota trident-csi -n trident
```

Normes de sécurité des modules (PSS) et contraintes de contexte de sécurité (SCC)

Les normes de sécurité des pods Kubernetes (PSS) et les politiques de sécurité des pods (PSP) définissent les niveaux d'autorisation et restreignent le comportement des pods. Les contraintes de contexte de sécurité OpenShift (SCC) définissent de la même manière les restrictions de pod spécifiques au moteur Kubernetes d'OpenShift. Pour permettre cette personnalisation, Trident active certaines autorisations lors de l'installation. Les sections suivantes détaillent les autorisations définies par Trident.



PSS remplace les politiques de sécurité des pods (PSP). PSP a été déprécié dans Kubernetes v1.21 et sera supprimé dans la v1.25. Pour plus d'informations, veuillez consulter "[Kubernetes : Sécurité](#)".

Contexte de sécurité Kubernetes requis et champs associés

Autorisation	Description
Privilégié	CSI exige que les points de montage soient bidirectionnels, ce qui signifie que le pod du nœud Trident doit exécuter un conteneur privilégié. Pour plus d'informations, veuillez consulter " Kubernetes : Propagation du montage ".
Réseau hôte	Requis pour le démon iSCSI. <code>iscsiadm</code> Gère les montages iSCSI et utilise le réseau hôte pour communiquer avec le démon iSCSI.
IPC de l'hôte	NFS utilise la communication interprocessus (IPC) pour communiquer avec le NFSD.
PID de l'hôte	Nécessaire pour démarrer <code>rpc-statd</code> pour NFS. Trident interroge les processus hôtes pour déterminer si <code>rpc-statd</code> s'exécute avant le montage des volumes NFS.
Capacités	Le <code>SYS_ADMIN</code> Cette fonctionnalité est fournie dans le cadre des fonctionnalités par défaut des conteneurs privilégiés. Par exemple, Docker définit les capacités suivantes pour les conteneurs privilégiés : <code>CapPrm: 0000003fffffffffff</code> <code>CapEff: 0000003fffffffffff</code>
Seccomp	Le profil Seccomp est toujours « non confiné » dans les conteneurs privilégiés ; par conséquent, il ne peut pas être activé dans Trident.
SELinux	Sur OpenShift, les conteneurs privilégiés sont exécutés dans le <code>spc_t</code> Le domaine (« Super Privileged Container ») et les conteneurs non privilégiés sont exécutés dans le domaine <code>container_t</code> domaine. Sur <code>containerd</code> , avec <code>container-selinux</code> installé, tous les conteneurs sont exécutés dans le <code>spc_t</code> domaine, ce qui désactive effectivement SELinux. Par conséquent, Trident n'ajoute pas <code>seLinuxOptions</code> aux conteneurs.
DAC	Les conteneurs privilégiés doivent être exécutés en tant que root. Les conteneurs non privilégiés s'exécutent en tant que root pour accéder aux sockets Unix requis par CSI.

Normes de sécurité des capsules (PSS)

Étiquette	Description	Défaut
pod-security.kubernetes.io/enforce-pod-security.kubernetes.io/enforce-version	Permet d'admettre le contrôleur Trident et les nœuds dans l'espace de noms d'installation. Ne modifiez pas l'étiquette de l'espace de noms.	enforce: privileged enforce-version: <version of the current cluster or highest version of PSS tested.>



Modifier les étiquettes d'espace de noms peut empêcher la planification des pods, entraîner une erreur lors de la création de : ... ou un avertissement : trident-csi-.... Si cela se produit, vérifiez si l'étiquette d'espace de noms pour `privileged` a été modifiée. Si c'est le cas, réinstallez Trident.

Politiques de sécurité des pods (PSP)

Champ	Description	Défaut
<code>allowPrivilegeEscalation</code>	Les conteneurs privilégiés doivent autoriser l'élévation de privilèges.	<code>true</code>
<code>allowedCSIDrivers</code>	Trident n'utilise pas de volumes éphémères CSI en ligne.	Vide
<code>allowedCapabilities</code>	Les conteneurs Trident non privilégiés ne nécessitent pas plus de capacités que l'ensemble par défaut, tandis que les conteneurs privilégiés se voient accorder toutes les capacités possibles.	Vide
<code>allowedFlexVolumes</code>	Trident n'utilise pas de "Pilote FlexVolume", par conséquent, ils ne sont pas inclus dans la liste des volumes autorisés.	Vide
<code>allowedHostPaths</code>	Le pod du nœud Trident monte le système de fichiers racine du nœud ; par conséquent, la configuration de cette liste ne présente aucun avantage.	Vide
<code>allowedProcMountTypes</code>	Trident n'utilise aucun <code>ProcMountTypes</code> .	Vide
<code>allowedUnsafeSysctls</code>	Trident ne nécessite aucune protection contre les risques. <code>sysctls</code> .	Vide
<code>defaultAddCapabilities</code>	Aucune fonctionnalité supplémentaire n'est requise pour les conteneurs privilégiés.	Vide
<code>defaultAllowPrivilegeEscalation</code>	L'autorisation d'élévation de privilèges est gérée dans chaque pod Trident.	<code>false</code>
<code>forbiddenSysctls</code>	Non <code>sysctls</code> sont autorisés.	Vide

Champ	Description	Défaut
<code>fsGroup</code>	Les conteneurs Trident s'exécutent en tant que root.	<code>RunAsAny</code>
<code>hostIPC</code>	Le montage de volumes NFS nécessite que l'IPC de l'hôte communique avec <code>nfsd</code>	<code>true</code>
<code>hostNetwork</code>	<code>iscsiadm</code> nécessite que le réseau hôte communique avec le démon iSCSI.	<code>true</code>
<code>hostPID</code>	Le PID de l'hôte est requis pour vérifier si <code>rpc-statd</code> est en cours d'exécution sur le nœud.	<code>true</code>
<code>hostPorts</code>	Trident n'utilise aucun port hôte.	Vide
<code>privileged</code>	Les pods de nœuds Trident doivent exécuter un conteneur privilégié pour pouvoir monter des volumes.	<code>true</code>
<code>readOnlyRootFilesystem</code>	Les pods de nœud Trident doivent écrire sur le système de fichiers du nœud.	<code>false</code>
<code>requiredDropCapabilities</code>	Les pods de nœuds Trident exécutent un conteneur privilégié et ne peuvent pas supprimer de fonctionnalités.	<code>none</code>
<code>runAsGroup</code>	Les conteneurs Trident s'exécutent en tant que root.	<code>RunAsAny</code>
<code>runAsUser</code>	Les conteneurs Trident s'exécutent en tant que root.	<code>runAsAny</code>
<code>runtimeClass</code>	Trident n'utilise pas <code>RuntimeClasses</code> .	Vide
<code>seLinux</code>	Trident ne définit pas <code>seLinuxOptions</code> car il existe actuellement des différences dans la manière dont les environnements d'exécution de conteneurs et les distributions Kubernetes gèrent SELinux.	Vide
<code>supplementalGroups</code>	Les conteneurs Trident s'exécutent en tant que root.	<code>RunAsAny</code>
<code>volumes</code>	Les modules Trident nécessitent ces plug-ins de volume.	<code>hostPath</code> , <code>projected</code> , <code>emptyDir</code>

Contraintes de contexte de sécurité (CCS)

Étiquettes	Description	Défaut
<code>allowHostDirVolumePlugin</code>	Les pods de nœud Trident montent le système de fichiers racine du nœud.	<code>true</code>
<code>allowHostIPC</code>	Le montage de volumes NFS nécessite que l'IPC de l'hôte communique avec <code>nfsd</code> .	<code>true</code>
<code>allowHostNetwork</code>	<code>iscsiadm</code> nécessite que le réseau hôte communique avec le démon iSCSI.	<code>true</code>
<code>allowHostPID</code>	Le PID de l'hôte est requis pour vérifier si <code>rpc-statd</code> est en cours d'exécution sur le nœud.	<code>true</code>
<code>allowHostPorts</code>	Trident n'utilise aucun port hôte.	<code>false</code>
<code>allowPrivilegeEscalation</code>	Les conteneurs privilégiés doivent autoriser l'élévation de privilèges.	<code>true</code>
<code>allowPrivilegedContainer</code>	Les pods de nœuds Trident doivent exécuter un conteneur privilégié pour pouvoir monter des volumes.	<code>true</code>
<code>allowedUnsafeSysctls</code>	Trident ne nécessite aucune protection contre les risques. <code>sysctls</code> .	<code>none</code>
<code>allowedCapabilities</code>	Les conteneurs Trident non privilégiés ne nécessitent pas plus de capacités que l'ensemble par défaut, tandis que les conteneurs privilégiés se voient accorder toutes les capacités possibles.	Vide
<code>defaultAddCapabilities</code>	Aucune fonctionnalité supplémentaire n'est requise pour les conteneurs privilégiés.	Vide
<code>fsGroup</code>	Les conteneurs Trident s'exécutent en tant que <code>root</code> .	<code>RunAsAny</code>
<code>groups</code>	Ce SCC est spécifique à Trident et est lié à son utilisateur.	Vide
<code>readOnlyRootFilesystem</code>	Les pods de nœud Trident doivent écrire sur le système de fichiers du nœud.	<code>false</code>
<code>requiredDropCapabilities</code>	Les pods de nœuds Trident exécutent un conteneur privilégié et ne peuvent pas supprimer de fonctionnalités.	<code>none</code>
<code>runAsUser</code>	Les conteneurs Trident s'exécutent en tant que <code>root</code> .	<code>RunAsAny</code>

Étiquettes	Description	Défaut
<code>seLinuxContext</code>	Trident ne définit pas <code>seLinuxOptions</code> car il existe actuellement des différences dans la manière dont les environnements d'exécution de conteneurs et les distributions Kubernetes gèrent SELinux.	Vide
<code>seccompProfiles</code>	Les conteneurs privilégiés s'exécutent toujours en mode « non confiné ».	Vide
<code>supplementalGroups</code>	Les conteneurs Trident s'exécutent en tant que root.	<code>RunAsAny</code>
<code>users</code>	Une entrée est prévue pour lier ce SCC à l'utilisateur Trident dans l'espace de noms Trident .	n / A
<code>volumes</code>	Les modules Trident nécessitent ces plug-ins de volume.	<code>hostPath</code> , <code>downwardAPI</code> , <code>projected</code> , <code>emptyDir</code>

Mentions légales

Les mentions légales donnent accès aux déclarations de droits d'auteur, aux marques déposées, aux brevets et bien plus encore.

Copyright

["https://www.netapp.com/company/legal/copyright/"](https://www.netapp.com/company/legal/copyright/)

Marques de commerce

NETAPP, le logo NETAPP et les marques répertoriées sur la page Marques NetApp sont des marques commerciales de NetApp, Inc. Les autres noms de sociétés et de produits peuvent être des marques commerciales de leurs propriétaires respectifs.

["https://www.netapp.com/company/legal/trademarks/"](https://www.netapp.com/company/legal/trademarks/)

Brevets

Une liste actuelle des brevets détenus par NetApp est disponible à l'adresse suivante :

<https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf>

Politique de confidentialité

["https://www.netapp.com/company/legal/privacy-policy/"](https://www.netapp.com/company/legal/privacy-policy/)

Open source

Vous pouvez consulter les droits d'auteur et les licences de tiers utilisés dans le logiciel NetApp pour Trident dans le fichier de notices de chaque version à l'adresse suivante : <https://github.com/NetApp/trident/> .

Informations sur le copyright

Copyright © 2026 NetApp, Inc. Tous droits réservés. Imprimé aux États-Unis. Aucune partie de ce document protégé par copyright ne peut être reproduite sous quelque forme que ce soit ou selon quelque méthode que ce soit (graphique, électronique ou mécanique, notamment par photocopie, enregistrement ou stockage dans un système de récupération électronique) sans l'autorisation écrite préalable du détenteur du droit de copyright.

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

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

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

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

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

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

Informations sur les marques commerciales

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