Documentation d'Astra Trident 24.02

Astra Trident

NetApp
April 18, 2024

Sommaire

Documentation d’Astra Trident 24.02

Notes de mise à jour

Quoi de neuf

Dans les notes de version, vous trouverez des informations sur les nouvelles fonctionnalités, les améliorations et les correctifs de bogues de la dernière version d'Astra Trident.

> Le `tridentctl` Binaire pour Linux fourni dans le fichier zip du programme 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 pas testée ou prise en charge.

Nouveautés de la version 24.02

Améliorations

- Prise en charge supplémentaire de Cloud Identity.
  - AKS avec ANF : Azure Workload Identity sera utilisé comme identité cloud.
  - EKS avec FSxN : le rôle IAM AWS sera utilisé comme identité cloud.
- Ajout de la prise en charge de l'installation d'Astra Trident en tant que module complémentaire sur le cluster EKS depuis la console EKS.
- Ajout de la possibilité de configurer et de désactiver l'auto-rétablissement iSCSI ("Question no 864").
- Ajout de la personnalité FSX aux pilotes ONTAP pour permettre l'intégration avec AWS IAM et SecretsManager et pour permettre à Astra Trident de supprimer des volumes FSX avec des sauvegardes ("Question no 453").

Kubernetes

- Prise en charge de Kubernetes 1.29.

Correctifs

- Correction des messages d'avertissement ACP lorsque ACP n'est pas activé ("Question no 866").
- Ajout d'un délai de 10 secondes avant d'effectuer une répartition des clones lors de la suppression d'un snapshot pour les pilotes ONTAP, lorsqu'un clone est associé au snapshot.

Dérations

- Suppression de l'infrastructure d'attepostes in-to des manifestes d'images multi-plates-formes.

Changements en 23.10

Correctifs

- Extension de volume fixe si la nouvelle taille demandée est inférieure à la taille totale des volumes pour les pilotes de stockage ontap-nas et ontap-nas-flexgroup ("Question no 834").
- Taille de volume fixe pour afficher uniquement la taille utilisable du volume lors de l'importation pour les pilotes de stockage ontap-nas et ontap-nas-flexgroup ("Question no 722").

- Conversion de noms FlexVol fixes pour ONTAP-NAS-Economy.
- Correction du problème d'initialisation d'Astra Trident sur un nœud Windows lors du redémarrage du nœud.

Améliorations

Kubernetes

Prise en charge de Kubernetes 1.28.

Astra Trident

- Ajout de la prise en charge de l'utilisation d'ami (Azure Managed identités) avec le pilote de stockage Azure-netapp-Files.
- Ajout de la prise en charge de NVMe over TCP pour le pilote ONTAP-SAN
- Ajout de la possibilité de suspendre le provisionnement d'un volume lorsque le back-end est défini sur suspendu par l'utilisateur ("Question no 558").

Fonctionnalités avancées disponibles dans Astra Control

Avec Astra Trident 23.10, un nouveau composant logiciel appelé Astra Control Provisioner est disponible pour les utilisateurs d'Astra Control sous licence. Ce mécanisme de provisionnement permet d'accéder à un ensemble de fonctionnalités avancées de gestion et de provisionnement du stockage qui vont au-delà de celles qu'Astra Trident prend en charge par elle-même. Pour la version 23.10, ces fonctionnalités comprennent :

- Fonctionnalités de sauvegarde et de restauration pour les applications avec des systèmes back-end de stockage ontap-nas économiques basés sur des pilotes
- Sécurité améliorée du back-end de stockage avec le chiffrement Kerberos 5
- Restauration des données à l'aide d'un snapshot
- Améliorations de SnapMirror

"En savoir plus sur Astra Control Provisioner."

Changements en 23.07.1

Kubernetes: Suppression fixe du démonset pour prendre en charge les mises à niveau sans temps d'arrêt ("Question no 740").

Changements en 23.07

Correctifs

Kubernetes

- Correction de la mise à niveau de Trident pour ignorer les anciens pods bloqués en état de terminaison ("Question no 740").
- Ajout d'une tolérance à la définition de « passagent-trident-version-pod » ("Question no 795").

Astra Trident

- Correction des demandes ZAPI ONTAP pour s'assurer que les numéros de série des LUN sont interrogés lors de l'obtention des attributs de LUN pour identifier et corriger les périphériques iSCSI fantômes pendant les opérations de stadification des nœuds.
- Correction de la gestion des erreurs dans le code du pilote de stockage ("Question no 816").
- Redimensionnement des quotas fixes lors de l'utilisation de pilotes ONTAP avec use-REST=true.
- Création de clones LUN fixes dans ontap-san-Economy.
- Annuler la publication du champ d'informations de `rawDevicePath` à `devicePath`; logique ajoutée pour remplir et récupérer (dans certains cas) `devicePath` légale.

Améliorations

Kubernetes

- Prise en charge supplémentaire de l'importation de snapshots préprovisionnés.
- Déploiement réduit et autorisations linux diaboconfigurées ("Question no 817").

Astra Trident

- Ne rapporte plus le champ d'état pour les volumes et les snapshots « en ligne ».
- Met à jour l'état du back-end si le back-end ONTAP est hors ligne ("Questions #801", "#543").
- Le numéro de série de la LUN est toujours récupéré et publié au cours du workflow ControllerVolumePublish.
- Ajout d'une logique supplémentaire pour vérifier le numéro de série et la taille du périphérique iSCSI à chemins d'accès multiples.
- Vérification supplémentaire des volumes iSCSI pour s'assurer que le périphérique multiacheminement correct n'est pas mis en place.

Amélioration expérimentale

Ajout de la prise en charge de la présentation technique de NVMe over TCP pour le pilote ONTAP-SAN.

Documentation

De nombreuses améliorations de l'organisation et du formatage ont été apportées.

Dérations

Kubernetes

- Suppression de la prise en charge des snapshots v1beta1.
- Suppression de la prise en charge des volumes et des classes de stockage pré-CSI.
- Mise à jour de la version 1.22 de Kubernetes minimale prise en charge.

Changements en 23.04

> Forcer le détachement de volume pour les volumes ONTAP-SAN-* est uniquement pris en charge avec les versions Kubernetes avec le volet fonctionnalité de fermeture de nœud non gracieuse activé. Le détachement forcé doit être activé au moment de l'installation à l'aide du `--enable-force-detach` Indicateur du programme d'installation Trident.

Correctifs

- Correction de l'opérateur Trident pour utiliser IPv6 localhost pour l'installation lorsqu'il est spécifié dans spec.
- Correction des autorisations de rôle de cluster de l'opérateur Trident pour qu'elles soient synchronisées avec les autorisations du bundle ("Question no 799").
- Résolution du problème de connexion d'un volume de bloc brut sur plusieurs nœuds en mode RWX.
- Prise en charge du clonage FlexGroup fixe et importation de volumes pour les volumes SMB.
- Résolution du problème où le contrôleur Trident n'a pas pu s'arrêter immédiatement ("Question no 811").
- Correctif ajouté pour afficher la liste de tous les noms de groupes initiateur associés à une LUN spécifiée provisionnée avec des pilotes ontap-san-*.
- Ajout d'un correctif pour permettre l'exécution des processus externes.
- Erreur de compilation corrigée pour l'architecture s390 ("Question no 537").
- Correction d'un niveau de journalisation incorrect lors des opérations de montage de volume ("Question no 781").
- Correction de l'erreur d'assertion de type de potentiel ("Question no 802").

Améliorations

- Kubernetes :
  - Prise en charge de Kubernetes 1.27.
  - Ajout de la prise en charge de l'importation de 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 de nœud non gracieuse.
  - Tous les volumes ONTAP-SAN-* utiliseront désormais les groupes initiateurs par nœud. Les LUN ne seront mappées qu'aux igroups dont la publication est active sur ces nœuds afin d'améliorer notre niveau de sécurité. Les volumes existants seront basculés de manière opportuniste vers le nouveau schéma d'igroup lorsque Trident détermine qu'il est possible de le faire sans incidence sur les workloads actifs ("Question no 758").
  - Amélioration de la sécurité de Trident en nettoyant les groupes initiateurs gérés par Trident non utilisés à partir de systèmes back-end 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 ("Question no 732").
- La procédure d'arrêt de Trident a été améliorée en désactivant d'abord les serveurs d'API ("Question no 811").

- Ajout de la prise en charge de la construction multi plate-forme pour les hôtes Windows et arm64 à Makefile ; voir BUILD.md.

Dérations

Kubernetes: les igroups Backend-scoped ne seront plus créés lors de la configuration de pilotes ontap-san et ontap-san-Economy ("Question no 758").

Changements en 23.01.1

Correctifs

- Correction de l'opérateur Trident pour utiliser IPv6 localhost pour l'installation lorsqu'il est spécifié dans spec.
- Correction des autorisations de rôle de cluster opérateur Trident synchronisées avec les autorisations de bundle "Question no 799".
- Ajout d'un correctif pour permettre l'exécution des processus externes.
- Résolution du problème de connexion d'un volume de bloc brut sur plusieurs nœuds en mode RWX.
- Prise en charge du clonage FlexGroup fixe et importation de volumes pour les volumes SMB.

Changements en 23.01

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

Correctifs

- Kubernetes : ajout d'options pour exclure la création de règles de sécurité du Pod pour réparer les installations Trident via Helm ("Questions #783, #794").

Améliorations

Kubernetes

- Prise en charge ajoutée de Kubernetes 1.26.
- Amélioration de l'utilisation globale des ressources RBAC Trident ("Numéro 757").
- Automatisation ajoutée 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 identifiants pour les volumes chiffrés LUKS.

Astra Trident

- Ajout de la prise en charge des volumes SMB avec Amazon FSX pour 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 le niveau de service CVS.
- Ajout de la prise en charge de l'utilisation facultative de flexgroupAgrégateList lors de la création de FlexGroups avec le pilote de stockage ontap-nas-flexgroup.

- Amélioration des performances du pilote de stockage économique ontap-nas lors de la gestion de plusieurs volumes FlexVol.
- Mises à jour des donnéesLIF activées pour tous les pilotes de stockage NAS de ONTAP.
- Mise à jour de la convention de nommage Trident Deployment and DemonSet afin de refléter le système d'exploitation du nœud hôte.

Dérations

- Kubernetes : mise à jour de Kubernetes minimale prise en charge vers la version 1.21.
- Les LIFs de données ne doivent plus être spécifiées lors de la configuration `ontap-san` ou `ontap-san-economy` pilotes.

Changements en 22.10

Vous devez lire les informations essentielles suivantes avant de passer à Astra Trident 22.10.

, la protection des données essentielles d'Astra Trident 22.10

- Kubernetes 1.25 est désormais pris en charge par Trident. Vous devez mettre à niveau Astra Trident vers 22.10 avant de procéder à la mise à niveau vers Kubernetes 1.25.
- Astra Trident applique désormais rigoureusement la configuration des chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Correctifs

- Problème spécifique au système ONTAP back-end créé à l'aide de `credentials` le champ ne s'est pas connecté pendant la mise à niveau 22.07.0 ("Numéro 759").
- **Docker:** correction d'un problème entraînant l'échec du démarrage du plug-in de volume Docker dans certains environnements ("Numéro 548" et "Numéro 760").
- Résolution du problème SLM spécifique aux systèmes back-end ONTAP pour garantir que seul un sous-ensemble de LIF de données appartenant aux nœuds de reporting est publié.
- Problème de performances résolu lors de la connexion d'un volume à des analyses inutiles des LUN iSCSI.
- Supprimez les tentatives granulaires dans le workflow iSCSI Astra Trident pour connaître un risque d'échec rapide et réduire les intervalles de tentatives externes.
- Résolution du problème lorsqu'une erreur a été renvoyée lors du vidage d'un périphérique iSCSI lorsque le périphérique multivoie correspondant a déjà été rincé.

Améliorations

- Kubernetes :
  - Prise en charge ajoutée de Kubernetes 1.25. Vous devez mettre à niveau Astra Trident vers 22.10 avant de procéder à la mise à niveau vers Kubernetes 1.25.

    - Ajout d'un ServiceAccount, ClusterRole et ClusterRoleBinding distincts pour Trident Deployment et DemonSet afin de permettre des améliorations futures des autorisations.
    - Prise en charge ajoutée de "partage de volume entre espaces de noms".
- Tout Trident `ontap-*` Les pilotes de stockage fonctionnent désormais avec l'API REST de ONTAP.
- Ajout d'un nouvel opérateur yaml (`bundle_post_1_25.yaml`) sans a. `PodSecurityPolicy` Pour la prise en charge de Kubernetes 1.25.
- Ajouté "Prise en charge des volumes LUKS-chiffrés" pour `ontap-san` et `ontap-san-economy` lecteurs de stockage
- Ajout de la prise en charge des nœuds Windows Server 2019.
- Ajouté "Prise en charge des volumes SMB sur les nœuds Windows" grâce au `azure-netapp-files` pilote de stockage
- La détection automatique du basculement MetroCluster pour les pilotes ONTAP est désormais disponible dans l'ensemble.

Dérations

- **Kubernetes:** mise à jour du nombre minimum de Kubernetes pris en charge vers 1.20.
- Suppression du pilote ADS (Data Store).
- Retrait du support pour `yes` et `smart` options pour `find_multipaths` Lors de la configuration des chemins d'accès multiples du nœud de travail pour iSCSI.

Changements en 22.07

Correctifs

Kubernetes

- Problème résolu pour gérer les valeurs booléennes et nombres pour le sélecteur de nœud lors de la configuration de Trident avec Helm ou l'opérateur Trident. ("Problème GitHub n° 700")
- Résolution du problème lors de la gestion des erreurs provenant d'un chemin non CHAP, de sorte que kubelet réessaie en cas d'échec. "Problème GitHub n° 736")

Améliorations

- Passer de k8s.gcr.io au registre.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 groupes initiateurs, tout en les ayant été publiés activement à ces nœuds pour améliorer notre sécurité. Les volumes existants sont basculés de manière opportuniste vers le nouveau modèle d'igroup lorsque Astra Trident détermine qu'il est possible de le faire en toute sécurité sans incidence sur les workloads actifs.
- Inclus un quota de Resourcequota avec les installations Trident pour s'assurer que Trident DemonSet est planifié lorsque la consommation PriorityClass est limitée par défaut.
- Ajout de la prise en charge des fonctions réseau au pilote Azure NetApp Files. ("Problème GitHub n° 717")
- Ajout de la détection automatique du basculement MetroCluster dans l'aperçu technique aux pilotes ONTAP. ("Problème GitHub n° 228")

Dérations

- **Kubernetes:** mise à jour du nombre minimum de Kubernetes pris en charge vers 1.19.
- La configuration backend n'autorise plus plusieurs types d'authentification dans la configuration unique.

Suppressions

- Le pilote CVS AWS (obsolète depuis 22.04) a été supprimé.
- Kubernetes
  - Suppression des capacités SYS_ADMIN inutiles des modules de nœud.
  - Réduit la préparation des nœuds afin de simplifier les informations sur l'hôte et la détection des services actifs pour obtenir la confirmation de la disponibilité des services NFS/iSCSI sur les nœuds workers.

Documentation

Une nouvelle "Normes de sécurité du pod" (PSS) a été ajouté en détail les autorisations activées par Astra Trident lors de l'installation.

Changements en 22.04

NetApp améliore et améliore continuellement ses produits et services. Voici quelques-unes des nouveautés d'Astra Trident. Pour les versions précédentes, reportez-vous à la section "Versions antérieures de la documentation".

> Si vous effectuez une mise à niveau à partir d'une version précédente de Trident et que vous utilisez Azure NetApp Files, le `location` le paramètre config est désormais un champ singleton obligatoire.

Correctifs

- Amélioration de l'analyse des noms d'initiateurs iSCSI. ("Problème GitHub n° 681")
- Problème résolu lorsque les paramètres de classe de stockage CSI n'étaient pas autorisés. ("Problème GitHub n° 598")
- Déclaration de clé en double fixe dans Trident CRD. ("Problème GitHub n° 671")
- Correction des journaux CSI instantanés erronés. ("Problème GitHub n° 629"))
- Résolution du problème lié à l'annulation de la publication des volumes sur les nœuds supprimés. ("Problème GitHub n° 691")
- Ajout de la gestion des incohérences du système de fichiers sur les périphériques en bloc. ("Problème GitHub n° 656")
- Problème résolu extraction automatique des images lors de la configuration du `imageRegistry` indicateur pendant l'installation. ("Problème GitHub n° 715")
- Résolution du problème d'échec du clonage d'un volume avec plusieurs règles d'exportation par le pilote Azure NetApp Files.

Améliorations

- Les connexions entrantes aux terminaux sécurisés de Trident requièrent désormais un minimum de TLS 1.3. ("Problème GitHub n° 698")

- Trident ajoute désormais des en-têtes HSTS aux réponses à partir de ses terminaux sécurisés.
- Trident tente désormais d'activer automatiquement la fonctionnalité d'autorisations unix Azure NetApp Files.
- **Kubernetes**: Trident demonset s'exécute maintenant dans la classe de priorité critique du nœud système. ("Problème GitHub n° 694")

Suppressions

Le pilote E-Series (désactivé depuis 20.07) a été supprimé.

Changements en 22.01.1

Correctifs

- Résolution du problème lié à l'annulation de la publication des volumes sur les nœuds supprimés. ("Problème GitHub n° 691")
- Panique fixe lors de l'accès aux champs nuls pour l'espace global dans les réponses de l'API ONTAP.

Changements en 22.01.0

Correctifs

- **Kubernetes:** augmentez le temps de rétentative de rétro-enregistrement des nœuds pour les grands clusters.
- Problème résolu dans lequel le pilote Azure-netapp-Files pourrait être confondu avec plusieurs ressources avec le même nom.
- Les LIF de données sur IPv6 SAN de ONTAP fonctionnent désormais si elles sont spécifiées avec des parenthèses.
- Problème résolu lors de la tentative d'importation d'un volume déjà importé renvoie EOF laissant le PVC à l'état en attente. ("Problème GitHub n° 489")
- Problème résolu lorsque la performance d'Astra Trident ralentit lorsque plus de 32 snapshots sont créés sur un volume SolidFire.
- SHA-1 remplacé par SHA-256 lors de la création du certificat SSL.
- Correction du pilote Azure NetApp Files pour permettre la duplication des noms de ressources et limiter les opérations à un seul emplacement.
- Correction du pilote Azure NetApp Files pour permettre la duplication des noms de ressources et limiter les opérations à un seul emplacement.

Améliorations

- Améliorations de Kubernetes :
  - Prise en charge ajoutée de Kubernetes 1.23.
  - Ajoutez des options de planification pour les pods Trident lorsqu'ils sont installés via l'opérateur Trident ou Helm. ("Problème GitHub n° 651")
- Autorisation des volumes inter-régions dans le pilote GCP ("Problème GitHub n° 633")
- Ajout de la prise en charge de l'option 'unixPermissionss' aux volumes Azure NetApp Files. ("Problème GitHub n° 666")

Dérations

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

Changements en 21.10.1

> La version v21.10.0 présente un problème qui peut placer le contrôleur Trident dans un état CrashLoopBackOff lorsqu'un nœud est supprimé, puis réintégré au cluster Kubernetes. Ce problème a été résolu dans la version 1.210.1 (édition GitHub 669).

Correctifs

- Condition de race potentielle fixe lors de l'importation d'un volume sur un back-end Cloud CVS GCP, entraînant l'échec de l'importation.
- Résolution d'un problème de mise en service du contrôleur Trident dans un état CashLoopBackOff lorsqu'un nœud est retiré, puis réintégré au cluster Kubernetes (problème GitHub 669).
- Problème résolu : les SVM n'ont plus été découverts si aucun nom de SVM n'a été spécifié (problème GitHub 612).

Changements en 21.10.0

Correctifs

- Problème résolu : les clones de volumes XFS n'ont pas pu être montés sur le même nœud que le volume source (problème GitHub 514).
- Résolution du problème pendant lequel Astra Trident a enregistré une erreur fatale lors de l'arrêt (problème GitHub 597).
- Correctifs liés à Kubernetes :
  - Renvoyer l'espace utilisé d'un volume comme taille de restauration minimale lors de la création de snapshots avec `ontap-nas` et `ontap-nas-flexgroup` Pilotes (problème GitHub 645).
  - Résolution du problème où `Failed to expand filesystem` Une erreur a été consignée après le redimensionnement du volume (problème GitHub 560).
  - Résolution du problème de blocage d'un module `Terminating` État (problème GitHub 572).
  - A résolu le cas où un `ontap-san-economy` FlexVol peut contenir des LUN de snapshot (GitHub : édition 533).
  - Résolution du problème d'installation YAML personnalisé avec une image différente (problème GitHub 613).
  - Calcul de la taille de snapshot fixe (problème GitHub 611).
  - Problème résolu : tous les installateurs Trident d'Astra pouvaient identifier Kubernetes ordinaire comme OpenShift (problème GitHub 639).
  - A corrigé l'opérateur Trident pour arrêter la réconciliation si le serveur d'API Kubernetes est inaccessible (problème GitHub 599).

Améliorations

- Prise en charge ajoutée de `unixPermissions` Option pour les volumes de performance GCP-CVS.
- Ajout de la prise en charge des volumes CVS optimisés pour l'évolutivité dans GCP dans la plage de 600

Gio à 1 Tio.

- Améliorations liées à Kubernetes :
  - Prise en charge ajoutée de Kubernetes 1.22.
  - Compatibilité de l'opérateur Trident et du tableau Helm avec Kubernetes 1.22 (problème GitHub 628).
  - Ajout d'une image opérateur à `tridentctl` Commande images (problème GitHub 570).

Améliorations expérimentales

- Ajout de la prise en charge de la réplication de volume dans `ontap-san` conducteur.
- Ajout de la prise en charge de REST * TECH Preview* pour `ontap-nas-flexgroup`, `ontap-san`, et `ontap-nas-economy` pilotes.

Problèmes connus

Les problèmes connus identifient les problèmes susceptibles de vous empêcher d'utiliser le produit avec succès.

- Lorsque vous mettez à niveau un cluster Kubernetes de 1.24 vers 1.25 ou version ultérieure sur lequel Astra Trident est installé, vous devez mettre à jour les valeurs.yaml pour les définir `excludePodSecurityPolicy` à `true` ou ajouter `--set excludePodSecurityPolicy=true` à la `helm upgrade` commande avant de pouvoir mettre à niveau le cluster.
- L'ASTRA Trident applique maintenant une blanc `fsType` (`fsType=""`) pour les volumes qui n'ont pas le `fsType` Spécifiés dans leur classe de stockage. Lorsque vous utilisez Kubernetes 1.17 ou version ultérieure, Trident prend en charge l'option vide `fsType` Pour les volumes NFS. Pour les volumes iSCSI, vous devez définir le `fsType` Sur votre classe de stockage lors de l'application d'un `fsGroup` Utilisation d'un contexte de sécurité.
- Lors de l'utilisation d'un système back-end pour plusieurs instances Trident d'Astra, chaque fichier de configuration back-end doit avoir un fichier différent `storagePrefix` Valeur pour les systèmes ONTAP back-end ou différente `TenantName` Pour les systèmes SolidFire back-end. Astra Trident ne peut pas détecter les volumes que d'autres instances d'Astra Trident ont créés. Tentative de création d'un volume existant sur un système back-end ONTAP ou SolidFire réussie, Astra Trident traite la création de volume comme une opération identente. Si `storagePrefix` ou `TenantName` n'en diffère pas, il peut y avoir des collisions de noms pour les volumes créés sur le même back-end.
- Lors de l'installation d'Astra Trident (à l'aide de `tridentctl` Ou l'opérateur Trident) et à l'aide de `tridentctl` Pour gérer Astra Trident, vous devez vous assurer que `KUBECONFIG` la variable d'environnement est définie. Cela est nécessaire pour indiquer le cluster Kubernetes `tridentctl` doit travailler contre. Lorsque vous utilisez plusieurs environnements Kubernetes, assurez-vous que `KUBECONFIG` le fichier est fourni avec précision.
- Pour réclamer de l'espace en ligne pour des volumes persistants iSCSI, le système d'exploitation sous-jacent du nœud worker peut nécessiter le passage des options de montage vers le volume. Ceci est vrai pour les instances RHEL/RedHat CoreOS qui requièrent le `discard` "option de montage"; Assurez-vous que le mountOption de mise au rebut est inclus dans votre[`StorageClass`^] pour prendre en charge le blocage en ligne.
- Si vous possédez plusieurs instances d'Astra Trident par cluster Kubernetes, Astra Trident ne peut pas communiquer avec d'autres instances et ne peut pas détecter les autres volumes qu'ils ont créés, ce qui entraîne un comportement inattendu et incorrect si plusieurs instances s'exécutent dans un cluster. Il ne devrait y avoir qu'une seule instance d'Astra Trident par cluster Kubernetes.
- Avec Astra Trident `StorageClass` Les objets sont supprimés de Kubernetes alors que Astra Trident est

hors ligne, Astra Trident ne supprime pas les classes de stockage correspondantes de la base de données lorsqu'elle est remise en ligne. Vous devez supprimer ces classes de stockage à l'aide de `tridentctl` Ou l'API REST.

- Si un utilisateur supprime un volume persistant provisionné par Astra Trident avant de supprimer le volume persistant correspondant, Astra Trident ne supprime pas automatiquement le volume de sauvegarde. Vous devez supprimer le volume via `tridentctl` Ou l'API REST.
- ONTAP ne peut pas provisionner simultanément plusieurs FlexGroup, sauf si l'ensemble d'agrégats est unique pour chaque demande de provisionnement.
- Lorsque vous utilisez Astra Trident sur IPv6, vous devez préciser `managementLIF` et `dataLIF` dans la définition du back-end entre crochets. Par exemple : `[fd20:8b1e:b258:2000:f816:3eff:feec:0]`.

> Vous ne pouvez pas spécifier `dataLIF` Sur un SAN backend ONTAP. Astra Trident détecte toutes les LIF iSCSI disponibles et les utilise pour établir la session multivoie.

- 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.

Trouvez plus d'informations

- "Astra Trident GitHub"
- "Blogs Trident d'Astra"

Versions antérieures de la documentation

Si vous n'exécutez pas Astra Trident 24.02, la documentation relative aux versions précédentes est disponible sur la base du "Cycle de vie de prise en charge d'Astra Trident".

- "ASTRA Trident 23.10"
- "ASTRA Trident 23.07"
- "ASTRA Trident 23.04"
- "ASTRA Trident 23.01"
- "Astra Trident 22.10"
- "Astra Trident 22.07"
- "Astra Trident 22.04"
- "Astra Trident 22.01"
- "Astra Trident 21.10"
- "Astra Trident 21.07"

Commencez

Découvrez Astra Trident

Découvrez Astra Trident

ASTRA Trident est un projet open source entièrement pris en charge et géré par NetApp dans le cadre du "Gamme de produits Astra". Il a été conçu pour vous aider à répondre aux exigences de persistance de vos applications conteneurisées en utilisant des interfaces standard telles que l'interface CSI (Container Storage interface).

Qu'est-ce qu'Astra ?

Astra facilite la gestion, la protection et le déplacement de leurs workloads riches en données qui s'exécutent sur Kubernetes, dans les clouds publics et sur site.

ASTRA provisionne et fournit un stockage persistant de conteneur basé sur Astra Trident. Il offre également des fonctionnalités avancées de gestion des données compatible avec les applications, telles que les copies Snapshot, la sauvegarde et la restauration, les journaux d'activité et le clonage actif pour la protection des données, la reprise d'activité et les données, l'audit des données et la migration pour les workloads Kubernetes.

En savoir plus sur "ASTRA ou inscrivez-vous pour un essai gratuit".

Qu'est-ce qu'Astra Trident ?

ASTRA 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 ONTAP (AFF, FAS, Select, Cloud, Amazon FSX pour NetApp ONTAP), Element (NetApp HCI, SolidFire), Azure NetApp Files service et Cloud Volumes Service sur Google Cloud.

ASTRA Trident est un orchestrateur de stockage dynamique conforme à CSI (Container Storage interface) qui s'intègre de manière native "Kubernetes". ASTRA Trident s'exécute en tant que pod unique de contrôleur et comme pod de nœud sur chaque nœud worker dans le cluster. Reportez-vous à la section "Architecture d'Astra Trident" pour plus d'informations.

ASTRA Trident assure également une intégration directe avec l'écosystème Docker pour les plateformes de stockage NetApp. Le plug-in de volume Docker (nDVP) de NetApp prend en charge le provisionnement et la gestion des ressources de stockage depuis la plateforme de stockage jusqu'aux hôtes Docker. Reportez-vous à la section "Déployez Astra Trident pour Docker" pour plus d'informations.

> Si vous utilisez Kubernetes pour la première fois, familiarisez-vous avec le "Concepts et outils Kubernetes".

Testez Astra Trident

Pour tester la solution, demandez l'accès au « déploiement et clonage simplifiés du stockage persistant pour les workloads conteneurisés ». "Test Drive NetApp" utilisation d'une image de laboratoire prête à l'emploi. Le test Drive fournit un environnement sandbox avec un cluster Kubernetes à trois nœuds et Astra Trident installé et configuré. C'est un excellent moyen de vous familiariser avec Astra Trident et d'explorer ses fonctionnalités.

Une autre option est la "Guide d'installation de kubeadm" Fourni par Kubernetes.

> N'utilisez pas un cluster Kubernetes que vous créez en utilisant ces instructions dans un environnement de production. Utilisez les guides de déploiement de production fournis par votre distributeur pour les clusters prêts à la production.

Intégration de Kubernetes avec les produits NetApp

Le portefeuille de produits de stockage NetApp s'intègre à de nombreux aspects des clusters Kubernetes avec des fonctionnalités avancées de gestion des données qui améliorent la fonctionnalité, la capacité, les performances et la disponibilité du déploiement Kubernetes.

Amazon FSX pour NetApp ONTAP

"Amazon FSX pour NetApp ONTAP" Est un service AWS entièrement géré qui vous permet de lancer et d'exécuter des systèmes de fichiers optimisés par le système d'exploitation du stockage NetApp ONTAP.

Azure NetApp Files

"Azure NetApp Files" Est un service de partage de fichiers Azure haute performance optimisé par NetApp. Vous pouvez exécuter les workloads basés sur des fichiers les plus exigeants dans Azure de façon native, avec les performances et les fonctionnalités avancées de gestion des données que vous attendez de NetApp.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP" Est une appliance de stockage exclusivement logicielle qui exécute le logiciel de gestion des données ONTAP dans le cloud.

Cloud Volumes Service pour Google Cloud

"NetApp Cloud Volumes Service pour Google Cloud" Est un service de fichiers cloud natif qui fournit des volumes NAS sur NFS et SMB avec des performances 100 % Flash.

Logiciel Element

"Elément" offre à l'administrateur du stockage la possibilité de consolider les charges de travail pour un encombrement du stockage simplifié et optimisé.

NetApp HCI

"NetApp HCI" simplifie la gestion et l'évolutivité du data center en automatisant les tâches de routine et en permettant aux administrateurs d'infrastructure de se concentrer sur des fonctions plus importantes.

ASTRA Trident peut provisionner et gérer des dispositifs de stockage pour des applications conteneurisées directement à partir de la plateforme de stockage NetApp HCI sous-jacente.

NetApp ONTAP

"NetApp ONTAP" Il s'agit du système d'exploitation de stockage unifié multiprotocole NetApp qui offre des fonctionnalités avancées de gestion des données pour toutes les applications.

Les systèmes ONTAP sont dotés de configurations 100 % Flash, hybrides ou 100 % HDD et proposent différents modèles de déploiement, notamment du matériel spécialisé (FAS et AFF), de l'infrastructure générique (ONTAP Select) et du cloud uniquement (Cloud Volumes ONTAP). ASTRA Trident prend en charge ces modèles de déploiement ONTAP.

Pour en savoir plus

- "Gamme NetApp Astra"
- "Documentation relative au service après-vente Astra Control"
- "Documentation Astra Control Center"
- "Documentation de l'API Astra"

Architecture d'Astra Trident

ASTRA Trident s'exécute en tant que pod unique de contrôleur et comme pod de nœud sur chaque nœud worker dans le cluster. Le pod de nœuds doit être exécuté sur n'importe quel hôte sur lequel vous souhaitez potentiellement monter un volume Astra Trident.

Présentation des pods de contrôleur et des nœuds

ASTRA Trident se déploie comme un seul système Pod du contrôleur Trident et un ou plusieurs Pods de nœuds Trident Sur le cluster Kubernetes et utilise des conteneurs Sidecar Kubernetes standard *CSI* pour simplifier le déploiement des plug-ins CSI. "Conteneurs Sidecar Kubernetes CSI" Sont gérés par la communauté Kubernetes Storage.

Kubernetes "sélecteurs de nœuds" et "tolérances et rejets" sont utilisés pour contraindre un pod à s'exécuter sur un nœud spécifique ou préféré. Vous pouvez configurer des sélecteurs de nœuds et des tolérances pour les pods de contrôleur et de nœud lors de l'installation d'Astra Trident.

- Le plug-in du contrôleur gère le provisionnement et la gestion des volumes, tels que les snapshots et le redimensionnement.
- Le plug-in du nœud permet d'attacher le stockage au nœud.

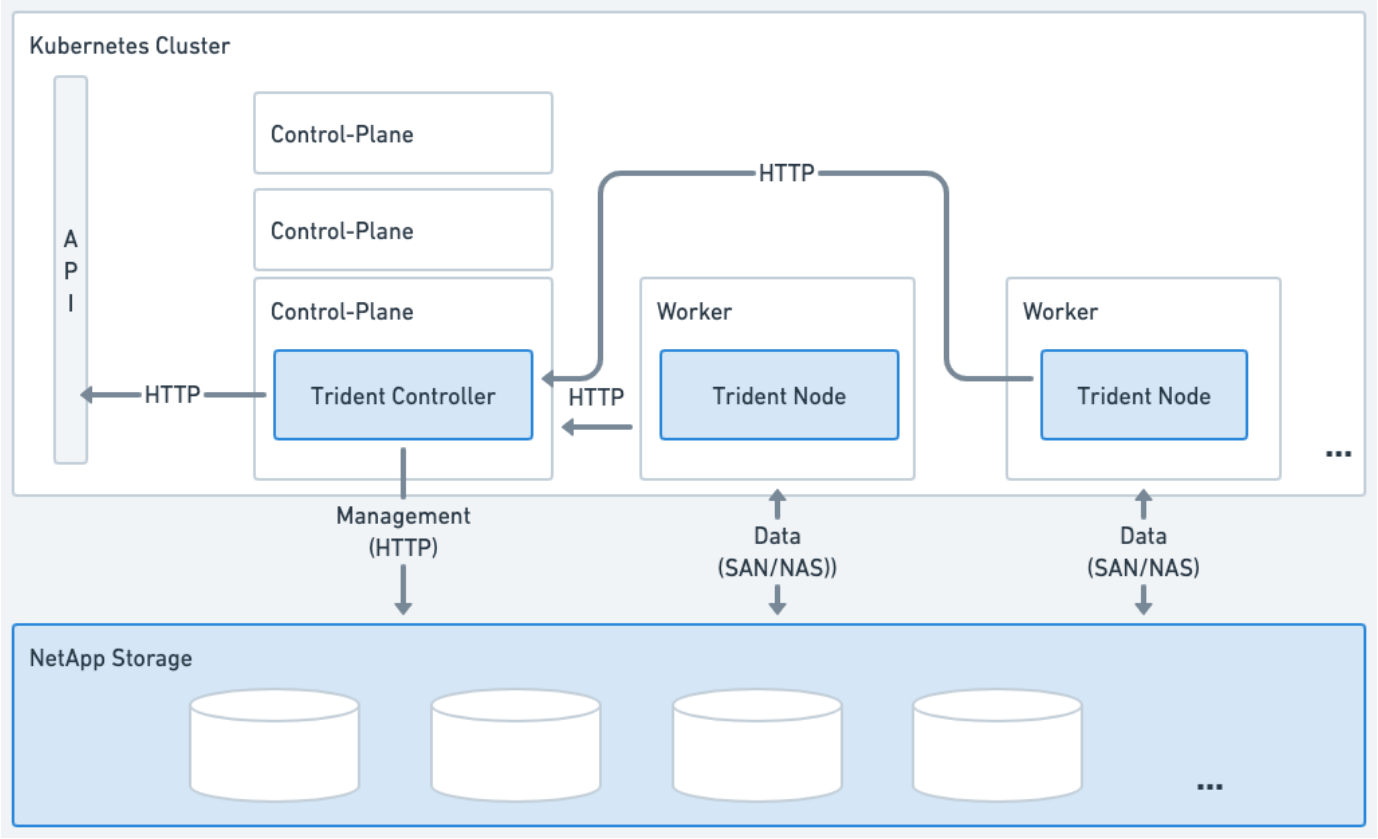


Figure 1. ASTRA Trident a été déployé sur le cluster Kubernetes

Pod du contrôleur Trident

Le pod du contrôleur Trident est un pod unique exécutant le plug-in du contrôleur CSI.

- Responsable du provisionnement et de la gestion des volumes dans le stockage NetApp
- Géré par un déploiement Kubernetes
- Peut s’exécuter sur le plan de contrôle ou les nœuds workers, selon les paramètres d’installation.

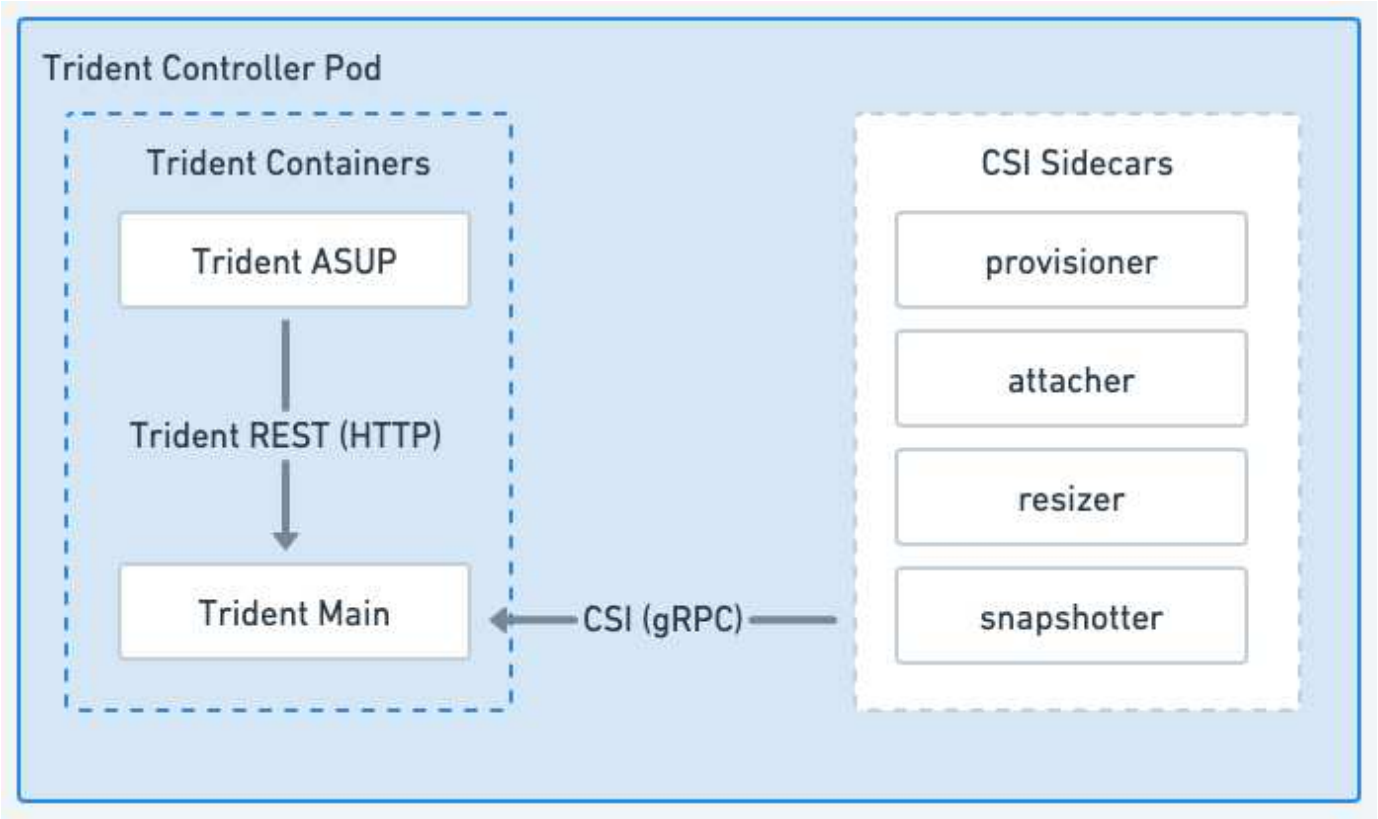


Figure 2. Diagramme du module de contrôleur Trident

Pods de nœuds Trident

Les pods de nœud Trident sont des pods privilégiés exécutant le plug-in CSI Node.

- Responsable du montage et du démontage du stockage des pods qui s'exécutent sur l'hôte
- Géré par un jeu de démonstration Kubernetes
- Doit s'exécuter sur n'importe quel nœud qui montera le stockage NetApp

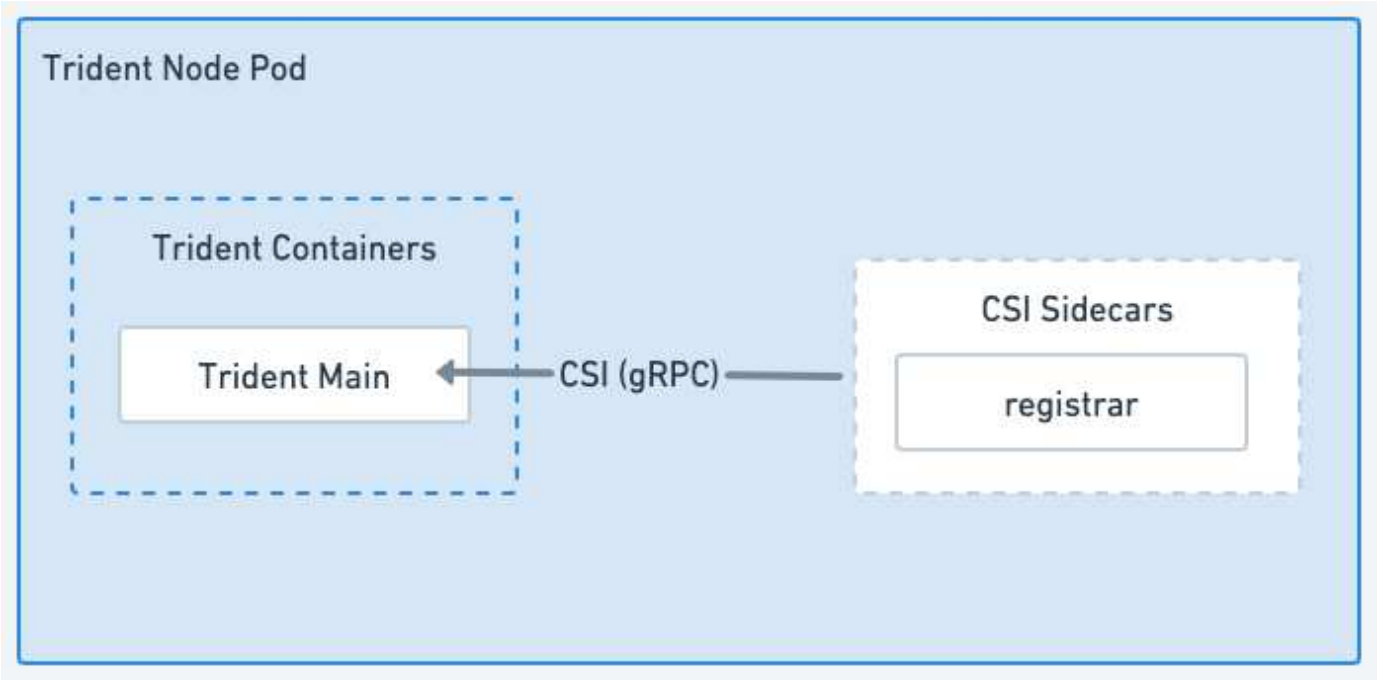


Figure 3. Diagramme Trident Node Pod

Architectures de cluster Kubernetes prises en charge

Astra Trident est pris en charge avec les architectures Kubernetes suivantes :

Architectures en cluster Kubernetes	Pris en charge	Installation par défaut
Maître unique, calcul	Oui.	Oui.
Plusieurs maîtres, calcul	Oui.	Oui.
Maître, `etcd`, calculer	Oui.	Oui.
Maîtrise, infrastructure, calcul	Oui.	Oui.

Concepts

Provisionnement

Le provisionnement avec Astra Trident a deux phases principales. La première phase associe une classe de stockage à l'ensemble des pools de stockage back-end appropriés et effectue la préparation nécessaire avant le provisionnement. La deuxième phase inclut la création du volume et nécessite le choix d'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 back-end à une classe de stockage repose sur les attributs demandés par la classe de stockage et sur leur `storagePools`, `additionalStoragePools`, et `excludeStoragePools` listes. Lorsque vous créez une classe de stockage, Trident compare les attributs et les pools proposés par

chacun de ses systèmes back-end à ceux requis 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, Astra Trident ajoute ce pool de stockage à l'ensemble des pools de stockage appropriés pour cette classe de stockage. D'autre part, Astra Trident ajoute tous les pools de stockage répertoriés dans le `additionalStoragePools` énumérez cet ensemble, même si leurs attributs ne remplissent pas tous ou aucun des attributs demandés par la classe de stockage. Vous devez utiliser le `excludeStoragePools` liste pour remplacer et supprimer les pools de stockage utilisés pour une classe de stockage. La solution Astra Trident effectue un processus similaire chaque fois que vous ajoutez un nouveau système back-end, 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 du volume

Astra 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, Astra Trident commence par obtenir l'ensemble des pools de stockage correspondant à la classe de stockage du volume. De plus, si vous spécifiez un protocole pour le volume, Astra Trident supprime les pools de stockage qui ne peuvent pas fournir le protocole demandé (par exemple, un back-end NetApp HCI/SolidFire ne peut pas fournir un volume basé sur les fichiers, tandis qu'un back-end ONTAP NAS ne peut pas fournir un volume basé sur les blocs). Astra Trident répartit de manière aléatoire l'ordre de ce jeu, afin de faciliter une distribution homogène des volumes, puis l'itérate via lui pour tenter de provisionner le volume sur chaque pool de stockage. S'il réussit sur un, il retourne avec succès, en enregistrant les échecs rencontrés dans le processus. Astra Trident renvoie une défaillance **uniquement si** il ne parvient pas à approvisionner **tous** les pools de stockage disponibles pour la classe et le protocole de stockage demandés.

Snapshots de volume

Découvrez comment Astra Trident gère la création de copies Snapshot de volume pour ses pilotes.

En savoir plus sur la création de snapshots de volume

- Pour le `ontap-nas`, `ontap-san`, `gcp-cvs`, et `azure-netapp-files` Chaque volume persistant est mappé à un FlexVol. Par conséquent, des snapshots de volume sont créés sous forme de snapshots NetApp. La technologie Snapshot de NetApp offre davantage de stabilité, d'évolutivité, de capacité de restauration et de performances que les technologies Snapshot concurrentes. Ces copies Snapshot sont extrêmement efficaces, aussi bien en termes de temps de création que d'espace de stockage.
- Pour le `ontap-nas-flexgroup` Chaque pilote de volume persistant est mappé à un FlexGroup. Par conséquent, des snapshots de volume sont créés sous forme de snapshots NetApp FlexGroup. La technologie Snapshot de NetApp offre davantage de stabilité, d'évolutivité, de capacité de restauration et de performances que les technologies Snapshot concurrentes. Ces copies Snapshot sont extrêmement efficaces, aussi bien en termes de temps de création que d'espace de stockage.
- Pour le `ontap-san-economy` Le pilote et les volumes persistants sont mis en correspondance avec les LUN créées sur les volumes FlexVol partagés. Les copies FlexClone de la LUN associée permettent d'obtenir les copies Snapshot VolumeSnapshot de la LUN associée. Grâce à la technologie FlexClone de ONTAP, il est possible de créer quasi instantanément des copies des jeux de données les plus volumineux, même les plus volumineux. Les copies partagent les blocs de données avec leurs parents. Aucun stockage n'est nécessaire, sauf pour les métadonnées.
- Pour le `solidfire-san` Pilote, chaque volume persistant est mappé sur une LUN créée dans le cluster NetApp Element/NetApp HCI. Les copies Snapshot VolumeCas sont représentées par des copies Snapshot Element de la LUN sous-jacente. Ces snapshots sont des copies à un point dans le temps et ne prennent en charge qu'une petite quantité de ressources et d'espace système.

- Lorsque vous travaillez avec le `ontap-nas` et `ontap-san` Les snapshots ONTAP sont des copies ponctuelles de la FlexVol et consomment de l'espace sur la FlexVol elle-même. Cela peut entraîner la quantité d'espace inscriptible dans le volume pour une réduction du temps lors de la création ou de la planification des snapshots. L'une des façons simples de résoudre ce problème est d'augmenter le volume en le redimensionnant via Kubernetes. Une autre option consiste à supprimer les snapshots qui ne sont plus nécessaires. Lors de la suppression d'un Snapshot VolumeCas créé via Kubernetes, Astra Trident supprime le snapshot ONTAP associé. Les snapshots ONTAP qui n'ont pas été créés par Kubernetes peuvent également être supprimés.

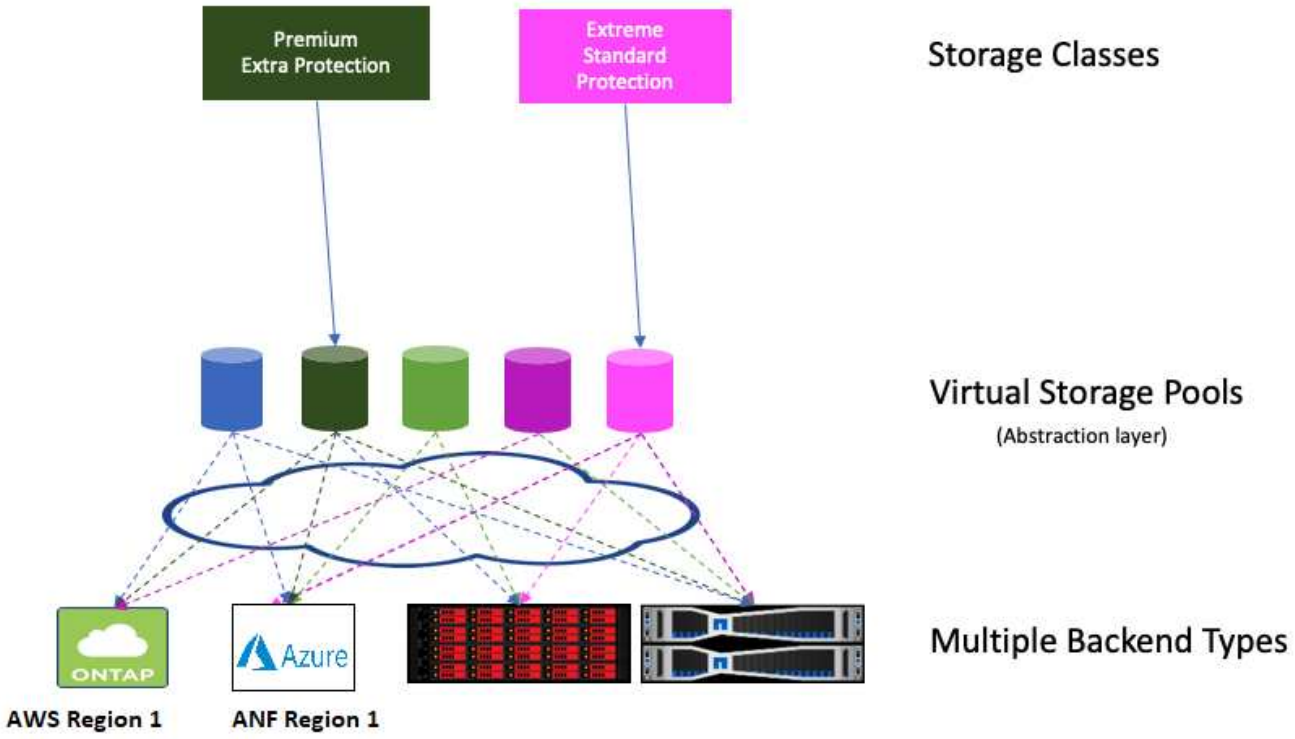
Avec Astra Trident, vous pouvez utiliser Volumesnapshots pour créer de nouveaux volumes persistants à partir d'entre eux. La création de volumes persistants est effectuée à partir de ces copies Snapshot à l'aide de la technologie FlexClone pour les systèmes back-end ONTAP et CVS pris en charge. Lors de la création d'un volume persistant à partir d'un snapshot, le volume de sauvegarde est un volume FlexClone du volume parent du snapshot. Le `solidfire-san` Le pilote utilise des clones de volumes logiciels Element pour créer des volumes persistants à partir de snapshots. Ici, cela crée un clone à partir du snapshot Element.

Pools virtuels

Les pools virtuels fournissent une couche d'abstraction entre les systèmes back-end de stockage Astra Trident et Kubernetes `StorageClasses`. Ils permettent à un administrateur de définir des aspects, tels que l'emplacement, les performances et la protection pour chaque système back-end, de façon commune et indépendante du système back-end `StorageClass` spécifiez le type de back-end physique, de pool back-end ou de type back-end à utiliser pour répondre aux critères souhaités.

En savoir plus sur les pools virtuels

L'administrateur du stockage peut définir des pools virtuels sur n'importe quel système back-end Trident Astra dans un fichier de définition JSON ou YAML.



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

- Lors de la définition de pools virtuels, n'essayez pas de réorganiser l'ordre des pools virtuels existants dans une définition backend.
- Nous vous conseillons de modifier les attributs d'un pool virtuel existant. Vous devez définir un nouveau pool virtuel pour apporter des modifications.

La plupart des aspects sont spécifiés dans des termes spécifiques au système back-end. Il est primordial que les valeurs de l'aspect ne soient pas exposées en dehors du conducteur du back-end et ne soient pas disponibles pour la correspondance dans `StorageClasses`. À la place, l'administrateur définit un ou plusieurs libellés pour chaque pool virtuel. Chaque étiquette est une paire clé:valeur et les étiquettes sont souvent répandues sur différents systèmes back-end. Tout comme les aspects, les étiquettes peuvent être spécifiées par pool ou globales au back-end. Contrairement aux aspects, qui ont des noms et des valeurs prédéfinis, l'administrateur dispose d'une entière discrétion pour définir les clés et les valeurs de libellé selon les besoins. Pour plus de commodité, les administrateurs du stockage peuvent définir des étiquettes par pool virtuel et les volumes de groupe par étiquette.

A `StorageClass` identifie le pool virtuel à utiliser en référençant les étiquettes dans un paramètre de sélection. Les sélecteurs de pool virtuel prennent en charge les opérateurs suivants :

Opérateur	Exemple	La valeur d'étiquette d'un pool doit :
=	performance=premium	Correspondance
! =	performance !=extrême	Ne correspond pas

Opérateur	Exemple	La valeur d'étiquette d'un pool doit :
`in`	emplacement à (est, ouest)	Être dans l'ensemble de valeurs
`notin`	performances notin (argent, bronze)	Ne pas être dans l'ensemble de valeurs
`<key>`	la protection	Existe avec n'importe quelle valeur
`!<key>`	!protection	N'existe pas

Groupes d'accès de volume

Découvrez l'utilisation d'Astra Trident "groupes d'accès de volume".

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

En savoir plus sur les groupes d'accès aux volumes

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

ASTRA Trident associe de nouveaux volumes aux groupes d'accès configurés, mais il ne crée pas et ne gère pas les groupes d'accès eux-mêmes. Les groupes d'accès doivent exister avant l'ajout du système back-end de stockage à Astra Trident et doivent contenir les IQN iSCSI de chaque nœud du cluster Kubernetes pouvant potentiellement monter les volumes provisionnés par ce back-end. Dans la plupart des installations, cela inclut tous les nœuds workers dans le cluster.

Pour les clusters Kubernetes de plus de 64 nœuds, vous devez 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 quatre groupes d'accès configurés au maximum, n'importe quel nœud d'un cluster de 256 nœuds maximum pourra accéder à n'importe quel volume. Pour connaître les dernières limites relatives aux groupes d'accès aux volumes, reportez-vous à la section "ici".

Si vous modifiez la configuration à partir d'une configuration utilisant la configuration par défaut `trident` Groupe d'accès à un groupe qui utilise également d'autres, inclure l'ID pour le `trident` groupe d'accès dans la liste.

Démarrage rapide d'Astra Trident

Vous pouvez installer Astra Trident et commencer à gérer les ressources de stockage en quelques étapes. Avant de commencer, consultez "Exigences d'Astra Trident".

> Pour Docker, reportez-vous à la section "Astra Trident pour Docker".

1 Installer Astra Trident

ASTRA Trident offre plusieurs méthodes et modes d'installation optimisés pour un large éventail d'environnements et d'entreprises.

"Installer Astra Trident"

2 Préparez le nœud de travail

Tous les nœuds workers du cluster Kubernetes doivent pouvoir monter les volumes provisionnés pour vos pods.

"Préparez le nœud de travail"

3 Créer un back-end

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

"Configurer un back-end" de votre système de stockage

4 Créez une classe de stockage Kubernetes

L'objet StorageClass Kubernetes spécifie Astra Trident en tant que mécanisme de provisionnement et permet de créer une classe de stockage pour provisionner des volumes avec des attributs personnalisables. ASTRA Trident crée une classe de stockage correspondante pour les objets Kubernetes pour spécifier le mécanisme de provisionnement Astra Trident.

"Créer une classe de stockage"

5 Provisionner un volume

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

Créez un volume persistant et une demande de volume persistant qui utilisent la classe de stockage Kubernetes configurée pour demander l'accès au volume persistant. Vous pouvez ensuite monter le volume persistant sur un pod.

"Provisionner un volume"

Et la suite ?

Vous pouvez à présent ajouter des systèmes back-end supplémentaires, gérer les classes de stockage, gérer les systèmes back-end et effectuer des opérations de volume.

De formation

Avant d'installer Astra Trident, il est recommandé de vérifier ces exigences système générales. Il se peut que les systèmes back-end spécifiques présentent des exigences supplémentaires.

Informations stratégiques sur Astra Trident

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

** informations pratiques sur le Tridécouvrez Astra **

- Kubernetes 1.29 est désormais pris en charge dans Trident. Mettez à niveau Astra Trident avant de mettre à niveau Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. ASTRA Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Systèmes front-end (orchestrateurs) pris en charge

Astra Trident prend en charge plusieurs moteurs et orchestrateurs de conteneur, notamment :

- Anthos sur site (VMware) et Anthos sur bare Metal 1.16
- Kubernetes 1.23 - 1.29
- OpenShift 4.10 - 4.14

L'opérateur de Trident est pris en charge par ces versions :

- Anthos sur site (VMware) et Anthos sur bare Metal 1.16
- Kubernetes 1.23 - 1.29
- OpenShift 4.10 - 4.14

ASTRA 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), Rancher et le portefeuille VMware Tanzu.

Pour laquelle Astra Trident et ONTAP peuvent être utilisés en tant que fournisseur de stockage "KubeVirt".

> Avant de mettre à niveau un cluster Kubernetes de la version 1.24 vers la version 1.25 ou ultérieure sur lequel Astra Trident est installé, reportez-vous à la section "Mettre à niveau une installation Helm".

Systèmes back-end pris en charge (stockage)

Pour utiliser Astra Trident, vous avez besoin d'un ou de plusieurs des systèmes back-end pris en charge :

- Amazon FSX pour NetApp ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP

- Cloud Volumes Service pour GCP
- FAS/AFF/Select 9.5 ou version ultérieure
- Baie SAN 100 % Flash (ASA) de NetApp
- Logiciel NetApp HCI/Element 11 ou version ultérieure

Configuration requise

Le tableau ci-dessous résume les fonctionnalités disponibles dans cette version d'Astra Trident et les versions de Kubernetes qu'il prend en charge.

Fonction	Version Kubernetes	Portes-fonctions requises ?
Astra Trident	1.23 - 1.29	Non
Snapshots de volume	1.23 - 1.29	Non
Volume persistant à partir des copies Snapshot des volumes	1.23 - 1.29	Non
Redimensionnement PV iSCSI	1.23 - 1.29	Non
Chap bidirectionnel ONTAP	1.23 - 1.29	Non
Règles d'exportation dynamiques	1.23 - 1.29	Non
Opérateur Trident	1.23 - 1.29	Non
Topologie CSI	1.23 - 1.29	Non

Systèmes d'exploitation hôtes testés

Bien que l'Astra Trident ne prenne pas officiellement en charge des systèmes d'exploitation spécifiques, il s'agit des éléments suivants qui fonctionnent :

- Versions de Red Hat CoreOS (RHCOS) prises en charge par OpenShift Container Platform (AMD64 et ARM64)
- RHEL 8+ (AMD64 ET ARM64)

  NVMe/TCP requiert RHEL 9 ou version ultérieure.

- Ubuntu 22.04 ou version ultérieure (AMD64 et ARM64)
- Windows Server 2019 (AMD64)

Par défaut, Astra Trident s'exécute dans un conteneur et s'exécute donc sur un utilisateur Linux. Cependant, ces employés doivent pouvoir monter les volumes qu'Astra Trident utilise le client NFS standard ou l'initiateur iSCSI, en fonction du système back-end utilisé.

Le `tridentctl` Utility s'exécute également sur l'une de ces distributions de Linux.

Configuration de l'hôte

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

"Préparez le nœud de travail"

Configuration du système de stockage

Il est possible qu'Astra Trident modifie le système de stockage avant qu'une configuration back-end ne puisse l'utiliser.

"Configuration des systèmes back-end"

Ports Trident d'Astra

L'Astra Trident doit accéder à des ports spécifiques pour la communication.

"Ports Trident d'Astra"

Images de conteneur et versions Kubernetes correspondantes

Pour les installations utilisant des systèmes à air comprimé, la liste suivante est une référence des images de conteneur nécessaires à l'installation d'Astra Trident. Utilisez le `tridentctl images` commande pour vérifier la liste des images de conteneur requises.

Version Kubernetes	Image de conteneur
v1.23.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)

Version Kubernetes	Image de conteneur
v1.24.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)
v1.25.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)
v1.26.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)

Version Kubernetes	Image de conteneur
v1.27.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)
v1.28.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v3.6.0 • registry.k8s.io/sig-storage/csi-attacher:v4.4.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.0 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.9.0 • docker.io/netapp/trident-operator:24.02.0 (en option)
v1.29.0	• docker.io/netapp/trident : 24.02.0 • docker.io/netapp/trident-autosupport:24.02 • registry.k8s.io/sig-storage/csi-provisionneur:v4.0.0 • registry.k8s.io/sig-storage/csi-attacher:v4.5.0 • registry.k8s.io/sig-storage/csi-resizer:v1.9.3 • registry.k8s.io/sig-storage/csi-snapshotter:v6.3.3 • registry.k8s.io/sig-storage/csi-node-driver-registratr:v2.10.0 • docker.io/netapp/trident-operator : 24.02.0 (facultatif)

Installer Astra Trident

Découvrez l'installation d'Astra Trident

Pour vérifier qu'Astra Trident peut être installé dans un grand nombre d'environnements et d'entreprises, NetApp propose plusieurs options d'installation. Vous pouvez installer Astra Trident à l'aide de l'opérateur Trident (manuellement ou à l'aide de Helm) ou à l'aide de `tridentctl`. Cette rubrique fournit des informations importantes pour sélectionner le processus d'installation qui vous convient.

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

** informations pratiques sur le Tridécouvrez Astra **

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Avant de commencer

Quel que soit votre chemin d'installation, vous devez avoir :

- Privilèges complets vers un cluster Kubernetes pris en charge exécutant une version prise en charge de Kubernetes et conditions requises pour les fonctionnalités activées. Vérifiez le "de formation" pour plus d'informations.
- Accès à un système de stockage NetApp pris en charge.
- Capacité de monter des volumes à partir de tous les nœuds de travail Kubernetes.
- Un hôte Linux avec `kubectl` (ou `oc`, Si vous utilisez OpenShift) installé et configuré pour gérer le cluster Kubernetes que vous souhaitez utiliser.
- Le `KUBECONFIG` Variable d'environnement qui pointe vers votre configuration de cluster Kubernetes.
- Si vous utilisez Kubernetes avec Docker Enterprise, "Suivez les étapes indiquées pour activer l'accès à l'interface de ligne de commande".

Si vous ne vous êtes pas familiarisé avec le "concepts de base", c'est le moment idéal pour le faire.

Choisissez votre méthode d'installation

Sélectionnez la méthode d'installation qui vous convient. Vous devez également examiner les considérations à prendre en compte pour "passage d'une méthode à l'autre" avant de prendre votre décision.

Utilisation de l'opérateur Trident

Que ce soit pour un déploiement manuel ou à l'aide de Helm, l'opérateur Trident est un excellent moyen de simplifier l'installation et de gérer dynamiquement les ressources Trident. Vous pouvez même "Personnalisez le déploiement de l'opérateur Trident" utilisation des attributs dans `TridentOrchestrator` Ressource personnalisée (CR).

L'utilisateur de Trident présente les avantages suivants :

de l'objet de la carte de Trident crece

L'opérateur Trident crée automatiquement les objets suivants pour votre version Kubernetes.

- ServiceAccount pour l'opérateur
- ClusterRole et ClusterRoleBinding au ServiceAccount
- Dedicated PodSecurityPolicy (pour Kubernetes 1.25 et versions antérieures)
- L'opérateur lui-même

compte : « »

L'opérateur Trident dont le périmètre est défini dans le cluster gère les ressources associées à une installation Astra Trident au niveau du cluster. Cela réduit les erreurs pouvant être provoquées lors de la maintenance des ressources du cluster-scoped à l'aide d'un opérateur namespace-scoped. Ceci est essentiel pour l'auto-rétablissement et l'application de correctifs.

- Capcuratif de la prise

L'opérateur surveille l'installation d'Astra Trident et prend activement des mesures pour résoudre les problèmes, par exemple lorsque le déploiement est supprimé ou lorsqu'il est modifié par erreur. A `trident-operator-<generated-id>` le pod est créé et associe un `TridentOrchestrator` CR avec une installation Astra Trident. Cela garantit qu'il n'y a qu'une seule instance d'Astra Trident dans le cluster et contrôle sa configuration, en s'assurant que l'installation est idemopuissante. Lorsque des modifications sont apportées à l'installation (par exemple, la suppression du déploiement ou du demonset de nœuds), l'opérateur les identifie et les corrige individuellement.

** mise à jour de l'installation de existante**

Vous pouvez facilement mettre à jour un déploiement existant avec l'opérateur. Il vous suffit de modifier le `TridentOrchestrator` CR pour effectuer des mises à jour d'une installation.

Prenons l'exemple d'un scénario dans lequel vous devez activer Astra Trident pour générer des journaux de débogage. Pour ce faire, patch de votre `TridentOrchestrator` à régler `spec.debug` à `true`:

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

Après `TridentOrchestrator` est mis à jour, l'opérateur traite les mises à jour et met à jour l'installation existante. Cela peut déclencher la création de nouveaux modules pour modifier l'installation en conséquence.

** **

L'opérateur Trident dont le périmètre est défini dans le cluster permet la suppression complète des ressources dont le périmètre est défini dans le cluster. Les utilisateurs peuvent désinstaller complètement Astra Trident et réinstaller facilement.

** : mise à niveau de **

Lorsque la version Kubernetes du cluster est mise à niveau vers une version prise en charge, l'opérateur met automatiquement à jour une installation Astra Trident existante et la modifie pour s'assurer qu'elle répond aux exigences de la version Kubernetes.

> Si le cluster est mis à niveau vers une version non prise en charge, l'opérateur empêche l'installation d'Astra Trident. Si Astra Trident a déjà été installé avec l'opérateur, un avertissement s'affiche pour indiquer que l'Astra Trident est installé sur une version Kubernetes non prise en charge.

Gestion de clusters ™ avec BlueXP (anciennement Cloud Manager)

Avec "Astra Trident avec BlueXP", Vous pouvez effectuer la mise à niveau vers la dernière version d'Astra Trident, ajouter et gérer des classes de stockage, les connecter aux environnements de travail et sauvegarder des volumes persistants à l'aide de Cloud Backup Service. BlueXP prend en charge le déploiement Astra Trident à l'aide de l'opérateur Trident, soit manuellement, soit via Helm.

À l'aide de `tridentctl`

Si votre déploiement existant doit être mis à niveau ou si vous cherchez à le personnaliser hautement, il est à votre disposition . Il s'agit de la méthode classique de déploiement d'Astra Trident.

C'est possible Générer les manifestes pour les ressources Trident. Cela inclut le déploiement, la demonset, le compte de service et le rôle de cluster qu'Astra Trident a créé dans le cadre de son installation.

> Depuis la version 22.04, les clés AES ne sont plus régénérées à chaque installation d'Astra Trident. Avec cette version, Astra Trident va installer un nouvel objet secret qui persiste dans toutes les installations. Cela signifie que `tridentctl` La version 22.04 peut désinstaller les versions précédentes de Trident, mais les versions antérieures ne peuvent pas désinstaller 22.04 installations. Sélectionnez l'installation appropriée *method*.

Choisissez votre mode d'installation

Déterminez votre processus de déploiement en fonction du *mode d'installation* (Standard, Offline ou Remote) requis par votre organisation.

Installation standard

Il s'agit du moyen le plus simple d'installer Astra Trident et qui fonctionne pour la plupart des environnements qui n'imposent pas de restrictions de réseau. Le mode d'installation standard utilise les registres par défaut pour stocker Trident requis (`docker.io`) Et CSI (`registry.k8s.io`) images.

Lorsque vous utilisez le mode standard, le programme d'installation d'Astra Trident :

- Extrait les images conteneur sur Internet
- Crée un déploiement ou un demonset de nœuds, qui fait tourner les pods Astra Trident sur tous les nœuds éligibles du cluster Kubernetes

Installation hors ligne

Le mode d'installation hors ligne peut être requis dans un emplacement rodé ou sécurisé. Dans ce scénario, vous pouvez créer un registre privé en miroir ou deux registres en miroir pour stocker les images Trident et CSI requises.

> Quelle que soit la configuration de votre registre, les images CSI doivent résider dans un registre.

Installation à distance

Voici une présentation générale du processus d'installation à distance :

- Déployez la version appropriée de `kubectl` Sur l'ordinateur distant d'où vous souhaitez déployer Astra Trident.
- Copiez les fichiers de configuration depuis le cluster Kubernetes et configurez le `KUBECONFIG` variable d'environnement sur la machine à distance.
- Lancer un `kubectl get nodes` Commande pour vérifier que vous pouvez vous connecter au cluster Kubernetes requis.
- Effectuez le déploiement à partir de la machine distante en suivant les étapes d'installation standard.

Sélectionnez le processus en fonction de votre méthode et de votre mode

Après avoir pris vos décisions, sélectionnez le processus approprié.

Méthode	Mode d'installation
Opérateur Trident (manuellement)	"Installation standard" "Installation hors ligne"
Opérateur Trident (Helm)	"Installation standard" "Installation hors ligne"
`tridentctl`	"Installation standard ou hors ligne"

Passage d'une méthode d'installation à l'autre

Vous pouvez décider de modifier votre méthode d'installation. Avant de procéder, prenez en compte les points suivants :

- Utilisez toujours la même méthode pour installer et désinstaller Astra Trident. Si vous avez déployé avec `tridentctl`, vous devez utiliser la version appropriée de l' `tridentctl` Binaire pour désinstaller Astra Trident. De même, si vous déployez avec l'opérateur, vous devez modifier le `TridentOrchestrator` CR et set `spec.uninstall=true` Pour désinstaller Astra Trident.
- Si vous avez un déploiement basé sur l'opérateur que vous souhaitez supprimer et utiliser à la place `tridentctl` Pour déployer Astra Trident, vous devez d'abord modifier `TridentOrchestrator` et jeu `spec.uninstall=true` Pour désinstaller Astra Trident. Puis supprimer `TridentOrchestrator` et le déploiement de l'opérateur. Vous pouvez ensuite installer à l'aide de `tridentctl`.
- Si vous disposez d'un déploiement manuel basé sur l'opérateur et que vous souhaitez utiliser le déploiement d'opérateurs Trident basé sur Helm, vous devez d'abord désinstaller manuellement l'opérateur, puis effectuer l'installation de Helm. Helm permet à l'opérateur Trident de déployer les étiquettes et les annotations requises. Si vous ne le faites pas, le déploiement d'un opérateur Trident basé sur Helm échoue en raison de l'erreur de validation des étiquettes et de l'erreur de validation des annotations. Si vous avez un `tridentctl`Le déploiement basé sur Helm permet d'utiliser un déploiement basé sur Helm sans s'exécuter dans les problèmes.

Autres options de configuration connues

Lors de l'installation d'Astra Trident sur les produits de la gamme VMware Tanzu :

- Le cluster doit prendre en charge les workloads privilégiés.
- Le `--kubelet-dir` l'indicateur doit être défini sur l'emplacement du répertoire kubelet. Par défaut, il s'agit de `/var/vcap/data/kubelet`.

  Spécifier l'emplacement du kubelet à l'aide de `--kubelet-dir` Est connu pour fonctionner avec l'opérateur Trident, Helm et `tridentctl` de nombreux déploiements.

Installer à l'aide de l'opérateur Trident

Déployer manuellement l'opérateur Trident (mode Standard)

Vous pouvez déployer manuellement l'opérateur Trident pour installer Astra Trident. Ce processus s'applique aux installations dans lesquelles les images de conteneur requises par Astra Trident ne sont pas stockées dans un registre privé. Si vous disposez d'un registre d'images privé, utilisez le "processus de déploiement hors ligne".

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

** informations pratiques sur le Tridécouvrez Astra **

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Déployer manuellement l'opérateur Trident et installer Trident

Révision "présentation de l'installation" pour vous assurer que les conditions préalables à l'installation sont respectées et que vous avez sélectionné l'option d'installation appropriée pour votre environnement.

Avant de commencer

Avant de commencer l'installation, connectez-vous à l'hôte Linux et vérifiez qu'il gère un travail. "Cluster Kubernetes pris en charge" et que vous disposez des privilèges nécessaires.

> Avec OpenShift, utilisez `oc` au lieu de `kubectl` dans tous les exemples qui suivent, et connectez-vous en tant que **system:admin** en premier lieu en cours d'exécution `oc login -u system:admin` ou `oc login -u kube-admin`.

1. Vérifiez votre version Kubernetes :

```
kubectl version
```

2. Vérifiez les privilèges d'administrateur du cluster :

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Vérifiez que vous pouvez lancer un pod qui utilise une image depuis Docker Hub et atteindre votre système de stockage sur le réseau du pod :

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Étape 1 : téléchargez le package du programme d'installation de Trident

Le package d'installation d'Astra Trident contient tout ce dont vous avez besoin pour déployer l'opérateur Trident et installer Astra Trident. Téléchargez et extrayez la dernière version du programme d'installation de Trident "La section *Assets* sur GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

Étape 2 : créez le `TridentOrchestrator` CRD

Créer le `TridentOrchestrator` Définition de ressource personnalisée (CRD). Vous allez créer un `TridentOrchestrator` Ressources personnalisées plus tard. Utilisez la version CRD YAML appropriée dans `deploy/crds` pour créer le `TridentOrchestrator` CRD.

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

Étape 3 : déployer l'opérateur Trident

Le programme d'installation d'Astra Trident fournit un fichier de bundle qui peut être utilisé pour installer l'opérateur et créer des objets associés. Le fichier bundle est un moyen simple de déployer l'opérateur et d'installer Astra Trident avec une configuration par défaut.

- Pour les clusters exécutant Kubernetes 1.24 ou version antérieure, utilisez `bundle_pre_1_25.yaml`.

- Pour les clusters exécutant Kubernetes 1.25 ou version ultérieure, utilisez `bundle_post_1_25.yaml`.

Avant de commencer

- Par défaut, le programme d'installation de Trident déploie l'opérateur dans `trident` espace de noms. Si le `trident` l'espace de noms n'existe pas. créez-le à l'aide des éléments suivants :

```
kubectl apply -f deploy/namespace.yaml
```

- Pour déployer l'opérateur dans un espace de nom autre que le `trident` espace de noms, mettre à jour `serviceaccount.yaml`, `clusterrolebinding.yaml` et `operator.yaml` et générez votre fichier de bundle à l'aide du `kustomization.yaml`.

  a. Créer le `kustomization.yaml` en utilisant la commande suivante où *<bundle.yaml>* est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` Basé sur votre version de Kubernetes.

  ```
  cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
  ```

  b. Compilez le bundle à l'aide de la commande suivante où *<bundle.yaml>* est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` Basé sur votre version de Kubernetes.

  ```
  kubectl kustomize deploy/ > deploy/<bundle.yaml>
  ```

Étapes

1. Créer les ressources et déployer l'opérateur :

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

2. Vérifiez que l'opérateur, le déploiement et les réplicateurs ont été créés.

```
kubectl get all -n <operator-namespace>
```

> Il ne doit y avoir que **une instance** de l'opérateur dans un cluster Kubernetes. Ne créez pas plusieurs déploiements de l'opérateur Trident.

Étape 4 : créez le `TridentOrchestrator` Et installer Trident

Vous pouvez maintenant créer le `TridentOrchestrator` Et installer Astra Trident. Si vous le souhaitez, vous pouvez "Personnalisez votre installation de Trident" utilisation des attributs dans `TridentOrchestrator` spécifications

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Debug:     true
  Namespace: trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:24.02.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Vérifiez l’installation

Il existe plusieurs façons de vérifier votre installation.

À l’aide de `TridentOrchestrator` état

Le statut de `TridentOrchestrator` Indique si l’installation a réussi et affiche la version de Trident installée.

Pendant l'installation, l'état de `TridentOrchestrator` modifications de `Installing` à `Installed`. Si vous observez l' `Failed` l'état et l'opérateur ne parvient pas à récupérer lui-même, "vérifiez les journaux".

État	Description
Installation	L'opérateur installe Astra Trident à l'aide de ce module `TridentOrchestrator` CR.
Installé	Astra Trident a été installé avec succès.
Désinstallation	L'opérateur désinstallant Astra Trident, car `spec.uninstall=true`.
Désinstallé	Astra Trident est désinstallé.
Échec	L'opérateur n'a pas pu installer, corriger, mettre à jour ou désinstaller Astra Trident. L'opérateur essaiera automatiquement de récupérer cet état. Si cet état persiste, vous devrez effectuer un dépannage.
Mise à jour	L'opérateur met à jour une installation existante.
Erreur	Le `TridentOrchestrator` n'est pas utilisé. Un autre existe déjà.

Utilisation du statut de création du pod

Vous pouvez vérifier que l'installation d'Astra Trident est terminée en consultant le statut des pods créés :

```
kubectl get pods -n trident

NAME                                      READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw       6/6     Running   0
1m
trident-node-linux-mr6zc                  2/2     Running   0
1m
trident-node-linux-xrp7w                  2/2     Running   0
1m
trident-node-linux-zh2jt                  2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv         1/1     Running   0
3m
```

À l'aide de `tridentctl`

Vous pouvez utiliser `tridentctl` Pour vérifier la version d'Astra Trident installée.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

Déploiement manuel de l'opérateur Trident (mode hors ligne)

Vous pouvez déployer manuellement l'opérateur Trident pour installer Astra Trident. Ce processus s'applique aux installations dans lesquelles les images de conteneur requises par Astra Trident sont stockées dans un registre privé. Si vous ne disposez pas d'un registre d'images privé, utilisez le "du déploiement standard".

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

** informations pratiques sur le Tridécouvrez Astra **

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Déployer manuellement l'opérateur Trident et installer Trident

Révision "présentation de l'installation" pour vous assurer que les conditions préalables à l'installation sont respectées et que vous avez sélectionné l'option d'installation appropriée pour votre environnement.

Avant de commencer

Connectez-vous à l'hôte Linux et vérifiez qu'il gère un environnement de travail et "Cluster Kubernetes pris en charge" et que vous disposez des privilèges nécessaires.

> Avec OpenShift, utilisez `oc` au lieu de `kubectl` dans tous les exemples qui suivent, et connectez-vous en tant que **system:admin** en premier lieu en cours d'exécution `oc login -u system:admin` ou `oc login -u kube-admin`.

1. Vérifiez votre version Kubernetes :

```
kubectl version
```

2. Vérifiez les privilèges d'administrateur du cluster :

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Vérifiez que vous pouvez lancer un pod qui utilise une image depuis Docker Hub et atteindre votre système de stockage sur le réseau du pod :

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Étape 1 : téléchargez le package du programme d'installation de Trident

Le package d'installation d'Astra Trident contient tout ce dont vous avez besoin pour déployer l'opérateur Trident et installer Astra Trident. Téléchargez et extrayez la dernière version du programme d'installation de Trident "La section *Assets* sur GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

Étape 2 : créez le `TridentOrchestrator` CRD

Créer le `TridentOrchestrator` Définition de ressource personnalisée (CRD). Vous allez créer un `TridentOrchestrator` Ressources personnalisées plus tard. Utilisez la version CRD YAML appropriée dans `deploy/crds` pour créer le `TridentOrchestrator` CRD :

```
kubectl create -f deploy/crds/<VERSION>.yaml
```

Étape 3 : mettez à jour l'emplacement du registre dans l'opérateur

Dans `/deploy/operator.yaml`, mettre à jour `image: docker.io/netapp/trident-operator:24.02.0` pour refléter l'emplacement de votre registre d'images. Votre "Images Trident et CSI" Peut être situé dans un registre ou dans des registres différents, mais toutes les images CSI doivent se trouver dans le même registre. Par exemple :

- `image: <your-registry>/trident-operator:24.02.0` si vos images sont toutes situées dans un même registre.

- `image: <your-registry>/netapp/trident-operator:24.02.0` Si votre image Trident se trouve dans un registre différent de vos images CSI.

Étape 4 : déploiement de l'opérateur Trident

Le programme d'installation d'Astra Trident fournit un fichier de bundle qui peut être utilisé pour installer l'opérateur et créer des objets associés. Le fichier bundle est un moyen simple de déployer l'opérateur et d'installer Astra Trident avec une configuration par défaut.

- Pour les clusters exécutant Kubernetes 1.24 ou version antérieure, utilisez `bundle_pre_1_25.yaml`.
- Pour les clusters exécutant Kubernetes 1.25 ou version ultérieure, utilisez `bundle_post_1_25.yaml`.

Avant de commencer

- Par défaut, le programme d'installation de Trident déploie l'opérateur dans `trident` espace de noms. Si le `trident` l'espace de noms n'existe pas. créez-le à l'aide des éléments suivants :

```
kubectl apply -f deploy/namespace.yaml
```

- Pour déployer l'opérateur dans un espace de nom autre que le `trident` espace de noms, mettre à jour `serviceaccount.yaml`, `clusterrolebinding.yaml` et `operator.yaml` et générez votre fichier de bundle à l'aide du `kustomization.yaml`.

  a. Créer le `kustomization.yaml` en utilisant la commande suivante où *<bundle.yaml>* est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` Basé sur votre version de Kubernetes.

  ```
  cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
  ```

  b. Compilez le bundle à l'aide de la commande suivante où *<bundle.yaml>* est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` Basé sur votre version de Kubernetes.

  ```
  kubectl kustomize deploy/ > deploy/<bundle.yaml>
  ```

Étapes

1. Créer les ressources et déployer l'opérateur :

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

2. Vérifiez que l'opérateur, le déploiement et les réplicateurs ont été créés.

```
kubectl get all -n <operator-namespace>
```

> Il ne doit y avoir que **une instance** de l'opérateur dans un cluster Kubernetes. Ne créez pas plusieurs déploiements de l'opérateur Trident.

Étape 5 : mettez à jour l'emplacement du registre d'images dans le `TridentOrchestrator`

Votre "Images Trident et CSI" Peut être situé dans un registre ou dans des registres différents, mais toutes les images CSI doivent se trouver dans le même registre. Mise à jour `deploy/crds/tridentorchestrator_cr.yaml` pour ajouter les spécifications d'emplacement supplémentaires en fonction de votre configuration de registre.

Images dans un registre

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:24.02"
tridentImage: "<your-registry>/trident:24.02.0"
```

Images dans différents registres

Vous devez ajouter `sig-storage` à la `imageRegistry` pour utiliser différents emplacements de registre.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:24.02"
tridentImage: "<your-registry>/netapp/trident:24.02.0"
```

Étape 6 : créez le `TridentOrchestrator` Et installer Trident

Vous pouvez maintenant créer le `TridentOrchestrator` Et installer Astra Trident. Si vous le souhaitez, vous pouvez aussi aller plus loin "Personnalisez votre installation de Trident" utilisation des attributs dans `TridentOrchestrator` spécifications L'exemple suivant montre une installation dans laquelle les images Trident et CSI se trouvent dans différents registres.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Autosupport Image:  <your-registry>/netapp/trident-autosupport:24.02
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:24.02.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                      true
    Http Request Timeout:       90s
    Image Pull Secrets:
    Image Registry:       <your-registry>/sig-storage
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/netapp/trident:24.02.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Vérifiez l'installation

Il existe plusieurs façons de vérifier votre installation.

À l'aide de `TridentOrchestrator` état

Le statut de `TridentOrchestrator` Indique si l'installation a réussi et affiche la version de Trident installée. Pendant l'installation, l'état de `TridentOrchestrator` modifications de `Installing` à `Installed`. Si vous observez l' `Failed` l'état et l'opérateur ne parvient pas à récupérer lui-même, "vérifiez les journaux".

État	Description
Installation	L'opérateur installe Astra Trident à l'aide de ce module `TridentOrchestrator` CR.
Installé	Astra Trident a été installé avec succès.
Désinstallation	L'opérateur désinstallant Astra Trident, car `spec.uninstall=true`.
Désinstallé	Astra Trident est désinstallé.
Échec	L'opérateur n'a pas pu installer, corriger, mettre à jour ou désinstaller Astra Trident. L'opérateur essaiera automatiquement de récupérer cet état. Si cet état persiste, vous devrez effectuer un dépannage.
Mise à jour	L'opérateur met à jour une installation existante.
Erreur	Le `TridentOrchestrator` n'est pas utilisé. Un autre existe déjà.

Utilisation du statut de création du pod

Vous pouvez vérifier que l'installation d'Astra Trident est terminée en consultant le statut des pods créés :

```
kubectl get pods -n trident

NAME                                      READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw       6/6     Running   0
1m
trident-node-linux-mr6zc                  2/2     Running   0
1m
trident-node-linux-xrp7w                  2/2     Running   0
1m
trident-node-linux-zh2jt                  2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv         1/1     Running   0
3m
```

À l'aide de `tridentctl`

Vous pouvez utiliser `tridentctl` Pour vérifier la version d'Astra Trident installée.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

Déploiement de l'opérateur Trident à l'aide de Helm (mode standard)

Vous pouvez déployer l'opérateur Trident et installer Astra Trident à l'aide de Helm. Ce processus s'applique aux installations dans lesquelles les images de conteneur requises par Astra Trident ne sont pas stockées dans un registre privé. Si vous disposez d'un registre d'images privé, utilisez le "processus de déploiement hors ligne".

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

informations pratiques sur le Tridécouvrez Astra

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Déployer l'opérateur Trident et installer Astra Trident à l'aide de Helm

Avec Trident "Graphique Helm" Vous pouvez déployer l'opérateur Trident et installer Trident en une étape.

Révision "présentation de l'installation" pour vous assurer que les conditions préalables à l'installation sont respectées et que vous avez sélectionné l'option d'installation appropriée pour votre environnement.

Avant de commencer

En plus du "conditions préalables au déploiement" dont vous avez besoin "Version 3 de Helm".

Étapes

1. Ajout du référentiel Astra Trident Helm :

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

2. Utiliser `helm install` et spécifiez un nom pour votre déploiement comme dans l'exemple suivant où `100.2402.0` Est la version d'Astra Trident que vous installez.

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace <trident-namespace>
```

> Si vous avez déjà créé un namespace pour Trident, le `--create-namespace` le paramètre ne crée pas d'espace de noms supplémentaire.

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

Transmettre les données de configuration pendant l'installation

Il existe deux façons de passer les données de configuration au cours de l'installation :

Option	Description
`--values` (ou `-f`)	Spécifiez un fichier YAML avec les remplacements. Ceci peut être spécifié plusieurs fois et le fichier le plus à droite sera prioritaire.
`--set`	Spécifiez les remplacements sur la ligne de commande.

Par exemple, pour modifier la valeur par défaut de `debug`, exécutez ce qui suit `--set` commande où `100.2402.0` Est la version d'Astra Trident que vous installez :

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace trident --set tridentDebug=true
```

Options de configuration

Ce tableau et le `values.yaml` Le fichier, qui fait partie du graphique Helm, fournit la liste des clés et leurs valeurs par défaut.

Option	Description	Valeur par défaut
`nodeSelector`	Libellés des nœuds pour l'affectation des pods	
`podAnnotations`	Annotations de pod	
`deploymentAnnotations`	Annotations de déploiement	

Option	Description	Valeur par défaut
`tolerations`	Tolérances pour l'affectation de pod	
`affinity`	Affinité pour l'affectation de pod	
`tridentControllerPluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Reportez-vous à la section Présentation des pods de contrôleur et des nœuds pour plus d'informations.	
`tridentControllerPluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Reportez-vous à la section Présentation des pods de contrôleur et des nœuds pour plus d'informations.	
`tridentNodePluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Reportez-vous à la section Présentation des pods de contrôleur et des nœuds pour plus d'informations.	
`tridentNodePluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Reportez-vous à la section Présentation des pods de contrôleur et des nœuds pour plus d'informations.	
`imageRegistry`	Identifie le registre du `trident-operator`, `trident`, et autres images. Laissez vide pour accepter la valeur par défaut.	`""`
`imagePullPolicy`	Définit la stratégie d'extraction d'image pour le `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Définit les secrets d'extraction d'image pour le `trident-operator`, `trident`, et autres images.	
`kubeletDir`	Permet de remplacer l'emplacement hôte de l'état interne du kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permet de définir le niveau du journal de l'opérateur Trident sur : `trace`, `debug`, `info`, `warn`, `error`, ou `fatal`.	`"info"`
`operatorDebug`	Permet de définir le niveau du journal de l'opérateur Trident sur DEBUG.	`true`
`operatorImage`	Permet la neutralisation complète de l'image pour `trident-operator`.	`""`
`operatorImageTag`	Permet de remplacer la balise du `trident-operator` image.	`""`
`tridentIPv6`	Permet à Astra Trident de fonctionner dans des clusters IPv6.	`false`
`tridentK8sTimeout`	Remplace le délai d'expiration par défaut de 30 secondes pour la plupart des opérations de l'API Kubernetes (s'il n'est pas égal à zéro, en secondes).	`0`
`tridentHttpRequestTimeout`	Remplace le délai par défaut de 90 secondes pour les requêtes HTTP, par `0s` étant une durée infinie pour le délai d'expiration. Les valeurs négatives ne sont pas autorisées.	`"90s"`

Option	Description	Valeur par défaut
`tridentSilenceAutosupport`	Permet de désactiver le reporting AutoSupport périodique d'Astra Trident.	`false`
`tridentAutosupportImageTag`	Permet de remplacer la balise de l'image pour le conteneur AutoSupport Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permet au conteneur AutoSupport Astra Trident de Phone Home via un proxy HTTP.	`""`
`tridentLogFormat`	Définit le format de journalisation d'Astra Trident (`text` ou `json`).	`"text"`
`tridentDisableAuditLog`	Désactive l'enregistreur d'audit Astra Trident.	`true`
`tridentLogLevel`	Permet de définir le niveau de journal d'Astra Trident sur : `trace`, `debug`, `info`, `warn`, `error`, ou `fatal`.	`"info"`
`tridentDebug`	Permet de définir le niveau du journal d'Astra Trident sur `debug`.	`false`
`tridentLogWorkflows`	Permet d'activer des workflows Astra Trident spécifiques pour la journalisation des traces ou la suppression de journaux.	`""`
`tridentLogLayers`	Permet d'activer des couches Astra Trident spécifiques pour la journalisation des traces ou la suppression des journaux.	`""`
`tridentImage`	Permet la neutralisation complète de l'image pour Astra Trident.	`""`
`tridentImageTag`	Permet de remplacer la balise de l'image pour Astra Trident.	`""`
`tridentProbePort`	Permet de remplacer le port par défaut utilisé pour les sondes de disponibilité/préparation Kubernetes.	`""`
`windows`	Permet d'installer Astra Trident sur le nœud de travail Windows.	`false`
`enableForceDetach`	Permet d'activer la fonction forcer le détachement.	`false`
`excludePodSecurityPolicy`	Exclut la stratégie de sécurité du module opérateur de la création.	`false`
`cloudProvider`	Réglez sur `"Azure"` Lors de l'utilisation d'identités gérées ou d'une identité de cloud sur un cluster AKS. Défini sur AWS lors de l'utilisation d'une identité de cloud sur un cluster EKS.	`""`
`cloudIdentity`	Défini sur l'identité de la charge de travail (« Azure.Workload.Identity/client-ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx ») lors de l'utilisation de l'identité cloud sur un cluster AKS. Définir sur le rôle IAM AWS (« eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role ») lors de l'utilisation de l'identité cloud sur un cluster EKS.	`""`

Présentation des pods de contrôleur et des nœuds

ASTRA Trident s'exécute comme un seul pod de contrôleur, plus un pod de nœud sur chaque nœud worker dans le cluster. Le pod de nœuds doit être exécuté sur n'importe quel hôte sur lequel vous souhaitez potentiellement monter un volume Astra Trident.

Kubernetes "sélecteurs de nœuds" et "tolérances et rejets" sont utilisés pour contraindre un pod à s'exécuter sur un nœud spécifique ou préféré. En utilisant le « ControllerPlugin » et `NodePlugin`, vous pouvez spécifier des contraintes et des remplacements.

- Le plug-in du contrôleur gère le provisionnement et la gestion des volumes, tels que les snapshots et le redimensionnement.
- Le plug-in du nœud permet d'attacher le stockage au nœud.

Déploiement de l'opérateur Trident à l'aide de Helm (mode hors ligne)

Vous pouvez déployer l'opérateur Trident et installer Astra Trident à l'aide de Helm. Ce processus s'applique aux installations dans lesquelles les images de conteneur requises par Astra Trident sont stockées dans un registre privé. Si vous ne disposez pas d'un registre d'images privé, utilisez le "du déploiement standard".

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

informations pratiques sur le Tridécouvrez Astra

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Déployer l'opérateur Trident et installer Astra Trident à l'aide de Helm

Avec Trident "Graphique Helm" Vous pouvez déployer l'opérateur Trident et installer Trident en une étape.

Révision "présentation de l'installation" pour vous assurer que les conditions préalables à l'installation sont respectées et que vous avez sélectionné l'option d'installation appropriée pour votre environnement.

Avant de commencer

En plus du "conditions préalables au déploiement" dont vous avez besoin "Version 3 de Helm".

Étapes

1. Ajout du référentiel Astra Trident Helm :

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

2. Utiliser `helm install` et spécifiez un nom pour votre déploiement et l'emplacement du registre d'images. Votre "Images Trident et CSI" Peut être situé dans un registre ou dans des registres différents, mais toutes les images CSI doivent se trouver dans le même registre. Dans les exemples, `100.2402.0` Est la version d'Astra Trident que vous installez.

Images dans un registre

```
helm install <name> netapp-trident/trident-operator --version
100.2402.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace>
```

Images dans différents registres

Vous devez ajouter `sig-storage` à la `imageRegistry` pour utiliser différents emplacements de registre.

```
helm install <name> netapp-trident/trident-operator --version
100.2402.0 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:24.02 --set tridentImage=<your-
registry>/netapp/trident:24.02.0 --create-namespace --namespace
<trident-namespace>
```

Si vous avez déjà créé un namespace pour Trident, le `--create-namespace` le paramètre ne crée pas d'espace de noms supplémentaire.

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

Transmettre les données de configuration pendant l'installation

Il existe deux façons de passer les données de configuration au cours de l'installation :

Option	Description
`--values` (ou `-f`)	Spécifiez un fichier YAML avec les remplacements. Ceci peut être spécifié plusieurs fois et le fichier le plus à droite sera prioritaire.
`--set`	Spécifiez les remplacements sur la ligne de commande.

Par exemple, pour modifier la valeur par défaut de `debug`, exécutez ce qui suit `--set` commande où `100.2402.0` Est la version d'Astra Trident que vous installez :

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--create-namespace --namespace trident --set tridentDebug=true
```

Options de configuration

Ce tableau et le `values.yaml` Le fichier, qui fait partie du graphique Helm, fournit la liste des clés et leurs valeurs par défaut.

Option	Description	Valeur par défaut
`nodeSelector`	Libellés des nœuds pour l'affectation des pods	
`podAnnotations`	Annotations de pod	
`deploymentAnnotations`	Annotations de déploiement	
`tolerations`	Tolérances pour l'affectation de pod	
`affinity`	Affinité pour l'affectation de pod	
`tridentControllerPluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Reportez-vous à la section "Présentation des pods de contrôleur et des nœuds" pour plus d'informations.	
`tridentControllerPluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Reportez-vous à la section "Présentation des pods de contrôleur et des nœuds" pour plus d'informations.	
`tridentNodePluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Reportez-vous à la section "Présentation des pods de contrôleur et des nœuds" pour plus d'informations.	
`tridentNodePluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Reportez-vous à la section "Présentation des pods de contrôleur et des nœuds" pour plus d'informations.	
`imageRegistry`	Identifie le registre du `trident-operator`, `trident`, et autres images. Laissez vide pour accepter la valeur par défaut.	« »

Option	Description	Valeur par défaut
`imagePullPolicy`	Définit la stratégie d'extraction d'image pour le `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Définit les secrets d'extraction d'image pour le `trident-operator`, `trident`, et autres images.	
`kubeletDir`	Permet de remplacer l'emplacement hôte de l'état interne du kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permet de définir le niveau du journal de l'opérateur Trident sur : `trace`, `debug`, `info`, `warn`, `error`, ou `fatal`.	`"info"`
`operatorDebug`	Permet de définir le niveau du journal de l'opérateur Trident sur DEBUG.	`true`
`operatorImage`	Permet la neutralisation complète de l'image pour `trident-operator`.	« »
`operatorImageTag`	Permet de remplacer la balise du `trident-operator` image.	« »
`tridentIPv6`	Permet à Astra Trident de fonctionner dans des clusters IPv6.	`false`
`tridentK8sTimeout`	Remplace le délai d'expiration par défaut de 30 secondes pour la plupart des opérations de l'API Kubernetes (s'il n'est pas égal à zéro, en secondes).	`0`
`tridentHttpRequestTimeout`	Remplace le délai par défaut de 90 secondes pour les requêtes HTTP, par `0s` étant une durée infinie pour le délai d'expiration. Les valeurs négatives ne sont pas autorisées.	`"90s"`
`tridentSilenceAutosupport`	Permet de désactiver le reporting AutoSupport périodique d'Astra Trident.	`false`
`tridentAutosupportImageTag`	Permet de remplacer la balise de l'image pour le conteneur AutoSupport Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permet au conteneur AutoSupport Astra Trident de Phone Home via un proxy HTTP.	« »

Option	Description	Valeur par défaut
`tridentLogFormat`	Définit le format de journalisation d'Astra Trident (`text` ou `json`).	`"text"`
`tridentDisableAuditLog`	Désactive l'enregistreur d'audit Astra Trident.	`true`
`tridentLogLevel`	Permet de définir le niveau de journal d'Astra Trident sur : `trace`, `debug`, `info`, `warn`, `error`, ou `fatal`.	`"info"`
`tridentDebug`	Permet de définir le niveau du journal d'Astra Trident sur `debug`.	`false`
`tridentLogWorkflows`	Permet d'activer des workflows Astra Trident spécifiques pour la journalisation des traces ou la suppression de journaux.	« »
`tridentLogLayers`	Permet d'activer des couches Astra Trident spécifiques pour la journalisation des traces ou la suppression des journaux.	« »
`tridentImage`	Permet la neutralisation complète de l'image pour Astra Trident.	« »
`tridentImageTag`	Permet de remplacer la balise de l'image pour Astra Trident.	« »
`tridentProbePort`	Permet de remplacer le port par défaut utilisé pour les sondes de disponibilité/préparation Kubernetes.	« »
`windows`	Permet d'installer Astra Trident sur le nœud de travail Windows.	`false`
`enableForceDetach`	Permet d'activer la fonction forcer le détachement.	`false`
`excludePodSecurityPolicy`	Exclut la stratégie de sécurité du module opérateur de la création.	`false`

Personnalisez l'installation de l'opérateur Trident

L'opérateur Trident vous permet de personnaliser l'installation d'Astra Trident à l'aide des attributs du `TridentOrchestrator` spécifications Si vous voulez personnaliser l'installation au-delà de ce qui est `TridentOrchestrator` les arguments permettent, envisagez d'utiliser `tridentctl` Pour générer des manifestes YAML personnalisés à modifier selon les besoins.

Présentation des pods de contrôleur et des nœuds

ASTRA Trident s'exécute comme un seul pod de contrôleur, plus un pod de nœud sur chaque nœud worker

dans le cluster. Le pod de nœuds doit être exécuté sur n'importe quel hôte sur lequel vous souhaitez potentiellement monter un volume Astra Trident.

Kubernetes "sélecteurs de nœuds" et "tolérances et rejets" sont utilisés pour contraindre un pod à s'exécuter sur un nœud spécifique ou préféré. En utilisant le « ControllerPlugin » et `NodePlugin`, vous pouvez spécifier des contraintes et des remplacements.

- Le plug-in du contrôleur gère le provisionnement et la gestion des volumes, tels que les snapshots et le redimensionnement.
- Le plug-in du nœud permet d'attacher le stockage au nœud.

Options de configuration

> `spec.namespace` est spécifié dans `TridentOrchestrator` Pour indiquer l'espace de noms dans lequel Astra Trident est installé. Ce paramètre **ne peut pas être mis à jour après l'installation d'Astra Trident**. Pour tenter de le faire, le `TridentOrchestrator` statut pour passer à `Failed`. Astra Trident n'est pas conçu pour être migré entre les espaces de noms.

Ce tableau est plus détaillé `TridentOrchestrator` attributs.

Paramètre	Description	Valeur par défaut
`namespace`	Espace de noms pour installer Astra Trident dans	`"default"`
`debug`	Activez le débogage pour Astra Trident	`false`
`enableForceDetach`	`ontap-san` et `ontap-san-economy` uniquement. Fonctionne avec Kubernetes non-Grass Node Shutdown (NGN) pour autoriser les administrateurs du cluster à migrer en toute sécurité les workloads avec des volumes montés vers de nouveaux nœuds en cas de problème.	`false`
`windows`	Réglage sur `true` Active l'installation sur les nœuds de travail Windows.	`false`
`cloudProvider`	Réglez sur `"Azure"` Lors de l'utilisation d'identités gérées ou d'une identité de cloud sur un cluster AKS. Défini sur AWS lors de l'utilisation d'une identité de cloud sur un cluster EKS.	`""`
`cloudIdentity`	Défini sur l'identité de la charge de travail (« Azure.Workload.Identity/client-ID: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxx ») lors de l'utilisation de l'identité cloud sur un cluster AKS. Défini sur le rôle IAM AWS (« eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role ») lors de l'utilisation de l'identité cloud sur un cluster EKS.	`""`
`IPv6`	Installez Astra Trident sur IPv6	faux
`k8sTimeout`	Délai d'expiration pour les opérations Kubernetes	`30sec`
`silenceAutosupport`	N'envoyez pas de packs AutoSupport à NetApp automatiquement	`false`

Paramètre	Description	Valeur par défaut
`autosupportImage`	Image conteneur pour la télémétrie AutoSupport	`"netapp/trident-autosupport:24.02"`
`autosupportProxy`	Adresse/port d'un proxy pour l'envoi de AutoSupport Télémétrie	`"http://proxy.example.com:8888"`
`uninstall`	Indicateur utilisé pour désinstaller Astra Trident	`false`
`logFormat`	Format de connexion Astra Trident à utiliser [text,json]	`"text"`
`tridentImage`	Image Astra Trident à installer	`"netapp/trident:24.02"`
`imageRegistry`	Chemin d'accès au registre interne, du format `<registry FQDN>[:port][/subpath]`	`"k8s.gcr.io/sig-storage"` (Kubernetes 1.19+) ou `"quay.io/k8scsi"`
`kubeletDir`	Chemin d'accès au répertoire kubelet de l'hôte	`"/var/lib/kubelet"`
`wipeout`	Liste des ressources à supprimer pour effectuer une suppression complète de Astra Trident	
`imagePullSecrets`	Secrets pour extraire des images d'un registre interne	
`imagePullPolicy`	Définit la stratégie de collecte d'image pour l'opérateur Trident. Les valeurs valides sont : `Always` pour toujours tirer l'image. `IfNotPresent` pour extraire l'image uniquement s'il n'existe pas déjà sur le nœud. `Never` pour ne jamais tirer l'image.	`IfNotPresent`
`controllerPluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Suit le même format que `pod.spec.nodeSelector`.	Pas de valeur par défaut ; facultatif
`controllerPluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Suit le même format que `pod.spec.Tolerations`.	Pas de valeur par défaut ; facultatif
`nodePluginNodeSelector`	Sélecteurs de nœuds supplémentaires pour les pods. Suit le même format que `pod.spec.nodeSelector`.	Pas de valeur par défaut ; facultatif
`nodePluginTolerations`	Remplace les tolérances Kubernetes pour les pods. Suit le même format que `pod.spec.Tolerations`.	Pas de valeur par défaut ; facultatif

Pour plus d'informations sur le formatage des paramètres du pod, reportez-vous à la section "Attribution de pods aux nœuds".

Détails sur le détachement forcé

Forcer le détachement est disponible pour `ontap-san` et `ontap-san-economy` uniquement. Avant d'activer le détachement forcé, l'arrêt non autorisé des nœuds (NGN) doit être activé sur le cluster Kubernetes. Pour plus d'informations, reportez-vous à la section "Kubernetes : arrêt du nœud sans interruption".

> Comme Astra Trident repose sur LES NGN Kubernetes, ne supprimez pas `out-of-service` elle est corrompue jusqu'à ce que toutes les charges de travail non tolérables soient replanifiées. L'application ou la suppression imprudemment de cet outil peut compromettre la protection des données back-end.

Lorsque l'administrateur du cluster Kubernetes a appliqué `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` taint au nœud et `enableForceDetach` est défini sur `true`, Astra Trident déterminera l'état du nœud et :

1. Cessez l'accès aux E/S back-end pour les volumes montés sur ce nœud.
2. Marquez l'objet de nœud Astra Trident en tant que `dirty` (pas sûr pour les nouvelles publications).

> Le contrôleur Trident rejette les nouvelles demandes de volume publiées jusqu'à ce que le nœud soit de nouveau qualifié (après avoir été marqué comme `dirty`) Par le pod du nœud Trident. Tous les workloads planifiés avec une demande de volume persistant montée (même lorsque le nœud du cluster est sain et prêt) ne seront pas acceptés tant qu'Astra Trident ne peut pas vérifier le nœud `clean` (sûr pour les nouvelles publications).

Lorsque l'intégrité du nœud est restaurée et que la taint est supprimée, Astra Trident :

1. Identifiez et nettoyez les chemins publiés obsolètes sur le nœud.
2. Si le nœud est dans un `cleanable` state (le taint hors service a été supprimé et le nœud est dans `Ready` État). Tous les chemins obsolètes et publiés sont propres. Astra Trident reprépare le nœud en tant que `clean` et autoriser les nouveaux volumes publiés sur le nœud.

Exemples de configurations

Vous pouvez utiliser les attributs dans Options de configuration lors de la définition `TridentOrchestrator` pour personnaliser votre installation.

Configuration personnalisée de base

Ceci est un exemple d'installation personnalisée de base.

```
cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
```

Sélecteurs de nœuds

Dans cet exemple, vous installez Astra Trident avec des sélecteurs de nœuds.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  controllerPluginNodeSelector:
    nodetype: master
  nodePluginNodeSelector:
    storage: netapp
```

Nœuds worker Windows

Cet exemple installe Astra Trident sur un nœud worker Windows.

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

Identités gérées sur un cluster AKS

Cet exemple installe Astra Trident pour activer les identités gérées sur un cluster AKS.

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

Identité cloud sur un cluster AKS

Cet exemple installe Astra Trident en vue d'une utilisation avec une identité de cloud sur un cluster AKS.

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

Identité cloud sur un cluster EKS

Cet exemple installe Astra Trident en vue d'une utilisation avec une identité de cloud sur un cluster AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Installation à l'aide de tridentctl

Installation à l'aide de tridentctl

Vous pouvez installer Astra Trident à l'aide de `tridentctl`. Ce processus s'applique aux installations où les images de conteneur requises par Astra Trident sont stockées dans un registre privé ou non. Pour personnaliser votre `tridentctl` déploiement, voir "Personnalisez le déploiement tridentctl".

Informations stratégiques sur Astra Trident 24.02

Vous devez lire les renseignements essentiels suivants sur Astra Trident.

informations pratiques sur le Tridécouvrez Astra

- Kubernetes 1.27 est désormais pris en charge dans Trident. Mise à niveau de Trident avant la mise à niveau de Kubernetes.
- Astra Trident applique rigoureusement l'utilisation de la configuration de chemins d'accès multiples dans les environnements SAN, avec la valeur recommandée de `find_multipaths: no` dans le fichier multipath.conf.

  Utilisation d'une configuration sans chemins d'accès multiples ou de l'utilisation de `find_multipaths: yes` ou `find_multipaths: smart` la valeur du fichier multipath.conf entraînera des échecs de montage. Trident a recommandé l'utilisation de `find_multipaths: no` depuis la version 21.07.

Installation d'Astra Trident à l'aide de `tridentctl`

Révision "présentation de l'installation" pour vous assurer que les conditions préalables à l'installation sont respectées et que vous avez sélectionné l'option d'installation appropriée pour votre environnement.

Avant de commencer

Avant de commencer l'installation, connectez-vous à l'hôte Linux et vérifiez qu'il gère un travail. "Cluster Kubernetes pris en charge" et que vous disposez des privilèges nécessaires.

> Avec OpenShift, utilisez `oc` au lieu de `kubectl` dans tous les exemples qui suivent, et connectez-vous en tant que **system:admin** en premier lieu en cours d'exécution `oc login -u system:admin` ou `oc login -u kube-admin`.

1. Vérifiez votre version Kubernetes :

```
kubectl version
```

2. Vérifiez les privilèges d'administrateur du cluster :

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Vérifiez que vous pouvez lancer un pod qui utilise une image depuis Docker Hub et atteindre votre système de stockage sur le réseau du pod :

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Étape 1 : téléchargez le package du programme d'installation de Trident

Le package du programme d'installation Astra Trident crée un pod Trident, configure les objets CRD utilisés pour maintenir son état et initialise les sidecars CSI pour effectuer des actions telles que le provisionnement et la connexion de volumes aux hôtes du cluster. Téléchargez et extrayez la dernière version du programme d'installation de Trident "La section *Assets* sur GitHub". Mettez à jour *<trident-installer-XX.XX.X.tar.gz>* dans l'exemple avec la version Trident Astra que vous avez sélectionnée.

```
wget https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

Étape 2 : installer Astra Trident

Installez Astra Trident dans l'espace de noms souhaité en exécutant le `tridentctl install` commande. Vous pouvez ajouter des arguments supplémentaires pour spécifier l'emplacement du registre d'images.

Mode standard

```
./tridentctl install -n trident
```

Images dans un registre

```
./tridentctl install -n trident --image-registry <your-registry>
--autosupport-image <your-registry>/trident-autosupport:24.02 --trident
-image <your-registry>/trident:24.02.0
```

Images dans différents registres

Vous devez ajouter `sig-storage` à la `imageRegistry` pour utiliser différents emplacements de registre.

```
./tridentctl install -n trident --image-registry <your-registry>/sig-
storage --autosupport-image <your-registry>/netapp/trident-
autosupport:24.02 --trident-image <your-
registry>/netapp/trident:24.02.0
```

L'état de votre installation devrait ressembler à ceci.

```
....
INFO Starting Trident installation.                namespace=trident
INFO Created service account.
INFO Created cluster role.
INFO Created cluster role binding.
INFO Added finalizers to custom resource definitions.
INFO Created Trident service.
INFO Created Trident secret.
INFO Created Trident deployment.
INFO Created Trident daemonset.
INFO Waiting for Trident pod to start.
INFO Trident pod started.                          namespace=trident
pod=trident-controller-679648bd45-cv2mx
INFO Waiting for Trident REST interface.
INFO Trident REST interface is up.                 version=24.02.0
INFO Trident installation succeeded.
....
```

Vérifiez l'installation

Vous pouvez vérifier votre installation à l'aide de l'état de création du pod ou `tridentctl`.

Utilisation du statut de création du pod

Vous pouvez vérifier que l'installation d'Astra Trident est terminée en consultant le statut des pods créés :

```
kubectl get pods -n trident

NAME                                   READY   STATUS    RESTARTS   AGE
trident-controller-679648bd45-cv2mx    6/6     Running   0          5m29s
trident-node-linux-vgc8n               2/2     Running   0          5m29s
```

> Si le programme d'installation ne s'est pas terminé correctement ou `trident-controller-<generated id>` (`trident-csi-<generated id>` Dans les versions antérieures à 23.01) n'ont pas d'état **en cours d'exécution**, la plate-forme n'était pas installée. Utiliser `-d` à "activer le mode débogage" et de résoudre le problème.

À l'aide de `tridentctl`

Vous pouvez utiliser `tridentctl` Pour vérifier la version d'Astra Trident installée.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

Exemples de configurations

Les exemples suivants présentent des exemples de configuration pour l'installation d'Astra Trident à l'aide de `tridentctl`.

Nœuds Windows

Pour permettre l'exécution d'Astra Trident sur les nœuds Windows :

```
tridentctl install --windows -n trident
```

Forcer le détachement

Pour plus d'informations sur le détachement forcé, voir "Personnalisez l'installation de l'opérateur Trident".

```
tridentctl install --enable-force-detach=true -n trident
```

Personnalisez l'installation tridentctl

Vous pouvez utiliser le programme d'installation d'Astra Trident pour personnaliser l'installation.

En savoir plus sur le programme d'installation

Le programme d'installation d'Astra Trident vous permet de personnaliser les attributs. Par exemple, si vous avez copié l'image Trident dans un référentiel privé, vous pouvez spécifier le nom de l'image à l'aide de `--trident-image`. Si vous avez copié l'image Trident ainsi que les images sidecar CSI nécessaires dans un référentiel privé, il est peut-être préférable de spécifier l'emplacement de ce référentiel à l'aide du `--image-registry` commutateur, qui prend la forme `<registry FQDN>[:port]`.

Si vous utilisez une distribution de Kubernetes, où `kubelet` conserve ses données sur un chemin différent de la normale `/var/lib/kubelet`, vous pouvez spécifier la trajectoire alternative en utilisant `--kubelet-dir`.

Si vous devez personnaliser l'installation au-delà de ce que les arguments du programme d'installation autorisent, vous pouvez également personnaliser les fichiers de déploiement. À l'aide du `--generate-custom-yaml` Le paramètre crée les fichiers YAML suivants dans le programme d'installation `setup` répertoire :

- `trident-clusterrolebinding.yaml`
- `trident-deployment.yaml`
- `trident-crds.yaml`
- `trident-clusterrole.yaml`
- `trident-daemonset.yaml`
- `trident-service.yaml`
- `trident-namespace.yaml`
- `trident-serviceaccount.yaml`
- `trident-resourcequota.yaml`

Après avoir généré ces fichiers, vous pouvez les modifier en fonction de vos besoins, puis les utiliser `--use-custom-yaml` pour installer votre déploiement personnalisé.

```
./tridentctl install -n trident --use-custom-yaml
```

Avec Astra Trident

Préparez le nœud de travail

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

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 RedHat CoreOS ont les outils installés 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 (Nonvolatile Memory Express) sur TCP (NVMe/TCP).

> ONTAP 9.12 ou version ultérieure est recommandé pour NVMe/TCP.

Détection des services de nœud

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

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

Révision des événements

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

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

Examiner les services découverts

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

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

Volumes NFS

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

RHEL 8+

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

Ubuntu

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

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

Volumes iSCSI

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

Fonctionnalités d'auto-rétablissement de l'iSCSI

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

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

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

Avec les fonctionnalités d'autorétablissement iSCSI d'Astra Trident,

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

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

- Sessions iSCSI manquantes
- LUN manquantes

Installez les outils iSCSI

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

Avant de commencer

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

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

- Lorsque vous utilisez des nœuds workers exécutant RHEL/RedHat CoreOS avec iSCSI PVS, spécifiez le `discard` MounOption dans la classe de stockage pour effectuer la réclamation d'espace en ligne. Reportez-vous à la section "Documentation Red Hat".

RHEL 8+

1. Installez les packages système suivants :

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

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

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

3. Définir la numérisation sur manuelle :

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

4. Activer les chemins d'accès multiples :

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

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

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

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

6. Activer et démarrer `iscsi`:

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

Ubuntu

1. Installez les packages système suivants :

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

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

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

3. Définir la numérisation sur manuelle :

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

4. Activer les chemins d'accès multiples :

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

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

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

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

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

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

Volumes NVMe/TCP

Installez les outils NVMe à l'aide des commandes correspondant à votre système d'exploitation.

> - NVMe requiert RHEL 9 ou 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 avec 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
```

Vérifiez l'installation

Après l'installation, vérifiez que chaque nœud du cluster Kubernetes dispose d'un NQN unique via la commande :

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

> ASTRA Trident modifie le `ctrl_device_tmo` Value pour garantir que NVMe ne renonce pas au chemin en cas de panne. Ne modifiez pas ce paramètre.

Configuration et gestion des systèmes back-end

Configuration des systèmes back-end

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

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

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

Azure NetApp Files

Configurer un back-end Azure NetApp Files

Vous pouvez configurer Azure NetApp Files en tant que back-end pour Astra Trident. Vous pouvez relier des volumes NFS et SMB à l'aide d'un back-end Azure NetApp Files. ASTRA Trident prend également en charge la gestion des identifiants à l'aide d'identités gérées pour les clusters Azure Kubernetes Services (AKS).

Détails du pilote Azure NetApp Files

ASTRA 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), *ReadOnlyMey* (ROX), *ReadWriteMaly* (RWX), *ReadWriteOncePod* (RWOP).

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

Considérations

- Le service Azure NetApp Files ne prend pas en charge des volumes de moins de 100 Go. ASTRA Trident crée automatiquement des volumes de 100 Gio si un volume plus petit est demandé.
- Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.

Identités gérées pour AKS

Prise en charge d'Astra Trident "identités gérées" Pour les clusters Azure Kubernetes Services. Pour tirer parti de la gestion rationalisée des informations d'identification offerte par les identités gérées, vous devez disposer des éléments suivants :

- Cluster Kubernetes déployé à l'aide d'AKS
- Identités gérées configurées sur le cluster AKS kubernetes
- ASTRA Trident a été installé et inclut le `cloudProvider` à spécifier `"Azure"`.

Opérateur Trident

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

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

Gouvernail

L'exemple suivant illustre l'installation des ensembles Astra Trident `cloudProvider` À Azure à l'aide de la variable d'environnement `$CP`:

```
helm install trident trident-operator-100.2402.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

Dans l'exemple suivant, Astra Trident et le système sont installés `cloudProvider` marquer 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 comme identité de workload au lieu de fournir des informations d'identification Azure explicites.

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

- Cluster Kubernetes déployé à l'aide d'AKS
- Identité de la charge de travail et émetteur oidc configurés sur le cluster AKS Kubernetes
- ASTRA Trident a été installé et inclut le `cloudProvider` à spécifier `"Azure"` et `cloudIdentity` spécification de l'identité du workload

Opérateur Trident

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

Par exemple :

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

Gouvernail

Définissez les valeurs des indicateurs **cloud-Provider (CP)** et **cloud-Identity (ci)** à l'aide des variables d'environnement suivantes :

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

Dans l'exemple suivant, vous installez Astra Trident et les ensembles `cloudProvider` À Azure à l'aide de la variable d'environnement `$CP` et définit le `cloudIdentity` à l'aide de la variable d'environnement `$CI`:

```
helm install trident trident-operator-100.2402.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI
```

<code>tridentctl</code>

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

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

Dans l'exemple suivant, Astra Trident et le système sont installés `cloud-provider` marquer à `$CP`, et `cloud-identity` à `$CI`:

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

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

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

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 requise pour configurer Azure NetApp Files et créer un volume NFS. Reportez-vous à la section "Azure : configurez Azure NetApp Files et créez un volume NFS".

Pour configurer et utiliser un "Azure NetApp Files" back-end, 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 de cloud sur un cluster AKS.

- Un pool de capacité. Reportez-vous à la section "Microsoft : créez un pool de capacité pour Azure NetApp Files".
- Sous-réseau délégué à Azure NetApp Files. Reportez-vous à la section "Microsoft : déléguer un sous-réseau à Azure NetApp Files".
- `subscriptionID` Depuis un abonnement Azure avec Azure NetApp Files activé.
- `tenantID`, `clientID`, et `clientSecret` à partir d'un "Enregistrement d'applications" Dans Azure Active Directory avec les autorisations suffisantes pour le service Azure NetApp Files. L'enregistrement de l'application doit utiliser l'une des options suivantes :
  - Rôle propriétaire ou contributeur "Prédéfinie par Azure".
  - A "Rôle de contributeur personnalisé" au niveau de l'abonnement (`assignableScopes`) Avec les autorisations suivantes qui sont limitées à ce qu'exige Astra Trident. Après avoir créé le rôle personnalisé, "Attribuez le rôle à l'aide du portail Azure".

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": []
            }
        ]
    }
}
```

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

Exigences supplémentaires pour les volumes SMB

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

- Active Directory configuré et connecté à Azure NetApp Files. Reportez-vous à la section "Microsoft : création et gestion des connexions Active Directory pour Azure NetApp Files".
- Cluster Kubernetes avec un nœud de contrôleur Linux et au moins un nœud worker Windows exécutant Windows Server 2019. Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.
- Au moins un secret Astra Trident contenant vos informations d'identification Active Directory pour que Azure NetApp Files puisse s'authentifier auprès d'Active Directory. Pour générer un secret `smbcreds`:

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

- Un proxy CSI configuré en tant que service Windows. Pour configurer un `csi-proxy`, voir "GitHub : proxy

CSI" ou "GitHub : proxy CSI pour Windows" Pour les nœuds Kubernetes s'exécutant sur Windows.

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

Découvrez les options de configuration NFS et SMB backend pour Azure NetApp Files et passez en revue les exemples de configuration.

Options de configuration du back-end

ASTRA Trident utilise votre configuration back-end (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 qui correspondent au niveau de service et au sous-réseau requis.

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

Les systèmes Azure NetApp Files back-end proposent ces options de configuration.

Paramètre	Description	Valeur par défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	« azure-netapp-files »
`backendName`	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + caractères aléatoires
`subscriptionID`	L'ID d'abonnement de votre abonnement Azure Facultatif lorsque les identités gérées sont activées sur un cluster AKS.	
`tenantID`	ID locataire d'un enregistrement d'application Facultatif lorsque des identités gérées ou des identités de cloud sont utilisées sur un cluster AKS.	
`clientID`	L'ID client d'un enregistrement d'application Facultatif lorsque des identités gérées ou des identités de cloud sont utilisées sur un cluster AKS.	
`clientSecret`	Secret client d'un enregistrement d'application Facultatif lorsque des identités gérées ou des identités de cloud sont utilisées sur un cluster AKS.	
`serviceLevel`	Un de `Standard`, `Premium`, ou `Ultra`	« » (aléatoire)

Paramètre	Description	Valeur par défaut
`location`	Nom de l'emplacement Azure dans lequel les nouveaux volumes seront créés Facultatif lorsque les identités gérées sont activées sur un cluster AKS.	
`resourceGroups`	Liste des groupes de ressources pour le filtrage des ressources découvertes	«[] » (sans filtre)
`netappAccounts`	Liste des comptes NetApp permettant de filtrer les ressources découvertes	«[] » (sans filtre)
`capacityPools`	Liste des pools de capacité pour le filtrage des ressources découvertes	«[] » (sans filtre, aléatoire)
`virtualNetwork`	Nom d'un réseau virtuel avec un sous-réseau délégué	« »
`subnet`	Nom d'un sous-réseau délégué à `Microsoft.Netapp/volumes`	« »
`networkFeatures`	L'ensemble des fonctions de vnet pour un volume peut être `Basic` ou `Standard`. Les fonctions réseau ne sont pas disponibles dans toutes les régions et peuvent être activées dans un abonnement. Spécification `networkFeatures` lorsque la fonctionnalité n'est pas activée, le provisionnement du volume échoue.	« »
`nfsMountOptions`	Contrôle précis des options de montage NFS. Ignoré pour les volumes SMB. Pour monter des volumes à l'aide de NFS version 4.1, incluez `nfsvers=4` Dans la liste des options de montage délimitées par des virgules, choisissez NFS v4.1. Les options de montage définies dans une définition de classe de stockage remplacent les options de montage définies dans la configuration backend.	« nfsvers=3 »
`limitVolumeSize`	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur	« » (non appliqué par défaut)

Paramètre	Description	Valeur par défaut
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple `\{"api": false, "method": true, "discovery": true}`. Ne l'utilisez pas à moins que vous ne soyez en mesure de résoudre les problèmes et que vous ayez besoin d'un vidage détaillé des journaux.	nul
`nasType`	Configurez la création de volumes NFS ou SMB. Les options sont `nfs`, `smb` ou nul. La valeur null par défaut sur les volumes NFS.	`nfs`

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

Autorisations et ressources requises

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

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

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

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

Provisionnement de volume

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

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

Exemples de configurations

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

Configuration minimale

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

Cette configuration est idéale lorsque vous commencez à utiliser Azure NetApp Files et que vous essayez d'autres fonctionnalités, mais dans la pratique, vous voudrez ajouter de l'étendue aux volumes que vous provisionnez.

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

Identités gérées pour AKS

Cette configuration back-end 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 back-end omet `tenantID`, `clientID`, et `clientSecret`, qui sont facultatives lors de l'utilisation d'une identité de nuage.

```
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 de niveau de service spécifique avec filtres de pool de capacité

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

Configuration avancée

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

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

Configuration de pool virtuel

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

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

Définitions des classes de stockage

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

Exemples de définitions utilisant `parameter.selector` légale

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

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

Exemples de définitions pour les volumes SMB

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

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}
```

Utilisation de secrets différents par volume

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

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

Créer le backend

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

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

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

```
tridentctl logs
```

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

Configurer un système Cloud Volumes Service pour Google Cloud backend

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

Détails du pilote Google Cloud

ASTRA Trident offre le `gcp-cvs` pour communiquer avec le bloc d'instruments. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMey* (ROX), *ReadWriteMaly* (RWX), *ReadWriteOncePod* (RWOP).

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

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

Astra Trident peut créer des volumes Cloud Volumes Service dans un des deux "types de service":

- **CVS-Performance** : le type de service Astra Trident par défaut. Ce type de service aux performances optimisées est parfaitement adapté aux charges de travail de production qui exigent des performances élevées. Le type de service CVS-Performance est une option matérielle prenant en charge les volumes d'une taille minimale de 100 Gio. Vous pouvez choisir l'une des options "trois niveaux de service":
  - `standard`
  - `premium`
  - `extreme`

- **CVS**: Le type de service CVS fournit une haute disponibilité zonale avec des niveaux de performance limités à modérés. Le type de service CVS est une option logicielle utilisant des pools de stockage pour prendre en charge des volumes de 1 Gio. Le pool de stockage peut contenir jusqu'à 50 volumes dans lesquels tous les volumes partagent la capacité et les performances du pool. Vous pouvez choisir l'une des options "deux niveaux de service":
  - `standardsw`
  - `zoneredundantstandardsw`

Ce dont vous avez besoin

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

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

Options de configuration du back-end

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

Paramètre	Description	Valeur par défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	« gcp-cvs »
`backendName`	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + partie de la clé API
`storageClass`	Paramètre facultatif utilisé pour spécifier le type de service CVS. Utiliser `software` Pour sélectionner le type de service CVS. Sinon, Astra Trident suppose un type de service CVS-Performance (`hardware`).	
`storagePools`	Type de service CVS uniquement. Paramètre facultatif utilisé pour spécifier les pools de stockage pour la création du volume.	
`projectNumber`	Numéro de projet de compte Google Cloud. La valeur est disponible sur la page d'accueil du portail Google Cloud.	
`hostProjectNumber`	Requis si l'utilisation d'un réseau VPC partagé. Dans ce scénario, `projectNumber` est le projet de service, et `hostProjectNumber` est le projet hôte.	

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

`allowedTopologies`	Pour activer l'accès inter-région, votre définition de classe de stockage pour `allowedTopologies` doit inclure toutes les régions. Par exemple :
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Options de provisionnement de volumes

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

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

Exemples de type de service CVS-Performance

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

Exemple 1 : configuration minimale

Il s’agit de la configuration back-end minimale avec le type de service CVS-Performance par défaut et 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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
```

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

Exemple 2 : configuration du niveau de service

Dans cet exemple, nous présentons les options de configuration du back-end, y compris les niveaux de service et les valeurs par défaut des volumes.

```
---
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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
```

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

Utilisation de cet échantillon `storage` pour configurer des pools virtuels et `StorageClasses` cela leur renvoie. Reportez-vous à la section Définitions des classes de stockage pour voir comment les classes de stockage ont été définies.

Ici, des valeurs par défaut spécifiques sont définies pour tous les pools virtuels, qui définissent le `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 sa propre définition `serviceLevel`, et certains pools remplacent les valeurs par défaut. Des étiquettes de pools virtuels ont été utilisées pour différencier les pools 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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
```

```
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
    performance: extreme
    protection: extra
  serviceLevel: extreme
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
    performance: extreme
    protection: standard
  serviceLevel: extreme
- labels:
    performance: premium
    protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
    performance: premium
    protection: standard
  serviceLevel: premium
- labels:
    performance: standard
```

```
    serviceLevel: standard
```

Définitions des classes de stockage

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

Exemple de classe de stockage

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: 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 au premier pool virtuel. Il s'agit du seul pool offrant des performances extrêmes avec une réserve Snapshot de 10 %.
- La dernière classe de stockage (`cvs-extra-protection`) appelle tout pool de stockage qui fournit une réserve d'instantanés de 10%. Astra Trident décide du pool virtuel sélectionné et s'assure que les exigences de la réserve de snapshots sont respectées.

Exemples de type de service CVS

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

Exemple 1 : configuration minimale

Il s’agit de la configuration back-end minimale 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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
```

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

Exemple 2 : configuration du pool de stockage

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

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQDaT+Oui9FBAw19
    L1AGEkrYU5xd9K5NlO5jMkIFND5wCD+Nv+jd1GvtFRLaLK5RvXyF5wzvztmODNS+
    qtScpQ+5cFpQkuGtv9U9+N6qtuVYYO3b504Kp5CtqVPJCgMJaK2j8pZTIqUiMum/
    5/Y9oTbZrjAHSMgJm2nHzFq2X0rqVMaHghI6ATm4DOuWx8XGWKTGIPlc0qPqJlqS
    LLaWOH4VIZQZCAyW5IUp9CAmwqHgdG0uhFNfCgMmED6PBUvVLsLvcq86X+QSWR9k
    ETqElj/sGCenPF7ti1DhGBFafd9hPnxg9PZY29ArEZwY9G/ZjZQX7WPgs0VvxiNR
    DxZRC3GXAgMBAAECggEACn5c59bG/qnVEVI1CwMAalM5M2z09JFhlLlljKwntNPj
    Vilw2eTW2+UE7HbJru/S7KQgA5Dnn9kvCraEahPRuddUMrD0vG4kTl/IODV6uFuk
    Y0sZfbqd4jMUQ21smvGsqFzwloYWS5qzO1W83ivXH/HW/iqkmY2eW+EPRS/hwSSu
    SscR+SojI7PB0BWSJhlV4yqYf3vcD/D95el2CVHfRCkL85DKumeZ+yHEnpiXGZAE
    t8xSs4a5OOPm6NHhevCw2a/UQ95/foXNUR450HtbjieJo5o+FF6EYZQGfU2ZHZO8
    37FBKuaJkdGW5xqaI9TL7aqkGkFMF4F2qvOZM+vy8QKBgQD4oVuOkJDlhkTHP86W
    esFlw1kpWyJR9ZA7LI0g/rVpslnX+XdDq0WQf4umdLNau5hYEH9LU6ZSGs1Xk3/B
    NHwR6OXFuqEKNiu83d0zSlHhTy7PZpOZdj5a/vVvQfPDMz7OvsqLRd7YCAbdzuQ0
    +Ahq0Ztwvg0HQ64hdW0ukpYRRwKBgQDgyHj98oqswoYuIa+pP1yS0pPwLmjwKyNm
    /HayzCp+Qjiyy7Tzg8AUqlH1Ou83XbV428jvg7kDhO7PCCKFq+mMmfqHmTpb0Maq
    KpKnZg4ipsqP1yHNNEoRmcailXbwIhCLewMqMrggUiLOmCw4PscL5nK+4GKu2XE1
    jLqjWAZFMQKBgFHkQ9XXRAJ1kR3XpGHoGN890pZOkCVSrqju6aUef/5KYlFCt8ew
    F/+aIxM2iQSvmWQYOvVCnhuY/F2GFaQ7d0om3decuwI0CX/xy7PjHMkLXa2uaZs4
    WR17sLduj62RqXRLX0c0QkwBiNFyHbRcpdkZJQujbYMhBa+7j7SxT4BtAoGAWMWT
    UucocRXZm/pdvz9wteNH3YDWnJLMxm1KC06qMXbBoYrliY4sm3ywJWMC+iCd/H8A
    Gecxd/xVu5mA2L2N3KMq18Zhz8Th0G5DwKyDRJgOQ0Q46yuNXOoYEjlo4Wjyk8Me
    +tlQ8iK98E0UmZnhTgfSpSNElbz2AqnzQ3MN9uECgYAqdvdVPnKGfvdtZ2DjyMoJ
    E89UIC41WjjJGmHsd8W65+3X0RwMzKMT6aZc5tK9J5dHvmWIETnbM+lTImdBBFga
    NWOC6f3r2xbGXHhaWSl+nobpTuvlo56ZRJVvVk7lFMsiddzMuHH8pxfgNJemwA4P
    ThDHCejv035NNV6KyoO0tA==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
```

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

Et la suite ?

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

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

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

```
tridentctl logs
```

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

Configurer un système NetApp HCI ou SolidFire backend

Découvrez comment créer et utiliser un back-end Element avec votre installation Astra Trident.

Détails du pilote d'élément

ASTRA Trident offre le `solidfire-san` pilote de stockage pour communiquer avec le cluster. Les modes d'accès pris en charge sont : *ReadWriteOnce* (RWO), *ReadOnlyMey* (ROX), *ReadWriteMaly* (RWX), *ReadWriteOncePod* (RWOP).

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

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

Avant de commencer

Vous aurez besoin des éléments suivants avant de créer un back-end d'élément.

- Système de stockage pris en charge exécutant le logiciel Element.
- Identifiants de locataire ou administrateur de cluster NetApp HCI/SolidFire pouvant gérer les volumes
- Tous vos nœuds workers Kubernetes doivent avoir installé les outils iSCSI appropriés. Reportez-vous à la section "informations de préparation du nœud de travail".

Options de configuration du back-end

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

Paramètre	Description	Valeur par défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	Toujours « solidfire-san ».
`backendName`	Nom personnalisé ou système back-end de stockage	"SolidFire_" + adresse IP de stockage (iSCSI)
`Endpoint`	MVIP pour le cluster SolidFire avec les identifiants de locataire	
`SVIP`	Port et adresse IP de stockage (iSCSI)	
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes.	« »
`TenantName`	Nom du locataire à utiliser (créé si introuvable)	
`InitiatorIFace`	Limitez le trafic iSCSI à une interface hôte spécifique	« par défaut »
`UseCHAP`	Utilisez CHAP pour authentifier iSCSI. ASTRA Trident utilise le protocole CHAP.	vrai
`AccessGroups`	Liste des ID de groupes d'accès à utiliser	Recherche l'ID d'un groupe d'accès nommé « trident »

Paramètre	Description	Valeur par défaut
`Types`	Spécifications de QoS	
`limitVolumeSize`	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur	« » (non appliqué par défaut)
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, {“api”:false, “méthode”:true}	nul

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

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

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

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

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

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

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

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

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: "<svip>:3260"
TenantName: "<tenant>"
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
- labels:
    performance: gold
    cost: '4'
  zone: us-east-1a
```

```
  type: Gold
- labels:
    performance: silver
    cost: '3'
  zone: us-east-1b
  type: Silver
- labels:
    performance: bronze
    cost: '2'
  zone: us-east-1c
  type: Bronze
- labels:
    performance: silver
    cost: '1'
  zone: us-east-1d
```

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

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

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

Trouvez plus d'informations

- "Groupes d'accès de volume"

Pilotes SAN de ONTAP

Présentation du pilote SAN ONTAP

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

Détails du pilote SAN ONTAP

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

> Si vous utilisez Astra Control pour la protection, la restauration et la mobilité, lisez Compatibilité du pilote Astra Control.

Conducteur	Protocole	Mode Volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-san`	ISCSI	Bloc	RWO, ROX, RWX, RWOP	Pas de système de fichiers, périphérique de bloc brut
`ontap-san`	ISCSI	Système de fichiers	RWO, RWOP ROX et RWX ne sont pas disponibles en mode de volume du système de fichiers.	`xfs`, `ext3`, `ext4`
`ontap-san`	NVMe/TCP Reportez-vous à la section Autres considérations relatives au NVMe/TCP.	Bloc	RWO, ROX, RWX, RWOP	Pas de système de fichiers, périphérique de bloc brut

Conducteur	Protocole	Mode Volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-san`	NVMe/TCP Reportez-vous à la section Autres considérations relatives au NVMe/TCP.	Système de fichiers	RWO, RWOP ROX et RWX ne sont pas disponibles en mode de volume du système de fichiers.	`xfs`, `ext3`, `ext4`
`ontap-san-economy`	ISCSI	Bloc	RWO, ROX, RWX, RWOP	Pas de système de fichiers, périphérique de bloc brut
`ontap-san-economy`	ISCSI	Système de fichiers	RWO, RWOP ROX et RWX ne sont pas disponibles en mode de volume du système de fichiers.	`xfs`, `ext3`, `ext4`

Compatibilité du pilote Astra Control

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

- Utiliser `ontap-san-economy` uniquement si le nombre d'utilisations du volume persistant doit être supérieur à "Limites de volume ONTAP prises en charge".
- Utiliser `ontap-nas-economy` uniquement si le nombre d'utilisations du volume persistant doit 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 d'avoir besoin en termes de protection des données, de reprise sur incident ou de mobilité.

Autorisations utilisateur

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

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

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

Autres considérations relatives au NVMe/TCP

ASTRA Trident prend en charge le protocole NVMe (non-volatile Memory Express) à l'aide du `ontap-san` pilote comprenant :

- IPv6
- Copies Snapshot et clones de volumes NVMe
- Redimensionnement d'un volume NVMe
- Importation d'un volume NVMe créé en dehors d'Astra Trident afin que son cycle de vie puisse être géré par Astra Trident
- Chemins d'accès multiples natifs NVMe
- Arrêt normal ou sans gracieuse des nœuds K8s (24.02)

ASTRA Trident ne prend pas en charge :

- DH-HMAC-CHAP pris en charge par NVMe de manière native
- Chemins d'accès multiples du mappeur de périphériques (DM)
- Cryptage LUKS

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

Découvrez les exigences et les options d'authentification pour la configuration d'un back-end ONTAP avec des pilotes SAN ONTAP.

De formation

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

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

Tous vos nœuds workers Kubernetes doivent avoir installé les outils iSCSI appropriés. Reportez-vous à la section "Préparez le nœud de travail" pour plus d'informations.

Authentifiez le back-end ONTAP

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

- Basé sur les informations d'identification : nom d'utilisateur et mot de passe pour un utilisateur ONTAP disposant des autorisations requises. Il est recommandé d'utiliser un rôle de connexion de sécurité

prédéfini, par exemple `admin` ou `vsadmin` Pour garantir une compatibilité maximale avec les versions ONTAP.

- Basé sur des certificats : Astra Trident peut également communiquer avec un cluster ONTAP à l'aide d'un certificat installé sur le système back-end. Dans ce cas, la définition backend doit contenir des valeurs encodées Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance, le cas échéant (recommandé).

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

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

Activer l'authentification basée sur les informations d'identification

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

Voici un exemple de définition du back-end :

YAML

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

JSON

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

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

Activer l'authentification basée sur certificat

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

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

Un flux de travail type comprend les étapes suivantes.

Étapes

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Authentifier les connexions avec le protocole CHAP bidirectionnel

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

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

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

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

Comment cela fonctionne

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

- Configuration du protocole CHAP sur le SVM :
  - Si le type de sécurité initiateur par défaut du SVM est none (défini par défaut) **et** il n'y a pas de LUN préexistantes déjà présentes dans le volume, Astra Trident définit le type de sécurité par défaut sur `CHAP` Et procédez à la configuration de l'initiateur CHAP et du nom d'utilisateur cible et des secrets.
  - Si le SVM contient des LUN, Astra Trident n'active pas le protocole CHAP sur le SVM. Cela garantit que l'accès aux LUNs déjà présentes sur le SVM n'est pas restreint.
- Configuration de l'initiateur CHAP et du nom d'utilisateur cible et des secrets ; ces options doivent être spécifiées dans la configuration backend (comme indiqué ci-dessus).

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

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

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

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

```
cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "password",
    "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 ne seront pas affectées. Elles restent actives si les identifiants sont mis à jour par Astra Trident sur le SVM. Les nouvelles connexions utiliseront les informations d'identification mises à jour et les connexions existantes continuent de rester actives. La déconnexion et la reconnexion des anciens volumes persistants se traduront par l'utilisation des identifiants mis à jour.

Options et exemples de configuration des SAN ONTAP

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

Options de configuration du back-end

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

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

Paramètre	Description	Valeur par défaut
`chapInitiatorSecret`	Secret de l'initiateur CHAP. Requis si `useCHAP=true`	« »
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »
`chapTargetInitiatorSecret`	Secret de l'initiateur cible CHAP. Requis si `useCHAP=true`	« »
`chapUsername`	Nom d'utilisateur entrant. Requis si `useCHAP=true`	« »
`chapTargetUsername`	Nom d'utilisateur cible. Requis si `useCHAP=true`	« »
`clientCertificate`	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
`clientPrivateKey`	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »
`trustedCACertificate`	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification basée sur des certificats.	« »
`username`	Le nom d'utilisateur devait communiquer avec le cluster ONTAP. Utilisé pour l'authentification basée sur les identifiants.	« »
`password`	Mot de passe requis pour communiquer avec le cluster ONTAP. Utilisé pour l'authentification basée sur les identifiants.	« »
`svm`	Serveur virtuel de stockage à utiliser	Dérivé d'un SVM `managementLIF` est spécifié
`storagePrefix`	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être modifié ultérieurement. Pour mettre à jour ce paramètre, vous devez créer un nouveau backend.	`trident`
`limitAggregateUsage`	Echec du provisionnement si l'utilisation est supérieure à ce pourcentage. Si vous utilisez un système Amazon FSX pour le système back-end NetApp ONTAP, ne spécifiez pas `limitAggregateUsage`. Le fourni `fsxadmin` et `vsadmin` Ne contiennent pas les autorisations requises pour récupérer l'utilisation d'agrégats et le limiter à l'aide d'Astra Trident.	« » (non appliqué par défaut)
`limitVolumeSize`	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN.	« » (non appliqué par défaut)
`lunsPerFlexvol`	Nombre maximal de LUN par FlexVol, doit être compris dans la plage [50, 200]	`100`

Paramètre	Description	Valeur par défaut
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, {"api":false, "method":true} Ne pas utiliser sauf si vous effectuez un dépannage et que vous avez besoin d'un vidage de journal détaillé.	`null`
`useREST`	Paramètre booléen pour utiliser les API REST de ONTAP. **Aperçu technique** `useREST` est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur `true`, Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au `ontap` client supplémentaire. Ceci est satisfait par le pré-défini `vsadmin` et `cluster-admin` rôles. `useREST` N'est pas pris en charge par MetroCluster. `useREST` Est pleinement qualifié pour NVMe/TCP.	`false`
`sanType`	Utilisez pour sélectionner `iscsi` Pour iSCSI ou `nvme` Pour NVMe/TCP.	`iscsi` si vide

Options de configuration back-end pour les volumes de provisionnement

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

Paramètre	Description	Valeur par défaut
`spaceAllocation`	Allocation d'espace pour les LUN	« vrai »
`spaceReserve`	Mode de réservation d'espace ; « aucun » (fin) ou « volume » (épais)	« aucun »
`snapshotPolicy`	Règle Snapshot à utiliser	« aucun »
`qosPolicy`	QoS policy group à affecter pour les volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end. Avec Astra Trident, les groupes de règles de QoS doivent être utilisés avec ONTAP 9.8 ou version ultérieure. Nous recommandons l'utilisation d'un groupe de règles de qualité de service non partagé et nous assurer que le groupe de règles est appliqué à chaque composant individuellement. Un groupe de règles de QoS partagé appliquera le plafond du débit total de toutes les charges de travail.	« »

Paramètre	Description	Valeur par défaut
`adaptiveQosPolicy`	Groupe de règles de QoS adaptative à attribuer aux volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end	« »
`snapshotReserve`	Pourcentage de volume réservé pour les snapshots	« 0 » si `snapshotPolicy` est « aucun », sinon « »
`splitOnClone`	Séparer un clone de son parent lors de sa création	« faux »
`encryption`	Activez NetApp Volume Encryption (NVE) sur le nouveau volume. La valeur par défaut est `false`. Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE".	« faux »
`luksEncryption`	Activez le cryptage LUKS. Reportez-vous à la section "Utiliser la configuration de clé unifiée Linux (LUKS)". Le cryptage LUKS n'est pas pris en charge pour NVMe/TCP.	« »
`securityStyle`	Style de sécurité pour les nouveaux volumes	`unix`
`tieringPolicy`	Règle de hiérarchisation à utiliser « aucun »	« Snapshot uniquement » pour la configuration SVM-DR antérieure à ONTAP 9.5

Exemples de provisionnement de volumes

Voici un exemple avec des valeurs par défaut définies :

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

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

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

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

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

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
          _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                   online     RW         10GB     5.00GB    0%
          _pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d
                                   online     RW       5.79GB     5.50GB    0%
          _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                   online     RW          1GB    511.8MB    0%
3 entries were displayed.
```

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

Exemples de configuration minimaux

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

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

Exemple de SAN ONTAP

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 d'économie SAN ONTAP

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

Exemple MetroCluster

Vous pouvez configurer le back-end pour éviter d'avoir à mettre à jour manuellement la définition du back-end après le basculement et le rétablissement pendant "Réplication et restauration des SVM".

Pour un basculement et un rétablissement fluides, préciser le SVM en utilisant `managementLIF` et omettre le `dataLIF` et `svm` paramètres. Par exemple :

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

Exemple d'authentification basée sur un certificat

Dans cet exemple de configuration de base `clientCertificate`, `clientPrivateKey`, et `trustedCACertificate` (Facultatif, si vous utilisez une autorité de certification approuvée) est renseigné `backend.json` Et prendre les valeurs codées en base64 du certificat client, de la clé privée et du certificat CA de confiance, respectivement.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Exemples CHAP bidirectionnels

Ces exemples créent un backend avec `useCHAP` réglez sur `true`.

Exemple CHAP de SAN ONTAP

```
---
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 CHAP d'économie SAN ONTAP

```
---
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

Un SVM doit être configuré avec NVMe sur votre back-end ONTAP. Il s'agit d'une configuration back-end de base pour 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
```

Exemples de systèmes back-end avec pools virtuels

Dans ces exemples de fichiers de définition back-end, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, tels que `spaceReserve` aucune, `spaceAllocation` lors de la fausse idée, et `encryption` faux. Les pools virtuels sont définis dans la section stockage.

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

Dans ces exemples, certains pools de stockage sont définis comme étant leurs propres `spaceReserve`, `spaceAllocation`, et `encryption` et certains pools remplacent les valeurs par défaut.

Exemple de SAN ONTAP

```
---
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 SAN ONTAP

```
---
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'
```

Mappage des systèmes back-end aux classes de stockage

Les définitions de classe de stockage suivantes font référence au Exemples de systèmes back-end avec pools virtuels. À l'aide du `parameters.selector` Chaque classe de stockage indique quels pools virtuels peuvent être utilisés pour héberger un volume. Les aspects définis dans le pool virtuel sélectionné seront définis pour le volume.

- Le `protection-gold` StorageClass est mappé sur le premier pool virtuel du `ontap-san` back-end. Il s'agit du seul pool offrant une protection de niveau Gold.

```
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 mappé au deuxième et au troisième pool virtuel dans `ontap-san` back-end. Ce sont les seuls pools offrant un niveau de protection autre que Gold.

```
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 mappé sur le troisième pool virtuel dans `ontap-san-economy` back-end. Il s'agit du seul pool offrant la configuration du pool de stockage pour l'application 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 mappé sur le second pool virtuel dans `ontap-san` back-end. Il s'agit de la seule piscine offrant une protection de niveau argent et 20000 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 mappé sur le troisième pool virtuel dans `ontap-san` back-end et le quatrième pool virtuel dans `ontap-san-economy` back-end. Il s'agit des seules offres de pool 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` La classe de stockage est mappée sur `testAPP` pool virtuel dans `ontap-san` pilote avec `sanType: nvme`. Il s'agit de la seule offre de piscine `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"
```

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

Pilotes NAS ONTAP

Présentation du pilote NAS ONTAP

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

Détails du pilote NAS ONTAP

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

> Si vous utilisez Astra Control pour la protection, la restauration et la mobilité, lisez Compatibilité du pilote Astra Control.

Conducteur	Protocole	Mode Volume	Modes d'accès pris en charge	Systèmes de fichiers pris en charge
`ontap-nas`	NFS PME	Système de fichiers	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`
`ontap-nas-economy`	NFS PME	Système de fichiers	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`
`ontap-nas-flexgroup`	NFS PME	Système de fichiers	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`

Compatibilité du pilote Astra Control

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

> - Utiliser `ontap-san-economy` uniquement si le nombre d'utilisations du volume persistant doit être supérieur à "Limites de volume ONTAP prises en charge".
> - Utiliser `ontap-nas-economy` uniquement si le nombre d'utilisations du volume persistant doit 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 d'avoir besoin en termes de protection des données, de reprise sur incident ou de mobilité.

Autorisations utilisateur

Astra Trident devrait être exécuté en tant qu'administrateur de ONTAP ou du SVM, généralement à l'aide du `admin` utilisateur du cluster ou un `vsadmin` Utilisateur d'un SVM ou un utilisateur avec un autre nom qui a le même rôle.

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

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

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

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

Découvrez les exigences, les options d'authentification et les règles d'exportation pour la configuration d'un back-end ONTAP avec des pilotes NAS ONTAP.

De formation

- Pour tous les systèmes back-end ONTAP, Astra Trident requiert au moins un agrégat affecté à la 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 le `ontap-nas` Pilote et une classe Bronze qui utilise le `ontap-nas-economy` une seule.
- Tous vos nœuds workers Kubernetes doivent avoir installé les outils NFS appropriés. Reportez-vous à la section "ici" pour en savoir plus.
- Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows. Reportez-vous à la section Préparez-vous au provisionnement des volumes SMB pour plus d'informations.

Authentifiez le back-end ONTAP

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

- Basé sur les informations d'identification : ce mode requiert des autorisations suffisantes pour le backend ONTAP. Il est recommandé d'utiliser un compte associé à un rôle de connexion de sécurité prédéfini, par exemple `admin` ou `vsadmin` Pour garantir une compatibilité maximale avec les versions ONTAP.
- Basé sur un certificat : ce mode nécessite l'installation d'un certificat sur le back-end pour qu'Astra Trident puisse communiquer avec un cluster ONTAP. Dans ce cas, la définition backend doit contenir des valeurs encodées Base64 du certificat client, de la clé et du certificat d'autorité de certification de confiance, le cas échéant (recommandé).

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

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

Activer l'authentification basée sur les informations d'identification

Astra Trident nécessite les identifiants d'un administrateur SVM-scoped/cluster-scoped pour communiquer avec le ONTAP backend. Il est recommandé d'utiliser des rôles standard prédéfinis tels que `admin` ou

`vsadmin`. Il est ainsi possible d'assurer une compatibilité avec les futures versions d'ONTAP et d'exposer les API de fonctionnalités à utiliser avec les futures versions d'Astra Trident. Un rôle de connexion de sécurité personnalisé peut être créé et utilisé avec Astra Trident, mais il n'est pas recommandé.

Voici un exemple de définition du back-end :

YAML

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

JSON

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

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

Activez l'authentification basée sur les certificats

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

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

Un flux de travail type comprend les étapes suivantes.

Étapes

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Gestion des règles d'exportation NFS

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

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

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

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

Gérez les règles d'exportation de manière dynamique

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

> N'utilisez pas NAT (Network Address Translation) lorsque vous utilisez des stratégies d'exportation dynamiques. Avec NAT, le contrôleur de stockage voit l'adresse NAT front-end et non l'adresse IP réelle de l'hôte. L'accès sera donc refusé lorsqu'aucune correspondance n'est trouvée dans les règles d'exportation.

Exemple

Deux options de configuration doivent être utilisées. Voici un exemple de définition de back-end :

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

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

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

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

individuelle à périmètre global présentes sur les nœuds du worker.

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

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

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

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

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

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

Pour les systèmes back-end existants, mise à jour du système back-end avec `tridentctl update backend` S'assure qu'Astra Trident gère automatiquement les règles d'exportation. Cela créera une nouvelle export policy nommée après que l'UUID du back-end et les volumes présents sur le back-end utiliseront la nouvelle export policy créée lorsqu'ils sont à nouveau montés.

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

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

Préparez-vous au provisionnement des volumes SMB

Avec un peu de préparation supplémentaire, vous pouvez provisionner des volumes SMB à l'aide de `ontap-nas` pilotes.

> On doit configurer les protocoles NFS et SMB/CIFS sur le SVM pour créer un `ontap-nas-economy` Volume SMB pour ONTAP sur site. La configuration de l'un de ces protocoles entraîne l'échec de la création du volume SMB.

Avant de commencer

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

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

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

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

Étapes

1. Pour ONTAP sur site, vous pouvez choisir de créer un partage SMB ou Astra 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 à l'aide de l' "Console de gestion Microsoft" Dossier partagé snap-in ou à l'aide de l'interface de ligne de commande ONTAP. Pour créer les partages SMB à l'aide de l'interface de ligne de commandes ONTAP :

   a. Si nécessaire, créez la structure du chemin d'accès au répertoire pour le partage.

      Le `vserver cifs share create` commande vérifie le chemin spécifié dans l'option -path lors de la

création du partage. Si le chemin spécifié n'existe pas, la commande échoue.

b. Créer un partage SMB associé au SVM spécifié :

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

c. Vérifiez que le partage a été créé :

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

> Reportez-vous à la section "Créez un partage SMB" pour en savoir plus.

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

Paramètre	Description	Exemple
`smbShare`	Vous pouvez indiquer 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 commandes ONTAP ; un nom permettant à Astra Trident de créer le partage SMB. Vous pouvez également laisser vide le paramètre pour empêcher l'accès à un partage commun aux volumes. Ce paramètre est facultatif pour les ONTAP sur site. Ce paramètre est requis pour Amazon FSX pour les systèmes back-end ONTAP et ne peut pas être vide.	`smb-share`
`nasType`	**Doit être défini sur `smb`.** si elle est nulle, la valeur par défaut est `nfs`.	`smb`
`securityStyle`	Style de sécurité pour les nouveaux volumes. **Doit être défini sur `ntfs` ou `mixed` Pour les volumes SMB.**	`ntfs` ou `mixed` Pour les volumes SMB
`unixPermissions`	Mode pour les nouveaux volumes. **Doit rester vide pour les volumes SMB.**	« »

Options et exemples de configuration du NAS ONTAP

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

Options de configuration du back-end

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

Paramètre	Description	Valeur par défaut
`version`		Toujours 1
`storageDriverName`	Nom du pilote de stockage	« ontap-nas », « ontap-nas-economy », « ontap-nas-flexgroup », « ontap-san », « ontap-san-economy »
`backendName`	Nom personnalisé ou système back-end de stockage	Nom du pilote + "_" + dataLIF
`managementLIF`	Adresse IP d'un cluster ou d'une LIF de gestion SVM Un nom de domaine complet (FQDN) peut être spécifié. Peut être défini pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide de l'indicateur IPv6. Les adresses IPv6 doivent être définies entre crochets, par exemple `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]`. Pour un basculement MetroCluster transparent, reportez-vous au Exemple MetroCluster.	« 10.0.0.1 », « [2001:1234:abcd::fefe] »
`dataLIF`	Adresse IP de la LIF de protocole. Nous vous recommandons de spécifier `dataLIF`. Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. Peut être modifié après le réglage initial. Reportez-vous à la section . Peut être défini pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide de l'indicateur IPv6. Les adresses IPv6 doivent être définies entre crochets, par exemple `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]`. **Omettre pour MetroCluster.** Voir Exemple MetroCluster.	Adresse spécifiée ou dérivée d'un SVM, si non spécifiée (non recommandé)

Paramètre	Description	Valeur par défaut
`svm`	Serveur virtuel de stockage à utiliser **Omettre pour MetroCluster.** Voir Exemple MetroCluster.	Dérivé d'un SVM `managementLIF` est spécifié
`autoExportPolicy`	Activer la création et la mise à jour automatiques des règles d'exportation [booléennes]. À l'aide du `autoExportPolicy` et `autoExportCIDRs` Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	faux
`autoExportCIDRs`	Liste des CIDR permettant de filtrer les adresses IP des nœuds Kubernetes par rapport à quand `autoExportPolicy` est activé. À l'aide du `autoExportPolicy` et `autoExportCIDRs` Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	["0.0.0.0/0", ":/0"]"
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »
`clientCertificate`	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
`clientPrivateKey`	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »
`trustedCACertificate`	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification par certificat	« »
`username`	Nom d'utilisateur pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants	
`password`	Mot de passe pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants	
`storagePrefix`	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être mis à jour une fois que vous l'avez défini	« trident »
`limitAggregateUsage`	Echec du provisionnement si l'utilisation est supérieure à ce pourcentage. **Ne s'applique pas à Amazon FSX pour ONTAP**	« » (non appliqué par défaut)
`limitVolumeSize`	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN, et la `qtreesPerFlexvol` L'option permet de personnaliser le nombre maximal de qtree par FlexVol.	« » (non appliqué par défaut)
`lunsPerFlexvol`	Nombre maximal de LUN par FlexVol, doit être compris dans la plage [50, 200]	« 100 »

Paramètre	Description	Valeur par défaut
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Exemple, {"api":false, "method":true} Ne pas utiliser `debugTraceFlags` à moins que vous ne soyez en mesure de dépanner et que vous ayez besoin d'un vidage détaillé des journaux.	nul
`nasType`	Configurez la création de volumes NFS ou SMB. Les options sont `nfs`, `smb` ou nul. La valeur null par défaut sur les volumes NFS.	`nfs`
`nfsMountOptions`	Liste des options de montage NFS séparée par des virgules. Les options de montage des volumes Kubernetes persistants sont généralement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Astra Trident utilisera les options de montage spécifiées dans le fichier de configuration du système back-end. Si aucune option de montage n'est spécifiée dans la classe de stockage ou le fichier de configuration, Astra Trident ne définit aucune option de montage sur un volume persistant associé.	« »
`qtreesPerFlexvol`	Nombre maximal de qtrees par FlexVol, qui doit être compris dans la plage [50, 300]	« 200 »
`smbShare`	Vous pouvez indiquer 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 commandes ONTAP ; un nom permettant à Astra Trident de créer le partage SMB. Vous pouvez également laisser vide le paramètre pour empêcher l'accès à un partage commun aux volumes. Ce paramètre est facultatif pour les ONTAP sur site. Ce paramètre est requis pour Amazon FSX pour les systèmes back-end ONTAP et ne peut pas être vide.	`smb-share`
`useREST`	Paramètre booléen pour utiliser les API REST de ONTAP. **Aperçu technique** `useREST` est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur `true`, Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au `ontap` client supplémentaire. Ceci est satisfait par le pré-défini `vsadmin` et `cluster-admin` rôles. `useREST` N'est pas pris en charge par MetroCluster.	faux

Options de configuration back-end pour les volumes de provisionnement

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

Paramètre	Description	Valeur par défaut
`spaceAllocation`	Allocation d'espace pour les LUN	« vrai »
`spaceReserve`	Mode de réservation d'espace ; « aucun » (fin) ou « volume » (épais)	« aucun »
`snapshotPolicy`	Règle Snapshot à utiliser	« aucun »
`qosPolicy`	QoS policy group à affecter pour les volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end	« »
`adaptiveQosPolicy`	Groupe de règles de QoS adaptative à attribuer aux volumes créés. Choisissez une de qosPolicy ou adaptiveQosPolicy par pool de stockage/back-end. Non pris en charge par l'économie ontap-nas.	« »
`snapshotReserve`	Pourcentage de volume réservé pour les snapshots	« 0 » si `snapshotPolicy` est « aucun », sinon « »
`splitOnClone`	Séparer un clone de son parent lors de sa création	« faux »
`encryption`	Activez NetApp Volume Encryption (NVE) sur le nouveau volume. La valeur par défaut est `false`. Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE".	« faux »
`tieringPolicy`	Règle de hiérarchisation à utiliser « aucun »	« Snapshot uniquement » pour la configuration SVM-DR antérieure à ONTAP 9.5
`unixPermissions`	Mode pour les nouveaux volumes	« 777 » pour les volumes NFS ; vide (non applicable) pour les volumes SMB
`snapshotDir`	Contrôle l'accès au `.snapshot` répertoire	« faux »
`exportPolicy`	Export policy à utiliser	« par défaut »
`securityStyle`	Style de sécurité pour les nouveaux volumes. Prise en charge de NFS `mixed` et `unix` styles de sécurité. SMB prend en charge `mixed` et `ntfs` styles de sécurité.	NFS par défaut est `unix`. SMB par défaut est `ntfs`.

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

Exemples de provisionnement de volumes

Voici un exemple avec des valeurs par défaut définies :

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

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

Le calcul est le suivant :

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

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

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
                 _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                      online     RW         10GB     5.00GB    0%
                 _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                      online     RW          1GB    511.8MB    0%
2 entries were displayed.
```

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

Exemples de configuration minimaux

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

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

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 FlexGroup NAS ONTAP

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

Exemple MetroCluster

Vous pouvez configurer le back-end pour éviter d'avoir à mettre à jour manuellement la définition du back-end après le basculement et le rétablissement pendant "Réplication et restauration des SVM".

Pour un basculement et un rétablissement fluides, préciser le SVM en utilisant `managementLIF` et omettre le `dataLIF` et `svm` paramètres. Par exemple :

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

Exemple de volumes SMB

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

Exemple d'authentification basée sur un certificat

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

```
---
version: 1
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 règle d'export automatique

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

```
---
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` Utilisation d'une adresse IPv6.

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

Exemple d'Amazon FSX pour ONTAP avec des volumes SMB

Le `smbShare` Paramètre obligatoire 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_
```

Exemples de systèmes back-end avec pools virtuels

Dans les exemples de fichiers de définition back-end présentés ci-dessous, des valeurs par défaut spécifiques sont définies pour tous les pools de stockage, tels que `spaceReserve` aucune, `spaceAllocation` lors de la fausse idée, et `encryption` faux. Les pools virtuels sont définis dans la section stockage.

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

volumes de groupe par étiquette.

Dans ces exemples, certains pools de stockage sont définis comme étant leurs propres `spaceReserve`, `spaceAllocation`, et `encryption` et certains pools remplacent les valeurs par défaut.

Exemple de NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: 'false'
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
    app: msoffice
    cost: '100'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
    adaptiveQosPolicy: adaptive-premium
- labels:
    app: slack
    cost: '75'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    department: legal
    creditpoints: '5000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    app: wordpress
```

```
    cost: '50'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
    app: mysqldb
    cost: '25'
  zone: us_east_1d
  defaults:
    spaceReserve: volume
    encryption: 'false'
    unixPermissions: '0775'
```

Exemple de FlexGroup NAS ONTAP

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

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

Exemple d'économie NAS ONTAP

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

```
    encryption: 'false'
    unixPermissions: '0775'
```

Mappage des systèmes back-end aux classes de stockage

Les définitions de classe de stockage suivantes se rapportent à Exemples de systèmes back-end avec pools virtuels. À l'aide du `parameters.selector` Chaque classe de stockage indique quels pools virtuels peuvent être utilisés pour héberger un volume. Les aspects définis dans le pool virtuel sélectionné seront définis pour le volume.

- Le `protection-gold` StorageClass sera mappé au premier et au deuxième pool virtuel de la `ontap-nas-flexgroup` back-end. Il s'agit des seuls pools offrant une protection de niveau Gold.

```
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 mappé au troisième et au quatrième pool virtuel du `ontap-nas-flexgroup` back-end. 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 mappé sur le quatrième pool virtuel du `ontap-nas` back-end. Il s'agit du seul pool offrant la configuration du pool de stockage pour l'application 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"
```

- TThe `protection-silver-creditpoints-20k` StorageClass sera mappé sur le troisième pool virtuel du `ontap-nas-flexgroup` back-end. Il s'agit de la seule piscine offrant une protection de niveau argent et 20000 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 mappé sur le troisième pool virtuel du `ontap-nas` back-end et le second pool virtuel dans `ontap-nas-economy` back-end. Il s'agit des seules offres de pool 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"
```

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

Mise à jour `dataLIF` après la configuration initiale

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

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

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

Amazon FSX pour NetApp ONTAP

Utilisez Astra Trident avec Amazon FSX pour NetApp ONTAP

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

Présentation

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

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

Considérations

- Volumes SMB :
  - Les volumes SMB sont pris en charge à l'aide de `ontap-nas` conducteur uniquement.
  - Les volumes SMB ne sont pas pris en charge par le module d'extension Astra Trident EKS.
  - Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows.
- Avant Astra Trident 24.02, les volumes créés sur des systèmes de fichiers Amazon FSX pour lesquels les sauvegardes automatiques sont activées ne pouvaient pas être supprimés par Trident. Pour éviter ce problème dans Astra Trident 24.02 ou version ultérieure, spécifiez le `fsxFilesystemID`, AWS `apiRegion`, AWS `apikey`Et AWS `secretKey` Dans le fichier de configuration back-end pour AWS FSX pour ONTAP.

> Si vous spécifiez un rôle IAM pour Astra Trident, vous pouvez omettre de spécifier le `apiRegion`, `apiKey`, et `secretKey` Champs vers Astra Trident de manière explicite. Pour plus d'informations, reportez-vous à la section "Exemples et options de configuration de FSX pour ONTAP".

Détails du pilote FSX pour ONTAP

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

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

Pour plus d'informations sur le pilote, reportez-vous à la section "Pilotes NAS" et "Pilotes SAN".

Authentification

Astra Trident propose deux modes d'authentification.

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

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

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

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

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

Identité cloud pour EKS

L'identité cloud permet aux pods Kubernetes d'accéder aux ressources AWS en s'authentifiant en tant que rôle IAM AWS au lieu de fournir des informations d'identification AWS explicites.

Pour tirer parti de l'identité cloud dans AWS, vous devez disposer des éléments suivants :

- Cluster Kubernetes déployé à l'aide d'EKS
- ASTRA Trident a été installé et inclut le `cloudProvider` spécification `"AWS"` et `cloudIdentity` Spécification du rôle IAM AWS.

Opérateur Trident

Pour installer Astra Trident à l'aide de l'opérateur Trident, modifiez `tridentorchestrator_cr.yaml` à régler `cloudProvider` à `"AWS"` et jeu `cloudIdentity` Vers le rôle IAM AWS.

Par exemple :

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Gouvernail

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

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Dans l'exemple suivant, vous installez Astra Trident et les ensembles `cloudProvider` à `AWS` à l'aide de la variable d'environnement `$CP` Et définit l'identité cloud à l'aide de la variable d'environnement `$CI`:

```
helm install trident trident-operator-100.2402.0.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="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Dans l'exemple suivant, Astra Trident et le système sont installés `cloud-provider` marquer à `$CP`, et `cloud-identity` à `$CI`:

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

Trouvez plus d'informations

- "Documentation Amazon FSX pour NetApp ONTAP"
- "Billet de blog sur Amazon FSX pour NetApp ONTAP"

Intégration d'Amazon FSX pour NetApp ONTAP

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

De formation

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

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

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

- Astra Trident prend en charge les volumes SMB montés sur des pods qui s'exécutent uniquement sur des nœuds Windows. Reportez-vous à la section Préparez-vous au provisionnement des volumes SMB pour plus d'informations.

Intégration des pilotes SAN et NAS de ONTAP

> Si vous configurez la configuration pour les volumes SMB, vous devez lire Préparez-vous au provisionnement des volumes SMB avant de créer le backend.

Étapes

1. Déployez Astra Trident avec l'un des "méthodes de déploiement".
2. Collectez votre nom DNS de la LIF de gestion du SVM. Par exemple, recherchez le sur l'interface de ligne de commandes AWS `DNSName` entrée sous `Endpoints` → `Management` après avoir exécuté la commande suivante :

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

3. Créer et installer des certificats pour "Authentification NAS backend" ou "Authentification SAN backend".

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

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

YAML

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

JSON

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

Vous pouvez également créer un fichier back-end en utilisant les informations d'identification du SVM (nom d'utilisateur et mot de passe) stockées dans AWS Secret Manager, comme illustré ci-dessous :

YAML

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

JSON

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

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

- "Configurer un système back-end avec les pilotes NAS ONTAP"
- "Configurer un système back-end avec les pilotes SAN ONTAP"

Préparez-vous au provisionnement des volumes SMB

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

Avant de commencer

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

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

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

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

Étapes

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

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

   b. Créer un partage SMB associé au SVM spécifié :

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

   c. Vérifiez que le partage a été créé :

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

> Reportez-vous à la section "Créez un partage SMB" pour en savoir plus.

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

Paramètre	Description	Exemple
`smbShare`	Vous pouvez indiquer l'un des éléments suivants : 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 nom permettant à Astra Trident de créer le partage SMB. Ce paramètre est requis pour Amazon FSX pour les systèmes back-end ONTAP.	`smb-share`
`nasType`	**Doit être défini sur `smb`.** si elle est nulle, la valeur par défaut est `nfs`.	`smb`
`securityStyle`	Style de sécurité pour les nouveaux volumes. **Doit être défini sur `ntfs` ou `mixed` Pour les volumes SMB.**	`ntfs` ou `mixed` Pour les volumes SMB
`unixPermissions`	Mode pour les nouveaux volumes. **Doit rester vide pour les volumes SMB.**	« »

Exemples et options de configuration de FSX pour ONTAP

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

Options de configuration du back-end

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

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

Paramètre	Description	Exemple
managementLIF	Adresse IP d'un cluster ou d'une LIF de gestion SVM Un nom de domaine complet (FQDN) peut être spécifié. Peut être défini pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide de l'indicateur IPv6. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	« 10.0.0.1 », « [2001:1234:abcd::fefe] »
dataLIF	Adresse IP de la LIF de protocole. **Pilotes NAS ONTAP**: Nous vous recommandons de spécifier dataLIF. Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. Peut être modifié après le réglage initial. Reportez-vous à la section . **Pilotes SAN ONTAP** : ne pas spécifier pour iSCSI. Astra Trident utilise le mappage de LUN sélectif de ONTAP pour découvrir les LIFs iSCI nécessaires pour établir une session multi-chemins. Un avertissement est généré si dataLIF est explicitement défini. Peut être défini pour utiliser des adresses IPv6 si Astra Trident a été installé à l'aide de l'indicateur IPv6. Les adresses IPv6 doivent être définies entre crochets, telles que [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	

Paramètre	Description	Exemple
`autoExportPolicy`	Activer la création et la mise à jour automatiques des règles d'exportation [booléennes]. À l'aide du `autoExportPolicy` et `autoExportCIDRs` Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	`false`
`autoExportCIDRs`	Liste des CIDR permettant de filtrer les adresses IP des nœuds Kubernetes par rapport à quand `autoExportPolicy` est activé. À l'aide du `autoExportPolicy` et `autoExportCIDRs` Avec Astra Trident, il peut gérer automatiquement les règles d'exportation.	« [« 0.0.0.0/0 », «:/0 »] »
`labels`	Ensemble d'étiquettes arbitraires au format JSON à appliquer aux volumes	« »
`clientCertificate`	Valeur encodée en Base64 du certificat client. Utilisé pour l'authentification par certificat	« »
`clientPrivateKey`	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification par certificat	« »
`trustedCACertificate`	Valeur encodée en Base64 du certificat CA de confiance. Facultatif. Utilisé pour l'authentification basée sur des certificats.	« »
`username`	Nom d'utilisateur pour la connexion au cluster ou au SVM. Utilisé pour l'authentification basée sur les identifiants. Par exemple, vsadmin.	
`password`	Mot de passe pour se connecter au cluster ou au SVM. Utilisé pour l'authentification basée sur les identifiants.	
`svm`	Serveur virtuel de stockage à utiliser	Dérivé si une LIF de gestion SVM est spécifiée.

Paramètre	Description	Exemple
`storagePrefix`	Préfixe utilisé pour le provisionnement des nouveaux volumes dans la SVM. Ne peut pas être modifié après sa création. Pour mettre à jour ce paramètre, vous devez créer un nouveau backend.	`trident`
`limitAggregateUsage`	**Ne pas spécifier pour Amazon FSX pour NetApp ONTAP** `fsxadmin` et `vsadmin` Ne contiennent pas les autorisations requises pour récupérer l'utilisation d'agrégats et le limiter à l'aide d'Astra Trident.	Ne pas utiliser.
`limitVolumeSize`	Echec du provisionnement si la taille du volume demandé est supérieure à cette valeur. Restreint également la taille maximale des volumes qu'il gère pour les qtrees et les LUN, et la `qtreesPerFlexvol` L'option permet de personnaliser le nombre maximal de qtree par FlexVol.	« » (non appliqué par défaut)
`lunsPerFlexvol`	Le nombre maximal de LUN par FlexVol doit être compris dans la plage [50, 200]. SAN uniquement.	`100`
`debugTraceFlags`	Indicateurs de débogage à utiliser lors du dépannage. Par exemple, {"api":false, "méthode":true} ne pas utiliser `debugTraceFlags` à moins que vous ne soyez en mesure de dépanner et que vous ayez besoin d'un vidage détaillé des journaux.	nul

Paramètre	Description	Exemple
`nfsMountOptions`	Liste des options de montage NFS séparée par des virgules. Les options de montage des volumes Kubernetes persistants sont généralement spécifiées dans les classes de stockage, mais si aucune option de montage n'est spécifiée dans une classe de stockage, Astra Trident utilisera les options de montage spécifiées dans le fichier de configuration du système back-end. Si aucune option de montage n'est spécifiée dans la classe de stockage ou le fichier de configuration, Astra Trident ne définit aucune option de montage sur un volume persistant associé.	« »
`nasType`	Configurez la création de volumes NFS ou SMB. Les options sont `nfs`, `smb`, ou nul. **Doit être défini sur `smb` Pour les volumes SMB.** la valeur NULL est définie par défaut sur les volumes NFS.	`nfs`
`qtreesPerFlexvol`	Nombre maximal de qtrees par FlexVol, qui doit être compris dans la plage [50, 300]	`200`
`smbShare`	Vous pouvez indiquer l'un des éléments suivants : 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 nom permettant à Astra Trident de créer le partage SMB. Ce paramètre est requis pour Amazon FSX pour les systèmes back-end ONTAP.	`smb-share`

Paramètre	Description	Exemple
`useREST`	Paramètre booléen pour utiliser les API REST de ONTAP. **Aperçu technique** `useREST` est fourni sous forme d'aperçu technique ** qui est recommandé pour les environnements de test et non pour les charges de travail de production. Lorsqu'il est réglé sur `true`, Astra Trident va utiliser les API REST de ONTAP pour communiquer avec le système back-end. Cette fonctionnalité requiert ONTAP 9.11.1 et versions ultérieures. En outre, le rôle de connexion ONTAP utilisé doit avoir accès au `ontap` client supplémentaire. Ceci est satisfait par le pré-défini `vsadmin` et `cluster-admin` rôles.	`false`
`aws`	Vous pouvez spécifier ce qui suit dans le fichier de configuration d'AWS FSX pour ONTAP : - `fsxFilesystemID`: Spécifiez l'ID du système de fichiers AWS FSX. - `apiRegion`: Nom de la région de l'API AWS. - `apikey`: Clé d'API AWS. - `secretKey`: Clé secrète AWS.	"" "" ""
`credentials`	Spécifiez les informations d'identification du SVM FSX à stocker dans AWS Secret Manager. - `name`: Amazon Resource Name (ARN) du secret, qui contient les informations d'identification de SVM. - `type`: Défini sur `awsarn`. Reportez-vous à la section "Créez un secret AWS secrets Manager" pour en savoir plus.	

Mise à jour `dataLIF` après la configuration initiale

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

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

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

Options de configuration back-end pour les volumes de provisionnement

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

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

Paramètre	Description	Valeur par défaut
`encryption`	Activez NetApp Volume Encryption (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 back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE".	`false`
`luksEncryption`	Activez le cryptage LUKS. Reportez-vous à la section "Utiliser la configuration de clé unifiée Linux (LUKS)". SAN uniquement.	« »
`tieringPolicy`	Règle de hiérarchisation à utiliser `none`	`snapshot-only` Pour la configuration SVM-DR antérieure à ONTAP 9.5
`unixPermissions`	Mode pour les nouveaux volumes. **Laisser vide pour les volumes SMB.**	« »
`securityStyle`	Style de sécurité pour les nouveaux volumes. Prise en charge de NFS `mixed` et `unix` styles de sécurité. SMB prend en charge `mixed` et `ntfs` styles de sécurité.	NFS par défaut est `unix`. SMB par défaut est `ntfs`.

Exemples de configurations

Configuration de la classe de stockage pour les volumes SMB

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

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

Configuration pour AWS FSX pour ONTAP avec le gestionnaire de code secret

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

Configurez le module complémentaire Astra Trident EKS version 23.10 sur le cluster EKS

ASTRA Trident simplifie la gestion du stockage Amazon FSX pour NetApp ONTAP dans Kubernetes pour que vos développeurs et administrateurs puissent donner la priorité au déploiement d'applications. Le module complémentaire Astra Trident EKS inclut les derniers correctifs de sécurité et de bogues, et il est validé par AWS pour fonctionner avec Amazon EKS. Le module complémentaire EKS vous permet de vous assurer de manière cohérente que vos clusters Amazon EKS sont sécurisés et stables et de réduire la quantité de travail à effectuer pour installer, configurer et mettre à jour des modules complémentaires.

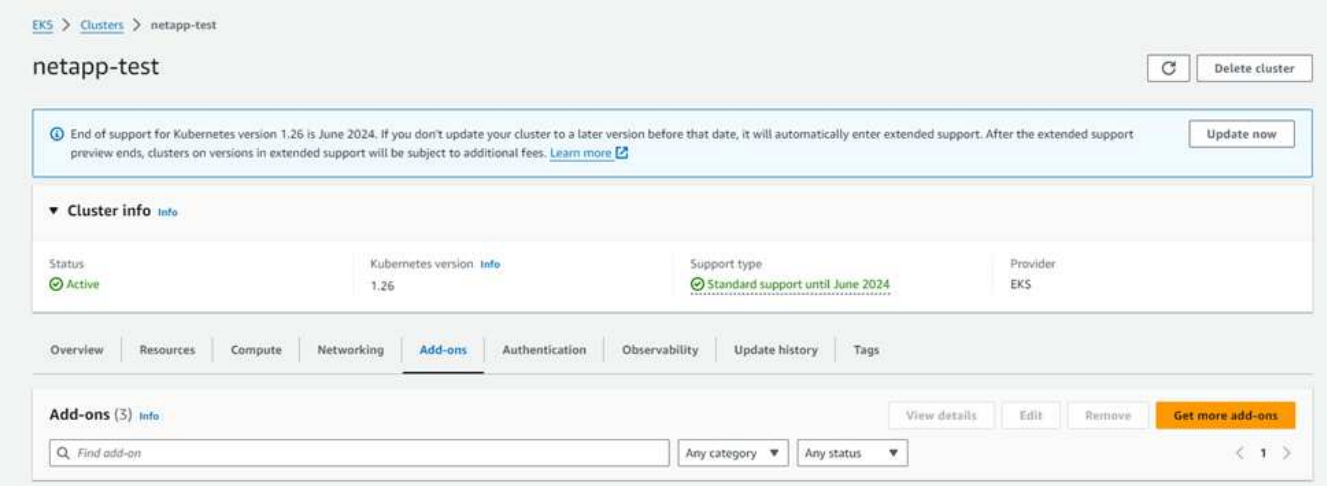
Prérequis

Vérifiez les points suivants avant de configurer le module complémentaire Astra Trident pour AWS EKS :

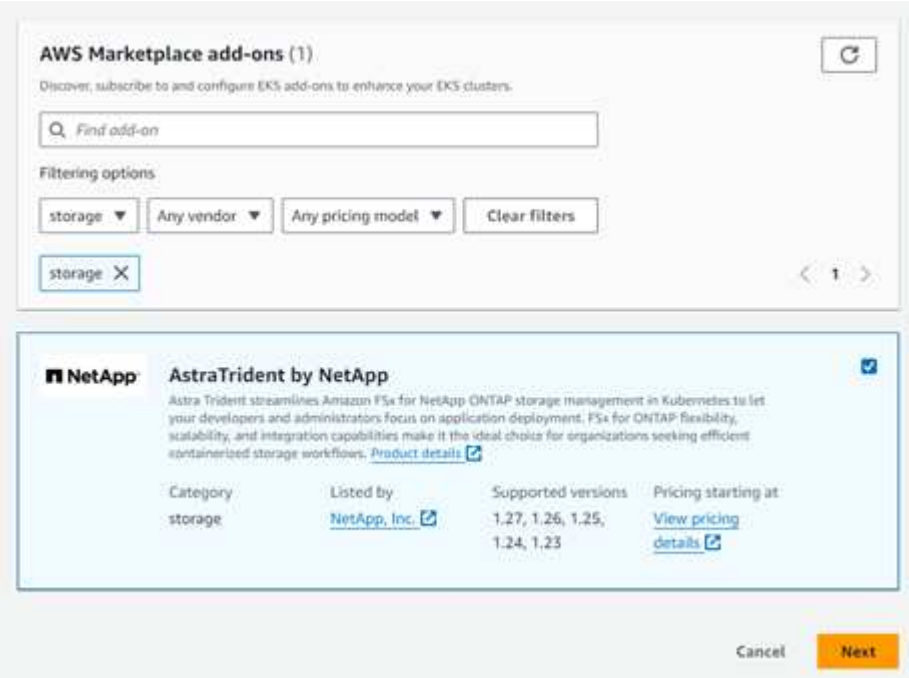
- Un compte de cluster Amazon EKS avec abonnement complémentaire
- Autorisations AWS sur AWS Marketplace :
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Type ami : Amazon Linux 2 (AL2_x86_64) ou Amazon Linux 2 Arm (AL2_ARM_64)
- Type de nœud : AMD ou ARM
- Un système de fichiers Amazon FSX pour NetApp ONTAP

Étapes

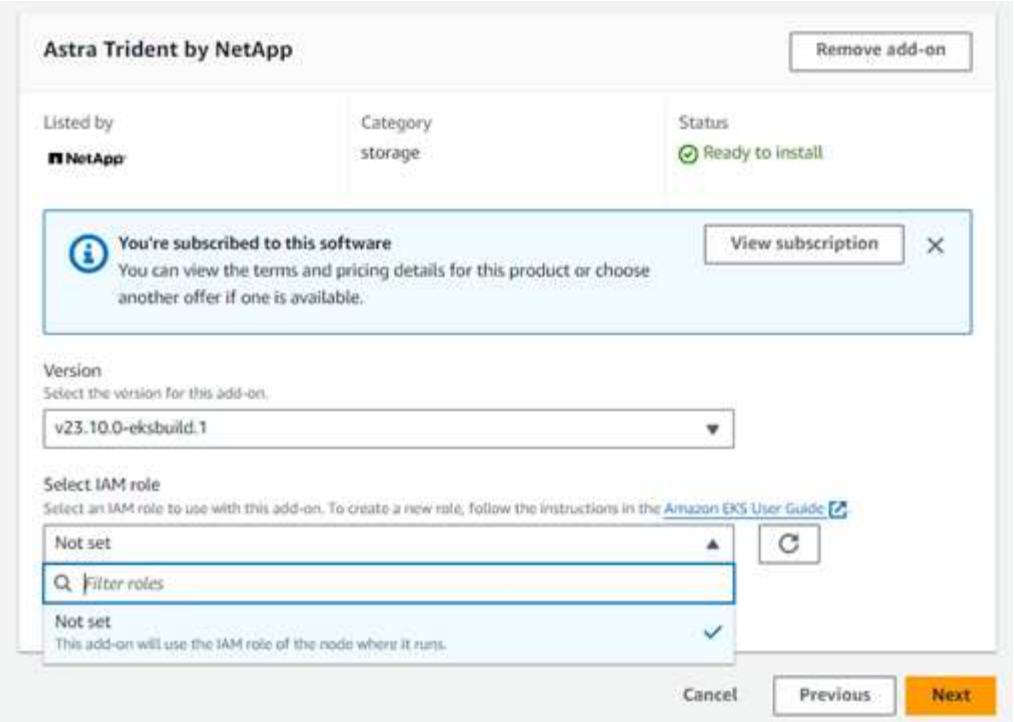
1. Sur votre cluster EKS Kubernetes, accédez à l'onglet **Add-ons**.



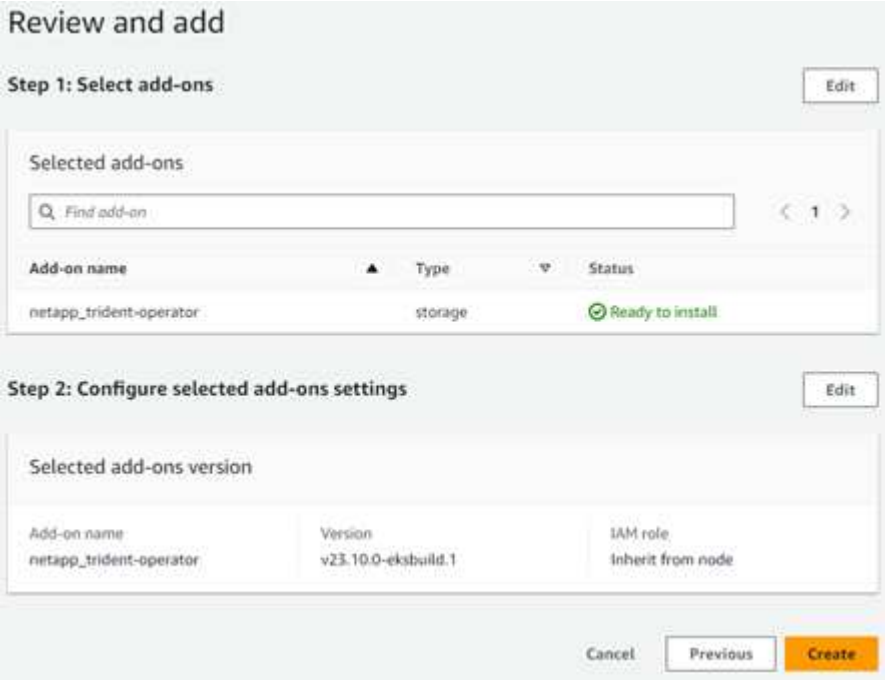
2. Accédez à **add-ons** AWS Marketplace et choisissez la catégorie *Storage*.



3. Localisez **AstraTrident par NetApp** et cochez la case correspondant au module complémentaire Astra Trident.
4. Choisissez la version souhaitée du module complémentaire.



5. Sélectionnez l'option rôle IAM à hériter du nœud.
6. Configurez tous les paramètres facultatifs selon les besoins et sélectionnez **Suivant**.



7. Sélectionnez **Créer**.
8. Vérifiez que l'état du complément est *Active*.

Installez/désinstallez le module complémentaire Astra Trident EKS à l'aide de l'interface de ligne de commande

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

Les exemples de commandes suivants installent le module complémentaire Astra Trident EKS :
`eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.`
`eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.1` (avec une version dédiée)

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

La commande suivante désinstalle le module complémentaire Astra Trident EKS :
`eksctl delete addon --cluster K8s-arm --name netapp_trident-operator`

Création de systèmes back-end avec kubectl

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

`TridentBackendConfig`

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

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

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

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

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

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

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

Vous pouvez également consulter les exemples de la "programme d'installation trident" répertoire des exemples de configuration pour la plate-forme/le service de stockage souhaité.

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

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

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

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

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

Présentation des étapes

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

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

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

Étape 1 : créez un code secret Kubernetes

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

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: t@Ax@7q(>
```

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

Description des champs secrets de la plate-forme de stockage	Secret	Description des champs
Azure NetApp Files	ID client	ID client d'un enregistrement d'application

Description des champs secrets de la plate-forme de stockage	Secret	Description des champs
Cloud Volumes Service pour GCP	id_clé_privée	ID de la clé privée. Partie de la clé API pour le compte de service GCP avec le rôle d'administrateur CVS
Cloud Volumes Service pour GCP	clé_privée	Clé privée. Partie de la clé API pour le compte de service GCP avec le rôle d'administrateur CVS
Element (NetApp HCI/SolidFire)	Point final	MVIP pour le cluster SolidFire avec les identifiants de locataire
ONTAP	nom d'utilisateur	Nom d'utilisateur pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants
ONTAP	mot de passe	Mot de passe pour la connexion au cluster/SVM. Utilisé pour l'authentification basée sur les identifiants
ONTAP	ClientPrivateKey	Valeur encodée en Base64 de la clé privée du client. Utilisé pour l'authentification basée sur des certificats
ONTAP	ChapUsername	Nom d'utilisateur entrant. Requis si useCHAP=vrai. Pour `ontap-san` et `ontap-san-economy`
ONTAP	Chapeau InitiatorSecret	Secret de l'initiateur CHAP. Requis si useCHAP=vrai. Pour `ontap-san` et `ontap-san-economy`
ONTAP	ChapTargetUsername	Nom d'utilisateur cible. Requis si useCHAP=vrai. Pour `ontap-san` et `ontap-san-economy`
ONTAP	ChapTargetInitiatorSecret	Secret de l'initiateur cible CHAP. Requis si useCHAP=vrai. Pour `ontap-san` et `ontap-san-economy`

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

Étape 2 : créez le `TridentBackendConfig` CR

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

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

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

Étape 3 : vérifier l'état du `TridentBackendConfig` CR

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

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

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

La phase peut prendre l'une des valeurs suivantes :

- `Bound`: Le `TridentBackendConfig` La demande de modification est associée à un back-end, et ce backend contient `configRef` réglez sur `TridentBackendConfig` ID de CR.
- `Unbound`: Représenté en utilisant `""`. Le `TridentBackendConfig` l'objet n'est pas lié à un back-end. Tout nouveau `TridentBackendConfig` Les CRS sont dans cette phase par défaut. Une fois la phase modifiée, elle ne peut plus revenir à Unbound.
- `Deleting`: Le `TridentBackendConfig` CR `deletionPolicy` a été configuré pour supprimer. Lorsque le `TridentBackendConfig` La demande de modification est supprimée, elle passe à l'état Suppression.
    - Si aucune demande de volume persistant n'existe sur le back-end, supprimez le

`TridentBackendConfig` Il en résultera la suppression du système back-end et du système Astra Trident `TridentBackendConfig` CR.

  - Si un ou plusieurs ESV sont présents sur le back-end, il passe à l'état de suppression. Le `TridentBackendConfig` La CR entre ensuite la phase de suppression. Le back-end et `TridentBackendConfig` Sont supprimés uniquement après la suppression de tous les ESV.

- `Lost`: Le back-end associé à l' `TridentBackendConfig` Le CR a été accidentellement ou délibérément supprimé et le `TridentBackendConfig` La CR a toujours une référence au back-end supprimé. Le `TridentBackendConfig` La CR peut toujours être supprimée, quel que soit le `deletionPolicy` valeur.
- `Unknown`: Astra Trident n'est pas en mesure de déterminer l'état ou l'existence du back-end associé au `TridentBackendConfig` CR. Par exemple, si le serveur d'API ne répond pas ou si `tridentbackends.trident.netapp.io` CRD manquant. Cela peut nécessiter une intervention.

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

(Facultatif) étape 4 : pour plus de détails

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

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

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

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

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

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

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

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

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

Gestion des systèmes back-end

Effectuer la gestion back-end avec kubectl

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

Supprimer un back-end

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

Exécutez la commande suivante :

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

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

Affichez les systèmes back-end existants

Exécutez la commande suivante :

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

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

Mettre à jour un back-end

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

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

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

- Les paramètres (tels que le nom du SVM ONTAP utilisé) doivent être mis à jour.
  - Vous pouvez mettre à jour `TridentBackendConfig` Objets directement dans 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 à l'aide de la commande suivante :

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

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

Gestion back-end avec tridentctl

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

Créer un back-end

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

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

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

```
tridentctl logs -n trident
```

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

Supprimer un back-end

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

1. Récupérer le nom du système back-end :

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

2. Supprimer le backend :

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

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

Affichez les systèmes back-end existants

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

- Pour obtenir un récapitulatif, exécutez la commande suivante :

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

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

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

Mettre à jour un back-end

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

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

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

```
tridentctl logs -n trident
```

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

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

Voici un exemple de questions que vous pouvez répondre avec le fichier JSON `tridentctl` sorties des objets back-end. Ceci utilise le `jq` utilitaire que vous devez installer.

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

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

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

Découvrez les différentes façons de gérer les systèmes back-end avec Astra Trident.

Options de gestion des systèmes back-end

Avec l'introduction de `TridentBackendConfig`, les administrateurs ont désormais deux méthodes uniques de gestion des systèmes back-end. Ceci pose les questions suivantes :

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

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

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

Cela s'applique aux scénarios suivants :

- Systèmes back-end existants, sans système `TridentBackendConfig` parce qu'ils ont été créés avec `tridentctl`.
- Nouveaux systèmes back-end créés avec `tridentctl`, tandis que d'autres `TridentBackendConfig` les objets existent.

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

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

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

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

Étape 0 : identifier le back-end

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

```
tridentctl get backend ontap-nas-backend -n trident
```

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

cat ontap-nas-backend.json

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

    "defaults": {
        "spaceReserve": "none",
        "encryption": "false"
    },
    "labels":{"store":"nas_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"msoffice", "cost":"100"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"app":"mysqldb", "cost":"25"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
```

```
            }
        }
    ]
}
```

Étape 1 : créez un code secret Kubernetes

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

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

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

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

Étape 2 : créer un `TridentBackendConfig` CR

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

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

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

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

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

Étape 3 : vérifier l'état du `TridentBackendConfig` CR

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

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

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                  UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

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

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

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

Étape 0 : identifier le back-end

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

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

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

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

Étape 1 : confirmer `deletionPolicy` est défini sur `retain`

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

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

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

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

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

Étape 2 : supprimez le `TridentBackendConfig` CR

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

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

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

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

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 Astra Trident comment provisionner les volumes.

Configuration d'un objet StorageClass Kubernetes

Le "Objet classe de stockage Kubernetes" Identifie Astra Trident en tant que mécanisme de provisionnement utilisé pour cette classe indique à Astra Trident comment provisionner un volume. Par exemple :

```
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
```

Reportez-vous à la section "Kubernetes et objets Trident" pour plus d'informations sur l'interaction des classes de stockage avec le `PersistentVolumeClaim` Et paramètres de contrôle du provisionnement des volumes par Astra Trident.

Créer une classe de stockage

Après avoir créé l'objet StorageClass, vous pouvez créer la classe de stockage. Échantillons de classe de stockage fournit des exemples de base que vous pouvez utiliser ou modifier.

Étapes

1. Il s'agit d'un objet Kubernetes, alors utilisez-le `kubectl` Pour la créer dans Kubernetes.

   ```
   kubectl create -f sample-input/storage-class-basic-csi.yaml
   ```

2. Vous devriez désormais voir une classe de stockage **Basic-csi** dans Kubernetes et Astra Trident. Astra Trident devrait avoir découvert les pools sur le système back-end.

```
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 de stockage

ASTRA Trident offre "définition simple de classes de stockage pour des systèmes back-end spécifiques".

Vous pouvez également modifier `sample-input/storage-class-csi.yaml.templ` fichier fourni avec le programme d'installation et de remplacement *`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 afficher les classes de stockage existantes, définir une classe de stockage par défaut, identifier le back-end de la classe de stockage et supprimer les 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 de la classe de stockage Kubernetes, exécutez la commande suivante :

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

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

```
tridentctl get storageclass
```

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

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

Définir une classe de stockage par défaut

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

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

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

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

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

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

> Votre cluster ne doit contenir qu'une seule classe de stockage par défaut à la fois. Kubernetes n'empêche pas techniquement d'en avoir plusieurs, mais il se comporte comme s'il n'existe aucune classe de stockage par défaut.

Identifier le système back-end pour une classe de stockage

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

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

Supprimer une classe de stockage

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

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

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

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

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

Provisionnement et gestion des volumes

Provisionner un volume

Créez un volume persistant et une demande de volume persistant qui utilisent la classe de stockage Kubernetes configurée pour demander l'accès au volume persistant. Vous pouvez ensuite monter le volume persistant sur un pod.

Présentation

A "*Volume persistant*" (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 volume persistant sur le cluster.

Le PVC peut être configuré pour demander un stockage d'une certaine taille ou d'un certain mode d'accès. À l'aide de la classe de stockage associée, l'administrateur du cluster peut contrôler plus que la taille du volume persistant et le mode d'accès, tels que les performances ou le niveau de service.

Après avoir créé le volume persistant et la demande de volume persistant, vous pouvez monter le volume dans un pod.

Exemples de manifestes

Exemple de manifeste de volume persistant

Cet exemple de manifeste montre un volume persistant de base de 10Gi associé à StorageClass `basic-csi`.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-storage
  labels:
    type: local
spec:
  storageClassName: basic-csi
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/my/host/path"
```

Exemples de manifestes de demande de volume persistant

Ces exemples présentent les options de configuration de base de la PVC.

PVC avec accès RWO

Cet exemple montre une demande de volume persistant de base avec accès RWO associée à une classe de stockage 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 présente une demande de volume persistant de base pour NVMe/TCP avec accès RWO associée à une classe de stockage nommée `protection-gold`.

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

Échantillons de manifeste de pod

Ces exemples présentent les configurations de base pour fixer la demande de volume persistant à un pod.

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
```

Configuration NVMe/TCP de base

```
---
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
    - image: nginx
      name: nginx
      resources: {}
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
  dnsPolicy: ClusterFirst
  restartPolicy: Always
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
      claimName: pvc-san-nvme
```

Créer le volume persistant et la demande de volume persistant

Étapes

1. Créer la PV.

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

2. Vérifiez l'état du PV.

```
kubectl get pv
NAME        CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM
STORAGECLASS  REASON  AGE
pv-storage  4Gi       RWO           Retain          Available
7s
```

3. Créer la PVC.

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

4. Vérifiez l'état de la demande de volume persistant.

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

5. Montez le volume dans un pod.

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

> Vous pouvez surveiller la progression à l'aide de `kubectl get pod --watch`.

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

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

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

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

Reportez-vous à la section "Kubernetes et objets Trident" pour plus d'informations sur l'interaction des classes de stockage avec le `PersistentVolumeClaim` Et paramètres de contrôle du provisionnement des volumes par Astra Trident.

Développement des volumes

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

Développez un volume iSCSI

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

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

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

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

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

Modifiez la définition de la demande de volume persistant 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
```

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

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

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

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

Reliez le volume persistant à un pod pour qu'il soit redimensionné. Lors du redimensionnement d'un volume persistant iSCSI, deux scénarios sont possibles :

- Si le volume persistant est connecté à un pod, Astra Trident étend le volume en back-end, reanalyse le système et redimensionne le système de fichiers.
- Pour redimensionner un volume persistant non connecté, Astra Trident étend le volume sur le back-end. Une fois le volume de volume persistant lié à un pod, Trident analyse de nouveau le périphérique et redimensionne le système de fichiers. Kubernetes met ensuite à jour la taille de la demande de volume persistant une fois l'opération d'extension terminée.

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

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

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

Étape 4 : développez le volume persistant

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

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

Étape 5 : valider l’extension

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

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

Développez un volume NFS

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

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

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

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

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

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

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

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

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

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

Étape 3 : développez le volume persistant

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

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

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

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

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

Importer des volumes

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

Présentation et considérations

Vous pouvez importer un volume dans Astra Trident et :

- Conteneurisation d'une application et réutilisation de son jeu de données existant
- Utilisez un clone d'un jeu de données pour une application éphémère
- Reconstruction d'un cluster Kubernetes en panne
- Migration des données applicatives pendant la reprise après incident

Considérations

Avant d'importer un volume, consultez les considérations suivantes.

- ASTRA Trident peut importer des volumes ONTAP de type RW (lecture-écriture) uniquement. Les volumes de type DP (protection des données) sont des volumes de destination SnapMirror. Vous devez rompre la relation de miroir avant d'importer le volume dans Astra Trident.

- Nous vous suggérons d'importer des volumes sans connexions actives. Pour importer un volume activement utilisé, clonez-le, puis effectuez l'importation.

  C'est particulièrement important pour les volumes en mode bloc, car Kubernetes ignorerait la connexion précédente et pourrait facilement relier un volume actif à un pod. Cela peut entraîner une corruption des données.

- Cependant `StorageClass` Doit être spécifié sur une demande de volume persistant, Astra Trident n'utilise pas ce paramètre lors de l'importation. Les classes de stockage sont utilisées lors de la création du volume pour sélectionner un pool disponible en fonction des caractéristiques de stockage. Comme le volume existe déjà, aucune sélection de pool n'est requise pendant l'importation. Par conséquent, l'importation n'échouera pas même si le volume existe sur un back-end ou un pool qui ne correspond pas à la classe de stockage spécifiée dans le PVC.
- La taille du volume existant est déterminée et définie dans la PVC. Une fois le volume importé par le pilote de stockage, le volume persistant est créé avec un SécurRef dans la demande de volume persistant.
  - La règle de récupération est initialement définie sur `retain` Dans la PV. Une fois que Kubernetes a réussi à relier la demande de volume persistant et le volume persistant, la règle de récupération est mise à jour pour correspondre à la règle de récupération de la classe de stockage.
  - Si la règle de récupération de la classe de stockage est `delete`, Le volume de stockage sera supprimé lorsque le volume persistant est supprimé.
- Par défaut, Astra Trident gère la demande de volume persistant et renomme le FlexVol et le LUN sur le back-end. Vous pouvez passer le `--no-manage` indicateur pour importer un volume non géré. Si vous utilisez `--no-manage`, Astra Trident n'effectue aucune opération supplémentaire sur la demande de volume persistant ou la demande de volume persistant pour le cycle de vie des objets. Le volume de stockage n'est pas supprimé lorsque le volume persistant est supprimé et d'autres opérations telles que le clone de volume et le redimensionnement de volume sont également ignorées.

  Cette option est utile si vous souhaitez utiliser Kubernetes pour des workloads conteneurisés, mais que vous souhaitez gérer le cycle de vie du volume de stockage en dehors de Kubernetes.

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

Importer un volume

Vous pouvez utiliser `tridentctl import` pour importer un volume.

Étapes

1. Création du fichier de demande de volume persistant (par exemple, `pvc.yaml`) Qui sera utilisé pour créer la PVC. Le fichier PVC doit inclure `name`, `namespace`, `accessModes`, et `storageClassName`. Vous pouvez également spécifier `unixPermissions` Dans votre définition de PVC.

   Voici un exemple de spécification minimale :

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

> N'incluez pas de paramètres supplémentaires tels que le nom du volume persistant ou la taille du volume. Cela peut entraîner l'échec de la commande d'importation.

2. Utilisez le `tridentctl import` Commande permettant de spécifier le nom du système back-end Astra Trident contenant le volume et le nom qui identifie de manière unique le volume sur le stockage (par exemple : ONTAP FlexVol, Element Volume, chemin Cloud Volumes Service). Le `-f` L'argument est requis pour spécifier le chemin d'accès au fichier PVC.

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

Exemples

Consultez les exemples d'importation de volume suivants pour les pilotes pris en charge.

NAS ONTAP et FlexGroup NAS ONTAP

ASTRA Trident prend en charge l'importation de volumes à l'aide du `ontap-nas` et `ontap-nas-flexgroup` pilotes.

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

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

Exemples NAS de ONTAP

Voici un exemple d'importation de volume géré et de volume non géré.

Volume géré

L'exemple suivant importe un volume nommé `managed_volume` sur un système back-end nommé `ontap_nas`:

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

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

Volume non géré

Lorsque vous utilisez le `--no-manage` Argument, Astra Trident ne renomme pas le volume.

L'exemple suivant importe `unmanaged_volume` sur le `ontap_nas` back-end :

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

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

SAN ONTAP

ASTRA Trident prend en charge l'importation de volumes à l'aide du `ontap-san` conducteur. L'importation de volume n'est pas prise en charge à l'aide du `ontap-san-economy` conducteur.

ASTRA Trident peut importer des volumes ONTAP SAN FlexVols qui contiennent un LUN unique. Ceci est cohérent avec le `ontap-san` Pilote, qui crée un FlexVol pour chaque demande de volume persistant et une LUN au sein de la FlexVol. ASTRA Trident importe le FlexVol et l'associe à la définition de l'ESV.

Exemples de SAN ONTAP

Voici un exemple d'importation de volume géré et de volume non géré.

Volume géré

Pour les volumes gérés, Astra Trident renomme le système FlexVol en `pvc-<uuid>` Formatez et la LUN au sein du FlexVol à `lun0`.

L'exemple suivant importe le `ontap-san-managed` FlexVol présent sur le `ontap_san_default` back-end :

```
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` back-end :

```
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 des LUN sont mappées à des igroups qui partagent un IQN avec un IQN de nœud Kubernetes, comme dans l'exemple suivant, l'erreur s'affiche : `LUN already mapped to initiator(s) in this group`. Vous devez supprimer l'initiateur ou annuler le mappage de la 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
```

Elément

ASTRA Trident prend en charge le logiciel NetApp Element et l'importation de volumes NetApp HCI à l'aide du `solidfire-san` conducteur.

> Le pilote d'élément prend en charge les noms de volume dupliqués. Toutefois, Astra Trident renvoie une erreur si des noms de volumes sont dupliqués. Pour contourner ce problème, clonez le volume, indiquez un nom de volume unique et importez le volume cloné.

Exemple d'élément

L'exemple suivant importe un `element-managed` volume sur le back-end `element_default`.

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

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

Google Cloud Platform

ASTRA Trident prend en charge l'importation de volumes à l'aide du `gcp-cvs` conducteur.

> Pour importer un volume soutenu par NetApp Cloud Volumes Service dans Google Cloud Platform, identifiez le volume par son chemin d'accès au volume. Le chemin du volume est la partie du chemin d'exportation du volume après 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 back-end `gcpcvs_YEppr` avec le chemin de volume de `adroit-jolly-swift`.

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

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

Azure NetApp Files

ASTRA Trident prend en charge l'importation de volumes à l'aide du `azure-netapp-files` conducteur.

> Pour importer un volume Azure NetApp Files, identifiez-le par son chemin d'accès au volume. Le chemin du volume est la partie du chemin d'exportation du volume après le `:/`. Par exemple, si le chemin de montage est `10.0.0.2:/importvol1`, le chemin du volume est `importvol1`.

Exemple Azure NetApp Files

L'exemple suivant importe un `azure-netapp-files` volume sur le back-end `azurenetappfiles_40517` avec le chemin de volume `importvol1`.

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

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

Partager un volume NFS entre les espaces de noms

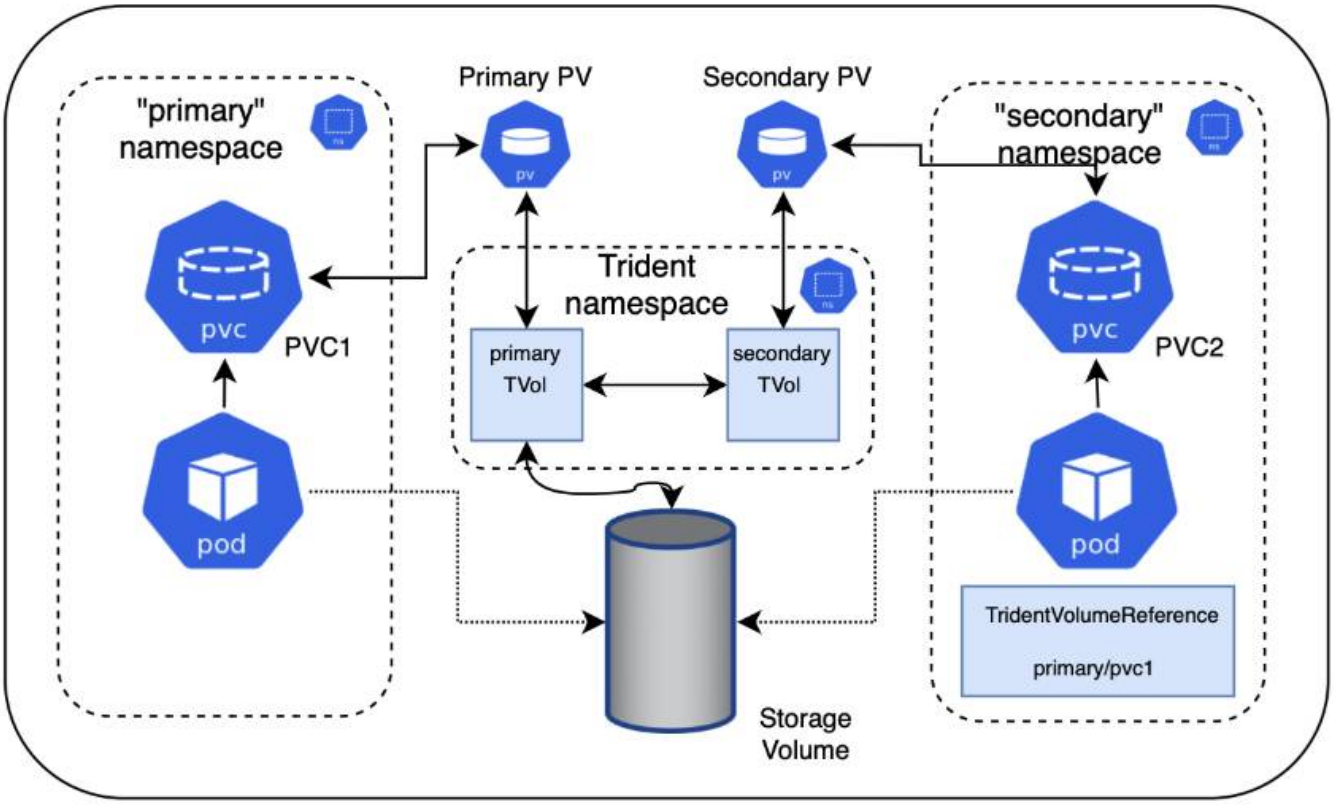
Avec Astra Trident, vous pouvez créer un volume dans un espace de noms principal et le partager dans un ou plusieurs espaces de noms secondaires.

Caractéristiques

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

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

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



Démarrage rapide

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

1. **Configurez la demande de volume persistant source pour partager le volume**

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

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

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

3 Créer TridentVolumeReference dans l'espace de noms de destination

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

4 Créez le PVC subalterne dans l'espace de noms de destination

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

Configurer les espaces de noms source et de destination

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

Étapes

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

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

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

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

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

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

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

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

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

Résultats

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

Supprimer un volume partagé

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

Utiliser `tridentctl get` pour interroger des volumes subordonnés

À l'aide du[`tridentctl` vous pouvez exécuter l' `get` commande pour obtenir des volumes subordonnés. Pour plus d'informations, consultez le lien :../trident-Reference/tridentctl.html[`tridentctl` commandes et options].

```
Usage:
  tridentctl get [option]
```

Alarmes :

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

Limites

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

Pour en savoir plus

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

- Visitez "Partage de volumes entre les espaces de noms : dites bonjour à l'accès aux volumes situés à l'échelle d'un espace de noms".
- Voir la démo "NetAppTV".

Utiliser la topologie CSI

Astra Trident peut créer et relier de façon sélective des volumes aux nœuds présents dans un cluster Kubernetes en utilisant le "Fonction de topologie CSI".

Présentation

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

En savoir plus sur la fonction de topologie CSI "ici".

Kubernetes propose deux modes de liaison de volumes :

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

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

Ce dont vous avez besoin

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

- Un cluster Kubernetes exécutant un "Version Kubernetes prise en charge"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

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

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

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

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

Voici un exemple de définition de back-end :

YAML

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

JSON

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

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

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

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-a
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-b
storage:
- labels:
    workload: production
  region: Iowa-DC
  zone: Iowa-DC-A
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
- labels:
    workload: dev
  region: Iowa-DC
  zone: Iowa-DC-B
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
```

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

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

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

Voir l'exemple suivant :

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

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

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

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

Voir l'exemple `spec` ci-dessous :

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

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

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

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

```
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

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

Voir le résultat suivant :

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

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

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

Trouvez plus d'informations

- "Gestion des ressources pour les conteneurs"
- "Outil de sélection de nœud"
- "Affinité et anti-affinité"
- "Teintes et tolérances"

Travailler avec des instantanés

Les copies Snapshot de volume Kubernetes de volumes persistants (PVS) permettent d'effectuer des copies instantanées de volumes. Vous pouvez créer une copie Snapshot d'un volume créé à l'aide d'Astra Trident, importer un snapshot créé hors d'Astra Trident, créer un nouveau volume à partir d'un snapshot existant et restaurer les données de volume à partir de snapshots.

Présentation

Le snapshot de volume est pris en charge par `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, et `azure-netapp-files` pilotes.

Avant de commencer

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

Si votre distribution Kubernetes n'inclut pas le contrôleur de snapshot et les CRD, reportez-vous à la Déployer un contrôleur de snapshot de volume.

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

Créer un snapshot de volume

Étapes

1. Créer un `VolumeSnapshotClass`. Pour plus d'informations, reportez-vous à la section "VolumeSnapshotClass".
   - Le `driver` Pointe vers le pilote Astra Trident CSI.
   - `deletionPolicy` peut être `Delete` ou `Retain`. Lorsqu'il est réglé sur `Retain`, le snapshot physique sous-jacent sur le cluster de stockage est conservé même lorsque `VolumeSnapshot` l'objet a été supprimé.

     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 snapshot d'une demande de volume persistant existante.

   Exemples

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

     ```
     cat snap.yaml
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Cet exemple crée un objet de snapshot de volume pour une demande de volume persistant nommée `pvc1` et le nom du snapshot est défini sur `pvc1-snap`. Un instantané VolumeSnapshot est similaire à une demande de volume persistant et est associé à une `VolumeSnapshotContent` objet qui représente le snapshot 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` objet pour le `pvc1-snap` VolumeSnapshot en le décrivant. Le `Snapshot Content Name` Identifie l'objet VolumeSnapshotContent qui sert ce snapshot. Le `Ready To Use` Paramètre indique que l'instantané peut être utilisé pour créer une nouvelle demande de volume persistant.

```
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 une demande de volume persistant à partir d'un snapshot de volume

Vous pouvez utiliser `dataSource` Pour créer une demande de volume persistant à l'aide d'un VolumeSnapshot nommé `<pvc-name>` comme source des données. Une fois la demande de volume persistant créée, elle peut être connectée à un pod et utilisée comme n'importe quel autre PVC.

> La demande de volume sera créée dans le même back-end que le volume source. Reportez-vous à la section "Base de connaissances : la création d'une demande de volume persistant à partir d'un Snapshot de volume persistant Trident ne peut pas être créée dans un autre back-end".

L'exemple suivant crée la demande de volume persistant à l'aide de `pvc1-snap` comme source de données.

```
cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importer un instantané de volume

ASTRA Trident prend en charge le "Processus Snapshot préprovisionné Kubernetes" pour permettre à l'administrateur du cluster de créer un `VolumeSnapshotContent` Objet et importation de snapshots créés en dehors d'Astra Trident.

Avant de commencer

ASTRA Trident doit avoir créé ou importé le volume parent du snapshot.

Étapes

1. **Cluster admin:** Créer un `VolumeSnapshotContent` objet qui fait référence au snapshot back-end. Cela lance le workflow de snapshot dans Astra Trident.

   - Spécifiez le nom du snapshot back-end dans `annotations` comme `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Spécifiez `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` dans `snapshotHandle`. Il s'agit des seules informations fournies à Astra Trident par le Snapshot externe du `ListSnapshots` appel.

     > Le `<volumeSnapshotContentName>` Impossible de toujours faire correspondre le nom du snapshot back-end en raison des contraintes de dénomination CR.

     Exemple

     L'exemple suivant crée un `VolumeSnapshotContent` objet qui fait référence au snapshot back-end `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>
```

2. **Cluster admin:** Créez le `VolumeSnapshot` CR qui fait référence au `VolumeSnapshotContent` objet. Cette opération demande l'accès à `VolumeSnapshot` dans un espace de noms donné.

   Exemple

   L'exemple suivant crée un `VolumeSnapshot` CR nommée `import-snap` qui fait référence au `VolumeSnapshotContent` nommé `import-snap-content`.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: import-snap
spec:
  # volumeSnapshotClassName: csi-snapclass (not required for pre-
provisioned or imported snapshots)
  source:
    volumeSnapshotContentName: import-snap-content
```

3. **Traitement interne (aucune action requise):** le snapshotter externe reconnaît le nouveau créé `VolumeSnapshotContent` et exécute le `ListSnapshots` appel. ASTRA Trident crée le `TridentSnapshot`.
   - Le snapshotter externe définit le `VolumeSnapshotContent` à `readyToUse` et le `VolumeSnapshot` à `true`.
   - Retour Trident `readyToUse=true`.
4. **Tout utilisateur :** Créer un `PersistentVolumeClaim` pour référencer le nouveau `VolumeSnapshot`, où `spec.dataSource` (ou `spec.dataSourceRef`) 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
```

Restaurez les données de volume à l'aide de snapshots

Le répertoire des snapshots est masqué par défaut pour faciliter la compatibilité maximale des volumes provisionnés à l'aide de `ontap-nas` et `ontap-nas-economy` pilotes. Activez le `.snapshot` répertoire permettant de restaurer directement les données à partir de snapshots.

Utilisez l'interface de ligne de commandes ONTAP de restauration de snapshot de volume pour restaurer un volume à un état enregistré dans un snapshot précédent.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

> Lorsque vous restaurez une copie Snapshot, la configuration de volume existante est écrasée. Les modifications apportées aux données de volume après la création de la copie Snapshot sont perdues.

Supprimez un volume persistant avec les snapshots associés

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

Déployer un contrôleur de snapshot de volume

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

Étapes

1. Création de CRD de snapshot de volume.

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

```
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, ouvrir `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` et mettre à jour `namespace` à votre espace de noms.

Liens connexes

- "Snapshots de volume"
- "VolumeSnapshotClass"

Gestion et contrôle d'Astra Trident

Mettez à niveau Astra Trident

Mettez à niveau Astra Trident

Astra Trident suit le rythme de sa mise à jour trimestrielle, fournissant quatre versions majeures chaque année. Chaque nouvelle version exploite les versions précédentes et fournit de nouvelles fonctionnalités, des améliorations de performances, des correctifs et des améliorations. Nous vous encourageons à effectuer une mise à niveau au moins une fois par an pour profiter des nouvelles fonctionnalités d'Astra Trident.

Considérations avant la mise à niveau

Lorsque vous mettez à niveau vers la dernière version d'Astra Trident, prenez en compte les points suivants :

- Il ne doit y avoir qu'une seule instance Astra Trident installée sur tous les namespaces d'un cluster Kubernetes donné.
- ASTRA Trident 23.07 et les versions ultérieures requièrent des snapshots de volume v1 et ne prennent plus en charge les snapshots alpha ou bêta.
- Si vous avez créé Cloud Volumes Service pour Google Cloud dans le "Type de service CVS", vous devez mettre à jour la configuration back-end pour utiliser le `standardsw` ou `zoneredundantstandardsw` Niveau de service lors de la mise à niveau à partir d'Astra Trident 23.01. Impossible de mettre à jour le `serviceLevel` dans le back-end, les volumes pourraient tomber en panne. Reportez-vous à la section "Exemples de type de service CVS" pour plus d'informations.
- Lors de la mise à niveau, il est important de fournir `parameter.fsType` dans `StorageClasses` Utilisé par Astra Trident. Vous pouvez supprimer et recréer `StorageClasses` sans interrompre les volumes existants.
  - Il s'agit d'une exigence ** pour l'application "contextes de sécurité" Pour les volumes SAN.
  - Le répertoire d'entrée sample contient des exemples, tels que `storage-class-basic.yaml.templ` et `storage-class-bronze-default.yaml`.
  - Pour plus d'informations, reportez-vous à la section "Problèmes connus".

Étape 1 : sélectionnez une version

Les versions d'Astra Trident suivent une date `YY.MM` convention de dénomination, où "YY" est les deux derniers chiffres de l'année et "MM" est le mois. Les versions point suivent un `YY.MM.X` convention, où « X » est le niveau de patch. Vous allez sélectionner la version à mettre à niveau en fonction de la version à partir de laquelle vous effectuez la mise à niveau.

- Vous pouvez effectuer une mise à niveau directe vers n'importe quelle version cible située dans une fenêtre à quatre versions de la version installée. Par exemple, vous pouvez effectuer une mise à niveau directe de la version 23.01 (ou de toute version 23.01 points) vers la version 24.02.
- Si vous effectuez une mise à niveau à partir d'une version en dehors de la fenêtre à quatre versions, effectuez une mise à niveau en plusieurs étapes. Suivez les instructions de mise à niveau du "version antérieure" vous effectuez une mise à niveau de vers la version la plus récente qui correspond à la fenêtre à quatre versions. Par exemple, si vous exécutez 22.01 et que vous souhaitez effectuer une mise à niveau vers 24.02 :

a. Première mise à niveau de 22.01 à 23.01.

b. Puis passez de 23.01 à 24.02.

> Lorsque vous effectuez une mise à niveau avec 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 sorti avec 21.01.0 contient un problème connu qui a été résolu en 21.01.1. Pour plus de détails, reportez-vous au "Consultez le document GitHub pour plus d'informations".

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

Pour déterminer quelle version vous avez utilisée pour l'installation initiale d'Astra Trident :

1. Utiliser `kubectl get pods -n trident` pour examiner les pods.
   - S'il n'y a pas de pod opérateur, Astra Trident a été installé avec `tridentctl`.
   - S'il existe un pod opérateur, Astra Trident a été installé à l'aide de l'opérateur Trident soit manuellement, soit à l'aide d'Helm.
2. S'il y a un boîtier opérateur, utiliser `kubectl describe torc` Pour déterminer si Astra Trident a été installé à l'aide d'Helm.
   - S'il y a une étiquette Helm, Astra Trident a été installée à l'aide d'Helm.
   - S'il n'y a pas d'étiquette Helm, Astra Trident a été installé manuellement à l'aide de l'opérateur Trident.

Étape 3 : sélectionnez une méthode de mise à niveau

En général, vous devez effectuer une mise à niveau en utilisant la même méthode que celle utilisée pour l'installation initiale, mais vous pouvez le faire "passer d'une méthode d'installation à l'autre". Il existe deux options de mise à niveau d'Astra Trident.

- "Mise à niveau à l'aide de l'opérateur Trident"

> Nous vous suggérons de revoir "Comprendre le workflow de mise à niveau de l'opérateur" avant la mise à niveau avec l'opérateur.

*

Mise à niveau avec l'opérateur

Comprendre le workflow de mise à niveau de l'opérateur

Avant d'utiliser l'opérateur Trident pour mettre à niveau Astra Trident, vous devez comprendre les processus en arrière-plan qui se produisent pendant la mise à niveau. Cela inclut les modifications apportées au contrôleur Trident, au pod du contrôleur et aux pods des nœuds, ainsi qu'au jeu de démonstration des nœuds qui activent les mises à jour en continu.

Gestion des mises à niveau par l'opérateur Trident

L'un des nombreux "Avantages de l'utilisation de l'opérateur Trident" Pour installer et mettre à niveau Astra Trident, il s'agit du traitement automatique des objets Astra Trident et Kubernetes sans interrompre les

volumes montés déjà en place. De cette façon, Astra Trident peut prendre en charge les mises à niveau sans interruption, ou "*mises à jour en continu*". En particulier, l'opérateur Trident communique avec le cluster Kubernetes pour :

- Supprimez et recréez le déploiement du contrôleur Trident et le nœud DemonSet.
- Remplacez l'afficheur de contrôleur Trident et les pods de nœud Trident par de nouvelles versions.
  - Si un nœud n'est pas mis à jour, il n'empêche pas la mise à jour des nœuds restants.
  - Seuls les nœuds exécutant Trident Node Pod peuvent monter des volumes.

> Pour en savoir plus sur l'architecture d'Astra Trident sur le cluster Kubernetes, reportez-vous à la section "Architecture d'Astra Trident".

Workflow de mise à niveau de l'opérateur

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

1. L'opérateur **Trident** :
   a. Détecte la version d'Astra Trident actuellement installée (version *n*).
   b. Mise à jour de tous les objets Kubernetes, y compris les CRD, RBAC et le service Trident.
   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 le pod du contrôleur Trident pour *n+1*.
3. L'opérateur **Trident** :
   a. Supprime le jeu de démonstration du nœud Trident pour *n*. L'opérateur n'attend pas la fin de Node Pod.
   b. Crée le dédéclenchement du nœud Trident pour *n+1*.
4. **Kubernetes** crée des pods de nœuds Trident sur les nœuds qui n'exécutent pas Trident Node Pod *n*. Cela permet de garantir qu'il n'y a jamais plus d'un pod de nœuds Trident, quelle que soit la version, sur un nœud.

Mettre à niveau une installation de l'opérateur Trident

Vous pouvez mettre à niveau Astra Trident à l'aide de l'opérateur Trident manuellement ou à l'aide d'Helm. Vous pouvez effectuer la mise à niveau d'une installation d'opérateur Trident vers une autre installation d'opérateur Trident ou à partir d'un `tridentctl` Installation sur une version opérateur Trident. Révision "Sélectionnez une méthode de mise à niveau" Avant de mettre à niveau l'installation d'un opérateur Trident.

Mettre à niveau une installation manuelle

Vous pouvez effectuer une mise à niveau d'une installation d'opérateur Trident dont le périmètre est défini dans le cluster vers une autre installation d'opérateur Trident dont le périmètre est défini dans le cluster. Toutes les versions 21.01 et supérieures d'Astra Trident utilisent un opérateur à périmètre de cluster.

> Pour effectuer la mise à niveau à partir d'Astra Trident qui a été installée à l'aide de l'opérateur dont le périmètre d'espace de noms est compris entre les versions 20.07 et 20.10, suivez les instructions de mise à niveau de "votre version installée" D'Astra Trident.

Description de la tâche

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

- Pour les clusters exécutant Kubernetes 1.24 ou version antérieure, utilisez "bundle_pre_1_25.yaml".
- Pour les clusters exécutant Kubernetes 1.25 ou version ultérieure, utilisez "bundle_post_1_25.yaml".

Avant de commencer

Assurez-vous d'utiliser un cluster Kubernetes en cours d'exécution "Version Kubernetes prise en charge".

Étapes

1. Vérifiez votre version d'Astra Trident :

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

2. Supprimez l'opérateur Trident qui a été utilisé pour installer l'instance Astra Trident actuelle. Par exemple, si vous mettez à niveau depuis 23.07, exécutez la commande suivante :

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

3. Si vous avez personnalisé votre installation initiale à l'aide de `TridentOrchestrator` attributs, vous pouvez modifier le `TridentOrchestrator` objet pour modifier les paramètres d'installation. Cela peut inclure des modifications visant à spécifier les registres d'images en miroir Trident et CSI pour le mode hors ligne, à activer les journaux de débogage ou à spécifier les secrets d'extraction d'images.
4. Installez Astra Trident à l'aide du fichier YAML de bundle approprié pour votre environnement, où *<bundle.yaml>* est `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` Basé sur votre version de Kubernetes. Par exemple, si vous installez Astra Trident 24.02, exécutez la commande suivante :

```
kubectl create -f 24.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

Mettre à niveau une installation Helm

Vous pouvez mettre à niveau une installation d'Astra Trident Helm.

> Lorsque vous mettez à niveau un cluster Kubernetes de 1.24 vers 1.25 ou version ultérieure sur lequel Astra Trident est installé, vous devez mettre à jour les valeurs.yaml pour les définir `excludePodSecurityPolicy` à `true` ou ajouter `--set excludePodSecurityPolicy=true` à la `helm upgrade` commande avant de pouvoir mettre à niveau le cluster.

Étapes

1. Si vous "Installez Astra Trident à l'aide d'Helm - effectué", vous pouvez utiliser `helm upgrade trident netapp-trident/trident-operator --version 100.2402.0` pour mettre à niveau en une seule

étape. Si vous n'avez pas ajouté le Helm repo ou si vous ne pouvez pas l'utiliser pour mettre à niveau :

a. Téléchargez la dernière version d'Astra Trident sur le site "La section *Assets* sur GitHub".

b. Utilisez le `helm upgrade` commande où `trident-operator-24.02.0.tgz` reflète la version vers laquelle vous souhaitez effectuer la mise à niveau.

```
helm upgrade <name> trident-operator-24.02.0.tgz
```

> Si vous définissez des options personnalisées lors de l'installation initiale (par exemple, si vous spécifiez des registres privés en miroir pour les images Trident et CSI), ajoutez le `helm upgrade` commande à l'aide de `--set` pour vous assurer que ces options sont incluses dans la commande de mise à niveau, sinon les valeurs sont réinitialisées sur les valeurs par défaut.

2. Courez `helm list` pour vérifier que le graphique et la version de l'application ont tous deux été mis à niveau. Courez `tridentctl logs` pour consulter les messages de débogage.

Mise à niveau à partir d'un `tridentctl` Installation sur l'opérateur Trident

Vous pouvez effectuer la mise à niveau vers la dernière version de l'opérateur Trident à partir d'un `tridentctl` installation. Les systèmes back-end et ESV existants seront automatiquement disponibles.

> Avant de passer d'une méthode d'installation à l'autre, consultez "Passage d'une méthode d'installation à l'autre".

Étapes

1. Téléchargez la dernière version d'Astra Trident.

```
# Download the release required [24.020.0]
mkdir 24.02.0
cd 24.02.0
wget
https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

2. Créer le `tridentorchestrator` CRD du manifeste.

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

3. Déployer l'opérateur cluster-scoped dans le même namespace.

```
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 d'Astra 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. Vérifiez que Trident a été mis à niveau vers la version prévue.

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v24.02.0
```

Mise à niveau avec tridentctl

Vous pouvez facilement mettre à niveau une installation Astra Trident existante à l'aide de `tridentctl`.

Description de la tâche

La désinstallation et la réinstallation d'Astra Trident fait office de mise à niveau. Lorsque vous désinstallez Trident, la demande de volume persistant et le volume persistant utilisés par l'Astra Trident. Les volumes persistants ayant déjà été provisionnés restent disponibles pendant la mise hors ligne d'Astra Trident, et Astra Trident provisionne les volumes pour les demandes de volume persistant créées dans l'intervalle une fois de nouveau en ligne.

Avant de commencer

Révision "Sélectionnez une méthode de mise à niveau" avant la mise à niveau avec `tridentctl`.

Étapes

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

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

2. Réinstallez Astra Trident. Reportez-vous à la section "Installation d'Astra Trident à l'aide de tridentctl".

> N'interrompez pas le processus de mise à niveau. Assurez-vous que le programme d'installation s'exécute jusqu'à la fin.

Gérez Astra Trident à l'aide de tridentctl

Le "Pack d'installation Trident" inclut le `tridentctl` Utilitaire de ligne de commande pour un accès simple à Astra Trident. Les utilisateurs Kubernetes disposant de privilèges suffisants peuvent l'utiliser pour installer Astra Trident ou gérer le namespace qui contient le pod Astra Trident.

Commandes et indicateurs globaux

Vous pouvez exécuter `tridentctl help` pour obtenir une liste des commandes disponibles pour `tridentctl` ou ajoutez le `--help` marquer n'importe quelle commande pour obtenir une liste d'options et d'indicateurs pour cette commande spécifique.

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

ASTRA Trident `tridentctl` l'utilitaire prend en charge les commandes et indicateurs globaux suivants.

Commandes

`create`
Ajoutez une ressource à Astra Trident.

`delete`
Supprimez une ou plusieurs ressources d'Astra Trident.

`get`
Obtenez une ou plusieurs ressources d'Astra Trident.

`help`
Aide sur n'importe quelle commande.

`images`
Imprimez un tableau des images de conteneur dont Astra Trident a besoin.

`import`
Importez une ressource existante dans Astra Trident.

`install`
Installer Astra Trident.

`logs`
Imprimez les journaux depuis Astra Trident.

`send`
Envoyer une ressource depuis Astra Trident.

`uninstall`
Désinstallez Astra Trident.

`update`
Modifier une ressource dans Astra Trident.

`update backend state`
Suspendre temporairement les opérations back-end.

`upgrade`
Mise à niveau d'une ressource dans Astra Trident

`version`
Imprimez la version d'Astra Trident.

Alarmes globales

`-d, --debug`

Sortie de débogage.

`-h, --help`

Aide pour `tridentctl`.

`-k, --kubeconfig string`

Spécifiez le `KUBECONFIG` Chemin d'accès pour exécuter des commandes localement ou d'un cluster Kubernetes vers un autre.

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

`-n, --namespace string`

Espace de noms du déploiement d'Astra Trident.

`-o, --output string`

Format de sortie. Un de json|yaml|nom|large|ps (par défaut).

`-s, --server string`

Adresse/port de l'interface REST d'Astra Trident.

> Vous pouvez configurer l'interface REST de Trident pour écouter et utiliser l'interface 127.0.0.1 (pour IPv4) ou [::1] (pour IPv6) uniquement.

Options et indicateurs de commande

création

Utilisez le `create` Commande d'ajout d'une ressource à Astra Trident.

```
tridentctl create [option]
```

Options

`backend`: Ajouter un backend à Astra Trident.

supprimer

Utilisez le `delete` Commande de supprimer une ou plusieurs ressources d'Astra Trident.

```
tridentctl delete [option]
```

Options

`backend`: Supprimer un ou plusieurs systèmes back-end de stockage d'Astra Trident.
`snapshot`: Supprimez un ou plusieurs instantanés de volume d'Astra Trident.

`storageclass`: Supprimez une ou plusieurs classes de stockage d'Astra Trident.
`volume`: Supprimer un ou plusieurs volumes de stockage d'Astra Trident.

obtenez

Utilisez le `get` Commander pour obtenir une ou plusieurs ressources d'Astra Trident.

```
tridentctl get [option]
```

Options

`backend`: Faites passer un ou plusieurs systèmes back-end de stockage à Astra Trident.
`snapshot`: Obtenez un ou plusieurs instantanés d'Astra Trident.
`storageclass`: Obtenez une ou plusieurs classes de stockage d'Astra Trident.
`volume`: Obtenez un ou plusieurs volumes d'Astra Trident.

Alarmes

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

images

Utiliser `images` Drapeaux pour imprimer un tableau des images de conteneur dont Astra Trident a besoin.

```
tridentctl images [flags]
```

Alarmes

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

importer le volume

Utilisez le `import volume` Commande permettant d'importer un volume existant vers Astra Trident.

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

Alias

`volume`, `v`

Alarmes

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

installer

Utilisez le `install` Drapeaux pour l'installation d'Astra Trident.

```
tridentctl install [flags]
```

Alarmes

`--autosupport-image string`: Image conteneur pour AutoSupport Telemetry (par défaut « netapp/trident AutoSupport:<current-version> »).
`--autosupport-proxy string`: Adresse/port d'un proxy pour l'envoi de télémétrie AutoSupport.
`--enable-node-prep`: Tentative d'installation des paquets requis sur les nœuds.
`--generate-custom-yaml`: Générer des fichiers YAML sans rien installer.
`-h, --help`: Aide pour l'installation.
`--http-request-timeout`: Override the HTTP request timeout for Trident Controller's REST API (Default 1m30s).
`--image-registry string`: Adresse/port d'un registre d'images interne.
`--k8s-timeout duration`: Délai d'expiration pour toutes les opérations Kubernetes (par défaut 3m0s).
`--kubelet-dir string`: L'emplacement hôte de l'état interne de kubelet (par défaut "/var/lib/kubelet").
`--log-format string`: Le format de consignation Astra Trident (texte, json) (par défaut "texte").
`--pv string`: Le nom de la PV héritée utilisée par Astra Trident, s'assure que cela n'existe pas (par défaut "trident").
`--pvc string`: Le nom du PVC hérité utilisé par Astra Trident, s'assure qu'il n'existe pas (par défaut "trident").
`--silence-autosupport`: N'envoyez pas automatiquement les packs AutoSupport à NetApp (valeur par défaut vraie).
`--silent`: Désactiver la plupart des sorties lors de l'installation.
`--trident-image string`: L'image Astra Trident à installer.
`--use-custom-yaml`: Utilisez tous les fichiers YAML existants qui existent dans le répertoire de configuration.
`--use-ipv6`: Utiliser IPv6 pour la communication d'Astra Trident.

journaux

Utiliser `logs` Drapeaux pour imprimer les journaux à partir d'Astra Trident.

```
tridentctl logs [flags]
```

Alarmes

`-a, --archive`: Créez une archive de support avec tous les journaux sauf indication contraire.
`-h, --help`: Aide pour les journaux.
`-l, --log string`: Astra Trident log à afficher. Un de trident|auto|trident-operator|All (auto par défaut).
`--node string`: Le nom du nœud Kubernetes à partir duquel recueillir les journaux de pod de nœud.
`-p, --previous`: Si elle existe, procurez-vous les journaux de l'instance de conteneur précédente.
`--sidecars`: Procurez-vous les bûches pour les conteneurs de sidecar.

envoyer

Utilisez le `send` Commande permettant d'envoyer une ressource à Astra Trident.

```
tridentctl send [option]
```

Options

`autosupport`: Envoyez une archive AutoSupport à NetApp.

désinstaller

Utiliser `uninstall` Drapeaux pour désinstaller Astra Trident.

```
tridentctl uninstall [flags]
```

Alarmes

`-h, --help`: Aide pour désinstaller.
`--silent`: Désactiver la plupart des résultats lors de la désinstallation.

mise à jour

Utilisez le `update` Commande permettant de modifier une ressource dans Astra Trident.

```
tridentctl update [option]
```

Options

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

mettre à jour l'état back-end

Utilisez le `update backend state` pour suspendre ou reprendre les opérations back-end.

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

Alarmes

`-h`, `--help`: Aide pour l'état back-end.
`--user-state`: Défini sur `suspended` pour interrompre les opérations back-end. Réglez sur `normal` pour reprendre les opérations back-end. Lorsqu'il est réglé sur `suspended`:

- `AddVolume`, `CloneVolume`, `Import Volume`, `ResizeVolume` sont en pause.
- `PublishVolume`, `UnPublishVolume`, `CreateSnapshot`, `GetSnapshot`, `RestoreSnapshot`, `DeleteSnapshot`, `RemoveVolume`, `GetVolumeExternal`, `ReconcileNodeAccess` et restent disponibles.

version

Utiliser `version` indicateurs pour imprimer la version de `tridentctl` Et le service exécutant Trident.

```
tridentctl version [flags]
```

Alarmes

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

Contrôle d'Astra Trident

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

Présentation

Avec les metrics d'Astra Trident, vous pouvez :

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

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

Ce dont vous avez besoin

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

Étape 1 : définir une cible Prometheus

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

Étape 2 : créer un ServiceMonitor Prometheus

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

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

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

Outre les metrics directement disponibles par Astra Trident, kubelet expose beaucoup `kubelet_volume_*` metrics via son propre terminal de metrics Kubelet peut fournir des informations sur les volumes reliés, ainsi que sur les pods et autres opérations internes qu'elle gère. Reportez-vous à la section "ici".

Étape 3 : interroger les mesures Trident avec PromQL

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

Voici quelques questions PromQL que vous pouvez utiliser :

Accédez aux informations sur l'état de santé de Trident

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

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

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

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

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

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

Découvrez les informations d'utilisation d'Astra Trident

- **Taille moyenne du volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

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

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

Utiliser individuellement le volume

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

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

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Découvrez la télémétrie AutoSupport d'Astra Trident

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

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

elle-même. Pour en savoir plus sur l'engagement de NetApp en matière de sécurité des données et de confiance "ici".

Voici un exemple de charge utile envoyée par Astra Trident :

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

- Les messages AutoSupport sont envoyés au terminal AutoSupport de NetApp. Si vous utilisez un registre privé pour stocker des images de conteneur, vous pouvez utiliser le `--image-registry` drapeau.
- Vous pouvez également configurer des URL proxy en générant les fichiers YAML d'installation. Pour ce faire, utilisez `tridentctl install --generate-custom-yaml` Pour créer les fichiers YAML et ajouter le `--proxy-url` argument pour le `trident-autosupport` conteneur `trident-deployment.yaml`.

Désactivation des metrics d'Astra Trident

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

Désinstaller Astra Trident

Vous devez utiliser la même méthode pour désinstaller Astra Trident que pour installer Astra Trident.

Description de la tâche

- Si vous avez besoin d'un correctif pour les bugs observés après une mise à niveau, des problèmes de dépendance ou une mise à niveau non réussie ou incomplète, désinstallez Astra Trident et réinstallez la version précédente en suivant les instructions spécifiques "version". Il s'agit de la seule méthode recommandée pour *rétrograder* vers une version antérieure.
- Pour faciliter la mise à niveau et la réinstallation, la désinstallation d'Astra Trident ne supprime pas les CRD ou les objets associés créés par Astra Trident. Si vous devez supprimer complètement Astra Trident et toutes ses données, reportez-vous à la section "Retirez complètement Astra Trident et les CRD".

Avant de commencer

Si vous désaffectez des clusters Kubernetes, vous devez supprimer toutes les applications qui utilisent des volumes créés par Astra Trident avant de procéder à la désinstallation. Cela permet de s'assurer que les ESV ne sont pas publiées sur les nœuds Kubernetes avant d'être supprimées.

Déterminez la méthode d'installation d'origine

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

1. Utiliser `kubectl get pods -n trident` pour examiner les pods.
   - S'il n'y a pas de pod opérateur, Astra Trident a été installé avec `tridentctl`.
   - S'il existe un pod opérateur, Astra Trident a été installé à l'aide de l'opérateur Trident soit manuellement, soit à l'aide d'Helm.
2. S'il y a un boîtier opérateur, utiliser `kubectl describe tproc trident` Pour déterminer si Astra Trident a été installé à l'aide d'Helm.
   - S'il y a une étiquette Helm, Astra Trident a été installée à l'aide d'Helm.
   - S'il n'y a pas d'étiquette Helm, Astra Trident a été installé manuellement à l'aide de l'opérateur Trident.

Désinstallez l'installation d'un opérateur Trident

Vous pouvez désinstaller manuellement l'installation d'un opérateur trident ou à l'aide d'Helm.

Désinstallez l'installation manuelle

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

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

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

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

2. **Supprimer `TridentOrchestrator`** : En retirant le `TridentOrchestrator` CR utilisé pour déployer Astra Trident, vous demandez à l'opérateur de désinstaller Trident. L'opérateur traite la dépose de `TridentOrchestrator` Il procède également au retrait du déploiement et de la demonset Astra Trident, en supprimant les pods Trident qu'il avait créés dans le cadre de l'installation.

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

Désinstallez l'installation d'Helm

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

```
#List the Helm release corresponding to the Astra 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
```

Désinstallez un `tridentctl` installation

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

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

Astra Trident pour Docker

Conditions préalables au déploiement

Vous devez installer et configurer les protocoles requis sur votre hôte avant de déployer Astra Trident.

Vérifier les exigences

- Vérifiez que votre déploiement répond à toutes les "de formation".
- Vérifiez que vous disposez d'une version prise en charge de Docker installée. Si votre version de Docker est obsolète, "installez-le ou mettez-le à jour".

```
docker --version
```

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

Outils NFS

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

RHEL 8+

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

Ubuntu

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

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

Outils iSCSI

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

RHEL 8+

1. Installez les packages système suivants :

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

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

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

3. Définir la numérisation sur manuelle :

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

4. Activer les chemins d'accès multiples :

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

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

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

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

6. Activer et démarrer `iscsi`:

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

Ubuntu

1. Installez les packages système suivants :

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

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

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

3. Définir la numérisation sur manuelle :

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

4. Activer les chemins d'accès multiples :

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

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

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

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

Outils NVMe

Installez les outils NVMe à l'aide des commandes correspondant à votre système d'exploitation.

> - NVMe requiert RHEL 9 ou 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 avec 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
```

Déployez Astra Trident

ASTRA Trident pour Docker offre 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, depuis la plateforme de stockage jusqu'aux hôtes Docker, par exemple, l'ajout de plateformes supplémentaires à l'avenir.

Plusieurs instances d'Astra Trident peuvent être exécutées simultanément sur le même hôte. Vous pouvez ainsi établir des connexions simultanées à plusieurs systèmes et types de stockage, et personnaliser le stockage utilisé pour les volumes Docker.

Ce dont vous avez besoin

Voir la "conditions préalables au déploiement". Une fois que vous avez rempli les conditions préalables, vous êtes prêt à déployer Astra Trident.

Méthode de plug-in géré Docker (version 1.13/17.03 et ultérieure)

Avant de commencer

Si vous avez utilisé Astra Trident pré Docker 1.13/17.03 dans la méthode du démon traditionnel, veillez à arrêter le processus Astra Trident et à redémarrer votre démon Docker avant d'utiliser la méthode du plug-in géré.

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

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Redémarrez Docker.

```
systemctl restart docker
```

3. Vérifiez que Docker Engine 17.03 (nouveau modèle 1.13) ou ultérieur est installé.

```
docker --version
```

Si votre version est obsolète, "installez ou mettez 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`, cependant, vous pouvez utiliser un nom quelconque en spécifiant le `config` avec le nom de fichier. Le fichier de configuration doit se trouver dans le `/etc/netappdvp` répertoire sur le système hôte.
   - `log-level`: Spécifiez le niveau de consignation (`debug`, `info`, `warn`, `error`, `fatal`). La valeur par défaut est `info`.
   - `debug`: Spécifiez si la journalisation de débogage est activée. La valeur par défaut est FALSE. Remplace le niveau de journalisation si vrai.
     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 Astra Trident à l'aide du système de plug-in géré. Remplacement `<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 Astra Trident pour consommer le stockage à partir du système configuré.

a. Créer un volume nommé « firstVolume » :

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

b. Créez un volume par défaut au démarrage du conteneur :

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

c. Supprimez le volume « firstVolume » :

```
docker volume rm firstVolume
```

Méthode traditionnelle (version 1.12 ou antérieure)

Avant de commencer

1. Vérifiez que Docker version 1.10 ou ultérieure est installé.

   ```
   docker --version
   ```

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

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

   Ou "suivez les instructions relatives à votre distribution".

2. Vérifiez que NFS et/ou iSCSI sont configurés pour votre système.

Étapes

1. Installez et configurez le plug-in de volume NetApp Docker :

   a. Téléchargez et déballez l'application :

   ```
   wget
   https://github.com/NetApp/trident/releases/download/v24.10.0/trident-
   installer-24.02.0.tar.gz
   tar zxf trident-installer-24.02.0.tar.gz
   ```

   b. Déplacer 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 de Docker

3. Créer un volume :

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

4. Provisionnement d'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
```

Commencez avec Astra Trident au démarrage du système

Un exemple de fichier d'unité pour les systèmes basés sur le système se trouve à l'adresse `contrib/trident.service.example` Dans le Git repo. Pour utiliser le fichier avec RHEL, procédez comme suit :

1. Copiez le fichier à l'emplacement correct.

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

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

2. Modifiez le fichier, modifiez la description (ligne 2) pour qu'elle corresponde au nom du pilote et au chemin du fichier de configuration (ligne 9) pour qu'elle corresponde à votre environnement.
3. Recharger le système pour qu'il ingère les modifications :

   ```
   systemctl daemon-reload
   ```

4. Activer le service.

   Ce nom varie en fonction de ce que vous avez nommé le fichier dans le `/usr/lib/systemd/system` répertoire.

   ```
   systemctl enable trident
   ```

5. Démarrer le service.

   ```
   systemctl start trident
   ```

6. Afficher l'état.

   ```
   systemctl status trident
   ```

> Chaque fois que vous modifiez le fichier d'unité, exécutez le `systemctl daemon-reload` commande pour que le service it soit conscient des modifications.

Mise à niveau ou désinstallation d'Astra Trident

Vous pouvez mettre à niveau Astra Trident pour Docker en toute sécurité, sans impact sur les volumes en cours d'utilisation. Pendant le processus de mise à niveau, il y aura une courte période où `docker volume` les commandes dirigées au niveau du plug-in ne réussiront pas et les applications ne pourront pas monter les volumes tant que le plug-in ne sera pas de nouveau exécuté. Dans la plupart des cas, c'est une question de secondes.

Mise à niveau

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

Étapes

1. Lister les volumes existants :

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

2. Désactivez le plug-in :

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

3. Mettez à niveau le plug-in :

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

> La version 18.01 d'Astra Trident remplace le nDVP. Vous devez mettre à niveau directement à partir du `netapp/ndvp-plugin` image vers le `netapp/trident-plugin` image.

4. Activer le plug-in :

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

5. Vérifiez que le plug-in est activé :

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

6. Vérifier que les volumes sont visibles :

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

> Si vous effectuez la mise à niveau d'une ancienne version d'Astra Trident (pré-20.10) vers Astra Trident 20.10 ou version ultérieure, vous risquez de vous produire une erreur. Pour plus d'informations, reportez-vous à la section "Problèmes connus". Si vous exécutez l'erreur, vous devez d'abord désactiver le plug-in, puis retirer le plug-in, puis installer la version Astra Trident requise en passant un paramètre de configuration supplémentaire : `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all-permissions config=config.json`

Désinstaller

Effectuez les opérations suivantes pour désinstaller Astra Trident pour Docker.

Étapes

1. Supprimez tous les volumes créés par le plug-in.
2. Désactivez le plug-in :

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

3. Retirez le plug-in :

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

Utilisation de volumes

Vous pouvez facilement créer, cloner et supprimer des volumes à l'aide de la norme `docker volume` Commandes avec le nom de pilote Astra Trident spécifié le cas échéant.

Créer un volume

- Créez un volume avec un pilote à l'aide du nom par défaut :

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

- Créez un volume avec une instance Astra Trident spécifique :

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

> Si vous n'en spécifiez aucun "options", les valeurs par défaut du pilote sont utilisées.

- Remplacer la taille du volume par défaut. Voir 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 en chaînes contenant une valeur entière avec des unités facultatives (par exemple : 10G, 20 Go, Tio). Si aucune unité n'est spécifiée, la valeur par défaut est G. Les unités de taille peuvent être exprimées en puissances de 2 (B, Kio, Mio, Gio, Tio) ou 10 (B, Ko, Mo, Go, To). Les unités de raccourci utilisent des puissances de 2 (G = Gio, T = Tio, …).

Supprimer un volume

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

```
docker volume rm firstVolume
```

> Lorsque vous utilisez le `solidfire-san` pilote, l'exemple ci-dessus supprime et purge le volume.

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

Clonez un volume

Lorsque vous utilisez le `ontap-nas`, `ontap-san`, `solidfire-san`, et `gcp-cvs storage drivers`, Astra Trident peut cloner des volumes. Lorsque vous utilisez le `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 entraîne la création d'un nouveau snapshot.

- Inspectez le volume pour énumérer les instantanés :

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

- Créer un nouveau volume à partir d'un volume existant. Cela entraîne 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 snapshot existant sur un volume. Cette opération ne crée 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 blocs créés en externe (ou à leurs clones) à l'aide de conteneurs utilisant Trident **uniquement**, s'ils ne possèdent aucune partition et si leur système de fichiers est pris en charge par Astra Trident (par exemple, un `ext4`-formaté `/dev/sdc1` Ne sera pas accessible via Astra Trident).

Options de volume spécifiques au conducteur

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

Ces options sont simples à utiliser lors de l'opération de création de volume. Indiquez l'option et la valeur à

l'aide de la `-o` Opérateur pendant le fonctionnement de l'interface de ligne de commande. Ces valeurs remplacent toute valeur équivalente du fichier de configuration JSON.

Options de volume ONTAP

Les options de création de volumes pour NFS et iSCSI sont les suivantes :

Option	Description
`size`	La taille du volume est de 1 Gio par défaut.
`spaceReserve`	Provisionnement fin ou non fin du volume, conversion par défaut en fin. Les valeurs valides sont `none` (provisionnement fin) et `volume` (provisionnement lourd).
`snapshotPolicy`	La règle de snapshot sera alors définie sur la valeur souhaitée. La valeur par défaut est `none`, cela signifie qu'aucun instantané ne sera automatiquement créé pour le volume. Sauf modification de la part de votre administrateur de stockage, une règle nommée « par défaut » existe sur tous les systèmes ONTAP qui créent et conserve six snapshots toutes les heures, deux par jour et deux fois par semaine. Vous pouvez restaurer les données conservées dans un snapshot en accédant au `.snapshot` dans n'importe quel répertoire du volume.
`snapshotReserve`	La réserve d'instantanés sera alors définie sur le pourcentage souhaité. La valeur par défaut n'est pas définie. Cela signifie que ONTAP sélectionne la fonction de copie instantanée (généralement 5 %) si vous avez sélectionné une stratégie de snapshots, ou 0 % si la stratégie de snapshots n'est pas définie. Vous pouvez définir la valeur par défaut des snapshots dans le fichier de configuration pour tous les systèmes back-end ONTAP. Vous pouvez l'utiliser comme option de création de volumes pour tous les systèmes back-end ONTAP, à l'exception des économies ontap-nas.
`splitOnClone`	Lors du clonage d'un volume, ONTAP va immédiatement séparer le clone de son volume parent. La valeur par défaut est `false`. Pour optimiser l'efficacité du stockage, il est préférable de séparer le clone de son parent dès sa création, car il est peu probable que cette utilisation soit utile. Par exemple, le clonage d'une base de données vide permet de gagner beaucoup de temps mais réduit les économies de stockage. Il est donc préférable de séparer immédiatement le clone.

Option	Description
`encryption`	Activez NetApp Volume Encryption (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 back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE".
`tieringPolicy`	Définit la règle de hiérarchisation à utiliser pour le volume. Cette décision détermine si les données sont déplacées vers le Tier cloud lorsqu'elles deviennent inactives.

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

Option	Description
`unixPermissions`	Cette option contrôle les autorisations définies pour le volume lui-même. Par défaut, les autorisations sont définies sur `---rwxr-xr-x, ou en notation numérique 0755, et `root` sera le propriétaire. Le format texte ou numérique fonctionnera.
`snapshotDir`	Régler sur `true` fera le `.snapshot` répertoire visible par les clients qui accèdent au volume. La valeur par défaut est `false`, ce qui signifie que la visibilité du `.snapshot` le répertoire est désactivé par défaut. Certaines images, par exemple l'image MySQL officielle, ne fonctionnent pas comme prévu quand le `.snapshot` le répertoire est visible.
`exportPolicy`	Définit l'export policy à utiliser pour le volume. La valeur par défaut est `default`.
`securityStyle`	Définit le style de sécurité à utiliser pour accéder au volume. La valeur par défaut est `unix`. Les valeurs valides sont `unix` et `mixed`.

Les options supplémentaires suivantes sont disponibles pour iSCSI **uniquement** :

Option	Description
`fileSystemType`	Définit le système de fichiers utilisé pour formater les volumes iSCSI. La valeur par défaut est `ext4`. Les valeurs valides sont `ext3`, `ext4`, et `xfs`.
`spaceAllocation`	Régler sur `false` Va désactiver la fonctionnalité d'allocation d'espace de la LUN. La valeur par défaut est `true`, Qui signifie que ONTAP notifie l'hôte lorsque l'espace du volume est insuffisant et que la LUN du volume ne peut pas accepter les écritures. Cette option permet également à ONTAP de récupérer automatiquement de l'espace lorsque votre hôte supprime des données.

Exemples

Voir les exemples ci-dessous :

- Création d'un volume de 10 Gio :

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

- Créez un volume de 100 Gio avec les snapshots :

```
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 20MiB.

Si la réserve Snapshot n'est pas spécifiée et que la règle Snapshot est `none`, Trident utilise une réserve Snapshot de 0 %.

- Créer un volume sans policy de snapshots et sans réserve de snapshots :

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

- Créer un volume sans policy snapshot et une réserve Snapshot personnalisée de 10 % :

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

- Créer un volume avec une règle Snapshot et une réserve Snapshot personnalisée de 10 % :

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

- Créer un volume avec une règle Snapshot et accepter la réserve Snapshot par défaut d'ONTAP (généralement 5 %) :

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

Options de volumes du logiciel Element

Les options du logiciel Element présentent les règles de taille et de qualité de services associées au volume. Lorsque le volume est créé, la politique de QoS associée à celui-ci est spécifiée à l'aide du `-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 minimum, maximum et en rafale associées à un nom dans le fichier de configuration.

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

Option	Description
`size`	La taille du volume, par défaut 1Gio ou entrée de configuration … "Par défaut": {"size": "5G"}.
`blocksize`	Utilisez 512 ou 4096, par défaut 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 règles : bronze, Silver et Gold. Ces noms sont arbitraires.

- Création d'un volume Gold de 10 Gio :

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

- Créez un volume Bronze de 100 Gio :

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

Collecte des journaux

Vous pouvez recueillir des journaux pour obtenir de l'aide en matière de dépannage. La méthode que vous utilisez pour collecter les journaux varie en fonction de l'exécution du plug-in Docker.

Collecte des journaux pour le dépannage

Étapes

1. Si vous exécutez Astra Trident à l'aide de la méthode de plug-in géré recommandée (par exemple, à l'aide de `docker plugin` les commandes), les afficher comme suit :

   ```
   docker plugin ls
   ID                  NAME                DESCRIPTION
   ENABLED
   4fb97d2b956b        netapp:latest       nDVP - NetApp Docker Volume
   Plugin   false
   journalctl -u docker | grep 4fb97d2b956b
   ```

   Le niveau d'enregistrement standard devrait vous permettre de diagnostiquer la plupart des problèmes. Si vous trouvez que ce n'est pas suffisant, vous pouvez activer la journalisation de débogage.

2. Pour activer la journalisation de débogage, installez le plug-in avec la journalisation de débogage activée :

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

   Ou activez la journalisation de débogage lorsque le plug-in est déjà installé :

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

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

Conseils généraux de dépannage

- Le problème le plus courant auquel les nouveaux utilisateurs se sont exécutés est une mauvaise configuration qui empêche le plug-in de s'initialiser. Lorsque cela se produit, vous verrez probablement un message tel que celui-ci lorsque vous essayez d'installer ou d'activer le plug-in :

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

Cela signifie que le plug-in n'a pas démarré. Heureusement, le plug-in a été conçu avec une fonctionnalité de journalisation complète qui devrait vous aider à diagnostiquer la plupart des problèmes que vous êtes susceptible de venir.

- En cas de problème de montage d'un PV sur un conteneur, vérifiez que `rpcbind` est installé et en cours d'exécution. Utilisez le gestionnaire de packages 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 un `systemctl status rpcbind` ou son équivalent.

Gérez plusieurs instances Trident d'Astra

Vous avez besoin de plusieurs instances de Trident lorsque vous souhaitez disposer de plusieurs configurations de stockage simultanément. La clé pour plusieurs instances est de leur donner des noms différents à l'aide de `--alias` avec le plug-in conteneurisé, ou `--volume-driver` Option lors de l'instanciation de Trident sur l'hôte.

Étapes du plug-in 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 autre alias et un fichier de configuration.

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

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

   Par exemple, pour le volume Gold :

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

   Par exemple, pour le volume Silver :

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

Étapes pour les versions traditionnelles (version 1.12 ou antérieure)

1. Lancez le plug-in avec une configuration NFS à l'aide d'un ID de pilote personnalisé :

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

2. Lancez le plug-in avec une configuration iSCSI à l'aide d'un ID de pilote personnalisé :

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

3. Provisionnement de 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

Découvrez les options de configuration disponibles pour vos configurations Astra Trident.

Options de configuration globale

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

Option	Description	Exemple
`version`	Numéro de version du fichier de configuration	`1`
`storageDriverName`	Nom du pilote de stockage	`ontap-nas`, `ontap-san`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `solidfire-san`
`storagePrefix`	Préfixe facultatif pour les noms de volumes. Valeur par défaut : `netappdvp_`.	`staging_`

Option	Description	Exemple
`limitVolumeSize`	Restriction facultative sur les tailles de volume. Par défaut : « » (non appliqué)	`10g`

> Ne pas utiliser `storagePrefix` (Y compris la valeur par défaut) pour les systèmes back-end Element. Par défaut, le `solidfire-san` le pilote ignore ce paramètre et n'utilise pas de préfixe. Nous vous recommandons d'utiliser un ID de tentID spécifique pour le mappage de volume Docker ou les données d'attributs renseignées par la version de Docker, les informations relatives au pilote et le nom brut de Docker dans les cas où il est possible d'utiliser une mundening de nom.

Les options par défaut sont disponibles pour éviter d'avoir à les spécifier sur chaque volume que vous créez. Le `size` option disponible pour tous les types de contrôleurs. Pour un exemple de définition de la taille de volume par défaut, reportez-vous à la section ONTAP configuration.

Option	Description	Exemple
`size`	Taille par défaut facultative pour les nouveaux volumes. Valeur par défaut : `1G`	`10G`

Configuration ONTAP

Outre les valeurs de configuration globale ci-dessus, lorsque vous utilisez ONTAP, les options de premier niveau suivantes sont disponibles.

Option	Description	Exemple
`managementLIF`	Adresse IP de la LIF de management ONTAP. Vous pouvez spécifier un nom de domaine complet (FQDN).	`10.0.0.1`

Option	Description	Exemple
`dataLIF`	Adresse IP de la LIF de protocole. **Pilotes NAS ONTAP**: Nous vous recommandons de spécifier `dataLIF`. Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. **Pilotes SAN ONTAP** : ne pas spécifier pour iSCSI. Astra Trident utilise "Mappage de LUN sélectif ONTAP" Pour découvrir les LIFs iSCI nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si `dataLIF` est explicitement défini.	`10.0.0.2`
`svm`	Storage Virtual machine à utiliser (requis, si la LIF de gestion est une LIF de cluster)	`svm_nfs`
`username`	Nom d'utilisateur pour la connexion au périphérique de stockage	`vsadmin`
`password`	Mot de passe pour se connecter au périphérique de stockage	`secret`
`aggregate`	Agrégat pour le provisionnement (facultatif ; si défini, doit être attribué au SVM) Pour le `ontap-nas-flexgroup` pilote, cette option est ignorée. Tous les agrégats affectés à un SVM sont utilisés pour provisionner un volume FlexGroup.	`aggr1`
`limitAggregateUsage`	Facultatif, le provisionnement échoue si l'utilisation est supérieure à ce pourcentage	`75%`

Option	Description	Exemple
`nfsMountOptions`	Contrôle granulaire des options de montage NFS ; par défaut : «-o nfssvers=3 ». **Disponible uniquement pour le `ontap-nas` et `ontap-nas-economy` pilotes**. "Pour plus d'informations sur la configuration de l'hôte NFS, consultez ici".	`-o nfsvers=4`
`igroupName`	ASTRA Trident crée et gère chaque nœud `igroups` comme `netappdvp`. Cette valeur ne peut pas être modifiée ou omise. **Disponible uniquement pour le `ontap-san` pilote**.	`netappdvp`
`limitVolumeSize`	Taille maximale du volume requesable et taille du volume parent qtree. **Pour le `ontap-nas-economy` Driver, cette option limite en outre la taille des volumes FlexVol qu'elle crée**.	`300g`
`qtreesPerFlexvol`	Le nombre maximal de qtrees par FlexVol doit être compris dans la plage [50, 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`	**Pris en charge pour `ontap-san` pilote uniquement.** Utilisez pour sélectionner `iscsi` Pour iSCSI ou `nvme` Pour NVMe/TCP.	`iscsi` si vide

Les 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` (provisionnement fin) ou `volume` (épais)	`none`

Option	Description	Exemple
`snapshotPolicy`	Règle Snapshot à utiliser ; la valeur par défaut est `none`	`none`
`snapshotReserve`	Pourcentage de réserve de snapshot. La valeur par défaut est « » pour accepter la valeur par défaut de ONTAP	`10`
`splitOnClone`	Séparer un clone de son parent lors de sa création. Par défaut, la valeur est `false`	`false`
`encryption`	Active NetApp Volume Encryption (NVE) sur le nouveau volume ; valeur par défaut sur `false`. Pour utiliser cette option, NVE doit être sous licence et activé sur le cluster. Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront activés par NAE. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE".	vrai
`unixPermissions`	L'option NAS pour les volumes NFS provisionnés est définie par défaut sur `777`	`777`
`snapshotDir`	Option NAS pour accéder à l' `.snapshot` répertoire, par défaut sur `false`	`true`
`exportPolicy`	L'option NAS pour la export policy NFS à utiliser est définie par défaut sur `default`	`default`
`securityStyle`	Option NAS pour l'accès au volume NFS provisionné. Prise en charge de NFS `mixed` et `unix` styles de sécurité. La valeur par défaut est `unix`.	`unix`
`fileSystemType`	OPTION SAN pour sélectionner le type de système de fichiers, par défaut sur `ext4`	`xfs`
`tieringPolicy`	Règle de Tiering à utiliser ; la règle par défaut est `none`; `snapshot-only` Pour la configuration SVM-DR antérieure à ONTAP 9.5	`none`

Options d'évolutivité

Le `ontap-nas` et `ontap-san` Les pilotes créent un ONTAP FlexVol pour chaque volume Docker. ONTAP prend en charge jusqu'à 1000 volumes FlexVol par nœud de cluster avec un cluster maximum de 12,000

volumes FlexVol. Si votre volume Docker répond à cette restriction, le `ontap-nas` Le pilote est la solution NAS préférée du fait des fonctionnalités supplémentaires offertes par les volumes FlexVol, telles que les snapshots et le clonage granulaires avec volume Docker.

Si vous avez besoin de plus de volumes Docker que ne peut pas être pris en charge par les limites FlexVol, choisissez la `ontap-nas-economy` ou le `ontap-san-economy` conducteur.

Le `ontap-nas-economy` Le pilote crée des volumes Docker en tant que qtrees ONTAP dans un pool de volumes FlexVol gérés automatiquement. Les qtrees offrent une évolutivité largement 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 le clonage ou les snapshots granulaires volume Docker.

> Le `ontap-nas-economy` Le pilote n'est pas pris en charge par Docker Swarm, car Swarm n'effectue pas la création de volumes entre plusieurs nœuds.

Le `ontap-san-economy` Le pilote crée des volumes Docker en tant que LUN ONTAP dans un pool partagé de volumes FlexVol gérés automatiquement. De cette façon, chaque FlexVol n'est pas limité à un seul LUN et offre une meilleure évolutivité pour les charges de travail SAN. Selon les baies de stockage, ONTAP prend en charge jusqu'à 16384 LUN par cluster. Comme les volumes sont sous-LUN, ce pilote prend en charge les snapshots et le clonage granulaires par volume Docker.

Choisissez le `ontap-nas-flexgroup` pilote pour augmenter le parallélisme vers un seul volume qui peut atteindre plusieurs pétaoctets avec des milliards de fichiers. Les utilisations idéales de FlexGroups sont l'IA, LE ML, le Big Data et l'analytique, les logiciels, le streaming, les référentiels de fichiers, etc. Trident utilise tous les agrégats attribués à un SVM lors du provisionnement d'un volume FlexGroup. La prise en charge d'FlexGroup dans Trident comporte également plusieurs considérations :

- Requiert ONTAP version 9.2 ou supérieure
- À ce jour, FlexGroups prend uniquement en charge NFS v3.
- Recommandé pour 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 FlexGroup volumes.

Pour plus d'informations sur FlexGroups et les workloads appropriés à FlexGroups, consultez le "NetApp FlexGroup Volume Guide des meilleures pratiques et de mise en œuvre".

Pour bénéficier de fonctionnalités avancées et d'une évolutivité massive dans le même environnement, vous pouvez exécuter plusieurs instances du plug-in de volume Docker, en utilisant une seule instance `ontap-nas` et une autre utilisation `ontap-nas-economy`.

Exemples de fichiers de configuration ONTAP

Exemple NFS pour le <code>ontap-nas</code> pilote

```
{
    "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 <code>ontap-nas-flexgroup</code> pilote

```
{
    "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 <code>ontap-nas-economy</code> pilote

```
{
    "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 <code>ontap-san</code> pilote

```
{
    "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 <code>ontap-san-economy</code> pilote

```
{
    "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 <code>ontap-san</code> pilote

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

Configuration logicielle Element

Outre les valeurs de configuration globale, lorsque le logiciel Element (NetApp HCI/SolidFire) est utilisé, ces options sont disponibles.

Option	Description	Exemple
`Endpoint`	https://<login>:<password>@<mvip>/json-rpc/<element-version>	https://admin:admin@192.168.160.3/json-rpc/8.0
`SVIP`	Port et adresse IP iSCSI	10.0.0.7:3260
`TenantName`	Locataire SolidFireF à utiliser (créé s'il n'est pas trouvé)	`docker`
`InitiatorIFace`	Spécifiez l'interface lors de la restriction du trafic iSCSI à une interface non-par défaut	`default`
`Types`	Spécifications de QoS	Voir l'exemple ci-dessous
`LegacyNamePrefix`	Préfixe des installations Trident mises à niveau. Si vous avez utilisé une version de Trident antérieure à la version 1.3.2 et effectué une mise à niveau avec des volumes existants, vous devez définir cette valeur pour accéder aux anciens volumes mappés avec la méthode nom-volume.	`netappdvp-`

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

Exemple de fichier de configuration du logiciel 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 et limites connus

Découvrez des informations sur les problèmes et les limites connus avec Astra Trident avec Docker.

La mise à niveau de Trident Docker Volume Plug-in vers la version 20.10 et ultérieure à partir des versions plus anciennes entraîne un échec de mise à niveau, sans erreur de fichier ou de répertoire de ce type.

Solution de contournement

1. Désactivez le plug-in.

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

2. Retirez le plug-in.

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

3. Réinstallez le plug-in en fournissant le complément `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 moins 2 caractères.

> Il s'agit d'une limitation client Docker. Le client interprète un seul nom de caractère comme étant un chemin Windows. "Voir bug 25773".

Docker Swarm dispose de certains comportements qui empêchent Astra Trident de les prendre en charge avec chaque combinaison de stockage et de pilote.

- Docker Swarm utilise actuellement le nom du volume, mais pas l'ID de volume, comme identifiant de volume unique.
- Les requêtes de volume sont envoyées simultanément à chaque nœud d'un cluster Swarm.
- Les plug-ins de volumes (y compris Astra Trident) doivent être exécutés de manière indépendante sur chaque nœud d'un cluster Swarm. Grâce au fonctionnement de ONTAP et à son mode de fonctionnement `ontap-nas` et `ontap-san` le conducteur fonctionne, ce sont les seuls qui peuvent être capables de fonctionner dans ces limites.

Les autres conducteurs sont sujets à des problèmes tels que les conditions de course qui peuvent entraîner la création d'un grand nombre de volumes pour une seule demande sans un « gagnant » clair ; par exemple, l'élément possède une fonctionnalité qui permet aux volumes d'avoir le même nom mais des ID différents.

NetApp a fourni des commentaires à l'équipe Docker, mais ne fournit aucune indication de recours futur.

Si un FlexGroup est provisionné, ONTAP ne provisionne pas un deuxième FlexGroup si le deuxième FlexGroup dispose d'un ou de plusieurs agrégats en commun avec la FlexGroup provisionnée.

Meilleures pratiques et recommandations

Déploiement

Utilisez les recommandations indiquées ici pour déployer Astra Trident.

Déploiement dans un namespace dédié

"Espaces de noms" séparation des tâches administratives entre les différentes applications et barrière au partage des ressources. Par exemple, un volume persistant ne peut pas être consommé depuis un autre espace de noms. Astra Trident fournit des ressources PV à tous les namespaces du cluster Kubernetes et exploite par conséquent un compte de service avec des privilèges élevés.

L'accès au pod Trident peut également permettre à un utilisateur d'accéder aux identifiants du système de stockage et à d'autres informations sensibles. Il est important de s'assurer que les utilisateurs d'applications et les applications de gestion ne peuvent pas accéder aux définitions d'objets Trident ou aux pods eux-mêmes.

Utilisez les quotas et les limites des plages pour contrôler la consommation du stockage

Kubernetes dispose de deux fonctionnalités qui, lorsqu'elles sont combinées, fournissent un mécanisme puissant pour limiter la consommation des ressources par les applications. Le "mécanisme de quotas de stockage" permet à l'administrateur d'implémenter des limites d'utilisation globales et spécifiques aux classes de stockage, à la capacité et au nombre d'objets, sur la base de chaque espace de noms. En outre, à l'aide d'un "limite de plage" Veille à ce que les demandes de volume persistant se situent dans une valeur minimale et maximale avant que la requête ne soit transférée au mécanisme de provisionnement.

Ces valeurs sont définies par espace de noms, ce qui signifie que chaque espace de noms doit avoir des valeurs définies qui correspondent à leurs besoins en ressources. Voir ici pour plus d'informations sur "comment exploiter les quotas".

Configuration de stockage sous-jacente

Chaque plateforme de stockage du portefeuille NetApp dispose de fonctionnalités uniques qui bénéficient aux applications, conteneurisées ou non.

Présentation de la plateforme

Trident fonctionne avec ONTAP et Element. Il n'existe pas de plate-forme mieux adaptée à toutes les applications et tous les scénarios qu'une autre. Cependant, les besoins de l'application et l'équipe chargée de l'administration du périphérique doivent être pris en compte lors du choix d'une plate-forme.

Vous devez suivre les meilleures pratiques de base du système d'exploitation hôte avec le protocole utilisé. Vous pouvez éventuellement envisager d'intégrer les meilleures pratiques des applications, le cas échéant, avec des paramètres de back-end, de classe de stockage et de volume persistant afin d'optimiser le stockage pour certaines applications.

Meilleures pratiques pour ONTAP et Cloud Volumes ONTAP

Découvrez les bonnes pratiques pour la configuration d'ONTAP et de Cloud Volumes ONTAP pour Trident.

Les recommandations suivantes sont des instructions de configuration de ONTAP pour les workloads conteneurisés, qui consomment des volumes provisionnés dynamiquement par Trident. Chaque élément doit être pris en compte et évalué en fonction de la pertinence dans votre environnement.

Utilisation de SVM(s) dédié(s) à Trident

Les machines virtuelles de stockage (SVM) assurent l'isolation et la séparation administrative entre les locataires sur un système ONTAP. La dédier un SVM aux applications permet de déléguer des privilèges et d'appliquer les meilleures pratiques en matière de limitation de la consommation des ressources.

Plusieurs options sont disponibles pour la gestion de la SVM :

- Fournir l'interface de gestion du cluster en configuration back-end avec les identifiants appropriés et spécifier le nom du SVM
- Créer une interface de gestion dédiée pour le SVM via ONTAP System Manager ou l'interface de ligne de commande.
- Partage du rôle de gestion avec une interface de données NFS

Dans chaque cas, l'interface doit être dans DNS et le nom DNS doit être utilisé lors de la configuration de Trident. Ainsi, certaines scénarios de reprise après incident, par exemple SVM-DR, sans conservation des identités de réseau.

Il n'existe aucune préférence entre une LIF de gestion dédiée ou partagée pour le SVM, cependant, vous devez vous assurer que vos stratégies de sécurité réseau sont en adéquation avec l'approche de votre choix. Indépendamment de la situation, le LIF de gestion doit être accessible via DNS pour faciliter une flexibilité maximale "SVM-DR" Utilisation en association avec Trident.

Limitez le nombre maximal de volumes

Les systèmes de stockage ONTAP disposent d'un nombre maximal de volumes, qui varie selon la version logicielle et la plateforme matérielle. Reportez-vous à la section "NetApp Hardware Universe" Pour votre plateforme et votre version ONTAP afin de déterminer les limites exactes. Lorsque le nombre de volumes est épuisé, les opérations de provisionnement échouent non seulement pour Trident, mais pour l'ensemble des requêtes de stockage.

Trident `ontap-nas` et `ontap-san` Des pilotes provisionnent un volume flexible pour chaque volume persistant Kubernetes créé. Le `ontap-nas-economy` Le pilote crée environ un FlexVolume pour chaque 200 PVS (configurable entre 50 et 300). Le `ontap-san-economy` Le pilote crée environ un FlexVolume pour chaque 100 PVS (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 la SVM. Vous pouvez le faire à partir de la ligne de commande :

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

La valeur pour `max-volumes` varie en fonction de plusieurs critères spécifiques 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
- Nombre de volumes persistants que les applications Kubernetes devraient consommer

Le `max-volumes` Valeur est le volume total provisionné sur tous les nœuds du cluster ONTAP et non sur un

nœud ONTAP individuel. Par conséquent, vous pouvez rencontrer des situations où un nœud de cluster ONTAP peut avoir 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 flexibles. Avoir le volume maximum réglé sur 1250 semble très raisonnable. Cependant, si seulement "64 bits" Depuis un nœud est attribué à la SVM, ou les agrégats attribués à partir d'un nœud ne peuvent pas être provisionnés sur (par exemple, en raison de la capacité). L'autre nœud devient alors la cible de tous les volumes provisionnés par Trident. Cela signifie que le volume peut être atteint en limite pour ce nœud avant le `max-volumes` La valeur est atteinte, ce qui affecte Trident et les autres opérations de volume utilisant ce nœud. **Vous pouvez éviter cette situation en vous assurant que les agrégats de chaque nœud du cluster sont attribués à la SVM utilisée par Trident en chiffres égaux.**

Limitez la taille maximale des volumes créés par Trident

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

Vous devez aussi exploiter les fonctionnalités Kubernetes pour contrôler la taille du volume au niveau de la baie de stockage.

Configurez Trident pour utiliser le protocole CHAP bidirectionnel

Vous pouvez spécifier l'initiateur CHAP et les noms d'utilisateur et mots de passe cibles dans votre définition du système back-end et activer Trident sur la SVM. À l'aide du `useCHAP` Paramètre dans votre configuration back-end, Trident authentifie les connexions iSCSI pour les systèmes back-end ONTAP avec CHAP.

Création et utilisation d'une politique de QoS de SVM

L'utilisation d'une politique de QoS de ONTAP appliquée au SVM limite le nombre de consommables d'IOPS par les volumes provisionnés par Trident. Cela permet de "évitez un tyran" Ou un conteneur hors contrôle de affectant les charges de travail en dehors du SVM Trident.

Vous pouvez créer une politique de QoS pour la SVM en quelques étapes. Consultez la documentation de votre version de ONTAP pour obtenir des informations précises. L'exemple ci-dessous crée une politique de 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>
```

Si votre version d'ONTAP la prend en charge, il est également possible d'utiliser un minimum de QoS pour garantir un débit minimum pour les workloads conteneurisés. La QoS adaptative n'est pas compatible avec une règle de niveau SVM.

Le nombre d'IOPS dédiées aux workloads conteneurisés dépend de plusieurs aspects. Entre autres choses :

- Autres charges de travail qui utilisent la baie de stockage. Si certaines charges de travail, autres que celles liées au déploiement Kubernetes avec les ressources de stockage, veillez à ne pas affecter

accidentellement ces charges de travail.

- Workloads attendus s'exécutant dans des conteneurs. Si des charges de travail qui exigent des IOPS élevées s'exécutent dans des conteneurs, une faible politique de QoS entraîne une mauvaise expérience.

Il est important de rappeler qu'une politique de QoS attribuée au niveau du SVM entraîne tous les volumes provisionnés sur la SVM et partageant le même pool d'IOPS. Si l'une des applications conteneurisées a une exigence d'IOPS élevées, elle pourrait devenir une force dominante pour les autres workloads conteneurisés. Dans ce cas, vous pourriez envisager d'utiliser l'automatisation externe pour attribuer des règles de QoS par volume.

> Vous devez affecter la « policy group » QoS à la SVM **Only** si la version de votre ONTAP est antérieure à 9.8.

Création de groupes de règles de QoS pour Trident

La qualité de service (QoS) garantit que les performances des workloads stratégiques ne sont pas dégradées par des charges de travail concurrentes. Les groupes de règles de QoS de ONTAP proposent des options de 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, reportez-vous à la section "Débit garanti avec la QoS". Vous pouvez spécifier des groupes de règles de QoS dans le back-end ou dans un pool de stockage, et ils sont appliqués à chaque volume créé dans ce pool ou back-end.

ONTAP propose deux types de groupes de règles de QoS : classiques et évolutifs. Les groupes de règles classiques fournissent un débit minimal (ou minimal, dans les versions ultérieures) plat en IOPS. La QoS adaptative ajuste automatiquement le débit en fonction de la taille du workload. Elle maintient le rapport entre les IOPS et les To|Go en fonction de l'évolution de la taille du workload. Vous pouvez ainsi gérer des centaines, voire des milliers de charges de travail dans le cadre d'un déploiement à grande échelle.

Avant de créer des groupes de règles de QoS, tenez compte des points suivants :

- Vous devez définir le `qosPolicy` saisissez le `defaults` bloc de la configuration back-end. Voir l'exemple de configuration back-end 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 « policy groups » par volume pour que chaque volume bénéficie de l'intégralité du débit spécifié par le « policy group ». Les groupes de stratégies partagés ne sont pas pris en charge.

Pour plus d'informations sur les groupes de règles de QoS, reportez-vous à la section "Commandes QoS de ONTAP 9.8".

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

La limitation de l'accès aux volumes NFS et aux LUN iSCSI créés par Trident est un composant stratégique du niveau de sécurité pour votre déploiement Kubernetes. En effet, les hôtes qui ne font pas partie du cluster Kubernetes n'accèdent pas aux volumes et peuvent modifier les données de façon inattendue.

Il est important de comprendre que les espaces de noms sont la limite logique des ressources dans Kubernetes. L'hypothèse est que les ressources dans un même espace de noms peuvent être partagées, mais, surtout, il n'existe aucune fonctionnalité de multi-espace de noms. Même si les volumes persistants sont des objets globaux, lorsqu'ils sont liés à une demande de volume persistant, ils ne sont accessibles que par des pods qui se trouvent dans le même espace de noms. **Il est essentiel de s'assurer que les espaces de noms sont utilisés pour fournir la séparation, le cas échéant.**

La préoccupation principale de la plupart des entreprises en ce qui concerne la sécurité des données dans un contexte Kubernetes est qu'un processus dans un conteneur peut accéder au stockage monté sur l'hôte, mais qui n'est pas destiné au conteneur. "Espaces de noms" sont conçus pour éviter ce type de compromis. Toutefois, il y a une exception : les conteneurs privilégiés.

Un conteneur privilégié est un conteneur exécuté avec beaucoup plus d'autorisations au niveau de l'hôte que la normale. Par défaut, ces dernières ne sont pas refusées. Veillez donc à désactiver cette fonctionnalité en utilisant "stratégies de sécurité des pods".

Pour les volumes pour lesquels l'accès est demandé depuis Kubernetes et des hôtes externes, le stockage doit être géré de manière classique, avec le volume persistant introduit par l'administrateur et non géré par

Trident. Cela garantit que le volume de stockage est détruit uniquement lorsque les hôtes Kubernetes et externes sont déconnectés et qu'ils n'utilisent plus le volume. En outre, il est possible d'appliquer une export policy personnalisée qui permet l'accès depuis les nœuds de cluster Kubernetes et les serveurs ciblés à l'extérieur du cluster Kubernetes.

Pour les déploiements avec des nœuds d'infrastructure dédiés (par exemple OpenShift) ou d'autres nœuds ne pouvant pas planifier les applications utilisateur, des règles d'exportation distinctes doivent être utilisées pour limiter davantage l'accès aux ressources de stockage. Cela inclut la création d'une export policy pour les services qui sont déployés sur ces nœuds d'infrastructure (par exemple les services OpenShift Metrics et Logging Services), ainsi que pour les applications standard déployées sur des nœuds non liés à l'infrastructure.

Utiliser une export policy dédiée

Vous devez vous assurer qu'il existe une export policy pour chaque backend qui autorise uniquement l'accès aux nœuds présents dans le cluster Kubernetes. Trident peut créer et gérer automatiquement des règles d'export. Trident limite ainsi l'accès aux volumes qu'il provisionne aux nœuds du cluster Kubernetes et simplifie l'ajout et la suppression des nœuds.

Vous pouvez également créer une export policy manuellement et la remplir à l'aide d'une ou plusieurs règles d'exportation qui traitent chaque demande d'accès de nœud :

- Utilisez le `vserver export-policy create` Commande CLI ONTAP pour créer l'export policy.
- Ajoutez des règles à la export policy à l'aide de `vserver export-policy rule create` Commande CLI ONTAP.

L'exécution de ces commandes vous permet de limiter l'accès aux données aux nœuds Kubernetes.

Désactiver `showmount` Pour le SVM applicatif

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 lancer le `showmount -e` Commande au niveau de la LIF de données et reçoit la liste des montages disponibles, y compris ceux auxquels elle n'a pas accès. Bien qu'il ne s'agisse pas d'un compromis sur la sécurité, cette solution fournit des informations inutiles susceptibles d'aider un utilisateur non autorisé à se connecter à une exportation NFS.

Vous devez désactiver `showmount` En utilisant la commande CLI ONTAP au niveau du SVM :

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

Les meilleures pratiques pour SolidFire

Découvrez les bonnes pratiques pour la configuration du stockage SolidFire pour Trident.

Créer un compte SolidFire

Chaque compte SolidFire représente un propriétaire de volume unique et reçoit ses propres informations d'identification CHAP (Challenge-Handshake Authentication Protocol). Vous pouvez accéder aux volumes affectés à un compte en utilisant le nom du compte et les informations d'identification CHAP relatives ou par le biais d'un groupe d'accès de volume. Un compte peut comporter jusqu'à deux milliers de volumes qui lui sont attribués, mais un volume ne peut appartenir qu'à un seul compte.

Création d'une règle de QoS

Utilisez les règles de QoS SolidFire pour créer et enregistrer des paramètres de qualité de service standardisés qui peuvent être appliqués à de nombreux volumes.

Vous pouvez définir des paramètres de QoS par volume. Les performances de chaque volume peuvent être garanties en définissant trois paramètres configurables pour définir les QoS : IOPS min, IOPS max et IOPS en rafale.

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

Paramètre IOPS	Définition	Minimum valeur	Valeur par défaut	Capacité Valeur (4 Ko)
IOPS min	Niveau de performance garanti pour un volume.	50	50	15000
IOPS max	La performance ne dépassera pas cette limite.	50	15000	200,000
IOPS en rafale	IOPS maximales autorisées en rafale,	50	15000	200,000

> Même si les IOPS maximales et en rafale peuvent être définies jusqu'à 200,000, les performances maximales réelles d'un volume sont limitées par l'utilisation du cluster et les performances par nœud.

La taille et la bande passante des blocs influencent directement le nombre d'opérations d'entrée/sortie par seconde. Lorsque la taille de bloc augmente, le système augmente la bande passante jusqu'au niveau nécessaire pour traiter les tailles de bloc de taille supérieure. Lorsque la bande passante augmente, le nombre d'IOPS augmente, le système peut atteindre une baisse. Reportez-vous à la section "Qualité de service SolidFire" Pour plus d'informations sur la qualité de service et les performances.

Authentification SolidFire

Element prend en charge deux méthodes d'authentification : CHAP et VAG (Volume Access Groups). CHAP utilise le protocole CHAP pour authentifier l'hôte au back-end. Les groupes d'accès de volume contrôlent l'accès aux volumes qu'ils provisionne. NetApp recommande d'utiliser le protocole CHAP pour l'authentification, car il est plus simple et ne comporte pas de limites d'évolutivité.

> Trident avec le mécanisme de provisionnement CSI amélioré prend en charge l'authentification CHAP. Les VAGs ne doivent être utilisés que dans le mode de fonctionnement traditionnel non CSI.

L'authentification CHAP (vérification que l'initiateur est l'utilisateur de volume prévu) n'est prise en charge qu'avec un contrôle d'accès basé sur le compte. Si vous utilisez CHAP pour l'authentification, deux options sont disponibles : CHAP unidirectionnel et CHAP bidirectionnel. L'authentification CHAP unidirectionnelle authentifie l'accès au volume à l'aide du nom du compte SolidFire et du secret de l'initiateur. L'option CHAP bidirectionnelle fournit le moyen le plus sûr d'authentifier le volume car le volume authentifie l'hôte via le nom du compte et le secret de l'initiateur, puis l'hôte authentifie le volume via le nom du compte et le secret cible.

Toutefois, si CHAP ne peut pas être activé et que VAGs sont requis, 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, le contrôle d'accès au groupe d'accès de volume est utilisé.

Où trouver plus d'informations ?

Une partie de la documentation sur les meilleures pratiques est présentée ci-dessous. Rechercher dans le "Bibliothèque NetApp" pour les versions les plus récentes.

ONTAP

- "Guide des meilleures pratiques et de mise en œuvre de NFS"
- "Guide d'administration DU SAN" (Pour iSCSI)
- "Configuration iSCSI Express pour RHEL"

Logiciel Element

- "Configuration de SolidFire pour Linux"

NetApp HCI

- "Conditions préalables au déploiement de NetApp HCI"
- "Accès au moteur de déploiement NetApp"

Information sur les pratiques exemplaires des applications

- "Bonnes pratiques pour MySQL sur ONTAP"
- "Bonnes pratiques pour MySQL sur SolidFire"
- "NetApp SolidFire et Cassandra"
- "Meilleures pratiques pour Oracle sur SolidFire"
- "Meilleures pratiques PostgreSQL sur SolidFire"

Toutes les applications ne disposent pas d'instructions spécifiques, il est important de collaborer avec votre équipe NetApp et d'utiliser le "Bibliothèque NetApp" pour trouver la documentation la plus récente.

Intégrez Astra Trident

Pour intégrer Astra Trident, les éléments de conception et d'architecture suivants nécessitent l'intégration : sélection des pilotes et déploiement, conception de la classe de stockage, conception de pool virtuel, impact de la demande de volume persistant sur le provisionnement du stockage, les opérations des volumes et le déploiement de services OpenShift avec Astra Trident.

Choix et déploiement du conducteur

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

Pilotes ONTAP backend

Les pilotes back-end ONTAP sont différenciés par le protocole utilisé et le mode de provisionnement des volumes sur le système de stockage. Par conséquent, réfléchissez bien au choix du conducteur à déployer.

À un niveau plus élevé, si votre application dispose de composants qui nécessitent un stockage partagé (plusieurs modules accédant au même volume de demande de volume persistant), les pilotes NAS seraient la solution par défaut, tandis que les pilotes iSCSI basés sur les blocs répondent aux besoins d'un stockage non partagé. Choisir le protocole en fonction des besoins de l'application et du niveau de confort des équipes chargées du stockage et de l'infrastructure. En règle générale, ces différences sont peu nombreuses pour la plupart des applications. La décision dépend donc souvent de la nécessité d'un stockage partagé (dans lequel plusieurs pods auront besoin d'un accès simultané).

Les pilotes ONTAP backend disponibles sont les suivants :

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

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

Il est à noter que dans les tableaux ci-dessous, toutes les fonctionnalités ne sont pas exposées par Astra Trident. L'administrateur du stockage doit appliquer une partie après le provisionnement si cette fonctionnalité est souhaitée. Les notes de bas de page en exposant distinguent les fonctionnalités par fonction et pilote.

Pilotes NAS ONTAP	Snapshots	Clones	Règles d'exportation dynamiques	Multi-attacher	La QoS	Redimensionner	La réplication
`ontap-nas`	Oui.	Oui.	Yes [5]	Oui.	Yes [1]	Oui.	Yes [1]
`ontap-nas-economy`	Yes [3]	Yes [3]	Yes [5]	Oui.	Yes [3]	Oui.	Yes [3]
`ontap-nas-flexgroup`	Yes [1]	Non	Yes [5]	Oui.	Yes [1]	Oui.	Yes [1]

Astra Trident propose 2 pilotes SAN pour ONTAP dont les fonctionnalités sont présentées ci-dessous.

Pilotes SAN de ONTAP	Snapshots	Clones	Multi-attacher	Chap bi-directionnel	La QoS	Redimensionner	La réplication
`ontap-san`	Oui.	Oui.	Yes [4]	Oui.	Yes [1]	Oui.	Yes [1]

Pilotes SAN de ONTAP	Snapshots	Clones	Multi-attacher	Chap bi-directionnel	La QoS	Redimensionner	La réplication
`ontap-san-economy`	Oui.	Oui.	Yes [4]	Oui.	Yes [3]	Oui.	Yes [3]

> Note de bas de page pour les tableaux ci-dessus :
>
> Yes [1] : non géré par Astra Trident
>
> Yes [2] : géré par Astra Trident, mais pas par volume persistant granulaire
>
> Yes [3] : non géré par Astra Trident et non granulaire par PV
>
> Yes [4] : pris en charge pour les volumes en mode bloc brut
>
> Yes [5] : pris en charge par Astra Trident

Les fonctionnalités qui ne sont pas granulaires volume persistant sont appliquées à l'ensemble du volume flexible et tous les volumes persistants (qtrees ou LUN inclus dans les volumes FlexVol partagés) partageront une planification commune.

Comme on peut le voir dans les tableaux ci-dessus, une grande partie des fonctionnalités entre `ontap-nas` et `ontap-nas-economy` est identique. Cependant, parce que le `ontap-nas-economy` Le pilote limite la capacité à contrôler la planification à la granularité par volume persistant, ce qui peut affecter en particulier la reprise après incident et la planification des sauvegardes. Pour les équipes de développement qui souhaitent exploiter la fonctionnalité de clonage PVC sur le stockage ONTAP, ce n'est possible que lorsque vous utilisez le `ontap-nas`, `ontap-san` ou `ontap-san-economy` pilotes.

> Le `solidfire-san` Le pilote est également capable de cloner des demandes de volume persistant.

Pilotes Cloud Volumes ONTAP backend

Cloud Volumes ONTAP assure le contrôle des données et des fonctionnalités de stockage haute performance dans divers cas d'utilisation, notamment pour les partages de fichiers et le stockage de niveau bloc qui servent les protocoles NAS et SAN (NFS, SMB/CIFS et iSCSI). Les pilotes compatibles avec Cloud Volume ONTAP sont les `ontap-nas`, `ontap-nas-economy`, `ontap-san` et `ontap-san-economy`. Applicable à Cloud Volume ONTAP pour Azure, Cloud Volume ONTAP pour GCP.

Pilotes backend Amazon FSX pour ONTAP

Avec Amazon FSX pour NetApp ONTAP, vous exploitez les fonctionnalités, les performances et les capacités d'administration d'NetApp que vous connaissez déjà, tout en profitant 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 de système de fichiers ONTAP et API d'administration. Les pilotes compatibles avec Cloud Volume ONTAP sont les `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `ontap-san` et `ontap-san-economy`.

Pilotes back-end NetApp HCI/SolidFire

Le `solidfire-san` Pilote utilisé avec les plateformes NetApp HCI/SolidFire pour aider l'administrateur à configurer un back-end Element pour Trident sur la base des limites de QoS. Si vous voulez concevoir votre système back-end pour définir les limites de QoS spécifiques sur les volumes provisionnés par Trident, utilisez la `type` paramètre dans le fichier backend. L'administrateur peut également restreindre la taille du volume pouvant être créé sur le stockage à l'aide de `limitVolumeSize` paramètre. Pour le moment, les fonctionnalités de stockage Element telles que le redimensionnement des volumes et la réplication des volumes ne sont pas prises en charge via le `solidfire-san` conducteur. Ces opérations doivent être effectuées manuellement via l'interface utilisateur Web du logiciel Element.

Pilote SolidFire	Snapshots	Clones	Multi-attacher	CHAP	La QoS	Redimensionner	La réplication
`solidfire-san`	Oui.	Oui.	Yes [2]	Oui.	Oui.	Oui.	Yes [1]

> Note de bas de page: Yes [1]: Non géré par Astra Trident Yes [2]: Pris en charge pour les volumes en blocs bruts

Pilotes Azure NetApp Files backend

Astra Trident utilise le `azure-netapp-files` pilote pour gérer le "Azure NetApp Files" services.

Vous trouverez plus d'informations sur ce pilote et sa configuration dans le "Configuration back-end d'Astra Trident pour Azure NetApp Files".

Pilote Azure NetApp Files	Snapshots	Clones	Multi-attacher	La QoS	Développement	La réplication
`azure-netapp-files`	Oui.	Oui.	Oui.	Oui.	Oui.	Yes [1]

> Note de bas de page: Yes [1]: Non géré par Astra Trident

Cloud Volumes Service sur le pilote back-end Google Cloud

Astra Trident utilise le `gcp-cvs` Pilote de liaison avec Cloud Volumes Service sur Google Cloud.

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

- Si des pools virtuels sont définis au niveau du système back-end, Astra Trident essaie de créer un volume dans les pools de stockage Google Cloud auxquels ces pools virtuels sont limités.
- Si des pools virtuels ne sont pas définis dans le système back-end, Astra Trident sélectionne un pool de stockage Google Cloud à partir des pools de stockage disponibles dans la région.

Pour configurer le back-end Google Cloud avec Astra Trident, vous devez préciser `projectNumber`, `apiRegion`, et `apiKey` dans le fichier backend. Le numéro de projet est indiqué dans la console Google Cloud. La clé API est utilisée depuis le 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 en savoir plus sur les types de services et les niveaux de service Cloud Volumes Service sur Google Cloud, consultez "En savoir plus sur la prise en charge d'Astra Trident pour CVS pour GCP".

Pilote Cloud Volumes Service pour Google Cloud	Snapshots	Clones	Multi-attacher	La QoS	Développement	La réplication
`gcp-cvs`	Oui.	Oui.	Oui.	Oui.	Oui.	Disponible uniquement sur le type de service CVS-Performance.

Notes de réplication

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

Conception de classe de stockage

Chaque classe de stockage doit être configurée et appliquée pour créer un objet de classe de stockage Kubernetes. Cette section décrit comment concevoir un système de stockage pour votre application.

Utilisation du système back-end spécifique

Le filtrage peut être utilisé au sein d'un objet de classe de stockage spécifique pour déterminer le pool de stockage ou l'ensemble de pools à utiliser 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` paramètre permet de limiter le stockage à l'ensemble de pools correspondant à tous les attributs spécifiés. Le `additionalStoragePools` Le paramètre est utilisé pour étendre l'ensemble de pools qu'Astra Trident utilisera pour le provisionnement ainsi que l'ensemble de 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` le paramètre est utilisé pour exclure spécifiquement l'ensemble de pools répertoriés qui correspondent aux attributs.

Émuler les règles de QoS

Si vous souhaitez concevoir des classes de stockage pour émuler les règles de qualité de service, créez une classe de stockage avec le `media` attribut en tant que `hdd` ou `ssd`. Basé sur `media` Attribut mentionné dans la classe de stockage, Trident sélectionne le back-end approprié qui sert `hdd` ou `ssd` les agrégats correspondent à l'attribut du support, puis dirigent le provisionnement des volumes sur l'agrégat spécifique. Nous pouvons donc créer une PRIME de classe de stockage qui aurait été nécessaire `media` attribut défini comme `ssd` Qui peuvent être classées comme politique DE qualité de service PREMIUM. Nous pouvons créer une autre NORME de classe de stockage dont l'attribut de support est défini comme `hdd', qui pourrait être classé comme règle de QoS STANDARD. Nous pourrions également utiliser l'attribut « IOPS » de la classe de stockage pour rediriger le provisionnement vers une appliance Element qui peut être définie comme une règle de QoS.

Utilisation du système back-end en fonction de fonctionnalités spécifiques

Les classes de stockage peuvent être conçues pour diriger le provisionnement des volumes sur un système back-end spécifique, où des fonctionnalités telles que le provisionnement fin et lourd, les copies Snapshot, les clones et le chiffrement sont activées. Pour spécifier le stockage à utiliser, créez des classes de stockage qui spécifient le back-end approprié avec la fonction requise activée.

Pools virtuels

Des pools virtuels sont disponibles pour tous les systèmes back-end Astra Trident. Vous pouvez définir des pools virtuels pour tout système back-end, à l'aide de n'importe quel pilote fourni par Astra Trident.

Les pools virtuels permettent à un administrateur de créer un niveau d'abstraction sur les systèmes back-end, qui peut être référencé via des classes de stockage, pour une plus grande flexibilité et un placement efficace des volumes dans les systèmes back-end. Différents systèmes back-end peuvent être définis avec la même classe de service. En outre, il est possible de créer plusieurs pools de stockage sur le même back-end, mais avec des caractéristiques différentes. Lorsqu'une classe de stockage est configurée avec un sélecteur portant les étiquettes spécifiques, Astra Trident choisit un système back-end correspondant à toutes les étiquettes de sélection pour placer le volume. Si les étiquettes de sélection de classe de stockage correspondent à plusieurs pools de stockage, Astra Trident choisira l'un d'entre eux pour provisionner le volume.

Conception de pool virtuel

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 système back-end avec les mêmes identifiants de stockage et avec un ensemble de paramètres différent. Grâce à l'introduction de pools virtuels, ce problème a été résolu. Les pools virtuels sont une abstraction de niveau introduite entre le back-end et la classe de stockage Kubernetes. L'administrateur peut ainsi définir des paramètres et des étiquettes que l'on peut référencer via les classes de stockage Kubernetes comme un sélecteur, de façon indépendante du back-end. Il est possible de définir des pools virtuels pour tous les systèmes back-end NetApp pris en charge avec Astra Trident. Il s'agit notamment des systèmes SolidFire/NetApp HCI, ONTAP, Cloud Volumes Service sur GCP et 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 backend. Il est également conseillé de ne pas modifier/modifier les attributs d'un pool virtuel existant et de définir un nouveau pool virtuel à la place.

Émulation de différents niveaux de service/QoS

Il est possible de concevoir des pools virtuels pour émuler des classes de service. Grâce à l'implémentation du pool virtuel pour Cloud volumes Service pour Azure NetApp Files, examinons comment nous pouvons configurer différentes classes de service. Configurer le back-end Azure NetApp Files avec plusieurs étiquettes représentant différents niveaux de performances. Réglez `servicelevel` aspect au niveau de performance approprié et ajouter d'autres aspects requis sous chaque étiquette. Créez désormais différentes classes de stockage Kubernetes qui seraient mappées sur différents pools virtuels. À l'aide du `parameters.selector` Chaque classe de stockage indique quels pools virtuels peuvent être utilisés pour héberger un volume.

Attribution d'un ensemble spécifique d'aspects

Il est possible de concevoir plusieurs pools virtuels, dont les aspects sont spécifiques, à partir d'un système back-end unique. Pour ce faire, configurez le back-end avec plusieurs étiquettes et définissez les aspects requis sous chaque étiquette. Créez désormais des classes de stockage Kubernetes différentes avec le `parameters.selector` champ correspondant à différents pools virtuels. Les volumes provisionnés sur le

back-end possèdent les aspects définis dans le pool virtuel choisi.

Caractéristiques des PVC qui affectent le provisionnement du stockage

Certains paramètres au-delà de la classe de stockage requise peuvent affecter le processus de décision de provisionnement Astra Trident lors de la création d'un volume persistant.

Mode d'accès

Lors de la demande de stockage via un PVC, l'un des champs obligatoires est le mode d'accès. Le mode désiré peut affecter le back-end sélectionné pour héberger la demande de stockage.

Astra Trident tentera de correspondre au protocole de stockage utilisé avec la méthode d'accès spécifiée dans la matrice suivante. Cette technologie est indépendante de la plateforme de stockage sous-jacente.

	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
ISCSI	Oui.	Oui.	Oui (bloc brut)
NFS	Oui.	Oui.	Oui.

Toute demande de volume persistant ReadWriteMany soumise à un déploiement Trident sans système back-end NFS configuré entraînera le provisionnement d'un volume. Pour cette raison, le demandeur doit utiliser le mode d'accès qui convient 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 règle de récupération et la taille peuvent être modifiées. Toutefois, certains aspects du volume ne peuvent pas être modifiés en dehors de Kubernetes. Vous pouvez ainsi personnaliser le volume pour des applications spécifiques, en veillant à ce que la capacité ne soit pas accidentellement consommée ou tout simplement pour déplacer le volume vers un autre contrôleur de stockage pour n'importe quelle raison.

> Les actuellement sur provisionnement des arborescences Kubernetes ne prennent pas en charge les opérations de redimensionnement des volumes pour les volumes NFS ou iSCSI PVS. Astra Trident prend en charge l'extension des volumes NFS et iSCSI.

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

Création de copies Snapshot de volume à la demande

Astra Trident prend en charge la création de copies Snapshot de volume à la demande et la création de demandes de volume persistant à partir de copies Snapshot via le framework CSI. Les snapshots constituent une méthode pratique de conservation des copies ponctuelles des données et ont un cycle de vie indépendant du volume persistant source dans Kubernetes. Ces snapshots peuvent être utilisés pour cloner des demandes de volume persistant.

Créer des volumes à partir de copies Snapshot

Astra Trident prend également en charge la création de volumes persistant à partir des snapshots de volume. Pour ce faire, il suffit de créer une demande de volume persistant et de mentionner le `datasource` l'instantané requis à partir duquel le volume doit être créé. Astra Trident va gérer ce volume de volume

persistant en créant un volume dont les données sont présentes sur le snapshot. Grâce à cette fonctionnalité, il est possible de dupliquer des données entre régions, de créer des environnements de test, de remplacer un volume de production endommagé ou corrompu dans son intégralité, ou de récupérer des fichiers et des répertoires spécifiques et de les transférer vers un autre volume attaché.

Déplacement des volumes dans le cluster

Les administrateurs du stockage peuvent déplacer des volumes entre les agrégats et les contrôleurs du cluster ONTAP sans interruption pour l'utilisateur du stockage. Cette opération n'affecte pas Astra Trident ou le cluster Kubernetes, tant que l'agrégat de destination est un auquel le SVM utilisé par Astra Trident a accès. Important : si l'agrégat a été récemment ajouté au SVM, le système back-end devra être actualisé en le ajoutant à Astra Trident. Cela déclenchera l'Astra Trident afin de réinventorier la SVM afin que le nouvel agrégat soit reconnu.

Néanmoins, Astra Trident ne prend pas automatiquement en charge le déplacement des volumes entre les systèmes back-end. Il s'agit notamment d'étendre les SVM au sein d'un même cluster, entre plusieurs clusters ou sur une autre plateforme de stockage (même si ce système est un SVM connecté à Astra Trident).

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

Développement des volumes

Astra Trident prend en charge le redimensionnement des volumes NFS et iSCSI PVS. Les utilisateurs peuvent ainsi 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 systèmes back-end Cloud Volumes Service. Pour permettre une extension possible ultérieurement, définissez `allowVolumeExpansion` à `true` Dans votre classe de stockage associée au volume. Lorsque le volume persistant doit être redimensionné, modifiez le `spec.resources.requests.storage` Annotation dans la demande de volume persistant vers la taille de volume requise. Trident s'occupe 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. Cette opération est actuellement prise en charge par `ontap-nas`, `ontap-nas-flexgroup`, `solidfire-san`, `azure-netapp-files`, et `gcp-cvs` pilotes. Cette fonctionnalité est utile lors du portage d'une application existante sur Kubernetes ou lors de scénarios de reprise après incident.

Lorsque vous utilisez ONTAP et `solidfire-san` pilotes, utilisez la commande `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` Pour importer un volume existant dans Kubernetes et le gérer par Astra Trident. Le fichier PVC YAML ou JSON utilisé dans la commande de volume d'importation pointe vers une classe de stockage qui identifie Astra Trident comme provisionneur. Si vous utilisez un système back-end NetApp HCI/SolidFire, assurez-vous que les noms des volumes sont uniques. Si les noms des volumes sont dupliqués, cloner le volume en un nom unique afin que la fonctionnalité d'importation des volumes puisse les distinguer.

Si le `azure-netapp-files` ou `gcp-cvs` pilote utilisé, utilisez la commande `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` Pour importer le volume dans Kubernetes qui sera géré par Astra Trident. Cela garantit une référence de volume unique.

À l'exécution de la commande ci-dessus, Astra Trident trouve le volume sur le back-end et lit sa taille. Il ajoute automatiquement (et écrase si nécessaire) la taille du volume de la demande de volume configurée. Astra Trident crée ensuite le nouveau volume persistant, et Kubernetes lie la demande de volume persistant au volume persistant.

Lorsqu'un conteneur a été déployé de façon à ce qu'il ait besoin de la demande de volume persistant importée spécifique, il resterait dans un état 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 liée, le conteneur doit s'installer, à condition qu'il n'y ait pas d'autres problèmes.

Le déploiement des services OpenShift

Les services de cluster à valeur ajoutée OpenShift offrent des fonctionnalités importantes aux administrateurs de clusters et aux applications hébergées. Le stockage utilisé par ces services peut être provisionné à l'aide des ressources locales. Toutefois, la capacité, la performance, la récupération et la durabilité du service sont souvent limitées. En tirant parti d'une baie de stockage d'entreprise pour fournir la capacité nécessaire à ces services, nous pouvons obtenir un service considérablement amélioré. Cependant, comme pour toutes les applications, OpenShift et les administrateurs de stockage doivent travailler en étroite collaboration afin de déterminer les options les plus adaptées à chacun d'entre eux. La documentation Red Hat doit être largement exploitée pour déterminer les exigences et s'assurer que les besoins en matière de dimensionnement et de performances sont satisfaits.

Service de registre

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 avec Ansible, avec les paramètres de configuration fournis par le fichier d'inventaire, également appelé hôtes, fournis avec le manuel de vente. Deux méthodes d'installation sont proposées : le déploiement de la journalisation lors de l'installation initiale d'OpenShift et le déploiement de la journalisation une fois OpenShift installé.

> À partir de la version 3.9 de Red Hat OpenShift, la documentation officielle recommande à NFS d'utiliser le service de journalisation en raison de problèmes de corruption des données. Ceci est basé sur les tests Red Hat de leurs produits. Le serveur NFS ONTAP ne présente pas ces problèmes et peut facilement soutenir un déploiement de journalisation. En fin de compte, le choix du protocole pour le service de journalisation constitue un bon choix. Il suffit de savoir que les deux fonctionneront bien avec les plateformes NetApp. Il n'y a aucune raison d'éviter NFS si c'est votre choix.

Si vous choisissez d'utiliser NFS avec le service de journalisation, vous devez définir la variable Ansible `openshift_enable_unsupported_configurations` à `true` pour éviter que le programme d'installation ne tombe en panne.

Commencez

Le service de journalisation peut, éventuellement, être déployé pour les deux applications ainsi que pour les opérations de base du cluster OpenShift. 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 des opérations contiennent des "OPS", alors que l'instance des applications ne le fait pas.

Il est important de configurer les variables Ansible selon la méthode de déploiement afin de s'assurer que le stockage approprié est utilisé par les services sous-jacents. Examinons les options de 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 trouverez d'autres options dans "Documentation de journalisation Red Hat OpenShift" quels domaines doivent être examinés, configurés et utilisés en fonction de votre déploiement ?

Les variables du tableau ci-dessous entraînent la création d'un volume persistant et de demande de volume persistant pour le service de journalisation à l'aide des informations fournies. Cette méthode est beaucoup moins flexible qu'avec le manuel d'installation des composants après l'installation d'OpenShift. Toutefois, si des volumes sont déjà disponibles, il s'agit d'une option.

Variable	Détails
`openshift_logging_storage_kind`	Réglez sur `nfs` Pour que le programme d'installation crée un volume persistant NFS pour le service de journalisation.
`openshift_logging_storage_host`	Le nom d'hôte ou l'adresse IP de l'hôte NFS. Il doit être défini sur la LIF de données pour votre machine virtuelle.
`openshift_logging_storage_nfs_directory`	Chemin de montage pour l'exportation NFS. Par exemple, si le volume est relié par jonction `/openshift_logging`, vous utiliserez ce chemin pour cette variable.
`openshift_logging_storage_volume_name`	Le nom, par exemple `pv_ose_logs`, De la PV à créer.
`openshift_logging_storage_volume_size`	Taille de l'exportation NFS, par exemple `100Gi`.

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 doivent être configurées.

Variable	Détails
`openshift_logging_es_pvc_dynamic`	Définis sur true pour l'utilisation de volumes provisionnés dynamiquement.
`openshift_logging_es_pvc_storage_class_name`	Nom de la classe de stockage qui sera utilisée dans le PVC.
`openshift_logging_es_pvc_size`	Taille du volume demandé dans la demande de volume persistant.
`openshift_logging_es_pvc_prefix`	Préfixe pour les ESV utilisés par le service de journalisation.
`openshift_logging_es_ops_pvc_dynamic`	Réglez sur `true` utilisation de volumes provisionnés dynamiquement pour l'instance de journalisation des opérations.
`openshift_logging_es_ops_pvc_storage_class_name`	Nom de la classe de stockage de l'instance de journalisation OPS.
`openshift_logging_es_ops_pvc_size`	Taille de la demande de volume pour l'instance OPS.
`openshift_logging_es_ops_pvc_prefix`	Préfixe pour les ESV de l'instance OPS.

Déploiement de la pile de consignation

Si vous déployez la connexion dans le cadre du processus d'installation initiale d'OpenShift, il vous suffit de suivre le processus de déploiement standard. Ansible configure et déploie les services et les objets OpenShift nécessaires, de sorte que le service soit disponible dès qu'Ansible se termine.

Cependant, si vous déployez après l'installation initiale, vous devez utiliser le PlayBook des composants Ansible. Ce processus peut légèrement évoluer avec différentes versions d'OpenShift, c'est pourquoi nous vous invitons à le lire et à le suivre "Documentation Red Hat OpenShift Container Platform 3.11" pour votre version.

Services de metrics

Le service de metrics fournit à l'administrateur des informations précieuses sur l'état, l'utilisation des ressources et la disponibilité du cluster OpenShift. Il est également nécessaire d'utiliser la fonctionnalité de montée en charge automatique des pods. De nombreuses entreprises utilisent les données issues du service de metrics pour leurs applications de refacturation et/ou de show-back.

Comme pour le service de journalisation, OpenShift dans son ensemble, Ansible est utilisé pour déployer le service de metrics. De même, tout comme le service de journalisation, le service de metrics peut être déployé lors de la configuration initiale du cluster ou après son fonctionnement à l'aide de la méthode d'installation des composants. Les tableaux suivants contiennent les variables importantes lors de la configuration du stockage persistant pour le service de metrics.

> Les tableaux ci-dessous contiennent uniquement les variables pertinentes pour la configuration du stockage car elles concernent le service de metrics. De nombreuses autres options sont disponibles dans la documentation qui doit être examinée, configurée et utilisée en fonction de votre déploiement.

Variable	Détails
`openshift_metrics_storage_kind`	Réglez sur `nfs` Pour que le programme d'installation crée un volume persistant NFS pour le service de journalisation.
`openshift_metrics_storage_host`	Le nom d'hôte ou l'adresse IP de l'hôte NFS. Il doit être défini sur la LIF de données pour votre SVM.
`openshift_metrics_storage_nfs_directory`	Chemin de montage pour l'exportation NFS. Par exemple, si le volume est relié par jonction `/openshift_metrics`, vous utiliserez ce chemin pour cette variable.
`openshift_metrics_storage_volume_name`	Le nom, par exemple `pv_ose_metrics`, De la PV à créer.
`openshift_metrics_storage_volume_size`	Taille de l'exportation NFS, par exemple `100Gi`.

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 doivent être configurées.

Variable	Détails
`openshift_metrics_cassandra_pvc_prefix`	Préfixe à utiliser pour les ESV de metrics.

Variable	Détails
`openshift_metrics_cassandra_pvc_size`	Taille des volumes à demander.
`openshift_metrics_cassandra_storage_type`	Le type de stockage à utiliser pour les metrics, doit être défini sur dynamique pour qu'Ansible crée des demandes de volume persistant avec la classe de stockage appropriée.
`openshift_metrics_cassanda_pvc_storage_class_name`	Nom de la classe de stockage à utiliser.

Déployez le service de metrics

Déployez le service à l'aide des variables Ansible appropriées définies dans votre fichier hôtes/d'inventaire. Si vous déployez au moment de l'installation d'OpenShift, le volume persistant est créé et utilisé automatiquement. Si vous déployez à l'aide des playbooks des composants après l'installation d'OpenShift, Ansible crée les demandes PVCS requises et, une fois qu'Astra Trident a provisionné le stockage pour eux, déploie le service.

Les variables ci-dessus et le processus de déploiement peuvent changer avec chaque version d'OpenShift. Vérifiez et suivez "Guide de déploiement OpenShift de Red Hat" pour votre version afin qu'elle soit configurée pour votre environnement.

Protection des données et reprise d'activité

Découvrez les options de protection et de restauration d'Astra Trident et des volumes créés à l'aide d'Astra Trident. Vous devez disposer d'une stratégie de protection et de restauration des données pour chaque application ayant des exigences de persistance.

Réplication et restauration d'Astra Trident

Vous pouvez créer une sauvegarde pour restaurer Astra Trident en cas d'incident.

Réplication Astra Trident

ASTRA Trident utilise des CRD Kubernetes pour stocker et gérer son propre état et le cluster Kubernetes etcd pour stocker ses métadonnées.

Étapes

1. Sauvegardez le cluster Kubernetes avec "Kubernetes : sauvegarde d'un cluster ETCD".
2. Placez les artéfacts de sauvegarde sur une FlexVol.

> Nous vous recommandons de protéger la SVM où réside la FlexVol avec une relation SnapMirror vers une autre SVM.

Restauration d'Astra Trident

À l'aide des CRD Kubernetes et du snapshot de groupe Kubernetes, vous pouvez restaurer Astra Trident.

Étapes

1. Depuis le SVM de destination, monter le volume qui contient les fichiers de données et les certificats Kubernetes sur l'hôte qui sera configuré en tant que nœud maître.
2. Copiez tous les certificats requis en rapport avec le 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 à l'aide de "Kubernetes : restauration d'un cluster ETCD".
4. Courez `kubectl get crd` Pour vérifier que toutes les ressources personnalisées Trident sont disponibles et récupérer les objets Trident afin de vérifier que toutes les données sont disponibles.

Réplication et restauration des SVM

ASTRA Trident ne peut pas configurer les relations de réplication, mais l'administrateur du stockage peut utiliser "SnapMirror ONTAP" Pour répliquer une SVM.

En cas d'incident, vous pouvez activer la SVM de destination SnapMirror pour démarrer le service des données. Vous pouvez revenir au système principal lorsque les systèmes sont restaurés.

Description de la tâche

Tenir compte des points suivants lors de l'utilisation de la fonction de réplication SVM SnapMirror :

- Vous devez créer un back-end distinct pour chaque SVM lorsque la fonction SVM-DR est activée.
- Configurez les classes de stockage pour sélectionner les systèmes back-end répliqués uniquement en cas de besoin, afin d'éviter que des volumes ne nécessitant pas de réplication provisionnée vers les systèmes back-end qui prennent en charge la SVM-DR.
- Les administrateurs d'applications doivent comprendre les coûts et la complexité supplémentaires associés à la réplication et tenir compte de leur plan de reprise avant de commencer ce processus.

Réplication SVM

Vous pouvez utiliser "ONTAP : réplication SVM SnapMirror" Pour créer la relation de réplication de SVM.

SnapMirror vous permet de définir des options pour contrôler ce qui doit être répliqué. Vous devez savoir quelles options vous avez sélectionnées lors du préformage Restauration d'un SVM avec Astra Trident.

- "-identité-préserver vrai" Réplique l'ensemble de la configuration du SVM.
- "-discard-configs réseau" Exclut les LIFs et les paramètres réseau associés.
- "-identity-preserve false" réplique uniquement les volumes et la configuration de sécurité.

Restauration d'un SVM avec Astra Trident

Astra Trident ne détecte pas automatiquement les défaillances du SVM. En cas d'incident, l'administrateur peut initier manuellement le basculement de Trident vers le nouveau SVM.

Étapes

1. Annuler les transferts SnapMirror planifiés et en cours, rompre la relation de réplication, arrêter la SVM source, puis activer la SVM de destination SnapMirror.
2. Si vous avez spécifié `-identity-preserve false` ou `-discard-config network` Lors de la configuration de la réplication de votre SVM, mettre à jour `managementLIF` et `dataLIF` Dans le fichier de définition du back-end Trident.

3. Confirmer `storagePrefix` Est présent dans le fichier de définition du back-end Trident. Ce paramètre ne peut pas être modifié. Omission `storagePrefix` provoque l'échec de la mise à jour du back-end.
4. Mettre à jour tous les systèmes back-end nécessaires pour indiquer le nom du nouveau SVM de destination à l'aide de :

```
./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 rebondir tous les pods d'application.

> Si vous avez spécifié `-identity-preserve true`, Tous les volumes provisionnés par Astra Trident commencent à transmettre des données lorsque le SVM de destination est activé.

Réplication et restauration de volume

ASTRA Trident ne peut pas configurer les relations de réplication SnapMirror, mais l'administrateur du stockage peut les utiliser "Réplication et restauration ONTAP SnapMirror" Pour répliquer les volumes créés par Astra Trident.

Vous pouvez ensuite importer les volumes récupérés dans Astra Trident à l'aide de "importation de volume tridentctl".

> L'importation n'est pas prise en charge sur `ontap-nas-economy`, `ontap-san-economy`, ou `ontap-flexgroup-economy` pilotes.

Protection des données Snapshot

Vous pouvez protéger et restaurer les données à l'aide des éléments suivants :

- Un contrôleur de snapshot externe et des CRD pour créer des copies Snapshot de volume Kubernetes de volumes persistants (PVS).

  "Snapshots de volume"

- Snapshots ONTAP pour restaurer le contenu complet d'un volume ou pour restaurer des fichiers individuels ou des LUN.

  "Snapshots ONTAP"

Réplication des applications Astra Control Center

Avec Astra Control, vous pouvez répliquer les modifications des données et des applications d'un cluster à un autre à l'aide des fonctionnalités de réplication asynchrone de SnapMirror.

"ASTRA Control : réplication d'applications sur un système distant à l'aide de la technologie SnapMirror"

Sécurité

Sécurité

Assurez-vous que l'installation d'Astra Trident est sécurisée à l'aide des recommandations indiquées ici.

Exécutez Astra 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 Astra Trident ou aux pods pour assurer un stockage fiable et bloquer tout risque d'activité malveillante.

Pour séparer les autres applications et utilisateurs d'Astra Trident, installez toujours Astra Trident dans son propre espace de noms Kubernetes (`trident`). L'utilisation d'Astra Trident dans son propre espace de noms garantit que seul le personnel d'administration Kubernetes a accès au pod Trident Astra et aux artéfacts (tels que les secrets d'arrière-plan et CHAP le cas échéant) stockés dans les objets CRD devant être namestes. Vous devez vous assurer que seuls les administrateurs ont accès à l'espace de noms Astra Trident et y ont donc accès `tridentctl` client supplémentaire.

Utilisez l'authentification CHAP avec les systèmes back-end ONTAP SAN

Astra Trident prend en charge l'authentification CHAP pour les workloads SAN de ONTAP (à l'aide du `ontap-san` et `ontap-san-economy` pilotes). NetApp recommande d'utiliser le protocole CHAP bidirectionnel avec Astra Trident pour l'authentification entre l'hôte et le système back-end de stockage.

Pour les systèmes ONTAP back-end qui utilisent les pilotes de stockage SAN, Astra Trident peut configurer le protocole CHAP bidirectionnel et gérer les noms d'utilisateur et les secrets CHAP via `tridentctl`. Reportez-vous à la section "" Pour comprendre comment Astra Trident configure le protocole CHAP sur les systèmes back-end ONTAP.

Utilisez l'authentification CHAP avec les systèmes back-end NetApp HCI et SolidFire

NetApp recommande de déployer le protocole CHAP bidirectionnel pour garantir l'authentification entre l'hôte et les systèmes back-end NetApp HCI et SolidFire. Astra Trident utilise un objet secret qui inclut deux mots de passe CHAP par locataire. Une fois Astra Trident installé, il gère les secrets CHAP et les stocke dans un `tridentvolume` Objet CR pour la PV correspondante. Lorsque vous créez un volume persistant, Astra Trident utilise les secrets CHAP pour initier une session iSCSI et communiquer avec le système NetApp HCI et SolidFire sur CHAP.

> Les volumes créés par Astra Trident ne sont associés à aucun groupe d'accès de volume.

Utilisez Astra Trident avec NVE et NAE

NetApp ONTAP assure le chiffrement des données au repos pour protéger les données sensibles en cas de vol, de retour ou de reconversion d'un disque. Pour plus de détails, reportez-vous à "Configurer la présentation de NetApp Volume Encryption".

- Si NAE est activé sur le back-end, tous les volumes provisionnés dans Astra Trident seront NAE.
- Si NAE n'est pas activé sur le back-end, les volumes provisionnés dans Astra Trident seront compatibles avec NVE, à moins que vous n'ayez défini le indicateur de chiffrement NVE sur `false` en configuration back-end.

> Les volumes créés dans Astra Trident sur un système back-end compatible NAE doivent être chiffrés NVE ou NAE.
>
> - Vous pouvez définir l'indicateur de chiffrement NVE sur `true` Dans la configuration back-end Trident pour remplacer le chiffrement NAE et utiliser une clé de chiffrement spécifique sur la base du volume.
> - Définition de l'indicateur de chiffrement NVE sur `false` Sur un système back-end NAE, un volume basé sur NAE est créé. 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 Astra Trident en définissant explicitement l'indicateur de chiffrement NVE sur `true`.

Pour plus d'informations sur les options de configuration du back-end, reportez-vous à :

- "Options de configuration du SAN ONTAP"
- "Options de configuration du stockage NAS ONTAP"

Configuration de clé unifiée Linux (LUKS)

Vous pouvez activer l'utilitaire Linux Unified Key Setup (LUKS) pour chiffrer les volumes SAN ONTAP et SAN ONTAP ÉCONOMIQUES sur Astra Trident. Astra Trident prend en charge la rotation de phrase secrète et l'extension de volume pour les volumes chiffrés LUKS.

Dans Astra Trident, les volumes chiffrés LUKS utilisent le sypher et le mode aes-xts-m64, comme recommandé par "NIST".

Avant de commencer

- Les nœuds worker doivent avoir cryptsetup 2.1 ou supérieur (mais inférieur à 3.0) installé. Pour plus d'informations, rendez-vous sur "Gitlab : cryptsetup".
- Pour des raisons de performances, nous recommandons aux nœuds workers de prendre en charge les nouvelles instructions AES-ni (Advanced Encryption Standard New instructions). Pour vérifier la prise en charge AES-ni, exécutez la commande suivante :

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

  Si rien n'est renvoyé, votre processeur ne prend pas en charge AES-ni. Pour plus d'informations sur AES-ni, visitez le site : "Intel : instructions AES-ni (Advanced Encryption Standard instructions)".

Activez le cryptage LUKS

Vous pouvez activer le chiffrement côté hôte par volume en utilisant Linux Unified Key Setup (LUKS) pour SAN ONTAP et les volumes ÉCONOMIQUES SAN ONTAP.

Étapes

1. Définissez les attributs de cryptage LUKS dans la configuration back-end. Pour plus d'informations sur les options de configuration des back-end pour SAN ONTAP, reportez-vous à "Options de configuration du SAN ONTAP".

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

2. Utiliser `parameters.selector` Pour définir les pools de stockage à l'aide du cryptage 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 qui contient 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 LUKS-chiffrés ne peuvent pas tirer parti de la déduplication et de la compression ONTAP.

Configuration back-end pour l'importation de volumes LUKS

Pour importer un volume LUKS, vous devez le définir `luksEncryption` à(`true` sur le back-end. Le `luksEncryption` Indique à Astra Trident si le volume est conforme LUKS (`true`) Ou non conforme LUKS (`false`) comme indiqué dans 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'
```

Faites pivoter une phrase de passe LUKS

Vous pouvez faire pivoter la phrase de passe LUKS et confirmer la rotation.

> N'oubliez pas une phrase de passe tant que vous n'avez pas vérifié qu'elle n'est plus référencée par un volume, un snapshot ou un secret. En cas de perte d'une phrase secrète référencée, vous risquez de ne pas pouvoir monter le volume et les données resteront cryptées et inaccessibles.

Description de la 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 pod, Astra Trident compare la phrase de passe LUKS sur le volume à la phrase de passe active dans le secret.

- Si la phrase de passe du volume ne correspond pas à la phrase de passe active dans le secret, la rotation se produit.
- Si la phrase de passe du volume correspond à la phrase de passe active dans le secret, le `previous-luks-passphrase` paramètre ignoré.

Étapes

1. Ajoutez le `node-publish-secret-name` et `node-publish-secret-namespace` Paramètres de classe de stockage. 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.

Volumétrie

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

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

Snapshot

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

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

3. Mettez à jour le secret LUKS pour le volume afin de spécifier les phrases de passe nouvelles et précédentes. Bien sûr `previous-luke-passphrase-name` et `previous-luks-passphrase` faites correspondre 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 qui monte le volume. Ceci est nécessaire pour lancer la rotation.
5. Vérifiez que la phrase de passe a été pivotée.

Volumétrie

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

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

Snapshot

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

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

Résultats

La phrase de passe a été pivotée lorsque seule la nouvelle phrase de passe est renvoyée sur le volume et le snapshot.

> Si deux phrases de passe sont renvoyées, par exemple `luksPassphraseNames: ["B", "A"]`, la rotation est incomplète. Vous pouvez déclencher un nouveau pod pour tenter de terminer la rotation.

Activer l'extension de volume

Vous pouvez activer l'extension de volume sur un volume chiffré LUKS.

Étapes

1. Activez le `CSINodeExpandSecret` feature gate (bêta 1.25+). Reportez-vous à la section "Kubernetes 1.25 : utilisez les secrets de l'extension des volumes CSI basée sur des nœuds" pour plus d'informations.
2. Ajoutez le `node-expand-secret-name` et `node-expand-secret-namespace` Paramètres de classe de stockage. 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 initiez l’extension du stockage en ligne, le kubelet transmet les identifiants appropriés au pilote.

Connaissances et support

Foire aux questions

Trouvez les réponses aux questions fréquemment posées concernant l'installation, la configuration, la mise à niveau et le dépannage d'Astra Trident.

Questions générales

À quelle fréquence Astra Trident est-il commercialisé ?

Astra Trident est commercialisé tous les trois mois : janvier, avril, juillet et octobre. Ceci fait un mois après une version de Kubernetes.

Est-ce que l'Astra Trident prend en charge toutes les fonctionnalités publiées dans une version particulière de Kubernetes ?

Astra Trident ne prend généralement pas en charge les fonctionnalités alpha dans Kubernetes. Trident peut prendre en charge les fonctionnalités bêta dans les deux versions de Trident, qui suivent la version bêta de Kubernetes.

Est-ce que l'Astra Trident se base sur d'autres produits NetApp pour son fonctionnement ?

Astra Trident ne fonctionne pas sous forme de dépendance vis-à-vis d'autres logiciels NetApp, et il fonctionne comme une application autonome. Toutefois, vous devez disposer d'un système de stockage back-end NetApp.

Comment obtenir les informations complètes sur la configuration d'Astra Trident ?

Utilisez le `tridentctl get` Commande pour obtenir plus d'informations sur la configuration d'Astra Trident.

Puis-je obtenir des metrics sur le provisionnement du stockage par Astra Trident ?

Oui. Les terminaux Prometheus peuvent être utilisés pour rassembler des informations sur les opérations d'Astra Trident, telles que le nombre de systèmes back-end gérés, le nombre de volumes provisionnés, d'octets consommés, etc. Vous pouvez également utiliser "Cloud Insights" pour la surveillance et l'analyse.

L'expérience utilisateur change-t-elle lors de l'utilisation d'Astra Trident en tant que provisionnement CSI ?

Non Il n'y a aucun changement en ce qui concerne l'expérience utilisateur et les fonctionnalités. Le nom de provisionnement utilisé est `csi.trident.netapp.io`. Cette méthode d'installation d'Astra Trident est recommandée si vous souhaitez utiliser toutes les nouvelles fonctionnalités fournies par les versions actuelles et futures.

Installez et utilisez Astra Trident sur un cluster Kubernetes

Est-ce que Astra Trident prend en charge une installation hors ligne à partir d'un registre privé ?

Oui, Astra Trident peut être installé hors ligne. Reportez-vous à la section "Découvrez l'installation d'Astra Trident".

Puis-je installer Astra Trident à distance ?

Oui. Avec Astra Trident 18.10 et les versions ultérieures, il prend en charge la fonctionnalité d'installation à distance à partir de n'importe quelle machine `kubectl` accès au cluster. Après `kubectl` vérification de l'accès (par exemple, lancement d'un `kubectl get nodes` commande de la machine à distance pour la vérification), suivre les instructions d'installation.

Puis-je configurer la haute disponibilité avec Astra Trident ?

Astra Trident est installé en tant que Kubernetes Deployment (ReplicaSet) avec une instance, et ce de même qu'il a intégré la haute disponibilité. Vous ne devez pas augmenter le nombre de répliques dans le déploiement. Si le nœud sur lequel Astra Trident est installé ou si le pod est inaccessible, Kubernetes redéploie automatiquement le pod sur un nœud en état de santé dans votre cluster. L'Astra Trident est uniquement du plan de contrôle. Les pods actuellement montés ne sont donc pas affectés en cas de redéploiement de l'Astra Trident.

Astra Trident a-t-il besoin d'accéder au namespace du système kube ?

L'Astra Trident effectue une lecture sur le serveur d'API Kubernetes afin de déterminer le moment où les applications demandent des nouveaux ESV. Il doit donc accéder à kube-System.

Quels sont les rôles et privilèges utilisés par Astra Trident ?

Le programme d'installation Trident crée un cluster Kubernetes ClusterRole, qui dispose d'un accès spécifique aux ressources PersistentVolume, PersistentVolumeClaim, StorageClass et Secret du cluster Kubernetes. Reportez-vous à la section "Personnalisez l'installation tridentctl".

Est-il possible de générer localement les fichiers de manifeste exacts qu'Astra Trident utilise pour l'installation ?

Vous pouvez générer et modifier localement les fichiers de manifeste exacts utilisés par Astra Trident pour l'installation, si nécessaire. Reportez-vous à la section "Personnalisez l'installation tridentctl".

Puis-je partager le même SVM back-end ONTAP pour deux instances Astra Trident distinctes pour deux clusters Kubernetes distincts ?

Bien qu'il ne soit pas conseillé, vous pouvez utiliser le même SVM back-end pour deux instances Astra Trident. Spécifiez un nom de volume unique pour chaque instance lors de l'installation et/ou spécifiez un nom unique `StoragePrefix` paramètre dans le `setup/backend.json` fichier. Ceci permet de s'assurer que le même FlexVol n'est pas utilisé pour les deux instances.

Est-il possible d'installer Astra Trident sous ContainerLinux (anciennement CoreOS) ?

Astra Trident est un simple pod Kubernetes. Il peut être installé quel que soit l'emplacement de Kubernetes.

Puis-je utiliser Astra Trident avec NetApp Cloud Volumes ONTAP ?

Oui, Astra Trident est pris en charge par AWS, Google Cloud et Azure.

Astra Trident fonctionne-t-il avec NetApp Cloud volumes Services ?

Oui. Astra Trident prend en charge le service Azure NetApp Files dans Azure ainsi que le service Cloud Volumes Service dans GCP.

Dépannage et support

NetApp prend-il en charge Astra Trident ?

Bien qu'Astra Trident soit open source et fourni gratuitement, NetApp le prend entièrement en charge à condition que votre système back-end NetApp soit pris en charge.

Comment puis-je soulever un dossier de demande de support ?

Pour soulever un dossier de support, effectuez l'une des opérations suivantes :

1. Contactez votre support Account Manager pour obtenir de l'aide pour créer un dossier.
2. Pour ouvrir un dossier de demande de support, contactez "Support NetApp".

Comment générer un bundle de journaux de support ?

Vous pouvez créer un bundle de support en exécutant `tridentctl logs -a`. Outre les journaux capturés dans le pack, capture le journal kubelet pour diagnostiquer les problèmes de montage côté Kubernetes. Les instructions d'obtention du journal kubelet varient en fonction de l'installation de Kubernetes.

Que faire si j'ai besoin de demander une nouvelle fonctionnalité ?

Créer un problème sur "Astra Trident Github" Et mentionner **RFE** dans le sujet et la description du problème.

Où puis-je soulever un défaut ?

Créer un problème sur "Astra Trident Github". Veillez à inclure toutes les informations et tous les journaux nécessaires concernant le problème.

Que se passe-t-il si j'ai une brève question sur Astra Trident et que j'ai besoin de précisions ? Y a-t-il une communauté ou un forum?

Si vous avez des questions, des problèmes ou des demandes, contactez-nous par le biais de notre Astra "Déroulez le canal" Ou GitHub.

Le mot de passe de mon système de stockage a changé et Astra Trident ne fonctionne plus. Comment puis-je le récupérer ?

Mettez à jour le mot de passe du back-end avec `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Remplacement `myBackend` dans l'exemple avec votre nom de back-end, et `/path/to_new_backend.json` avec le chemin d'accès correct `backend.json` fichier.

Astra Trident ne trouve pas le nœud Kubernetes. Comment résoudre ce problème ?

Il existe deux scénarios possibles pour savoir pourquoi Astra Trident ne peut pas trouver un nœud Kubernetes. Elle peut être due à un problème de mise en réseau dans Kubernetes ou DNS. Le demonset de nœuds Trident qui s'exécute sur chaque nœud Kubernetes doit pouvoir communiquer avec le contrôleur Trident pour enregistrer le nœud avec Trident. Si des modifications de mise en réseau ont eu lieu après l'installation d'Astra Trident, ce problème se produit uniquement avec les nouveaux nœuds Kubernetes qui sont ajoutés au cluster.

Si le pod Trident est détruit, ces données seront-elles perdues ?

Les données ne seront pas perdues si le pod Trident est détruit. Les métadonnées Trident sont stockées dans des objets CRD. Tous les volumes persistants provisionnés par Trident fonctionneront normalement.

Mettez à niveau Astra Trident

Est-il possible de mettre à niveau une version plus ancienne directement vers une version plus récente (sans passer par quelques versions) ?

NetApp prend en charge la mise à niveau d'Astra Trident d'une version majeure à la prochaine version majeure. Vous pouvez effectuer la mise à niveau de la version 18.xx vers la version 19.xx, 19.xx vers la version 20.xx, etc. Il est conseillé de tester la mise à niveau dans un 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 les bugs observés après une mise à niveau, des problèmes de dépendance ou une mise à niveau infructueuse ou incomplète, vous devez "Désinstallez Astra 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.

Gestion des systèmes back-end et des volumes

Dois-je définir à la fois des LIF de données et de gestion dans un fichier de définition du back-end ONTAP ?

Le LIF de gestion est obligatoire. Data LIF varie :

- San ONTAP : ne spécifiez pas pour iSCSI. Astra Trident utilise "Mappage de LUN sélectif ONTAP" Pour découvrir les LIFs iSCI nécessaires à l'établissement d'une session multi-chemin. Un avertissement est généré si `dataLIF` est explicitement défini. Reportez-vous à la section "Options et exemples de configuration des SAN ONTAP" pour plus d'informations.
- NAS ONTAP : spécification recommandée `dataLIF`. Si elle n'est pas fournie, Astra Trident extrait les LIF de données du SVM. Vous pouvez spécifier un nom de domaine complet (FQDN) à utiliser pour les opérations de montage NFS, permettant de créer un DNS Round-Robin pour équilibrer la charge sur plusieurs LIF de données. Reportez-vous à la section "Options et exemples de configuration du NAS ONTAP" pour plus d'informations

L'Astra Trident peut-il configurer le protocole CHAP pour les systèmes back-end ONTAP ?

Oui. ASTRA Trident prend en charge le protocole CHAP bidirectionnel pour les systèmes ONTAP back-end. Ceci nécessite un paramètre `useCHAP=true` dans votre configuration back-end.

Comment gérer les règles d'exportation avec Astra Trident ?

Astra Trident peut créer et gérer de manière dynamique des règles d'exportation à partir de la version 20.04. Cela permet à l'administrateur de stockage de fournir un ou plusieurs blocs CIDR dans leur configuration backend et de laisser Trident ajouter des adresses IP de nœud comprise dans ces plages à une export policy créée. Ainsi, Astra Trident gère automatiquement l'ajout et la suppression de règles pour les nœuds dont les adresses IP sont comprises dans les rapports CIDR donnés.

Les adresses IPv6 peuvent-elles être utilisées pour les LIF de données et de gestion ?

Astra Trident prend en charge la définition des adresses IPv6 pour :

- `managementLIF` et `dataLIF` Pour les systèmes NAS ONTAP.
- `managementLIF` Pour les systèmes back-end ONTAP SAN. Vous ne pouvez pas spécifier `dataLIF` Sur un SAN backend ONTAP.

ASTRA Trident doit être installé à l'aide du drapeau `--use-ipv6` (pour `tridentctl` installation), `IPv6` (Pour l'opérateur Trident), ou `tridentTPv6` (Pour l'installation Helm) pour qu'il fonctionne sur IPv6.

Est-il possible de mettre à jour la LIF de gestion en back-end ?

Oui, il est possible de mettre à jour la LIF de management back-end à l'aide de `tridentctl update backend` commande.

Est-il possible de mettre à jour la LIF de données sur le backend ?

Vous pouvez mettre à jour la LIF de données sur `ontap-nas` et `ontap-nas-economy` uniquement.

Est-il possible de créer plusieurs systèmes back-end dans Astra Trident pour Kubernetes ?

Astra Trident peut prendre en charge de nombreux systèmes back-end simultanément, avec le même pilote ou des pilotes différents.

Comment Astra Trident stocke-t-il les identifiants back-end ?

Astra Trident stocke les identifiants back-end sous le titre de secrets de Kubernetes.

Comment l'Astra Trident sélectionne-t-il un système back-end spécifique ?

Si les attributs back-end ne peuvent pas être utilisés pour sélectionner automatiquement les pools appropriés pour une classe, l' `storagePools` et `additionalStoragePools` les paramètres sont utilisés pour sélectionner un ensemble spécifique de pools.

Comment s'assurer qu'Astra Trident ne provisionne pas d'un back-end spécifique ?

Le `excludeStoragePools` Paramètre utilisé pour filtrer l'ensemble de pools qu'Astra Trident utilisera pour le provisionnement et supprimera tous les pools correspondant.

Si plusieurs systèmes back-end sont de même type, comment Astra Trident sélectionne-il le back-end à utiliser ?

Si plusieurs systèmes back-end configurés du même type sont configurés, Astra Trident sélectionne le back-end approprié en fonction des paramètres présents dans `StorageClass` et `PersistentVolumeClaim`. Par exemple, si il existe plusieurs pilotes back-end ontap-nas, Astra Trident tente de correspondre aux paramètres dans le `StorageClass` et `PersistentVolumeClaim` combiné et correspondre à un système back-end capable de fournir les exigences répertoriées dans `StorageClass` et `PersistentVolumeClaim`. Si plusieurs systèmes back-end correspondent à la demande, l'Astra Trident est sélectionnée de manière aléatoire.

Astra Trident prend-il en charge le protocole CHAP bidirectionnel avec Element/SolidFire ?

Oui.

Comment Astra Trident déploie-t-il des qtrees sur un volume ONTAP ? Combien de qtrees peuvent-ils être déployés sur un seul volume ?

Le `ontap-nas-economy` Le pilote crée jusqu'à 200 qtrees dans le même FlexVol (configurables entre 50 et 300), 100,000 qtrees par nœud de cluster et 2,4 millions par cluster. Lorsque vous saisissez un nouveau `PersistentVolumeClaim` Le pilote cherche à voir si un FlexVol existe déjà pour le service du nouveau qtree. Si la FlexVol n'existe pas qui peut traiter le qtree, un nouveau FlexVol est créé.

Comment définir des autorisations Unix pour les volumes provisionnés sur ONTAP NAS ?

Vous pouvez définir des autorisations Unix sur le volume provisionné par Astra Trident en définissant un paramètre dans le fichier de définition backend.

Comment configurer un ensemble explicite d'options de montage NFS ONTAP lors du provisionnement d'un volume ?

Par défaut, Astra Trident ne définit pas d'option de montage sur aucune valeur avec Kubernetes. Pour spécifier les options de montage dans la classe de stockage Kubernetes, suivez l'exemple donné "ici".

Comment définir les volumes provisionnés sur une export policy spécifique ?

Pour permettre aux hôtes appropriés d'accéder à un volume, utilisez le `exportPolicy` paramètre configuré dans le fichier de définition backend.

Comment définir le chiffrement de volume avec Astra Trident et ONTAP ?

Vous pouvez définir le chiffrement sur le volume provisionné par Trident à l'aide du paramètre de chiffrement dans le fichier de définition back-end. Pour plus d'informations, se reporter à : "Fonctionnement d'Astra Trident avec NVE et NAE"

Quelle est la meilleure façon d'implémenter la QoS pour ONTAP avec Astra Trident ?

Utiliser `StorageClasses` Afin d'implémenter la QoS pour ONTAP.

Comment puis-je spécifier le provisionnement fin ou non fin avec Astra Trident ?

Les pilotes ONTAP prennent en charge le provisionnement fin ou non fin. Le provisionnement fin est par défaut pour les pilotes ONTAP. Si un provisionnement lourd est souhaité, vous devez configurer le fichier de définition backend ou le `StorageClass`. Si les deux sont configurés, `StorageClass` a priorité. Configurez les éléments suivants pour ONTAP :

1. Marche `StorageClass`, réglez le `provisioningType` attribuer comme épaisseur.
2. Dans le fichier de définition back-end, activez les volumes épais par définition `backend spaceReserve parameter` comme volume.

Comment puis-je m'assurer que les volumes utilisés ne sont pas supprimés même si je supprime accidentellement le volume de volume persistant ?

La protection contre la demande de volume persistant est automatiquement activée sur Kubernetes à partir de la version 1.10.

Puis-je augmenter les demandes de volume persistant NFS créées par Astra Trident ?

Oui. Vous pouvez développer un volume de volume persistant créé par Astra Trident. Notez que la croissance automatique de volume est une fonctionnalité ONTAP qui n'est pas applicable à Trident.

Puis-je importer un volume en mode SnapMirror Data protection (DP) ou hors ligne ?

L'importation du volume échoue si le volume externe est en mode DP ou est 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 un quota de ressources est-il traduit-il vers un cluster NetApp ?

Le quota de ressources de stockage Kubernetes doit fonctionner tant que le stockage NetApp possède de la capacité. Lorsque le stockage NetApp ne peut pas respecter les paramètres de quota Kubernetes en raison d'un manque de capacité, Astra Trident tente d'effectuer le provisionnement, mais s'y efforce d'erreurs.

Est-il possible de créer des copies Snapshot de volume avec Astra Trident ?

Oui. La création à la demande de copies Snapshot de volume et de volumes persistants à partir de copies Snapshot est prise en charge par Astra Trident. Pour créer des volumes persistants à partir de snapshots, assurez-vous que l' `VolumeSnapshotDataSource` la porte de fonction a été activée.

Quels sont les pilotes qui prennent en charge les copies Snapshot de volume Astra Trident ?

Depuis, nous proposons aujourd'hui la prise en charge de snapshots à la demande `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, et `azure-netapp-files` pilotes backend.

Comment effectuer une sauvegarde Snapshot d'un volume provisionné par Astra Trident avec ONTAP ?

Cette option est disponible sur `ontap-nas`, `ontap-san`, et `ontap-nas-flexgroup` pilotes. Vous pouvez également spécifier un `snapshotPolicy` pour le `ontap-san-economy` Pilote au niveau FlexVol.

Cette fonction est également disponible sur le `ontap-nas-economy` Pilotes mais au niveau de la granularité FlexVol, pas au niveau de la granularité qtree. Pour permettre la création de copies Snapshot provisionnées par Astra Trident, définissez le paramètre back-end `snapshotPolicy` À la politique de snapshot souhaitée, telle que définie sur le back-end ONTAP. Tout snapshot effectué par le contrôleur de stockage ne est pas connu d'Astra Trident.

Puis-je définir un pourcentage de réserve de snapshot pour un volume provisionné via Astra Trident ?

Oui. Il est possible de réserver un pourcentage spécifique d'espace disque pour le stockage des copies Snapshot via Astra Trident en configurant le `snapshotReserve` attribut dans le fichier de définition backend. Si vous avez configuré `snapshotPolicy` et `snapshotReserve` dans le fichier de définition backend, le

pourcentage de réserve de snapshot est défini en fonction de la `snapshotReserve` pourcentage indiqué dans le fichier back-end. Si le `snapshotReserve` Le pourcentage de nombre n'est pas indiqué, ONTAP occupe par défaut le pourcentage de réserve Snapshot comme 5. Si le `snapshotPolicy` l'option est définie sur aucune, le pourcentage de réserve snapshot est défini sur 0.

Puis-je accéder directement au répertoire de snapshot de volume et copier les fichiers ?

Oui, vous pouvez accéder au répertoire de snapshots sur le volume provisionné par Trident en paramétrant le `snapshotDir` paramètre dans le fichier de définition backend.

Puis-je configurer SnapMirror pour des volumes avec Astra Trident ?

Actuellement, SnapMirror doit être défini en externe via l'interface de ligne de commande ONTAP ou OnCommand System Manager.

Comment restaurer des volumes persistants à un snapshot ONTAP spécifique ?

Pour restaurer un volume sur un snapshot ONTAP, effectuez les opérations suivantes :

1. Arrêter le pod d'application qui utilise le volume persistant.
2. Restaurez les données vers le snapshot requis via l'interface de ligne de commande de ONTAP ou OnCommand System Manager.
3. Redémarrez le pod d'application.

Trident peut-il provisionner des volumes sur des SVM dont un miroir de partage de charge est configuré ?

Des miroirs de partage de charge peuvent être créés pour les volumes root des SVM qui fournissent des données sur NFS. ONTAP met automatiquement à jour les miroirs de partage de charge pour les volumes qui ont été créés par Trident. Cela peut entraîner des retards dans le montage des volumes. Lorsque plusieurs volumes sont créés via 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 de la classe de stockage pour chaque client/locataire ?

Kubernetes n'autorise pas les classes de stockage dans les espaces de noms. Toutefois, vous pouvez utiliser Kubernetes pour limiter l'utilisation d'une classe de stockage spécifique par espace de noms à l'aide de quotas de ressources de stockage, qui sont par espace de noms. Pour refuser un accès d'espace de noms spécifique à un stockage spécifique, définissez le quota de ressources sur 0 pour cette classe de stockage.

Dépannage

Utilisez les pointeurs indiqués ici pour résoudre les problèmes que vous pourriez rencontrer lors de l'installation et de l'utilisation d'Astra Trident.

Dépannage général

- Si le pod Trident ne fonctionne pas correctement (par exemple, lorsque le pod Trident est coincé dans le `ContainerCreating` phase avec moins de deux conteneurs prêts à l'emploi), en cours d'exécution `kubectl -n trident describe deployment trident` et `kubectl -n trident describe pod trident-**` peut fournir des informations exploitables supplémentaires. Obtenir des journaux kubelet (par exemple, via `journalctl -xeu kubelet`) peut également être utile.

- Si les journaux Trident ne contient pas suffisamment d'informations, vous pouvez essayer d'activer le mode de débogage pour Trident en passant le `-d` permet d'indiquer le paramètre d'installation en fonction de votre option d'installation.

  Vérifiez ensuite que le débogage est défini à l'aide de `./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 modules Trident, ce qui peut prendre plusieurs secondes. Vous pouvez le vérifier en observant la colonne « ÂGE » dans la sortie de `kubectl get pod -n trident`.

  Pour utilisation d'Astra Trident 20.07 et 20.10 `tprov` à la place 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 back-end en incluant `debugTraceFlags` dans votre définition de back-end. Par exemple, incluez `debugTraceFlags: {"api":true, "method":true,}` Pour obtenir des appels d'API et des transits de méthode dans les journaux Trident. Systèmes back-end existants peuvent avoir lieu `debugTraceFlags` configuré avec un `tridentctl backend update`.
- Lorsque vous utilisez RedHat CoreOS, assurez-vous que cela `iscsid` est activé sur les nœuds workers et démarré par défaut. Pour ce faire, utilisez OpenShift MachineConfiguration ou modifiez les modèles d'allumage.
- Un problème courant que vous pouvez rencontrer avec Trident "Azure NetApp Files" lorsque les secrets de locataire et de client proviennent d'un enregistrement d'application avec des autorisations insuffisantes. Pour obtenir la liste complète de la configuration requise pour Trident, reportez-vous à la section "Azure NetApp Files" configuration.
- En cas de problème de montage d'un PV sur un conteneur, vérifiez que `rpcbind` est installé et en cours d'exécution. Utilisez le gestionnaire de packages requis pour le système d'exploitation hôte et vérifiez si `rpcbind` est en cours d'exécution. Vous pouvez vérifier le statut de l' `rpcbind` service en exécutant un `systemctl status rpcbind` ou son équivalent.
- Si un système Trident indique qu'il se trouve dans le `failed` État bien qu'il ait auparavant travaillé, il est probable que cela soit causé par la modification des identifiants SVM/admin associés au back-end. Mise à jour des informations du back-end à l'aide de `tridentctl update backend` Vous pouvez également rebondir sur le pod Trident pour résoudre ce problème.

- Si vous rencontrez des problèmes d'autorisation lors de l'installation de Trident avec Docker comme conteneur d'exécution, essayez d'installer Trident avec le `--in cluster=false` drapeau. Ceci n'utilise pas de module d'installation et évite les problèmes de permission observés en raison de l' `trident-installer` utilisateur.
- Utilisez le `uninstall parameter <Uninstalling Trident>` pour le nettoyage après un échec d'exécution. Par défaut, le script ne supprime pas les CRD créés par Trident, ce qui rend possible leur désinstallation et leur installation en toute sécurité, même dans le cadre d'un déploiement en cours d'exécution.
- Si vous souhaitez effectuer une mise à niveau vers une version antérieure de Trident, exécutez d'abord le `tridentctl uninstall` Commande de suppression de Trident. Télécharger le fichier désiré "Version Trident" et installer à l'aide de `tridentctl install` commande.
- Après une installation réussie, si un PVC est bloqué dans le `Pending` phase, exécution `kubectl describe pvc` Peut fournir des informations supplémentaires sur les raisons pour lesquelles Trident n'a pas pu provisionner un volume persistant pour cette demande de volume persistant.

Échec du déploiement de Trident avec l'opérateur

Si vous déployez Trident à l'aide de l'opérateur, le statut de `TridentOrchestrator` modifications de `Installing` à `Installed`. Si vous observez l' `Failed` status, et l'opérateur ne peut pas récupérer en lui-même, il est recommandé de vérifier les journaux de l'opérateur en exécutant la commande suivante :

```
tridentctl logs -l trident-operator
```

Traînant les journaux du conteneur de l'opérateur trident peut pointer vers l'emplacement où se trouve le problème. Par exemple, un tel problème pourrait être l'impossibilité d'extraire les images de conteneur requises des registres en amont dans un environnement mis à l'air.

Pour comprendre pourquoi l'installation de Trident n'a pas été effectuée, consultez le `TridentOrchestrator` état.

```
kubectl 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`Utilisé pour installer Trident. Étant donné que chaque cluster Kubernetes ne peut avoir qu'une seule instance de Trident, l'opérateur s'assure qu'une seule instance active existe à un instant donné `TridentOrchestrator` qu'il peut créer.

De plus, l'observation de l'état des pods Trident peut souvent indiquer si quelque chose n'est pas approprié.

```
kubectl get pods -n trident

NAME                                READY   STATUS             RESTARTS
AGE
trident-csi-4p5kq                   1/2     ImagePullBackOff   0
5m18s
trident-csi-6f45bfd8b6-vfrkw        4/5     ImagePullBackOff   0
5m19s
trident-csi-9q5xc                   1/2     ImagePullBackOff   0
5m18s
trident-csi-9v95z                   1/2     ImagePullBackOff   0
5m18s
trident-operator-766f7b8658-ldzsv   1/1     Running            0
8m17s
```

Vous pouvez clairement voir que les modules ne peuvent pas être initialisés complètement parce qu'une ou plusieurs images de conteneur n'ont pas été extraites.

Pour résoudre le problème, vous devez modifier le `TridentOrchestrator` CR. Vous pouvez également supprimer `TridentOrchestrator`, et en créer un nouveau avec la définition modifiée et précise.

Échec du déploiement de Trident avec `tridentctl`

Pour vous aider à déterminer ce qui s'est mal passé, vous pouvez exécuter à nouveau le programme d'installation à l'aide du `-d` argument, qui active le mode débogage et vous aide à comprendre le problème :

```
./tridentctl install -n trident -d
```

Après avoir résolu le problème, vous pouvez nettoyer l'installation comme suit, puis exécuter le `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.
```

Retirez complètement Astra Trident et les CRD

Vous pouvez supprimer complètement Astra Trident, tous les CRD créés et les ressources personnalisées associées.

Cette opération ne peut pas être annulée. Ne le faites que si vous souhaitez une toute nouvelle installation d'Astra Trident. Pour désinstaller Astra Trident sans supprimer les CRD, reportez-vous à la section "Désinstaller Astra Trident".

Opérateur Trident

Pour désinstaller Astra 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}}'
```

Gouvernail

Pour désinstaller Astra Trident et supprimer complètement les CRD à l'aide de Helm :

```
kubectl patch torc trident --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

<code>tridentctl</code>

Pour supprimer complètement les CRD après la désinstallation d'Astra Trident à l'aide de `tridentctl`

```
tridentctl obliviate crd
```

Échec de l'annulation du transfert de nœud NVMe avec les espaces de noms de bloc bruts RWX o Kubernetes 1.26

Si vous exécutez Kubernetes 1.26, l'annulation de l'environnement de nœud peut échouer lors de l'utilisation de NVMe/TCP avec les espaces de noms de bloc bruts RWX. Les scénarios suivants offrent une solution de contournement à la défaillance. Vous pouvez également mettre à niveau Kubernetes vers la version 1.27.

Espace de noms et pod supprimés

Imaginez un espace de noms géré Astra Trident (volume persistant NVMe) attaché à un pod. Si vous supprimez l'espace de nom directement du back-end ONTAP, le processus de déstaging est bloqué après la tentative de suppression du pod. Ce scénario n'a aucun impact sur le cluster Kubernetes ou tout autre fonctionnement.

Solution de contournement

Démontez le volume persistant (correspondant à cet espace de noms) du nœud respectif et supprimez-le.

DataLIFs bloquées

```
 If you block (or bring down) all the dataLIFs of the NVMe Astra 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
Afficher les dataLIFS pour restaurer toutes les fonctionnalités.
```

Mappage de l'espace de noms supprimé

```
 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.
```

Assistance

NetApp prend en charge Astra Trident de différentes manières. De nombreuses options d'auto-assistance gratuites sont disponibles 24 h/24 et 7 j/7, comme des articles de la base de connaissances (KB) et un canal discorde.

Cycle de vie de prise en charge d'Astra Trident

ASTRA Trident offre trois niveaux de support en fonction de votre version. Reportez-vous à la section "Prise en charge de la version du logiciel NetApp pour les définitions".

Support complet
: ASTRA Trident offre un support complet pendant douze mois à compter de la date de sortie.

Prise en charge limitée
: ASTRA Trident offre un support limité pour les mois 13 à 24 à compter de la date de sortie.

Auto-assistance
: La documentation d'Astra Trident est disponible pour les mois 25 à 36 à compter de la date de publication.

Version	Support complet	Prise en charge limitée	Auto-assistance
"24.02"	Février 2025	Février 2026	Février 2027
"23.10"	Octobre 2024	Octobre 2025	Octobre 2026
"23.07"	Juillet 2024	Juillet 2025	Juillet 2026
"23.04"	Avril 2024	Avril 2025	Avril 2026

Version	Support complet	Prise en charge limitée	Auto-assistance
"23.01"	—	Janvier 2025	Janvier 2026
"22.10"	—	Octobre 2024	Octobre 2025
"22.07"	—	Juillet 2024	Juillet 2025
"22.04"	—	Avril 2024	Avril 2025
"22.01"	—	—	Janvier 2025
"21.10"	—	—	Octobre 2024
"21.07"	—	—	Juillet 2024

Auto-assistance

Pour obtenir une liste complète des articles de dépannage, reportez-vous à la section "Base de connaissances NetApp (identifiant requis)". Vous trouverez également des informations sur le dépannage des problèmes liés à Astra "ici".

Soutien de la communauté

Notre projet Astra comprend une communauté publique très vivante d'utilisateurs de conteneurs (y compris les développeurs Astra Trident) "Déroulez le canal". C'est un endroit idéal pour poser des questions d'ordre général sur le projet et discuter de sujets connexes avec des pairs partageant des mêmes idées.

Support technique NetApp

Pour obtenir de l'aide avec Astra Trident, créez un bundle de support à l'aide de `tridentctl logs -a -n trident` et envoyez-le à `NetApp Support <Getting Help>`.

Pour en savoir plus

- "Blogs Astra"
- "Blogs Trident d'Astra"
- "Kubernetes Hub"
- "NetApp.io"

Référence

Ports Trident d'Astra

Découvrez les ports qu'Astra Trident utilise pour la communication.

Ports Trident d'Astra

Astra Trident communique sur les ports suivants :

Port	Objectif
8443	HTTPS backChannel
8001	Terminal des metrics Prometheus
8000	Serveur REST Trident
17546	Port de sonde de liaison/préparation utilisé par les modules de démonset Trident

> Le port de la sonde de liaison/préparation peut être modifié lors de l'installation à l'aide du `--probe-port` drapeau. Il est important de s'assurer que ce port n'est pas utilisé par un autre processus sur les nœuds worker.

API REST d'Astra Trident

Pendant "commandes et options tridentctl" Vous pouvez utiliser le terminal REST directement si vous le souhaitez. Pour interagir avec l'API REST Astra Trident,

Quand utiliser l'API REST

Il est utile pour les installations avancées qui utilisent Astra Trident en tant que binaire autonome dans les déploiements non Kubernetes.

Avec Astra Trident, qui offre une meilleure sécurité `REST API` est limité à localhost par défaut lors de l'exécution dans un pod. Pour changer ce comportement, vous devez définir Astra Trident `-address` dans sa configuration pod.

Avec l'API REST

Pour des exemples de la façon dont ces API sont appelées, passez le débogage (`-d`) drapeau. Pour plus d'informations, reportez-vous à la section "Gérez Astra Trident à l'aide de tridentctl".

L'API fonctionne comme suit :

OBTENEZ

`GET <trident-address>/trident/v1/<object-type>`

Répertorie tous les objets de ce type.

GET <trident-address>/trident/v1/<object-type>/<object-name>

Obtient les détails de l'objet nommé.

POST

POST <trident-address>/trident/v1/<object-type>

Crée un objet du type spécifié.

- Nécessite une configuration JSON pour que l'objet soit créé. Pour la spécification de chaque type d'objet, voir "Gérez Astra Trident à l'aide de tridentctl".
- Si l'objet existe déjà, le comportement varie : les systèmes back-end mettent à jour l'objet existant, tandis que tous les autres types d'objet échoueront.

SUPPRIMER

DELETE <trident-address>/trident/v1/<object-type>/<object-name>

Supprime la ressource nommée.

> Les volumes associés aux systèmes back-end ou aux classes de stockage continueront d'exister. Ils doivent être supprimés séparément. Pour plus d'informations, reportez-vous à la section "Gérez Astra Trident à l'aide de tridentctl".

Options de ligne de commande

Astra Trident expose plusieurs options de ligne de commande pour l'orchestrateur Trident. Vous pouvez utiliser ces options pour modifier votre déploiement.

Journalisation

-debug

Active la sortie de débogage.

-loglevel <level>

Définit le niveau de journalisation (débogage, info, avertissement, erreur, fatal). La valeur par défaut est INFO.

Kubernetes

-k8s_pod

Utilisez cette option ou `-k8s_api_server` Pour activer la prise en charge de Kubernetes. La configuration de cette configuration entraîne l'utilisation par Trident des identifiants du compte de service Kubernetes du pod qui y est associé pour contacter le serveur d'API. Cela fonctionne uniquement lorsque Trident s'exécute en tant que pod dans un cluster Kubernetes avec les 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. Lorsqu'il est spécifié, Trident se connecte au serveur API Kubernetes à l'aide de l'adresse et du port non sécurisés fournis. Trident peut donc être déployé en dehors d'un pod, mais il ne prend uniquement en charge les connexions non sécurisées vers le serveur API. Pour vous connecter de manière sécurisée, déployez Trident dans un

pod avec le `-k8s_pod` option.

Docker

`-volume_driver <name>`

Nom du pilote utilisé lors de l'enregistrement du plug-in Docker. La valeur par défaut est `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 vers un fichier de configuration back-end.

REPOS

`-address <ip-or-host>`

Spécifie l'adresse à laquelle le serveur REST de Trident doit écouter. Par défaut, localhost. Lorsque vous écoutez sur localhost et exécutez-les dans un pod Kubernetes, l'interface REST n'est pas directement accessible depuis l'extérieur du pod. Utiliser `-address ""` Pour rendre l'interface REST accessible depuis l'adresse IP du pod.

> Vous pouvez configurer l'interface REST de Trident pour écouter et utiliser l'interface 127.0.0.1 (pour IPv4) ou [::1] (pour IPv6) uniquement.

`-port <port-number>`

Spécifie le port sur lequel le serveur REST de Trident doit écouter. La valeur par défaut est 8000.

`-rest`

Active l'interface REST. Valeur true par défaut.

Kubernetes et objets Trident

Vous pouvez interagir avec Kubernetes et Trident à l'aide des API REST en lisant et en écrivant des objets de ressource. La relation entre Kubernetes et Trident, Trident et le stockage, ainsi que Kubernetes et le stockage est établie avec plusieurs objets de ressources. Certains de ces objets sont gérés par Kubernetes et d'autres sont gérés à l'aide de Trident.

Comment les objets interagissent-ils les uns avec les autres ?

La manière la plus simple de comprendre les objets, leur rôle et leur interaction consiste à suivre une seule demande de stockage auprès d'un utilisateur Kubernetes :

1. Un utilisateur crée un `PersistentVolumeClaim` demander un nouveau `PersistentVolume` D'une taille spécifique dans un Kubernetes `StorageClass` qui a été précédemment configuré par l'administrateur.
2. Le Kubernetes `StorageClass` Identifie Trident comme mécanisme de provisionnement et inclut des paramètres indiquant à Trident comment provisionner un volume pour la classe demandée.

3. Trident s'occupe par lui-même `StorageClass` avec 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 back-end correspondant et crée deux objets : un `PersistentVolume` Dans Kubernetes qui indique à Kubernetes comment rechercher, monter et traiter le volume, et à un volume dans Trident qui conserve la relation entre le système `PersistentVolume` et le stockage réel.
5. Kubernetes lie le `PersistentVolumeClaim` vers le nouveau `PersistentVolume`. Des modules qui incluent `PersistentVolumeClaim` Montez le volume persistant sur n'importe quel hôte sur lequel il s'exécute.
6. Un utilisateur crée un `VolumeSnapshot` D'un volume persistant existant, à l'aide d'un `VolumeSnapshotClass` Ce que nous pointe vers Trident.
7. Trident identifie le volume associé à la demande de volume persistant et crée un snapshot du volume sur son back-end. Elle crée également un `VolumeSnapshotContent` Cela indique à Kubernetes comment identifier le Snapshot.
8. Un utilisateur peut créer un `PersistentVolumeClaim` à l'aide de `VolumeSnapshot` en tant que source.
9. Trident identifie le snapshot requis et effectue les mêmes étapes que lors de la création d'un `PersistentVolume` et a `Volume`.

Pour en savoir plus sur les objets Kubernetes, nous vous recommandons vivement de lire le "Volumes persistants" Section de la documentation Kubernetes.

Kubernetes `PersistentVolumeClaim` objets

Un Kubernetes `PersistentVolumeClaim` Cet objet est une demande de stockage faite par un utilisateur du cluster Kubernetes.

Outre la spécification standard, Trident permet aux utilisateurs de spécifier les annotations spécifiques au volume suivantes s'ils veulent remplacer les valeurs par défaut que vous définissez dans la configuration back-end :

Annotation	Option de volume	Pilotes pris en charge
trident.netapp.io/fileSystem	Système de fichiers	ontap-san, solidfire-san, ontap-san-economy
trident.netapp.io/cloneFromPVC	Volume cloneSourceVolume	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, gcp-cvs ontap-san-économie
trident.netapp.io/splitOnClone	SplitOnClone	ontap-nas, ontap-san
trident.netapp.io/protocol	protocole	toutes
trident.netapp.io/exportPolicy	ExportPolicy	ontap-nas, économie ontap-nas, ontap-nas-flexgroup
trident.netapp.io/snapshotPolicy	Politique de snapshots	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san
trident.netapp.io/snapshotReserve	Réserve de snapshots	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs

Annotation	Option de volume	Pilotes pris en charge
trident.netapp.io/snapshotDirectory	Répertoire de snapshots	ontap-nas, économie ontap-nas, ontap-nas-flexgroup
trident.netapp.io/unixPermissions	Autorisations unix	ontap-nas, économie ontap-nas, ontap-nas-flexgroup
trident.netapp.io/blockSize	Taille de bloc	solidfire-san

Si le volume persistant créé est de `Delete` Lors de la récupération de la règle, Trident supprime le volume persistant et le volume de sauvegarde lorsque le volume persistant est libéré (c'est-à-dire lors de la suppression de la demande de volume persistant). En cas d'échec de l'action de suppression, Trident marque le volume persistant comme tel et tente régulièrement l'opération jusqu'à ce qu'il réussisse ou que le volume persistant soit supprimé manuellement. Si le PV utilise le `Retain` La règle, Trident l'ignore et suppose que l'administrateur l'nettoie depuis Kubernetes et le back-end, permettant ainsi de sauvegarder ou d'inspecter le volume avant sa suppression. Notez que la suppression du volume persistant n'entraîne pas la suppression du volume de sauvegarde par Trident. Vous devez le supprimer à l'aide de l'API REST (`tridentctl`).

Trident prend en charge la création de copies Snapshot de volumes à l'aide de la spécification CSI : vous pouvez créer un Snapshot de volume et l'utiliser comme source de données pour cloner des demandes de volume existantes. Ainsi, des copies instantanées de volumes persistants peuvent être exposées à Kubernetes sous forme de snapshots. Les snapshots peuvent ensuite être utilisés pour créer de nouveaux volumes persistants. Découvrez-en plus `On-Demand Volume Snapshots` pour voir comment cela fonctionne.

Trident fournit également le système `cloneFromPVC` et `splitOnClone` annotations pour la création de clones. Vous pouvez utiliser ces annotations pour cloner une demande de volume persistant sans avoir à utiliser l'implémentation CSI.

Voici un exemple : si un utilisateur a déjà un volume persistant appelé `mysql`, L'utilisateur peut créer un nouveau PVC appelé `mysqlclone` en utilisant l'annotation, par exemple `trident.netapp.io/cloneFromPVC: mysql`. Avec ce jeu d'annotations, Trident clone le volume correspondant à la demande de volume mysql au lieu de provisionner un volume entièrement.

Prenez en compte les points suivants :

- Nous vous recommandons de cloner un volume inactif.
- Un volume persistant et son clone doivent se trouver dans le même namespace Kubernetes et avoir la même classe de stockage.
- Avec le `ontap-nas` et `ontap-san` Pilotes, il peut être souhaitable de définir l'annotation PVC `trident.netapp.io/splitOnClone` en conjonction avec `trident.netapp.io/cloneFromPVC`. Avec `trident.netapp.io/splitOnClone` réglez sur `true`, Trident divise le volume cloné du volume parent et, par conséquent, découplant complètement le cycle de vie du volume cloné de sa parent, au détriment de la perte de l'efficacité du stockage. Pas de réglage `trident.netapp.io/splitOnClone` ou le définir sur `false` cette baisse de la consommation d'espace sur le back-end implique des frais de création des dépendances entre les volumes parent et clone de sorte que le volume parent ne puisse pas être supprimé, à moins que le clone ne soit supprimé en premier. Si le fractionnement du clone s'avère judicieux, il s'agit de cloner un volume de base de données vide où l'on peut attendre du volume et de son clone pour diverger considérablement, et ne bénéficier pas des fonctionnalités d'efficacité du stockage offertes par ONTAP.

Le `sample-input` Le répertoire contient des exemples de définitions de volume persistant à utiliser avec Trident. Reportez-vous à la section Pour obtenir une description complète des paramètres et des paramètres

associés aux volumes Trident.

Kubernetes `PersistentVolume` objets

Un Kubernetes `PersistentVolume` Cet objet représente un élément de stockage mis à disposition du cluster Kubernetes. Il dispose d'un cycle de vie indépendant du pod qui l'utilise.

> Création de Trident `PersistentVolume` Les objets et les enregistre automatiquement avec le cluster Kubernetes en fonction des volumes qu'il provisionne. Vous n'êtes pas censé les gérer vous-même.

Lorsque vous créez une demande de volume persistant faisant référence à une configuration Trident `StorageClass`, Trident provisionne un nouveau volume en utilisant la classe de stockage correspondante et enregistre un nouveau volume persistant pour ce volume. Lors de la configuration du volume provisionné et du volume persistant correspondant, Trident respecte les règles suivantes :

- Trident génère un nom de volume persistant pour Kubernetes et un nom interne utilisé pour le provisionnement du stockage. Dans les deux cas, il garantit que les noms sont uniques dans leur périmètre.
- La taille du volume correspond le plus possible à la taille demandée dans le PVC, bien qu'elle puisse être arrondie à la quantité la plus proche, selon la plate-forme.

Kubernetes `StorageClass` objets

Kubernetes `StorageClass` les objets sont spécifiés par le nom dans `PersistentVolumeClaims` provisionner le stockage avec un ensemble de propriétés. La classe de stockage elle-même identifie le mécanisme de provisionnement à utiliser et définit cet ensemble de propriétés, comme le mécanisme de provisionnement le comprend.

Il s'agit de l'un des deux objets de base qui doivent être créés et gérés par l'administrateur. L'autre est l'objet back-end Trident.

Un Kubernetes `StorageClass` Voici quelques aspects d'un objet qui utilise Trident :

```
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 des volumes pour la classe.

Les paramètres de classe de stockage sont les suivants :

Attribut	Type	Obligatoire	Description
attributs	chaîne map[string]	non	Voir la section attributs ci-dessous
StoragePools	Mapper[string]StringList	non	Mappage des noms backend avec les listes de pools de stockage dans
Des médutiquesde stockage	Mapper[string]StringList	non	Mappage des noms backend avec les listes de pools de stockage dans
Exclus du stockagePools	Mapper[string]StringList	non	Mappage des noms backend avec les listes de pools de stockage dans

Les attributs de stockage et leurs valeurs possibles peuvent être classés en attributs de sélection des pools 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 les volumes d'un type donné.

Attribut	Type	Valeurs	Offre	Demande	Pris en charge par
support[1]	chaîne	hdd, hybride, ssd	Le pool contient des supports de ce type ; hybride signifie les deux	Type de support spécifié	ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san, solidfire-san
Type de provisionnement	chaîne	fin, épais	Le pool prend en charge cette méthode de provisionnement	Méthode de provisionnement spécifiée	thick : tous les systèmes ONTAP ; thin : tous les systèmes ONTAP et solidfire-san
Type de dos	chaîne	ontap-nas, économie ontap-nas, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Le pool appartient à ce type de système back-end	Backend spécifié	Tous les conducteurs

Attribut	Type	Valeurs	Offre	Demande	Pris en charge par
snapshots	bool	vrai, faux	Le pool prend en charge les volumes dotés de snapshots	Volume sur lequel les snapshots sont activés	ontap-nas, ontap-san, solidfire-san, gcp-cvs
clones	bool	vrai, faux	Le pool prend en charge les volumes de clonage	Volume sur lequel les clones sont activés	ontap-nas, ontap-san, solidfire-san, gcp-cvs
le cryptage	bool	vrai, faux	Le pool prend en charge les volumes chiffrés	Volume avec chiffrement activé	ontap-nas, économie ontap-nas, ontap-nas-flexgroups, ontap-san
D'IOPS	int	entier positif	Le pool est en mesure de garantir l'IOPS dans cette plage	Volume garanti ces IOPS	solidfire-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, la demande d'un provisionnement lourd entraîne un volume approvisionné. Un pool de stockage Element utilise ses IOPS minimales et maximales pour définir des valeurs de QoS plutôt que la valeur demandée. Dans ce cas, la valeur demandée est utilisée uniquement pour sélectionner le pool de stockage.

Idéalement, vous pouvez l'utiliser `attributes` modélisez les qualités de stockage dont vous avez besoin pour répondre à vos besoins. Trident détecte et sélectionne automatiquement les pools de stockage qui correspondent à *All* du `attributes` que vous spécifiez.

Si vous vous trouvez incapable d'utiliser `attributes` pour sélectionner automatiquement les pools appropriés pour une classe, vous pouvez utiliser le `storagePools` et `additionalStoragePools` paramètres pour affiner davantage les pools ou même pour sélectionner un ensemble spécifique de pools.

Vous pouvez utiliser le `storagePools` paramètre pour restreindre davantage l'ensemble de pools correspondant à n'importe quel spécifié `attributes`. En d'autres termes, Trident utilise l'intersection des pools identifiés par le `attributes` et `storagePools` paramètres de provisionnement. Vous pouvez utiliser les paramètres seuls ou les deux ensemble.

Vous pouvez utiliser le `additionalStoragePools` Paramètre pour étendre l'ensemble de pools utilisés par Trident pour le provisionnement, quels que soient les pools sélectionnés par le système `attributes` et `storagePools` paramètres.

Vous pouvez utiliser le `excludeStoragePools` Paramètre pour filtrer l'ensemble des pools utilisés par Trident pour le provisionnement. L'utilisation de ce paramètre supprime tous les pools correspondant.

Dans le `storagePools` et `additionalStoragePools` paramètres, chaque entrée prend la forme `<backend>:<storagePoolList>`, où `<storagePoolList>` est une liste de pools de stockage séparés par des virgules pour le back-end spécifié. Par exemple, une valeur pour `additionalStoragePools` peut-être cela `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`. Ces listes

acceptent les valeurs regex tant pour le back-end que pour les valeurs de liste. Vous pouvez utiliser `tridentctl get backend` pour obtenir la liste des systèmes back-end et leurs pools.

Attributs Kubernetes

Ces attributs n'ont aucun impact sur la sélection des pools de stockage/systèmes back-end par Trident lors du provisionnement dynamique. En effet, ces attributs fournissent simplement les paramètres pris en charge par les volumes persistants de Kubernetes. Les nœuds worker sont responsables des opérations de création de système de fichiers et peuvent nécessiter des utilitaires de système de fichiers, tels que xfsprogs.

Attribut	Type	Valeurs	Description	Facteurs pertinents	Version Kubernetes
Fstype	chaîne	ext4, ext3, xfs, etc	Type de système de fichiers pour les volumes en mode bloc	solidfire-san, ontap-nas, ontap-nas-économie, ontap-nas-flexgroup, ontap-san, ontap-san-économie	Tout
Volumeallowexpansion	booléen	vrai, faux	Activez ou désactivez la prise en charge pour augmenter la taille de la demande de volume persistant	ontap-nas, économie ontap-nas, ontap-nas-flexgroup, ontap-san, ontap-san-économie, solidfire-san, gcp-cvs, azure-netapp-files	1.11+
Volume Bindingmode	chaîne	Immédiat, WaitForFirstConsumer	Sélectionnez le moment où la liaison des volumes et le provisionnement dynamique se produisent	Tout	1.19 - 1.26

- Le `fsType` Paramètre permet de contrôler le type de système de fichiers souhaité pour les LUN SAN. Kubernetes utilise également la présence de `fsType` dans une classe de stockage pour indiquer qu'un système de fichiers existe. Vous pouvez contrôler la propriété de volume à l'aide du `fsGroup` contexte de sécurité d'un pod uniquement si `fsType` est défini. Reportez-vous à la section "Kubernetes : configurez un contexte de sécurité pour un pod ou un conteneur" pour une vue d'ensemble de la définition de la propriété de volume à l'aide de l' `fsGroup` contexte. Kubernetes applique le `fsGroup` valeur uniquement si :
  - `fsType` est défini dans la classe de stockage.
  - Le mode d'accès PVC est RWO.

  Pour les pilotes de stockage NFS, un système de fichiers existe déjà dans le cadre de l'exportation NFS. Pour l'utilisation `fsGroup` la classe de stockage doit toujours spécifier un `fsType`. Vous pouvez le définir sur `nfs` ou toute valeur non nulle.
- Reportez-vous à la section "Développement des volumes" pour plus de détails sur l'extension du volume.
- Le bundle d'installation Trident propose plusieurs exemples de définitions de classes de stockage à utiliser avec Trident dans `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 similaires à `StorageClasses`. Ils aident à définir plusieurs classes de stockage. Ils sont référencés par les snapshots de volume pour associer le snapshot à la classe d'instantanés requise. Chaque snapshot de volume est associé à une classe de snapshot de volume unique.

A `VolumeSnapshotClass` doit être défini par un administrateur pour créer des instantanés. Une classe de snapshots 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` Spécifie à Kubernetes que demande des snapshots de volume du `csi-snapclass` Ces classes sont gérées par Trident. Le `deletionPolicy` spécifie l'action à effectuer lorsqu'un instantané doit être supprimé. Quand `deletionPolicy` est défini sur `Delete`, les objets de snapshot de volume ainsi que le snapshot sous-jacent du cluster de stockage sont supprimés lorsqu'un snapshot est supprimé. Vous pouvez également le régler sur `Retain` signifie que `VolumeSnapshotContent` et le snapshot physique sont conservés.

Kubernetes `VolumeSnapshot` objets

Un Kubernetes `VolumeSnapshot` objet est une demande de création d'un snapshot de volume. Tout comme

un volume persistant représente une demande de copie Snapshot d'un volume effectuée par un utilisateur, une copie Snapshot de volume est une demande de création d'un snapshot d'une demande de volume persistant existante.

Lorsqu'une requête de snapshot de volume est fournie, Trident gère automatiquement la création du snapshot du volume sur le back-end et expose le snapshot en créant un seul snapshot `VolumeSnapshotContent` objet. Vous pouvez créer des instantanés à partir de ESV existantes et les utiliser comme source de données lors de la création de nouveaux ESV.

> Le silecyle d'un VolumeSnapshot est indépendant de la demande de volume persistant source : un snapshot persiste même après la suppression de la demande de volume persistant source. Lors de la suppression d'un volume persistant qui possède des snapshots associés, Trident marque le volume de sauvegarde de ce volume persistant dans un état **Suppression**, mais ne le supprime pas complètement. Le volume est supprimé lorsque tous les snapshots associés sont supprimés.

Kubernetes `VolumeSnapshotContent` objets

Un Kubernetes `VolumeSnapshotContent` objet représente un snapshot pris à partir d'un volume déjà provisionné. Il est similaire à un `PersistentVolume` la désignation rr signifie un snapshot provisionné sur le cluster de stockage. Similaire à `PersistentVolumeClaim` et `PersistentVolume` lors de la création d'un snapshot, le `VolumeSnapshotContent` l'objet conserve un mappage un-à-un avec le `VolumeSnapshot` objet, qui avait demandé la création de snapshot.

Le `VolumeSnapshotContent` l'objet contient des détails qui identifient de manière unique le snapshot, comme le `snapshotHandle`. C'est ça `snapshotHandle` Est une combinaison unique du nom du PV et du nom du `VolumeSnapshotContent` objet.

Lorsqu'une requête de snapshot est fournie, Trident crée le snapshot sur le back-end. Une fois le snapshot créé, Trident configure un `VolumeSnapshotContent` Objet et donc expose le snapshot à l'API Kubernetes.

> En général, il n'est pas nécessaire de gérer le `VolumeSnapshotContent` objet. Une exception à cette règle est le moment où vous le souhaitez "importer un instantané de volume" Créé en dehors d'Astra Trident.

Kubernetes `CustomResourceDefinition` objets

Les ressources personnalisées Kubernetes sont des terminaux 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 le stockage d'une collection d'objets. Vous pouvez obtenir ces définitions de ressources en cours d'exécution `kubectl get crds`.

Les définitions de ressources personnalisées (CRD) et les métadonnées d'objet associées sont stockées sur le magasin de métadonnées Kubernetes. Ce qui évite d'avoir recours à un magasin séparé pour Trident.

Astra Trident utilise `CustomResourceDefinition` Objets pour préserver l'identité des objets Trident, tels que les systèmes back-end Trident, les classes de stockage Trident et les volumes Trident. Ces objets sont gérés par Trident. En outre, la structure d'instantané de volume CSI introduit quelques CRD nécessaires pour définir des instantanés de volume.

Les CRDS sont une construction Kubernetes. Les objets des ressources définies ci-dessus sont créés par Trident. À titre d'exemple simple, lorsqu'un système back-end est créé à l'aide de `tridentctl`, un

correspondant `tridentbackends` L'objet CRD est créé pour la consommation par Kubernetes.

Voici quelques points à garder à l'esprit sur les CRD de Trident :

- Lorsque Trident est installé, un ensemble de CRD est créé et peut être utilisé comme tout autre type de ressource.
- Lors de la désinstallation de Trident à l'aide de `tridentctl uninstall` Les pods Trident sont supprimés, mais les CRD créés ne sont pas nettoyés. Reportez-vous à la section "Désinstaller Trident" Afin de comprendre comment Trident peut être entièrement supprimé et reconfiguré de zéro.

Astra Trident `StorageClass` objets

Trident crée des classes de stockage correspondantes pour Kubernetes `StorageClass` objets spécifiés `csi.trident.netapp.io` dans leur champ de provisionnement. Le nom de classe de stockage correspond à celui du système Kubernetes `StorageClass` objet qu'il représente.

> Avec Kubernetes, ces objets sont créés automatiquement lorsqu'un système Kubernetes est activé `StorageClass` Qui utilise Trident comme mécanisme de provisionnement est enregistré.

Les classes de stockage comprennent un ensemble d'exigences pour les volumes. Trident mappe ces exigences avec les attributs présents dans chaque pool de stockage. S'ils correspondent, ce pool de stockage est une cible valide pour le provisionnement des volumes qui utilisent cette classe de stockage.

Vous pouvez créer des configurations de classes de stockage afin de définir directement des classes de stockage à l'aide de l'API REST. Toutefois, dans le cas des déploiements Kubernetes, nous attendons d'eux qu'ils soient créés lors de l'enregistrement du nouveau Kubernetes `StorageClass` objets.

Objets back-end Astra Trident

Les systèmes back-end représentent les fournisseurs de stockage au-dessus desquels Trident provisionne des volumes. Une instance Trident unique peut gérer un nombre illimité de systèmes back-end.

> Il s'agit de l'un des deux types d'objet que vous créez et gérez vous-même. L'autre est le Kubernetes `StorageClass` objet.

Pour plus d'informations sur la construction de ces objets, voir "configuration des systèmes back-end".

Astra Trident `StoragePool` objets

Les pools de stockage représentent les emplacements distincts disponibles pour le provisionnement sur chaque système back-end. Pour ONTAP, ces derniers correspondent à des agrégats dans des SVM. Pour NetApp HCI/SolidFire, ils correspondent aux bandes QoS spécifiées par l'administrateur. Pour Cloud Volumes Service, ces régions correspondent à des régions du fournisseur cloud. Chaque pool de stockage dispose d'un ensemble d'attributs de stockage distincts, qui définissent ses caractéristiques de performances et ses caractéristiques de protection des données.

Contrairement aux autres objets ici, les candidats au pool de stockage sont toujours découverts et gérés automatiquement.

Astra Trident `Volume` objets

Les volumes sont l'unité de provisionnement de base, comprenant les terminaux back-end, tels que les partages NFS et les LUN iSCSI. Dans Kubernetes, ces derniers correspondent directement à `PersistentVolumes`. Lorsque vous créez un volume, assurez-vous qu'il possède une classe de stockage, qui détermine l'emplacement de provisionnement de ce volume, ainsi que sa taille.

- Dans Kubernetes, ces objets sont gérés automatiquement. Vous pouvez les afficher pour voir le provisionnement Trident.
- Lors de la suppression d'un volume persistant avec des snapshots associés, le volume Trident correspondant est mis à jour avec un état **Suppression**. Pour que le volume Trident soit supprimé, vous devez supprimer les snapshots du volume.

Une configuration de volume définit les propriétés qu'un volume provisionné doit avoir.

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
Volume cloneSourceVolume	chaîne	non	ONTAP (nas, san) et SolidFire-* : nom du volume à cloner
SplitOnClone	chaîne	non	ONTAP (nas, san) : séparer le clone de son parent
Politique de snapshots	chaîne	non	ONTAP-* : stratégie d'instantané à utiliser
Réserve de snapshots	chaîne	non	ONTAP-* : pourcentage de volume réservé pour les snapshots
ExportPolicy	chaîne	non	ontap-nas* : export policy à utiliser

Attribut	Type	Obligatoire	Description
Répertoire de snapshots	bool	non	ontap-nas* : indique si le répertoire des snapshots est visible
Autorisations unix	chaîne	non	ontap-nas* : autorisations UNIX initiales
Taille de bloc	chaîne	non	SolidFire-*: Taille de bloc/secteur
Système de fichiers	chaîne	non	Type de système de fichiers

Génération de Trident `internalName` lors de la création du volume. Il s'agit de deux étapes. Tout d'abord, il prétermine le préfixe de stockage (soit le préfixe par défaut `trident` ou le préfixe de la configuration back-end) au nom du volume, ce qui produit un nom du formulaire `<prefix>-<volume-name>`. Il procède ensuite à la désinfection du nom en remplaçant les caractères non autorisés dans le back-end. Pour les systèmes ONTAP back-end, il remplace les tirets par des traits de soulignement (ainsi, le nom interne devient `<prefix>_<volume-name>`). Pour les systèmes back-end Element, il remplace les tirets de traits de soulignement.

Vous pouvez utiliser les configurations de volumes pour provisionner directement des volumes à l'aide de l'API REST, mais dans les déploiements Kubernetes, la plupart des utilisateurs utilisent le protocole Kubernetes standard `PersistentVolumeClaim` méthode. Trident crée automatiquement cet objet volume dans le cadre du provisionnement.

Astra Trident `Snapshot` objets

Les snapshots sont une copie de volumes à un point dans le temps, qui peut être utilisée pour provisionner de nouveaux volumes ou restaurer l'état de ces volumes. Dans Kubernetes, ces derniers correspondent directement à `VolumeSnapshotContent` objets. Chaque snapshot est associé à un volume, qui est la source des données du snapshot.

Chacun `Snapshot` l'objet inclut les propriétés répertoriées ci-dessous :

Attribut	Type	Obligatoire	Description
version	Chaîne	Oui.	Version de l'API Trident (« 1 »)
nom	Chaîne	Oui.	Nom de l'objet snapshot Trident
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 le snapshot est créé
Volume Nom interne	Chaîne	Oui.	Nom de l'objet volume Trident associé sur le système de stockage

Dans Kubernetes, ces objets sont gérés automatiquement. Vous pouvez les afficher pour voir le provisionnement Trident.

Lorsqu'un Kubernetes `VolumeSnapshot` La requête d'objet est créée, Trident crée un objet de snapshot sur le système de stockage secondaire. Le `internalName` cet objet de snapshot est généré en combinant le préfixe `snapshot-` avec le `UID` du `VolumeSnapshot` objet (par exemple, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` et `volumeInternalName` sont renseignées en obtenant les détails du volume de sauvegarde.

Astra Trident `ResourceQuota` objet

La déamOnset Trident utilise un `system-node-critical` Classe de priorité—la classe de priorité la plus élevée disponible dans Kubernetes—pour s'assurer que Astra Trident peut identifier et nettoyer les volumes lors de l'arrêt normal des nœuds. Il permet également aux pods Trident de s'assurer que ces derniers anticipent les charges de travail dans les clusters où la pression des ressources est élevée.

Astra Trident utilise un pour atteindre ces objectifs `ResourceQuota` Objet garantissant la satisfaction d'une classe de priorité « système-nœud-critique » sur le jeu de démonéset Trident. Avant de déployer et de diaboset, Astra 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 configurez le `ResourceQuota` Objet utilisant le graphique Helm.

Voici un exemple de `Resourcequota"objet hiérarchisant le demonset 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, voir "Kubernetes : quotas de ressources".

Nettoyez `ResourceQuota` si l'installation échoue

Dans les rares cas où l'installation échoue après le `ResourceQuota` l'objet est créé, commencez par essayer "désinstallation" puis réinstaller.

Si cela ne fonctionne pas, supprimez manuellement le `ResourceQuota` objet.

Déposer `ResourceQuota`

Si vous préférez contrôler l'allocation de vos ressources, vous pouvez supprimer Astra Trident `ResourceQuota` objet utilisant la commande :

```
kubectl delete quota trident-csi -n trident
```

Normes de sécurité de pod (PSS) et contraintes de contexte de sécurité (SCC)

Les normes de sécurité de Kubernetes Pod (PSS) et les règles de sécurité de Pod (PSP) définissent des niveaux d'autorisation et limitent le comportement des pods. OpenShift Security Context Constraints (SCC) définit de façon similaire les restrictions de pod spécifiques à OpenShift Kubernetes Engine. Pour offrir cette personnalisation, Astra Trident autorise certaines autorisations lors de l'installation. Les sections suivantes décrivent en détail les autorisations définies par Astra Trident.

> PSS remplace les politiques de sécurité Pod (PSP). La PSP est obsolète dans Kubernetes v1.21 et elle sera supprimée dans la version 1.25. Pour plus d'informations, reportez-vous à la section "Kubernetes : sécurité".

Contexte de sécurité Kubernetes requis et champs associés

Autorisations	Description
Privilégié	CSI nécessite que les points de montage soient bidirectionnels, ce qui signifie que le pod de nœud Trident doit exécuter un conteneur privilégié. Pour plus d'informations, reportez-vous à la section "Kubernetes : propagation du montage".
Mise en réseau d'hôtes	Requis pour le démon iSCSI. `iscsiadm` Gère les montages iSCSI et utilise la mise en réseau hôte pour communiquer avec le démon iSCSI.
IPC hôte	Le NFS utilise la communication interprocess (IPC) pour communiquer avec le NFSD.
PID hôte	Nécessaire pour démarrer `rpc-statd` Pour NFS. Astra Trident interroge les processus hôte pour déterminer si `rpc-statd` Est en cours d'exécution avant le montage de volumes NFS.

Capacités	Le `SYS_ADMIN` la fonctionnalité fait partie des fonctionnalités par défaut pour les conteneurs privilégiés. Par exemple, Docker définit ces fonctionnalités pour les conteneurs privilégiés :
	`CapPrm: 0000003ffffffff`
	`CapEff: 0000003ffffffff`

Autorisations	Description
Seccomp	Le profil Seccomp est toujours « confiné » dans des conteneurs privilégiés. Par conséquent, il ne peut pas être activé sur Astra Trident.
SELinux	Sur OpenShift, des conteneurs privilégiés sont exécutés dans le `spc_t` (« conteneur super privilégié ») domaine, et les conteneurs non privilégiés sont exécutés dans le `container_t` domaine. Marche `containerd`, avec `container-selinux` tous les conteneurs sont installés dans le `spc_t` Domaine, qui désactive réellement SELinux. Astra Trident n'ajoute donc pas `seLinuxOptions` aux conteneurs.
DAC	Les conteneurs privilégiés doivent être exécutés en tant que root. Les conteneurs non privilégiés s'exécutent comme root pour accéder aux sockets unix requis par CSI.

Normes de sécurité du pod (PSS)

Étiquette	Description	Valeur par défaut

`pod-security.kubernetes.io/enforce`
 	
`pod-security.kubernetes.io/enforce-version`
	Permet au contrôleur et aux nœuds Trident d'être admis dans le namespace d'installation. Ne modifiez pas le libellé de l'espace de noms.	`enforce: privileged`
		
		`enforce-version: <version of the current cluster or highest version of PSS tested.>`

> La modification des étiquettes de l'espace de noms peut entraîner l'absence de planification des modules, un "erreur de création: …" ou un "avertissement: trident-csi-…". Si cela se produit, vérifiez si le libellé de l'espace de noms pour `privileged` a été modifié. Si c'est le cas, réinstallez Trident.

Politiques de sécurité des pods (PSP)

Champ	Description	Valeur par défaut
`allowPrivilegeEscalation`	Les conteneurs privilégiés doivent autoriser l'escalade des privilèges.	`true`
`allowedCSIDrivers`	Trident n'utilise pas les volumes éphémères CSI en ligne.	Vide
`allowedCapabilities`	Les conteneurs Trident non privilégiés ne nécessitent pas de fonctionnalités supérieures à celles des ensembles par défaut et les conteneurs privilégiés se voient accorder toutes les capacités possibles.	Vide

Champ	Description	Valeur par défaut
`allowedFlexVolumes`	Trident n'utilise pas de système "Pilote FlexVolume", par conséquent, ils ne sont pas inclus dans la liste des volumes autorisés.	Vide
`allowedHostPaths`	Le pod des nœuds Trident monte le système de fichiers racine du nœud, ce qui ne permet donc pas de définir cette liste.	Vide
`allowedProcMountTypes`	Trident n'utilise aucun `ProcMountTypes`.	Vide
`allowedUnsafeSysctls`	Trident n'exige aucun niveau de sécurité `sysctls`.	Vide
`defaultAddCapabilities`	Aucune fonctionnalité n'est requise pour être ajoutée aux conteneurs privilégiés.	Vide
`defaultAllowPrivilegeEscalation`	L'autorisation de réaffectation des privilèges est gérée dans chaque pod Trident.	`false`
`forbiddenSysctls`	Non `sysctls` sont autorisés.	Vide
`fsGroup`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`
`hostIPC`	Le montage des volumes NFS requiert la communication du IPC hôte avec `nfsd`	`true`
`hostNetwork`	Iscsiadm nécessite que le réseau hôte communique avec le démon iSCSI.	`true`
`hostPID`	Le PID hôte est requis pour vérifier si `rpc-statd` est en cours d'exécution sur le nœud.	`true`
`hostPorts`	Trident n'utilise aucun port hôte.	Vide
`privileged`	Les pods de nœuds Trident doivent exécuter un conteneur privilégié pour monter des volumes.	`true`
`readOnlyRootFilesystem`	Les pods de nœuds Trident doivent écrire dans le système de fichiers de nœuds.	`false`
`requiredDropCapabilities`	Les pods de nœuds Trident exécutent un conteneur privilégié et ne peuvent pas supprimer de fonctionnalités.	`none`
`runAsGroup`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`

Champ	Description	Valeur par défaut
`runAsUser`	Les conteneurs Trident s'exécutent en tant que root.	`runAsAny`
`runtimeClass`	Trident n'utilise pas `RuntimeClasses`.	Vide
`seLinux`	Trident n'est pas défini `seLinuxOptions` Car il existe actuellement des différences dans le mode de gestion des conteneurs et de distribution Kubernetes de SELinux.	Vide
`supplementalGroups`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`
`volumes`	Les pods Trident requièrent ces plug-ins de volume.	`hostPath, projected, emptyDir`

Contraintes de contexte de sécurité (SCC)

Étiquettes	Description	Valeur par défaut
`allowHostDirVolumePlugin`	Les pods des nœuds Trident montent le système de fichiers racine du nœud.	`true`
`allowHostIPC`	Le montage des volumes NFS requiert la communication du IPC hôte avec `nfsd`.	`true`
`allowHostNetwork`	Iscsiadm nécessite que le réseau hôte communique avec le démon iSCSI.	`true`
`allowHostPID`	Le PID hôte est requis pour vérifier si `rpc-statd` est en cours d'exécution sur le nœud.	`true`
`allowHostPorts`	Trident n'utilise aucun port hôte.	`false`
`allowPrivilegeEscalation`	Les conteneurs privilégiés doivent autoriser l'escalade des privilèges.	`true`
`allowPrivilegedContainer`	Les pods de nœuds Trident doivent exécuter un conteneur privilégié pour monter des volumes.	`true`
`allowedUnsafeSysctls`	Trident n'exige aucun niveau de sécurité `sysctls`.	`none`

Étiquettes	Description	Valeur par défaut
`allowedCapabilities`	Les conteneurs Trident non privilégiés ne nécessitent pas de fonctionnalités supérieures à celles des ensembles par défaut et les conteneurs privilégiés se voient accorder toutes les capacités possibles.	Vide
`defaultAddCapabilities`	Aucune fonctionnalité n'est requise pour être ajoutée aux conteneurs privilégiés.	Vide
`fsGroup`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`
`groups`	Ce SCC est spécifique à Trident et lié à son utilisateur.	Vide
`readOnlyRootFilesystem`	Les pods de nœuds Trident doivent écrire dans le système de fichiers de nœuds.	`false`
`requiredDropCapabilities`	Les pods de nœuds Trident exécutent un conteneur privilégié et ne peuvent pas supprimer de fonctionnalités.	`none`
`runAsUser`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`
`seLinuxContext`	Trident n'est pas défini `seLinuxOptions` Car il existe actuellement des différences dans le mode de gestion des conteneurs et de distribution Kubernetes de SELinux.	Vide
`seccompProfiles`	Les conteneurs privilégiés s'exécutent toujours « sans limite ».	Vide
`supplementalGroups`	Les conteneurs Trident s'exécutent en tant que root.	`RunAsAny`
`users`	Une entrée est fournie pour lier ce SCC à l'utilisateur Trident dans l'espace de noms Trident.	s/o
`volumes`	Les pods Trident requièrent ces plug-ins de volume.	`hostPath, downwardAPI, projected, emptyDir`

Mentions légales

Les mentions légales donnent accès aux déclarations de copyright, aux marques, aux brevets, etc.

Droits d'auteur

"https://www.netapp.com/company/legal/copyright/"

Marques déposées

NetApp, le logo NETAPP et les marques mentionnées sur la page des marques commerciales 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/"

Brevets

Vous trouverez une liste actuelle des brevets appartenant à NetApp 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/"

Source ouverte

Vous pouvez consulter les informations de copyright et licences tierces utilisées dans le logiciel NetApp pour Astra Trident dans le fichier d'avis correspondant à chaque version, à l'adresse https://github.com/NetApp/trident/.