Gérer et surveiller Trident

Trident

NetApp
September 26, 2025

Sommaire

Gérer et surveiller Trident

Mettez à niveau Trident

Mettez à niveau Trident

À compter de la version 24.02, Trident suit une cadence de quatre mois pour publier trois versions majeures chaque année civile. 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 de Trident.

Considérations avant la mise à niveau

Lorsque vous effectuez une mise à niveau vers la dernière version de Trident, tenez compte des points suivants :

- Il ne doit y avoir qu'une seule instance Trident installée sur tous les namespaces d'un cluster Kubernetes donné.
- Trident 23.07 et versions ultérieures requièrent des instantanés de volume v1 et ne prend plus en charge les instantanés 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` niveau de service ou `zoneredundantstandardsw` lors de la mise à niveau à partir de Trident 23.01. L'échec de la mise à jour du système `serviceLevel` dans le back-end peut entraîner l'échec des volumes. Voir "Exemples de type de service CVS" pour plus de détails.
- Lors de la mise à niveau, il est important que vous fournissez `parameter.fsType` dans `StorageClasses` utilisé par Trident. Vous pouvez supprimer et recréer des données `StorageClasses` sans interrompre les volumes préexistants.
  - Il s'agit d'une **exigence** pour l'application "contextes de sécurité" pour les volumes SAN.
  - Le répertoire sample input contient des exemples, tels que https://github.com/NetApp/Trident/blob/master/Trident-installer/sample-input/Storage-class-samples/Storage-class-Basic.yaml.templ ^] et link:https://github.com/NetApp/Trident/blob/master/Trident-input-default-class-samples-Storage-sample-than-class-than-default-[`storage-class-bronze-default.yaml`class[`storage-class-basic.yaml.templ` ^.
  - Pour plus d'informations, reportez-vous "Problèmes connus"à .

Étape 1 : sélectionnez une version

Les versions Trident suivent une convention de dénomination basée sur la date `YY.MM`, où « YY » correspond aux deux derniers chiffres de l'année et « MM » au mois. Les versions de points suivent une `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 23.04 (ou de toute version 23.04 points) vers la version 24.06.
- 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 "version antérieure" niveau de pour effectuer la mise à niveau vers la version la plus récente qui s'adapte à la fenêtre à quatre versions. Par exemple, si vous exécutez 22.01 et que vous souhaitez effectuer une mise à niveau vers 24.06 :

a. Première mise à niveau de 22.07 à 23.04.

b. Puis passez de 23.04 à 24.06.

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

1. Utilisez `kubectl get pods -n trident` pour examiner les modules.
   - S'il n'y a pas de module opérateur, Trident a été installé à l'aide de `tridentctl`.
   - S'il existe un module opérateur, Trident a été installé à l'aide de l'opérateur Trident soit manuellement, soit à l'aide de l'assistant.
2. S'il y a un module opérateur, utilisez `kubectl describe torc` pour déterminer si Trident a été installé à l'aide de l'assistant.
   - S'il y a une étiquette Helm, Trident a été installé à l'aide de Helm.
   - S'il n'y a pas d'étiquette Helm, Trident a été installé manuellement à l'aide de l'opérateur Trident.

Étape 3 : sélectionnez une méthode de mise à niveau

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

- "Mise à niveau à l'aide de l'opérateur Trident"

> Nous vous conseillons de passer en revue "Comprendre le workflow de mise à niveau de l'opérateur" avant de procéder à 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 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'une des nombreuses "Avantages de l'utilisation de l'opérateur Trident" à installer et à mettre à niveau Trident est la gestion automatique des objets Trident et Kubernetes sans interrompre les volumes montés existants. De cette façon, Trident peut prendre en charge les mises à niveau sans temps d'indisponibilité, 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 plus d'informations sur l'architecture Trident sur le cluster Kubernetes, reportez-vous à "Architecture Trident"la .

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 actuellement installée de Trident (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.

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

Vous pouvez mettre à niveau Trident à l'aide de l'opérateur Trident manuellement ou à l'aide d'Helm. Vous pouvez effectuer une mise à niveau d'une installation opérateur Trident vers une autre installation opérateur Trident ou passer d'une `tridentctl` installation à une version opérateur Trident. Révision "Sélectionnez une méthode de mise à niveau" avant la mise à niveau d'une installation de l'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 de Trident 21.01 et supérieures utilisent un opérateur cluster-scoped.

Pour mettre à niveau à partir de Trident qui a été installé à l'aide de l'opérateur Namespace-scoped (versions 20.07 à 20.10), utilisez les instructions de mise à niveau pour "votre version installée" de 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, 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 de Trident :

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

2. Supprimez l'opérateur Trident qui a été utilisé pour installer l'instance 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 d' `TridentOrchestrator`attributs, vous pouvez modifier l' `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 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 Trident 24.10, exécutez la commande suivante :

```
kubectl create -f 24.10.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

Mettre à niveau une installation Helm

Vous pouvez mettre à niveau une installation Trident Helm.

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

Si vous avez déjà mis à niveau votre cluster Kubernetes de 1.24 à 1.25 sans mettre à niveau le contrôleur Trident Helm, la mise à niveau Helm échoue. Pour effectuer la mise à niveau de Helm, effectuez les étapes suivantes en tant que conditions préalables :

1. Installez le plug-in Helm-mapkubeapis à partir de https://github.com/helm/helm-mapkubeapis.
2. Effectuez une exécution à sec pour la version Trident dans l'espace de nom où Trident est installé. Cette liste répertorie les ressources qui seront nettoyées.

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

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

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

Étapes

1. Si vous "Installez Trident à l'aide de Helm - effectué", vous pouvez utiliser `helm upgrade trident netapp-trident/trident-operator --version 100.2410.0` pour effectuer une mise 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 de Trident sur "La section *Assets* sur GitHub".
   b. Utilisez `helm upgrade` la commande où reflète la version vers laquelle `trident-operator-24.10.0.tgz` vous souhaitez effectuer la mise à niveau.

```
helm upgrade <name> trident-operator-24.10.0.tgz
```

> Si vous définissez des options personnalisées lors de l'installation initiale (par exemple, spécification de registres privés, en miroir pour les images Trident et CSI), ajoutez la commande à l'aide de `--set` pour vous assurer que ces options sont incluses dans la commande de mise à niveau, sinon les valeurs seront réinitialisées `helm upgrade` à leur valeur par défaut.

2. Exécutez `helm list` pour vérifier que le graphique et la version de l'application ont tous deux été mis à niveau. Exécutez `tridentctl logs` pour consulter tous les messages de débogage.

Mise à niveau d'une `tridentctl` installation vers un opérateur Trident

Vous pouvez effectuer une mise à niveau vers la dernière version de l'opérateur Trident à partir d'une `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 de Trident.

```
# Download the release required [24.10.0]
mkdir 24.10.0
cd 24.10.0
wget
https://github.com/NetApp/trident/releases/download/v24.10.0/trident-
installer-24.10.0.tar.gz
tar -xf trident-installer-24.10.0.tar.gz
cd trident-installer
```

2. Créez le `tridentorchestrator` CRD à partir 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. Créez une `TridentOrchestrator` CR pour installer 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:                v24.10.0
```

Mise à niveau avec tridentctl

Vous pouvez facilement mettre à niveau une installation Trident existante à l'aide de `tridentctl`.

Description de la tâche

La désinstallation et la réinstallation de Trident agit comme une mise à niveau. Lorsque vous désinstallez Trident, la demande de volume persistant et le volume persistant utilisés par le déploiement Trident ne sont pas supprimés. Les volumes persistants déjà provisionnés restent disponibles pendant que Trident est hors ligne et Trident provisionne les volumes pour toutes les demandes de volume persistant créées entre la remise en ligne.

Avant de commencer

Révision "Sélectionnez une méthode de mise à niveau" avant la mise à niveau à l'aide de `tridentctl`.

Étapes

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

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

2. Réinstallez Trident. Reportez-vous à la "Installez 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 Trident à l'aide de tridentctl

Le "Pack d'installation Trident" comprend l' `tridentctl`utilitaire de ligne de commande qui permet un accès simple à Trident. Les utilisateurs Kubernetes disposant de suffisamment de Privileges peuvent l'utiliser pour installer Trident ou gérer le namespace qui contient le pod Trident.

Commandes et indicateurs globaux

Vous pouvez exécuter `tridentctl help` pour obtenir une liste des commandes disponibles pour `tridentctl` ou ajouter l' `--help`indicateur à n'importe quelle commande afin d'obtenir une liste d'options et d'indicateurs pour cette commande spécifique.

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

L'utilitaire Trident `tridentctl` prend en charge les commandes et indicateurs globaux suivants.

Commandes

`create`
Ajouter une ressource à Trident.

`delete`
Supprimez une ou plusieurs ressources de Trident.

`get`
Obtenez une ou plusieurs ressources de Trident.

`help`
Aide sur n'importe quelle commande.

`images`
Imprimez un tableau des images de conteneur dont Trident a besoin.

`import`
Importer une ressource existante dans Trident.

`install`
Installation de Trident.

`logs`
Imprimez les journaux depuis Trident.

`send`
Envoyer une ressource à partir de Trident.

`uninstall`
Désinstallez Trident.

`update`
Modifier une ressource dans Trident.

`update backend state`
Suspendre temporairement les opérations back-end.

`upgrade`
Mettre à niveau une ressource dans Trident.

`version`
Imprimez la version de Trident.

Alarmes globales

`-d, --debug`
Sortie de débogage.

`-h, --help`
Aide pour `tridentctl`.

`-k, --kubeconfig string`
Spécifiez le `KUBECONFIG` chemin d'exécution des commandes en local ou d'un cluster Kubernetes vers un autre.

> Vous pouvez également exporter la `KUBECONFIG` variable pour pointer vers un cluster Kubernetes spécifique et lancer des `tridentctl` commandes vers ce cluster.

`-n, --namespace string`
Espace de noms du déploiement 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 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 `create` la commande pour ajouter une ressource à Trident.

```
tridentctl create [option]
```

Options

`backend`: Ajouter un backend à Trident.

supprimer

Utilisez `delete` la commande pour supprimer une ou plusieurs ressources de Trident.

```
tridentctl delete [option]
```

Options

`backend`: Supprimez un ou plusieurs systèmes back-end de Trident.
`snapshot`: Supprimer un ou plusieurs instantanés de volume de Trident.
`storageclass`: Supprimer une ou plusieurs classes de stockage de Trident.

`volume`: Supprimer un ou plusieurs volumes de stockage de Trident.

obtenez

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

```
tridentctl get [option]
```

Options

`backend`: Obtenez un ou plusieurs systèmes back-end de stockage Trident.
`snapshot`: Obtenir un ou plusieurs instantanés de Trident.
`storageclass`: Obtenir une ou plusieurs classes de stockage de Trident.
`volume`: Obtenir un ou plusieurs volumes de Trident.

Alarmes

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

images

Utilisez `images` des indicateurs pour imprimer un tableau des images de conteneur dont 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

Utiliser `import volume` la commande pour importer un volume existant dans Trident.

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

Alias

`volume`, `v`

Alarmes

`-f`, `--filename string`: Chemin d'accès au fichier ESV 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 les `install` indicateurs pour installer Trident.

```
tridentctl install [flags]
```

Alarmes

`--autosupport-image string`: L'image conteneur pour la télémétrie AutoSupport (par défaut "NetApp/Trident AutoSupport:<current-version>").
`--autosupport-proxy string`: Adresse/port d'un proxy pour l'envoi de la télémétrie AutoSupport.
`--enable-node-prep`: Tentative d'installation des modules requis sur les nœuds.
`--generate-custom-yaml`: Générer des fichiers YAML sans rien installer.
`-h, --help`: Aide pour l'installation.
`--http-request-timeout`: Remplacer le délai d'expiration de la requête HTTP pour l'API REST du contrôleur Trident (1m30s par défaut).
`--image-registry string`: Adresse/port d'un registre d'images interne.
`--k8s-timeout duration`: Délai d'expiration pour toutes les opérations Kubernetes (3m0s par défaut).
`--kubelet-dir string`: Emplacement de l'hôte de l'état interne de kubelet (par défaut "/var/lib/kubelet").
`--log-format string`: Le format d'enregistrement Trident (texte, json) (par défaut "texte").
`--node-prep`: Permet à Trident de préparer les nœuds du cluster Kubernetes pour la gestion des volumes à l'aide du protocole de stockage de données spécifié. **Actuellement, `iscsi` est la seule valeur prise en charge.**
`--pv string`: le nom du PV hérité utilisé par Trident, s'assure que cela n'existe pas (par défaut "Trident").
`--pvc string`: Le nom du PVC existant utilisé par Trident, s'assure qu'il n'existe pas (par défaut "Trident").
`--silence-autosupport`: N'envoyez pas automatiquement de paquets AutoSupport à NetApp (valeur par défaut true).
`--silent`: Désactivez la plupart des sorties pendant l'installation.
`--trident-image string`: L'image Trident à installer.
`--use-custom-yaml`: Utilisez tous les fichiers YAML existants qui existent dans le répertoire d'installation.
`--use-ipv6`: Utiliser IPv6 pour la communication de Trident.

journaux

Utilisez `logs` des indicateurs pour imprimer les journaux à partir de 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`: Journal Trident à afficher. L'une des options Trident|auto|Trident-operator|All (par défaut, « auto »).
`--node string`: Nom du nœud Kubernetes à partir duquel collecter les journaux du pod du nœud.
`-p, --previous`: Si elle existe, obtenez les journaux de l'instance de conteneur précédente.
`--sidecars`: Obtenir les billes pour les conteneurs sidecar.

envoyer

Utilisez `send` la commande pour envoyer une ressource à partir de Trident.

```
tridentctl send [option]
```

Options

`autosupport`: Envoyer une archive AutoSupport à NetApp.

désinstaller

Utilisez `uninstall` des indicateurs pour désinstaller Trident.

```
tridentctl uninstall [flags]
```

Alarmes

`-h, --help`: Aide pour la désinstallation.
`--silent`: Désactivez la plupart des sorties lors de la désinstallation.

mise à jour

Utiliser `update` la commande pour modifier une ressource dans Trident.

```
tridentctl update [option]
```

Options

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

mettre à jour l'état back-end

Utiliser `update backend state` la commande pour suspendre ou reprendre les opérations back-end.

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

Points à prendre en compte

- Si un backend est créé à l'aide d'une TridentBackendConfig (tbc), le backend ne peut pas être mis à jour à l'aide d'un `backend.json` fichier.
- Si le `userState` a été défini dans un tbc, il ne peut pas être modifié à l'aide de la `tridentctl update backend state <backend-name> --user-state suspended/normal` commande.
- Pour rétablir la possibilité de définir le `userState` via tridentctl après avoir été défini via tbc, le `userState` champ doit être supprimé du tbc. Cela peut être fait à l'aide de la `kubectl edit tbc` commande. Une fois le `userState` champ supprimé, vous pouvez utiliser `tridentctl update backend state` la commande pour modifier le `userState` d'un back-end.
- Utilisez les `tridentctl update backend state` pour modifier le `userState`. Vous pouvez également mettre à jour le `userState` fichier en utilisant `TridentBackendConfig` ou `backend.json` ; ceci déclenche une réinitialisation complète du back-end et peut prendre du temps.

  #### Alarmes

  `-h, --help`: Aide pour l'état back-end.
  `--user-state`: Défini sur `suspended` pour interrompre les opérations back-end. Défini sur `normal` pour reprendre les opérations back-end. Si réglé sur `suspended`:

- `AddVolume` et `Import Volume` sont en pause.
- `CloneVolume`, `ResizeVolume` `PublishVolume`, , `UnPublishVolume`, `CreateSnapshot` `GetSnapshot` `RestoreSnapshot`, `DeleteSnapshot`, , `RemoveVolume`, `GetVolumeExternal` `ReconcileNodeAccess` et restent disponibles.

Vous pouvez également mettre à jour l'état du back-end à l'aide du `userState` champ dans le fichier de configuration du back-end `TridentBackendConfig` ou `backend.json`. Pour plus d'informations, reportez-

vous à "Options de gestion des systèmes back-end" et "Effectuer la gestion back-end avec kubectl".

Exemple:

JSON

Procédez comme suit pour mettre à jour `userState` à l'aide du `backend.json` fichier :

1. Modifiez le `backend.json` fichier pour inclure le `userState` champ avec sa valeur définie sur « terminé ».
2. Mettez à jour le backend à l'aide de la `tridentctl backend update` commande et du chemin d'accès au fichier mis à jour `backend.json` .

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

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

YAML

Vous pouvez modifier la commande tbc une fois qu'elle a été appliquée à l'aide de la `kubectl edit <tbc-name> -n <namespace>` commande. L'exemple suivant met à jour l'état back-end pour qu'il soit suspendu à l'aide de l' `userState: suspended` option :

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

version

Utilisez `version` des indicateurs pour imprimer la version de `tridentctl` et le service Trident en cours d'exécution.

```
tridentctl version [flags]
```

Alarmes

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

Prise en charge des plug-ins

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

Surveillez Trident

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

Présentation

Grâce aux mesures fournies par Trident, vous pouvez :

- Surveillez l'état et la configuration de 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. Vous pouvez examiner le temps nécessaire à Trident pour communiquer avec les systèmes back-end et effectuer les opérations.

> Par défaut, les mesures de Trident sont exposées sur le port cible `8001` au niveau du `/metrics` noeud final. Ces mesures sont **activées par défaut** lors de l'installation de Trident.

Ce dont vous avez besoin

- Cluster Kubernetes avec Trident installé.
- Instance Prometheus. Ce peut être un "Déploiement conteneurisé par Prometheus" ou vous pouvez choisir d'exécuter Prometheus comme un "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. "Blog"Vous apprendrez ainsi à utiliser

Prometheus et Grafana avec 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 Trident.

Étape 2 : créer un ServiceMonitor Prometheus

Pour consommer les metrics Trident, vous devez créer un ServiceMonitor Prometheus qui surveille le `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` service et recherche spécifiquement le `metrics` point final du service. Par conséquent, Prometheus est désormais configuré pour comprendre les metrics de Trident.

Outre les mesures directement disponibles auprès de Trident, kubelet expose de nombreuses `kubelet_volume_*` mesures 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 "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 de Trident**

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

- **Pourcentage de réponses de REPOS de Trident via 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 Trident**

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

Obtenez des informations sur l'utilisation de Trident

- **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
```

En savoir plus sur la télémétrie Trident AutoSupport

Par défaut, Trident envoie chaque jour des metrics Prometheus et des informations de base sur le back-end à NetApp.

- Pour empêcher Trident d'envoyer des metrics Prometheus et des informations back-end de base à NetApp, transmettez le `--silence-autosupport` drapeau pendant l'installation de Trident.

- Trident peut également envoyer des journaux de conteneur au support NetApp à la demande via `tridentctl send autosupport`. Vous devrez déclencher Trident pour télécharger ses journaux. Avant de soumettre des journaux, vous devez accepter les fichiers NetApp "politique de confidentialité".
- Sauf mention contraire, Trident récupère les journaux des 24 dernières heures.
- Vous pouvez spécifier la durée de conservation du journal avec l' `--since`indicateur. Par exemple : `tridentctl send autosupport --since=1h`. Ces informations sont collectées et envoyées via un `trident-autosupport` conteneur installé en même temps que Trident. Vous pouvez obtenir l'image du conteneur à l'adresse "AutoSupport Trident".
- Le AutoSupport Trident ne collecte pas et ne transmet pas d'informations à caractère personnel (PII) ou de données personnelles. Il est fourni avec un "CLUF" qui ne s'applique pas à l'image du conteneur Trident. Pour en savoir plus sur l'engagement de NetApp en faveur de la sécurité et de la confiance des données "ici".

Voici un exemple de charge envoyée par 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 l' `--image-registry`indicateur.
- Vous pouvez également configurer des URL proxy en générant les fichiers YAML d'installation. Ceci peut être fait en utilisant `tridentctl install --generate-custom-yaml` pour créer les fichiers YAML et en ajoutant l' `--proxy-url`argument pour le `trident-autosupport` conteneur dans `trident-deployment.yaml`.

Désactivez les mesures Trident

Pour que **disable** metrics ne soient pas signalés, vous devez générer des YAML personnalisés (à l'aide de `--generate-custom-yaml` l'indicateur) et les modifier pour supprimer l' `--metrics`indicateur d'appel pour le `trident-main` conteneur.

Désinstaller Trident

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

Description de la tâche

- Si vous avez besoin d'un correctif pour les bogues observés après une mise à niveau, des problèmes de dépendance ou une mise à niveau non réussie ou incomplète, désinstallez Trident et réinstallez la version précédente en suivant les instructions spécifiques à cette mise à niveau "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 de Trident ne supprime pas les CRD ou les objets associés créés par Trident. Si vous devez supprimer complètement Trident et toutes ses données, reportez-vous à la "Retirez complètement les 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 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 Trident que celle utilisée pour l'installer. Avant de procéder à la désinstallation, vérifiez la version que vous avez utilisée pour installer Trident à l'origine.

1. Utilisez `kubectl get pods -n trident` pour examiner les modules.
   - S'il n'y a pas de module opérateur, Trident a été installé à l'aide de `tridentctl`.
   - S'il existe un module opérateur, Trident a été installé à l'aide de l'opérateur Trident soit manuellement, soit à l'aide de l'assistant.
2. S'il y a un module opérateur, utilisez `kubectl describe tproc trident` pour déterminer si Trident a été installé à l'aide de l'assistant.
   - S'il y a une étiquette Helm, Trident a été installé à l'aide de Helm.
   - S'il n'y a pas d'étiquette Helm, Trident a été installé manuellement à l'aide de l'opérateur Trident.

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 Trident à l'aide de l'opérateur, vous pouvez le désinstaller en effectuant l'une des opérations suivantes :

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

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

Lorsque l'`uninstall` indicateur est défini sur `true`, l'opérateur Trident désinstalle Trident, mais ne supprime pas l'orchestrateur TridentOrchestrator lui-même. Vous devez nettoyer TridentOrchestrator et en créer un nouveau si vous souhaitez réinstaller Trident.

2. **Supprimer `TridentOrchestrator`** : en supprimant la `TridentOrchestrator` CR utilisée pour déployer Trident, vous demandez à l'opérateur de désinstaller Trident. L'opérateur procède à la suppression du `TridentOrchestrator` déploiement et du démonset 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 Trident à l'aide de Helm, vous pouvez le désinstaller en utilisant `helm uninstall`.

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

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

Désinstallez une `tridentctl` installation

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

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

Informations sur le copyright

Informations sur les marques commerciales