Los geht's

Astra Trident

NetApp
April 16, 2024

Inhalt

Los geht's

Probieren Sie es aus

NetApp stellt ein sofort einsatzbereites Lab-Image bereit, das Sie über anfordern können "NetApp Testversion".

Erfahren Sie mehr über die Testversion

Die Testversion bietet eine Sandbox-Umgebung, die mit einem Kubernetes Cluster mit drei Nodes und Astra Trident installiert und konfiguriert ist. Es ist eine gute Möglichkeit, sich mit Astra Trident vertraut zu machen und die Funktionen zu erkunden.

Eine weitere Option ist, die zu sehen "Installationsanleitung für kubeadm" Von Kubernetes bereitgestellt.

> In der Produktion sollte nicht das Kubernetes-Cluster, das Sie erstellen, mit diesen Anweisungen verwendet werden. Verwenden Sie die von der Distribution bereitgestellten Leitfäden zur Implementierung der Produktionsumgebung, um Cluster zu erstellen, die produktionsbereit sind.

Wenn Sie Kubernetes zum ersten Mal verwenden, sollten Sie sich mit den Konzepten und Tools vertraut machen "Hier".

Anforderungen

Vor der Installation von Astra Trident sollten Sie diese allgemeinen Systemanforderungen überprüfen. Spezifische Back-Ends können zusätzliche Anforderungen haben.

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TripereIT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Unterstützte Frontends (Orchestrators)

Astra Trident unterstützt mehrere Container-Engines und Orchestrierungslösungen. Dazu gehören:

- Anthos On-Prem (VMware) und Anthos auf Bare Metal 1.9, 1.10 und 1.11

- Kubernetes 1.21 - 1.26
- Mirantis Kubernetes Engine 3.5
- OpenShift 4.9 - 4.12

Der Trident-Operator wird durch folgende Versionen unterstützt:

- Anthos On-Prem (VMware) und Anthos auf Bare Metal 1.9, 1.10 und 1.11
- Kubernetes 1.21 - 1.26
- OpenShift 4.9 - 4.12

Astra Trident ist auch mit einer Vielzahl weiterer vollständig gemanagter und selbst verwalteter Kubernetes-Angebote kompatibel, darunter Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Rancher und VMware Tanzu Portfolio.

Bevor Sie ein Kubernetes Cluster von 1.24 auf 1.25 oder höher aktualisieren, auf die Astra Trident installiert ist, finden Sie unter "Aktualisieren einer Helm-basierten Bedienerinstallation".

Unterstützte Back-Ends (Storage)

Zur Verwendung von Astra Trident benötigen Sie ein oder mehrere der folgenden unterstützten Back-Ends:

- Amazon FSX für NetApp ONTAP
- Azure NetApp Dateien
- Cloud Volumes ONTAP
- Cloud Volumes Service für GCP
- FAS/All Flash FAS/Select 9.5 oder höher
- NetApp All-SAN-Array (ASA)
- NetApp HCI/Element Software 11 oder höher

Anforderungen an die Funktionen

Die nachfolgende Tabelle enthält einen Überblick über die Funktionen dieser Version von Astra Trident und die von ihm unterstützten Versionen von Kubernetes.

Merkmal	Kubernetes-Version	Funktionstore erforderlich?
CSI Trident	1.21 - 1.26	Nein
Volume Snapshots	1.21 - 1.26	Nein
PVC aus Volume Snapshots	1.21 - 1.26	Nein
ISCSI PV-Größe	1.21 - 1.26	Nein
Bidirektionales ONTAP-CHAP	1.21 - 1.26	Nein

Merkmal	Kubernetes-Version	Funktionstore erforderlich?
Dynamische Exportrichtlinien	1.21 - 1.26	Nein
Trident Operator	1.21 - 1.26	Nein
CSI-Topologie	1.21 - 1.26	Nein

Getestete Host-Betriebssysteme

Der Astra Trident unterstützt zwar bestimmte Betriebssysteme offiziell nicht, doch ist es bekannt, dass folgende Betriebssysteme funktionieren:

- Versionen von redhat CoreOS (RHCOS) werden von OpenShift Container Platform unterstützt
- RHEL 8 ODER HÖHER
- Ubuntu 22.04 oder höher
- Windows Server 2019

Standardmäßig wird Astra Trident in einem Container ausgeführt und läuft daher auf jedem Linux-Mitarbeiter. Diese Mitarbeiter müssen jedoch in der Lage sein, die Volumes, die Astra Trident bietet, je nach den von Ihnen verwendeten Back-Ends mit dem standardmäßigen NFS-Client oder iSCSI-Initiator zu mounten.

Der `tridentctl` Utility läuft auch auf jeder dieser Linux-Distributionen.

Host-Konfiguration

Alle Worker-Nodes im Kubernetes-Cluster müssen in der Lage sein, die Volumes, die Sie für Ihre Pods bereitgestellt haben, zu mounten. Um die Worker-Knoten vorzubereiten, müssen Sie die NFS- oder iSCSI-Tools auf der Grundlage Ihrer Treiberauswahl installieren.

"Bereiten Sie den Knoten „Worker“ vor"

Konfiguration des Storage-Systems

Astra Trident erfordert möglicherweise Änderungen an einem Storage-System, bevor es mit einer Backend-Konfiguration verwendet werden kann.

"Back-Ends konfigurieren"

Astra Trident-Ports

Astra Trident erfordert Zugriff auf spezifische Ports für die Kommunikation.

"Astra Trident-Ports"

Container-Images und entsprechende Kubernetes-Versionen

Bei luftvergaschten Installationen ist die folgende Liste eine Referenz für Container-Images, die für die Installation von Astra Trident erforderlich sind. Verwenden Sie die `tridentctl images` Befehl zum Überprüfen der Liste der erforderlichen Container-Images.

Kubernetes-Version	Container-Image
V1.21,0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)
V1.22.0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)
V1.23.0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)

Kubernetes-Version	Container-Image
V1.24.0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)
V1.25.0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)
V1.26.0	• docker.io/netapp/Trident:23.01.1 • docker.io/netapp/Trident-AutoSupport:23.01 • Registry.k8s.io/SIG-Storage/csi-provisioner:v3.4.0 • Registry.k8s.io/SIG-Storage/csi-Attacher:v4.1.0 • Registry.k8s.io/SIG-Storage/csi-resizer:v1.7.0 • Registry.k8s.io/SIG-Storage/csi-snapshotter:v6.2.1 • Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.7.0 • docker.io/netapp/Trident-Operator:23.01.1 (optional)

Verwenden Sie in Kubernetes ab Version 1.21 das validierte `registry.k8s.gcr.io/sig-storage/csi-snapshotter:v6.x` Bild nur, wenn der `v1` Version stellt den bereit `volumesnapshots.snapshot.storage.k8s.gcr.io` CRD.- Wenn der `v1beta1` Die Version dient der CRD mit/ohne dem `v1` Verwenden Sie die validierte Version `registry.k8s.gcr.io/sig-storage/csi-snapshotter:v3.x` Bild:

Installation Von Astra Trident

Erfahren Sie mehr über die Installation von Astra Trident

Damit Astra Trident in einer Vielzahl von Umgebungen und Organisationen installiert werden kann, bietet NetApp diverse Installationsoptionen an. Sie können Astra Trident über den Trident-Operator (manuell oder mit Helm) oder mit installieren `tridentctl`. In diesem Thema finden Sie wichtige Informationen zur Auswahl des richtigen Installationsprozesses.

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TripereIT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Bevor Sie beginnen

Unabhängig von Ihrem Installationspfad müssen Sie Folgendes haben:

- Vollständige Berechtigungen für einen unterstützten Kubernetes-Cluster, auf dem eine unterstützte Version von Kubernetes und aktivierte Funktionsanforderungen ausgeführt werden. Überprüfen Sie die "Anforderungen" Entsprechende Details.
- Zugriff auf ein unterstütztes NetApp Storage-System.
- Kann Volumes von allen Kubernetes Worker-Nodes aus mounten
- Einem Linux-Host mit `kubectl` (Oder `oc`, Falls Sie OpenShift nutzen) ist installiert und konfiguriert, um den Kubernetes-Cluster zu managen, den Sie verwenden möchten.
- Der `KUBECONFIG` Umgebungsvariable auf die Kubernetes-Cluster-Konfiguration verweisen.
- Bei Verwendung von Kubernetes mit Docker Enterprise "Führen Sie die entsprechenden Schritte aus, um den CLI-Zugriff zu aktivieren".

Wenn Sie sich nicht mit dem vertraut gemacht haben "Grundkonzepte", Ist jetzt eine tolle Zeit, um das zu tun.

Wählen Sie Ihre Installationsmethode

Wählen Sie die für Sie richtige Installationsmethode aus. Sie sollten auch die Überlegungen zu prüfen "Bewegen zwischen Methoden" Bevor Sie Ihre Entscheidung treffen.

Verwenden des Betreibers von Trident

Ob manuell oder mit Hilfe von Helm – der Trident Operator ist ein hervorragender Weg, die Installation zu vereinfachen und Astra Trident Ressourcen dynamisch zu managen. Das können Sie sogar "Individuelle Anpassung der Trident Implementierung" Verwenden der Attribute im `TridentOrchestrator` Benutzerdefinierte Ressource (CR).

Die Vorteile der Verwendung des Trident-Mitarbeiters:

** für Objekte aus Trident**

Der Trident Operator erstellt automatisch die folgenden Objekte für Ihre Kubernetes-Version.

- Servicekonto für den Betreiber
- ClusterRole und ClusterRoleBinding an das ServiceAccount
- Dedizierte PodSecurityPolicy (für Kubernetes 1.25 und früher)
- Der Bediener selbst

** Verheilen cappecackelT **

Der Bediener überwacht die Installation von Astra Trident und ergreift aktiv Maßnahmen, um Probleme wie das Löschen der Implementierung oder das versehentliche Ändern der Implementierung zu beheben. A `trident-operator-<generated-id>` Pod wird erstellt, der A zugeordnet `TridentOrchestrator` CR mit einer Astra Trident Installation. Dadurch wird sichergestellt, dass nur eine Instanz von Astra Trident im Cluster vorhanden ist und das Setup kontrolliert, um sicherzustellen, dass die Installation idempotent ist. Wenn Änderungen an der Installation vorgenommen werden (z. B. Löschen der Bereitstellung oder Knotendemonsatz), identifiziert der Bediener diese und korrigiert sie einzeln.

** Vermittlhat Updates für vorhandene InstalleIT **

Sie können eine vorhandene Implementierung einfach mit dem Bediener aktualisieren. Sie müssen nur die bearbeiten `TridentOrchestrator` CR, um Aktualisierungen für eine Installation durchzuführen.

Betrachten Sie zum Beispiel ein Szenario, bei dem Sie Astra Trident aktivieren müssen, um Debug-Protokolle zu generieren. Um dies zu tun, patchen Sie Ihre `TridentOrchestrator` Einstellen `spec.debug` Bis `true`:

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

Nachher `TridentOrchestrator` Wird aktualisiert, verarbeitet der Bediener die Updates und Patches für die vorhandene Installation. Dies kann dazu führen, dass neue Pods erstellt werden, um die Installation entsprechend zu ändern.

** für Kubernetes Upgrade**

Wenn die Kubernetes-Version des Clusters auf eine unterstützte Version aktualisiert wird, aktualisiert der Operator automatisch eine bestehende Astra Trident-Installation und ändert sie, um sicherzustellen, dass sie die Anforderungen der Kubernetes-Version erfüllt.

> Wenn das Cluster auf eine nicht unterstützte Version aktualisiert wird, verhindert der Operator die Installation von Astra Trident. Falls Astra Trident bereits mit dem Operator installiert wurde, wird eine Warnmeldung angezeigt, die angibt, dass Astra Trident auf einer nicht unterstützten Kubernetes-Version installiert ist.

** Cluster Management unter Verwendung von BlueXP (früher Cloud Manager)**

Mit "Astra Trident mit BlueXP", Sie können ein Upgrade auf die neueste Version von Astra Trident durchführen, Storage-Klassen hinzufügen und managen, mit Arbeitsumgebungen verbinden und persistente Volumes mit Cloud Backup Service sichern. BlueXP unterstützt die Astra Trident-Implementierung mithilfe des Trident-Operators entweder manuell oder über Helm.

Wird Verwendet `tridentctl`

Wenn Sie bereits eine Implementierung haben, die ein Upgrade durchgeführt werden muss oder eine stark angepasste Implementierung benötigen, sollten Sie dies in Betracht ziehen . Dies ist die herkömmliche Methode der Implementierung von Astra Trident.

Das können Sie Generierung der Manifeste für Trident Ressourcen: Dies umfasst die Implementierung, das Demonet, das Servicekonto und die Cluster-Rolle, die Astra Trident im Rahmen der Installation erstellt.

> Ab Version 22.04 werden die AES-Schlüssel nicht mehr bei jeder Installation von Astra Trident neu generiert. Mit dieser Version installiert Astra Trident ein neues geheimes Objekt, das bei den Installationen fortbesteht. Das bedeutet, `tridentctl` In 22.04 können frühere Versionen von Trident deinstalliert werden, ältere Versionen können jedoch nicht 22.04 Installationen deinstallieren. Wählen Sie die entsprechende Installation_method_ aus.

Wählen Sie den Installationsmodus aus

Bestimmen Sie Ihren Bereitstellungsprozess auf der Grundlage des von Ihrem Unternehmen benötigten *Installations-Modus* (Standard, Offline oder Remote).

Standardinstallation

Dies ist der einfachste Weg, Astra Trident zu installieren und funktioniert für die meisten Umgebungen, die keine Netzwerkeinschränkungen auferlegen. Im Standardinstallationsmodus werden standardmäßig erforderliche Trident-Datenbanken verwendet (`docker.io`) Und CSI (`registry.k8s.io`) Bilder.

Wenn Sie den Standardmodus verwenden, können Sie das Astra Trident-Installationsprogramm:

- Ruft die Container-Images über das Internet ab
- Erstellt eine Implementierung oder Node-Demonset, bei dem Astra Trident Pods auf allen teilnahmeberechtigten Nodes im Kubernetes Cluster gespinnt werden

Offline-Installation

Der Offline-Installationsmodus kann an einem luftgekapselten oder sicheren Ort erforderlich sein. In diesem Szenario können Sie eine einzelne private, gespiegelte Registry oder zwei gespiegelte Registryrien erstellen, um die erforderlichen Trident- und CSI-Images zu speichern.

> Unabhängig von Ihrer Registrierungskonfiguration müssen CSI-Bilder in einer Registrierung enthalten sein.

Remote-Installation

Hier finden Sie einen allgemeinen Überblick über den Remote-Installationsprozess:

- Stellen Sie die entsprechende Version von bereit `kubectl` Auf dem Remote-Rechner, von wo aus Sie Astra Trident implementieren möchten.
- Kopieren Sie die Konfigurationsdateien aus dem Kubernetes-Cluster und legen Sie die fest `KUBECONFIG` Umgebungsvariable auf dem Remotecomputer.
- Initiieren Sie A `kubectl get nodes` Befehl zum Überprüfen, ob eine Verbindung mit dem erforderlichen Kubernetes-Cluster hergestellt werden kann.
- Führen Sie die Implementierung von der Remote-Maschine aus, indem Sie die standardmäßigen Installationsschritte verwenden.

Wählen Sie den Prozess basierend auf Methode und Modus aus

Nachdem Sie Ihre Entscheidungen getroffen haben, wählen Sie den entsprechenden Prozess aus.

Methode	Installationsmodus
Trident-Operator (manuell)	"Standardinstallation" "Offline-Installation"

Methode	Installationsmodus
Betreiber von Trident (Helm)	"Standardinstallation" "Offline-Installation"
`tridentctl`	"Standard- oder Offline-Installation"

Wechseln zwischen den Installationsmethoden

Sie können sich entscheiden, Ihre Installationsmethode zu ändern. Bevor Sie dies tun, sollten Sie folgendes bedenken:

- Verwenden Sie immer die gleiche Methode für die Installation und Deinstallation von Astra Trident. Wenn Sie mit bereitgestellt haben `tridentctl`, Sie sollten die entsprechende Version des verwenden `tridentctl` Binary zur Deinstallation von Astra Trident. Ebenso sollten Sie bei der Bereitstellung mit dem Operator die bearbeiten `TridentOrchestrator` CR und Set `spec.uninstall=true` Um Astra Trident zu deinstallieren.
- Wenn Sie über eine bedienerbasierte Bereitstellung verfügen, die Sie stattdessen entfernen und verwenden möchten `tridentctl` Bei der Implementierung von Astra Trident sollten Sie zuerst bearbeiten `TridentOrchestrator` Und gesetzt `spec.uninstall=true` Um Astra Trident zu deinstallieren. Löschen Sie dann `TridentOrchestrator` Und die Bedienerbereitstellung. Sie können dann mit installieren `tridentctl`.
- Wenn Sie über eine manuelle, bedienerbasierte Implementierung verfügen und die Helm-basierte Trident Operator-Implementierung verwenden möchten, sollten Sie zuerst den Operator manuell deinstallieren und dann die Helm-Installation durchführen. So kann Helm den Trident-Operator mit den erforderlichen Beschriftungen und Anmerkungen implementieren. Wenn dies nicht der Fall ist, schlägt die Bereitstellung des Helm-basierten Trident-Operators mit einem Fehler bei der Labelvalidierung und einem Validierungsfehler bei der Annotation fehl. Wenn Sie eine haben `tridentctl`-Basierte Bereitstellung, können Sie Helm-basierte Implementierung nutzen, ohne Probleme zu verursachen.

Andere bekannte Konfigurationsoptionen

Bei der Installation von Astra Trident auf VMware Tanzu Portfolio Produkten:

- Das Cluster muss privilegierte Workloads unterstützen.
- Der `--kubelet-dir` Flag sollte auf den Speicherort des kubelet-Verzeichnisses gesetzt werden. Standardmäßig ist dies `/var/vcap/data/kubelet`.

  Festlegen der Kubelet-Position unter Verwendung `--kubelet-dir` Ist für Trident Operator, Helm und bekannt `tridentctl` Implementierungen.

Installation über den Trident Operator

Manuelle Implementierung des Trident-Mitarbeiters (Standard-Modus)

Sie können den Trident-Operator manuell implementieren, um Astra Trident zu installieren. Dieser Prozess gilt für Installationen, bei denen die von Astra Trident benötigten Container-Images nicht in einer privaten Registrierung gespeichert werden.

Wenn Sie über eine private Bildregistrierung verfügen, verwenden Sie das "Prozess für Offline-Implementierung".

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TripereIT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Trident-Operator kann manuell implementiert und Trident installiert werden

Prüfen "Die Übersicht über die Installation" Um sicherzustellen, dass Sie die Installationsvoraussetzungen erfüllt haben, und die richtige Installationsoption für Ihre Umgebung ausgewählt haben.

Bevor Sie beginnen

Melden Sie sich vor der Installation beim Linux-Host an, und überprüfen Sie, ob er einen funktionierenden "Unterstützter Kubernetes-Cluster" Und dass Sie die erforderlichen Berechtigungen haben.

Mit OpenShift, verwenden `oc` Statt `kubectl` In allen folgenden Beispielen, und melden Sie sich als **System:admin** zuerst mit dem Ausführen an `oc login -u system:admin` Oder `oc login -u kube-admin`.

1. Überprüfen Sie Ihre Kubernetes Version:

```
kubectl version
```

2. Überprüfung der Berechtigungen für Cluster-Administratoren:

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

3. Überprüfen Sie, ob Sie einen Pod starten können, der ein Image aus dem Docker Hub verwendet, und ob er das Storage-System über das POD-Netzwerk erreichen kann:

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

Schritt 1: Laden Sie das Trident Installer-Paket herunter

Das Astra Trident Installationspaket enthält alles, was Sie für die Bereitstellung des Trident-Operators und die Installation von Astra Trident benötigen. Laden Sie die neueste Version des Trident Installationsprogramms herunter und extrahieren Sie sie aus "Die Sektion *Assets* auf GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v23.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
```

Schritt 2: Erstellen Sie die `TridentOrchestrator` CRD.

Erstellen Sie die `TridentOrchestrator` Benutzerdefinierte Ressourcendefinition (CRD). Sie werden ein erstellen `TridentOrchestrator` Benutzerdefinierte Ressourcen später. Verwenden Sie die entsprechende CRD YAML-Version in `deploy/crds` Um die zu erstellen `TridentOrchestrator` CRD.-

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

Schritt 3: Implementieren Sie den Trident-Operator

Das Astra Trident-Installationsprogramm stellt eine Paketdatei bereit, mit der der Operator installiert und zugehörige Objekte erstellt werden können. Die Bundle-Datei ist eine einfache Möglichkeit, den Operator zu implementieren und Astra Trident mit einer Standardkonfiguration zu installieren.

- Verwenden Sie für Cluster mit Kubernetes 1.24 oder älter `bundle_pre_1_25.yaml`.

- Verwenden Sie für Cluster mit Kubernetes 1.25 oder höher `bundle_post_1_25.yaml`.

Das Trident-Installationsprogramm implementiert den Operator im `trident` Namespace. Wenn der `trident` Der Namespace ist nicht vorhanden, verwenden Sie `kubectl apply -f deploy/namespace.yaml` Um sie zu erstellen.

Schritte

1. Erstellen Sie die Ressourcen und stellen Sie den Operator bereit:

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

> Um den Operator in einem anderen Namespace als dem bereitzustellen `trident` Namespace, Update `serviceaccount.yaml`, `clusterrolebinding.yaml` Und `operator.yaml` Und erstellen Sie Ihre Bundle-Datei mit `kustomization.yaml`:
>
> ```
> kubectl kustomize deploy/ > deploy/<bundle>.yaml
> ```

2. Überprüfen Sie, ob der Bediener bereitgestellt wurde.

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

NAME                READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator    1/1     1            1           3m
```

> Es sollte nur eine Instanz* des Operators in einem Kubernetes-Cluster geben. Erstellen Sie nicht mehrere Implementierungen des Trident-Operators.

Schritt 4: Erstellen Sie die `TridentOrchestrator` Und Trident installieren

Sie können jetzt die erstellen `TridentOrchestrator` Und Installation von Astra Trident durchführen. Optional können Sie "Anpassung der Trident Installation" Verwenden der Attribute im `TridentOrchestrator` Spez.

```
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:23.01
    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:23.01.1
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v23.01.1
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Überprüfen Sie die Installation

Die Installation kann auf verschiedene Weise überprüft werden.

Wird Verwendet `TridentOrchestrator` Status

Der Status von `TridentOrchestrator` Gibt an, ob die Installation erfolgreich war und zeigt die installierte

Version von Trident an. Während der Installation den Status von `TridentOrchestrator` Änderungen von `Installing` Bis `Installed`. Wenn Sie die beobachten `Failed` Der Status und der Operator kann sich nicht selbst wiederherstellen. "Prüfen Sie die Protokolle".

Status	Beschreibung
Installation	Der Betreiber installiert damit den Astra Trident `TridentOrchestrator` CR.
Installiert	Astra Trident wurde erfolgreich installiert.
Deinstallation	Der Betreiber deinstalliert den Astra Trident, denn `spec.uninstall=true`.
Deinstalliert	Astra Trident ist deinstalliert.
Fehlgeschlagen	Der Operator konnte Astra Trident nicht installieren, patchen, aktualisieren oder deinstallieren; der Operator versucht automatisch, aus diesem Zustand wiederherzustellen. Wenn dieser Status weiterhin besteht, müssen Sie eine Fehlerbehebung durchführen.
Aktualisierung	Der Bediener aktualisiert eine vorhandene Installation.
Fehler	Der `TridentOrchestrator` Wird nicht verwendet. Eine weitere ist bereits vorhanden.

Den Status der Pod-Erstellung verwenden

Überprüfen Sie den Status der erstellten Pods, ob die Astra Trident-Installation abgeschlossen wurde:

```
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
```

Wird Verwendet `tridentctl`

Verwenden Sie können `tridentctl` Um die installierte Version von Astra Trident zu überprüfen.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.01.1        | 23.01.1        |
+----------------+----------------+
```

Wie es weiter geht

Das ist jetzt möglich "Erstellen Sie ein Back-End und eine Storage-Klasse, stellen Sie ein Volume bereit und mounten Sie das Volume in einem POD".

Manuelles Bereitstellen des Trident-Mitarbeiters (Offline-Modus)

Sie können den Trident-Operator manuell implementieren, um Astra Trident zu installieren. Dieser Prozess gilt für Installationen, bei denen die von Astra Trident benötigten Container-Images in einer privaten Registrierung gespeichert werden. Wenn Sie keine private Bildregistrierung besitzen, verwenden Sie das "Standardimplementierung einsetzen".

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

Informationen über Astra TriperelT

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Trident-Operator kann manuell implementiert und Trident installiert werden

Prüfen "Die Übersicht über die Installation" Um sicherzustellen, dass Sie die Installationsvoraussetzungen erfüllt haben, und die richtige Installationsoption für Ihre Umgebung ausgewählt haben.

Bevor Sie beginnen

Melden Sie sich beim Linux-Host an, und überprüfen Sie, ob er einen funktionierenden und verwaltet "Unterstützter Kubernetes-Cluster" Und dass Sie die erforderlichen Berechtigungen haben.

Mit OpenShift, verwenden `oc` Statt `kubectl` In allen folgenden Beispielen, und melden Sie sich als **System:admin** zuerst mit dem Ausführen an `oc login -u system:admin` Oder `oc login -u kube-admin`.

1. Überprüfen Sie Ihre Kubernetes Version:

```
kubectl version
```

2. Überprüfung der Berechtigungen für Cluster-Administratoren:

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

3. Überprüfen Sie, ob Sie einen Pod starten können, der ein Image aus dem Docker Hub verwendet, und ob er das Storage-System über das POD-Netzwerk erreichen kann:

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

Schritt 1: Laden Sie das Trident Installer-Paket herunter

Das Astra Trident Installationspaket enthält alles, was Sie für die Bereitstellung des Trident-Operators und die Installation von Astra Trident benötigen. Laden Sie die neueste Version des Trident Installationsprogramms herunter und extrahieren Sie sie aus "Die Sektion *Assets* auf GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v23.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
```

Schritt 2: Erstellen Sie die `TridentOrchestrator` CRD.

Erstellen Sie die `TridentOrchestrator` Benutzerdefinierte Ressourcendefinition (CRD). Sie werden ein erstellen `TridentOrchestrator` Benutzerdefinierte Ressourcen später. Verwenden Sie die entsprechende CRD YAML-Version in `deploy/crds` Um die zu erstellen `TridentOrchestrator` CRD:

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

Schritt 3: Aktualisieren Sie den Registrierungsort im Operator

In `/deploy/operator.yaml`, Update `image: docker.io/netapp/trident-operator:23.01.1` So geben Sie den Speicherort Ihrer Bildregistrierung an. Ihr "Trident und CSI-Images" Kann in einer Registrierung

oder in verschiedenen Registern gefunden werden, aber alle CSI-Images müssen sich in derselben Registrierung befinden. Beispiel:

- `image: <your-registry>/trident-operator:23.01.1` Wenn Ihre Bilder alle in einer Registrierung gespeichert sind.
- `image: <your-registry>/netapp/trident-operator:23.01.1` Wenn sich Ihr Trident-Image in einer anderen Registrierung als Ihre CSI-Images befindet.

Schritt 4: Implementieren des Trident-Operators

Das Trident-Installationsprogramm implementiert den Operator im `trident` Namespace. Wenn der `trident` Der Namespace ist nicht vorhanden, verwenden Sie `kubectl apply -f deploy/namespace.yaml` Um sie zu erstellen.

Um den Operator in einem anderen Namespace als dem bereitzustellen `trident` Namespace, Update `serviceaccount.yaml`, `clusterrolebinding.yaml` Und `operator.yaml` Vor der Bereitstellung des Bedieners.

1. Erstellen Sie die Ressourcen und stellen Sie den Operator bereit:

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

> Das Astra Trident-Installationsprogramm stellt eine Paketdatei bereit, mit der der Operator installiert und zugehörige Objekte erstellt werden können. Die Bundle-Datei ist eine einfache Möglichkeit, den Operator zu implementieren und Astra Trident mit einer Standardkonfiguration zu installieren.
>
> - Verwenden Sie für Cluster mit Kubernetes 1.24 oder älter `bundle_pre_1_25.yaml`.
> - Verwenden Sie für Cluster mit Kubernetes 1.25 oder höher `bundle_post_1_25.yaml`.

2. Überprüfen Sie, ob der Bediener bereitgestellt wurde.

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

NAME                READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator    1/1     1            1           3m
```

> Es sollte nur eine Instanz* des Operators in einem Kubernetes-Cluster geben. Erstellen Sie nicht mehrere Implementierungen des Trident-Operators.

Schritt 5: Aktualisieren Sie den Speicherort der Bildregistrierung im `TridentOrchestrator`

Ihr "Trident und CSI-Images" Kann in einer Registrierung oder in verschiedenen Registern gefunden werden, aber alle CSI-Images müssen sich in derselben Registrierung befinden. Aktualisierung `deploy/crds/tridentorchestrator_cr.yaml` So fügen Sie zusätzliche Standortspezifikationen basierend auf Ihrer Registrierungskonfiguration hinzu.

Bilder in einer Registrierung

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:23.01"
tridentImage: "<your-registry>/trident:23.01.1"
```

Bilder in verschiedenen Registern

Sie müssen anhängen `sig-storage` Bis zum `imageRegistry` Um unterschiedliche Registrierungsstandorte zu verwenden.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:23.01"
tridentImage: "<your-registry>/netapp/trident:23.01.1"
```

Schritt 6: Erstellen Sie die `TridentOrchestrator` Und Trident installieren

Sie können jetzt die erstellen `TridentOrchestrator` Und Installation von Astra Trident durchführen. Optional können Sie weiter "Anpassung der Trident Installation" Verwenden der Attribute im `TridentOrchestrator` Spez. Das folgende Beispiel zeigt eine Installation, bei der sich Trident- und CSI-Bilder in verschiedenen Registern befinden.

```
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:23.01
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:23.01.1
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:23.01
    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:23.01.1
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v23.01.1
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Überprüfen Sie die Installation

Die Installation kann auf verschiedene Weise überprüft werden.

Wird Verwendet `TridentOrchestrator` Status

Der Status von `TridentOrchestrator` Gibt an, ob die Installation erfolgreich war und zeigt die installierte Version von Trident an. Während der Installation den Status von `TridentOrchestrator` Änderungen von `Installing` Bis `Installed`. Wenn Sie die beobachten `Failed` Der Status und der Operator kann sich nicht selbst wiederherstellen. "Prüfen Sie die Protokolle".

Status	Beschreibung
Installation	Der Betreiber installiert damit den Astra Trident `TridentOrchestrator` CR.
Installiert	Astra Trident wurde erfolgreich installiert.
Deinstallation	Der Betreiber deinstalliert den Astra Trident, denn `spec.uninstall=true`.
Deinstalliert	Astra Trident ist deinstalliert.
Fehlgeschlagen	Der Operator konnte Astra Trident nicht installieren, patchen, aktualisieren oder deinstallieren; der Operator versucht automatisch, aus diesem Zustand wiederherzustellen. Wenn dieser Status weiterhin besteht, müssen Sie eine Fehlerbehebung durchführen.
Aktualisierung	Der Bediener aktualisiert eine vorhandene Installation.
Fehler	Der `TridentOrchestrator` Wird nicht verwendet. Eine weitere ist bereits vorhanden.

Den Status der Pod-Erstellung verwenden

Überprüfen Sie den Status der erstellten Pods, ob die Astra Trident-Installation abgeschlossen wurde:

```
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
```

Wird Verwendet `tridentctl`

Verwenden Sie können `tridentctl` Um die installierte Version von Astra Trident zu überprüfen.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.01.1        | 23.01.1        |
+----------------+----------------+
```

Wie es weiter geht

Das ist jetzt möglich "Erstellen Sie ein Back-End und eine Storage-Klasse, stellen Sie ein Volume bereit und mounten Sie das Volume in einem POD".

Trident Operator mit Helm (Standard-Modus) implementieren

Sie können den Trident-Operator implementieren und Astra Trident mithilfe von Helm installieren. Dieser Prozess gilt für Installationen, bei denen die von Astra Trident benötigten Container-Images nicht in einer privaten Registrierung gespeichert werden. Wenn Sie über eine private Bildregistrierung verfügen, verwenden Sie das "Prozess für Offline-Implementierung".

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TripereIT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Setzen Sie den Trident-Operator ein und installieren Sie Astra Trident mit Helm

Verwendung von Trident "Steuerruderdiagramm" Sie können den Trident Operator implementieren und Trident in einem Schritt installieren.

Prüfen "Die Übersicht über die Installation" Um sicherzustellen, dass Sie die Installationsvoraussetzungen erfüllt haben, und die richtige Installationsoption für Ihre Umgebung ausgewählt haben.

Bevor Sie beginnen

Zusätzlich zum "Voraussetzungen für die Implementierung" Die Sie benötigen "Helm Version 3".

Schritte

1. Fügen Sie das Helm Repository von Astra Trident hinzu:

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

2. Nutzung `helm install` Und geben Sie einen Namen für Ihre Bereitstellung an, wie im folgenden Beispiel, wo `23.01.1` Ist die Version des Astra Trident, die Sie installieren.

```
helm install <name> netapp-trident/trident-operator --version 23.01.1
--create-namespace --namespace <trident-namespace>
```

> Wenn Sie bereits einen Namespace für Trident erstellt haben, wird der `--create -namespace` Parameter erstellt keinen zusätzlichen Namespace.

Verwenden Sie können `helm list` So prüfen Sie Installationsdetails wie Name, Namespace, Diagramm, Status, App-Version, Und Revisionsnummer.

Konfigurationsdaten während der Installation übergeben

Während der Installation gibt es zwei Möglichkeiten, die Konfigurationsdaten zu übergeben:

Option	Beschreibung
`--values` (Oder `-f`)	Geben Sie eine YAML-Datei mit Überschreibungen an. Dies kann mehrfach angegeben werden, und die rechteste Datei hat Vorrang.
`--set`	Geben Sie Überschreibungen in der Befehlszeile an.

Um beispielsweise den Standardwert von zu ändern `debug`, Ausführen Sie das folgende `--set` Befehl wo `23.01.1` Ist die Version von Astra Trident, die Sie installieren:

```
helm install <name> netapp-trident/trident-operator --version 23.01.1
--create-namespace --namespace --set tridentDebug=true
```

Konfigurationsoptionen

Diese Tabelle und die `values.yaml` Datei, die Teil des Helm-Diagramms ist, enthält die Liste der Schlüssel und ihre Standardwerte.

Option	Beschreibung	Standard
`nodeSelector`	Node-Etiketten für Pod-Zuweisung	
`podAnnotations`	Pod-Anmerkungen	
`deploymentAnnotations`	Anmerkungen zur Bereitstellung	
`tolerations`	Toleranzen für Pod-Zuweisung	
`affinity`	Affinität für Pod-Zuweisung	
`tridentControllerPluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentControllerPluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentNodePluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentNodePluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	

Option	Beschreibung	Standard
`imageRegistry`	Identifiziert die Registrierung für den `trident-operator`, `trident`, Und andere Bilder. Lassen Sie das Feld leer, um die Standardeinstellung zu übernehmen.	„“
`imagePullPolicy`	Legt die Richtlinie zum Abziehen von Bildern für den fest `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Legt die Abzugsgeheimnisse für das Bild fest `trident-operator`, `trident`, Und andere Bilder.	
`kubeletDir`	Ermöglicht das Überschreiben der Hostposition des internen Status von kubelet.	'“/var/lib/kubelet“`
`operatorLogLevel`	Ermöglicht die Einstellung der Protokollebene des Trident-Operators auf: `trace`, `debug`, `info`, `warn`, `error`, Oder `fatal`.	`"info"`
`operatorDebug`	Ermöglicht es, die Protokollebene des Trident-Operators auf Debug zu setzen.	`true`
`operatorImage`	Ermöglicht die vollständige Überschreibung des Bildes für `trident-operator`.	„“
`operatorImageTag`	Ermöglicht das Überschreiben des Tags des `trident-operator` Bild:	„“
`tridentIPv6`	Ermöglicht die Aktivierung von Astra Trident in IPv6-Clustern.	`false`
`tridentK8sTimeout`	Setzt das standardmäßige 30-Sekunden-Zeitlimit für die meisten Kubernetes-API-Vorgänge außer Kraft (wenn nicht Null, in Sekunden).	`0`
`tridentHttpRequestTimeout`	Setzt das standardmäßige 90-Sekunden-Timeout für die HTTP-Anforderungen mit außer Kraft `0s` Ist eine unendliche Dauer für das Timeout. Negative Werte sind nicht zulässig.	`"90s"`
`tridentSilenceAutosupport`	Ermöglicht die Deaktivierung von regelmäßigen AutoSupport Berichten für Astra Trident.	`false`

Option	Beschreibung	Standard
`tridentAutosupportImageTag`	Ermöglicht das Überschreiben des Tags des Images für den Astra Trident AutoSupport-Container.	`<version>`
`tridentAutosupportProxy`	Der Astra Trident AutoSupport Container kann über einen HTTP-Proxy nach Hause telefonieren.	„“
`tridentLogFormat`	Legt das Astra Trident Protokollierungsformat fest (`text` Oder `json`).	`"text"`
`tridentDisableAuditLog`	Deaktiviert den Astra Trident Audit-Logger.	`true`
`tridentLogLevel`	Ermöglicht die Festlegung der Protokollebene von Astra Trident auf: `trace`, `debug`, `info`, `warn`, `error`, Oder `fatal`.	`"info"`
`tridentDebug`	Ermöglicht das Festlegen der Protokollebene für Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Ermöglicht die Aktivierung bestimmter Astra Trident Workflows für die Trace-Protokollierung oder Protokollunterdrückung.	„“
`tridentLogLayers`	Ermöglicht die Aktivierung bestimmter Astra Trident-Ebenen für die Trace-Protokollierung oder Protokollunterdrückung.	„“
`tridentImage`	Ermöglicht die vollständige Überschreibung des Images für Astra Trident.	„“
`tridentImageTag`	Ermöglicht das Überschreiben des Tags des Images für Astra Trident.	„“
`tridentProbePort`	Ermöglicht das Überschreiben des Standardports, der für Kubernetes Liveness/Readiness-Sonden verwendet wird.	„“
`windows`	Ermöglicht die Installation von Astra Trident auf einem Windows Worker-Node.	`false`
`enableForceDetach`	Ermöglicht die Aktivierung der Funktion zum Abtrennen erzwingen.	`false`
`excludePodSecurityPolicy`	Schließt die Sicherheitsrichtlinie des Operator POD von der Erstellung aus.	`false`

Allgemeines zu Controller-Pods und Node-Pods

Astra Trident wird als einzelner Controller-Pod ausgeführt sowie als Node-Pod auf jedem Worker-Node im Cluster. Der Node Pod muss auf jedem Host ausgeführt werden, auf dem Sie ein Astra Trident Volume mounten möchten.

Kubernetes "Knotenauswahl" Und "Toleranzen und Verfleckungen" Werden verwendet, um die Ausführung eines Pod auf einem bestimmten oder bevorzugten Node einzuschränken. Verwenden von`ControllerPlugin` und `NodePlugin`, Sie können Bedingungen und Überschreibungen festlegen.

- Das Controller-Plug-in übernimmt Volume-Bereitstellung und -Management, beispielsweise Snapshots und Größenanpassungen.
- Das Node-Plug-in verarbeitet das Verbinden des Speichers mit dem Node.

Wie es weiter geht

Das ist jetzt möglich "Erstellen Sie ein Back-End und eine Storage-Klasse, stellen Sie ein Volume bereit und mounten Sie das Volume in einem POD".

Trident-Operator mit Helm (Offline-Modus) implementieren

Sie können den Trident-Operator implementieren und Astra Trident mithilfe von Helm installieren. Dieser Prozess gilt für Installationen, bei denen die von Astra Trident benötigten Container-Images in einer privaten Registrierung gespeichert werden. Wenn Sie keine private Bildregistrierung besitzen, verwenden Sie das "Standardimplementierung einsetzen".

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TriperelT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Setzen Sie den Trident-Operator ein und installieren Sie Astra Trident mit Helm

Verwendung von Trident "Steuerruderdiagramm" Sie können den Trident Operator implementieren und Trident in einem Schritt installieren.

Prüfen "Die Übersicht über die Installation" Um sicherzustellen, dass Sie die Installationsvoraussetzungen erfüllt haben, und die richtige Installationsoption für Ihre Umgebung ausgewählt haben.

Bevor Sie beginnen

Zusätzlich zum "Voraussetzungen für die Implementierung" Die Sie benötigen "Helm Version 3".

Schritte

1. Fügen Sie das Helm Repository von Astra Trident hinzu:

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

2. Nutzung `helm install` Und geben Sie einen Namen für die Bereitstellung und den Speicherort der Image-Registrierung an. Ihr "Trident und CSI-Images" Kann in einer Registrierung oder in verschiedenen Registern gefunden werden, aber alle CSI-Images müssen sich in derselben Registrierung befinden. In den Beispielen: `23.01.1` Ist die Version des Astra Trident, die Sie installieren.

Bilder in einer Registrierung

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

Bilder in verschiedenen Registern

Sie müssen anhängen `sig-storage` Bis zum `imageRegistry` Um unterschiedliche Registrierungsstandorte zu verwenden.

```
helm install <name> netapp-trident/trident-operator --version
23.01.1 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:23.01.1 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:23.01 --set tridentImage=<your-
registry>/netapp/trident:23.01.1 --create-namespace --namespace
<trident-namespace>
```

Wenn Sie bereits einen Namespace für Trident erstellt haben, wird der `--create-namespace` Parameter erstellt keinen zusätzlichen Namespace.

Verwenden Sie können `helm list` So prüfen Sie Installationsdetails wie Name, Namespace, Diagramm, Status, App-Version, Und Revisionsnummer.

Konfigurationsdaten während der Installation übergeben

Während der Installation gibt es zwei Möglichkeiten, die Konfigurationsdaten zu übergeben:

Option	Beschreibung
`--values` (Oder `-f`)	Geben Sie eine YAML-Datei mit Überschreibungen an. Dies kann mehrfach angegeben werden, und die rechteste Datei hat Vorrang.
`--set`	Geben Sie Überschreibungen in der Befehlszeile an.

Um beispielsweise den Standardwert von zu ändern `debug`, Ausführen Sie das folgende `--set` Befehl wo `23.01.1` Ist die Version von Astra Trident, die Sie installieren:

```
helm install <name> netapp-trident/trident-operator --version 23.01.1
--create-namespace --namespace --set tridentDebug=true
```

Konfigurationsoptionen

Diese Tabelle und die `values.yaml` Datei, die Teil des Helm-Diagramms ist, enthält die Liste der Schlüssel und ihre Standardwerte.

Option	Beschreibung	Standard
`nodeSelector`	Node-Etiketten für Pod-Zuweisung	
`podAnnotations`	Pod-Anmerkungen	
`deploymentAnnotations`	Anmerkungen zur Bereitstellung	
`tolerations`	Toleranzen für Pod-Zuweisung	
`affinity`	Affinität für Pod-Zuweisung	
`tridentControllerPluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentControllerPluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentNodePluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	
`tridentNodePluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Siehe Allgemeines zu Controller-Pods und Node-Pods Entsprechende Details.	

Option	Beschreibung	Standard
`imageRegistry`	Identifiziert die Registrierung für den `trident-operator`, `trident`, Und andere Bilder. Lassen Sie das Feld leer, um die Standardeinstellung zu übernehmen.	„“
`imagePullPolicy`	Legt die Richtlinie zum Abziehen von Bildern für den fest `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Legt die Abzugsgeheimnisse für das Bild fest `trident-operator`, `trident`, Und andere Bilder.	
`kubeletDir`	Ermöglicht das Überschreiben der Hostposition des internen Status von kubelet.	'“/var/lib/kubelet“`
`operatorLogLevel`	Ermöglicht die Einstellung der Protokollebene des Trident-Operators auf: `trace`, `debug`, `info`, `warn`, `error`, Oder `fatal`.	`"info"`
`operatorDebug`	Ermöglicht es, die Protokollebene des Trident-Operators auf Debug zu setzen.	`true`
`operatorImage`	Ermöglicht die vollständige Überschreibung des Bildes für `trident-operator`.	„“
`operatorImageTag`	Ermöglicht das Überschreiben des Tags des `trident-operator` Bild:	„“
`tridentIPv6`	Ermöglicht die Aktivierung von Astra Trident in IPv6-Clustern.	`false`
`tridentK8sTimeout`	Setzt das standardmäßige 30-Sekunden-Zeitlimit für die meisten Kubernetes-API-Vorgänge außer Kraft (wenn nicht Null, in Sekunden).	`0`
`tridentHttpRequestTimeout`	Setzt das standardmäßige 90-Sekunden-Timeout für die HTTP-Anforderungen mit außer Kraft `0s` Ist eine unendliche Dauer für das Timeout. Negative Werte sind nicht zulässig.	`"90s"`
`tridentSilenceAutosupport`	Ermöglicht die Deaktivierung von regelmäßigen AutoSupport Berichten für Astra Trident.	`false`

Option	Beschreibung	Standard
`tridentAutosupportImageTag`	Ermöglicht das Überschreiben des Tags des Images für den Astra Trident AutoSupport-Container.	`<version>`
`tridentAutosupportProxy`	Der Astra Trident AutoSupport Container kann über einen HTTP-Proxy nach Hause telefonieren.	„“
`tridentLogFormat`	Legt das Astra Trident Protokollierungsformat fest (`text` Oder `json`).	`"text"`
`tridentDisableAuditLog`	Deaktiviert den Astra Trident Audit-Logger.	`true`
`tridentLogLevel`	Ermöglicht die Festlegung der Protokollebene von Astra Trident auf: `trace`, `debug`, `info`, `warn`, `error`, Oder `fatal`.	`"info"`
`tridentDebug`	Ermöglicht das Festlegen der Protokollebene für Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Ermöglicht die Aktivierung bestimmter Astra Trident Workflows für die Trace-Protokollierung oder Protokollunterdrückung.	„“
`tridentLogLayers`	Ermöglicht die Aktivierung bestimmter Astra Trident-Ebenen für die Trace-Protokollierung oder Protokollunterdrückung.	„“
`tridentImage`	Ermöglicht die vollständige Überschreibung des Images für Astra Trident.	„“
`tridentImageTag`	Ermöglicht das Überschreiben des Tags des Images für Astra Trident.	„“
`tridentProbePort`	Ermöglicht das Überschreiben des Standardports, der für Kubernetes Liveness/Readiness-Sonden verwendet wird.	„“
`windows`	Ermöglicht die Installation von Astra Trident auf einem Windows Worker-Node.	`false`
`enableForceDetach`	Ermöglicht die Aktivierung der Funktion zum Abtrennen erzwingen.	`false`
`excludePodSecurityPolicy`	Schließt die Sicherheitsrichtlinie des Operator POD von der Erstellung aus.	`false`

Allgemeines zu Controller-Pods und Node-Pods

Astra Trident wird als einzelner Controller-Pod ausgeführt sowie als Node-Pod auf jedem Worker-Node im Cluster. Der Node Pod muss auf jedem Host ausgeführt werden, auf dem Sie ein Astra Trident Volume mounten möchten.

Kubernetes "Knotenauswahl" Und "Toleranzen und Verfleckungen" Werden verwendet, um die Ausführung eines Pod auf einem bestimmten oder bevorzugten Node einzuschränken. Verwenden von`ControllerPlugin` und `NodePlugin`, Sie können Bedingungen und Überschreibungen festlegen.

- Das Controller-Plug-in übernimmt Volume-Bereitstellung und -Management, beispielsweise Snapshots und Größenanpassungen.
- Das Node-Plug-in verarbeitet das Verbinden des Speichers mit dem Node.

Wie es weiter geht

Das ist jetzt möglich "Erstellen Sie ein Back-End und eine Storage-Klasse, stellen Sie ein Volume bereit und mounten Sie das Volume in einem POD".

Anpassen der Trident Operator-Installation

Über den Trident-Operator können Sie die Astra Trident-Installation anhand der Attribute im anpassen `TridentOrchestrator` Spez. Wenn Sie die Installation über die von Ihnen gewünschte hinaus anpassen möchten `TridentOrchestrator` Argumente erlauben, verwenden Sie `tridentctl` Um benutzerdefinierte YAML-Manifeste zu erzeugen, die bei Bedarf geändert werden sollen.

Allgemeines zu Controller-Pods und Node-Pods

Astra Trident wird als einzelner Controller-Pod ausgeführt sowie als Node-Pod auf jedem Worker-Node im Cluster. Der Node Pod muss auf jedem Host ausgeführt werden, auf dem Sie ein Astra Trident Volume mounten möchten.

Kubernetes "Knotenauswahl" Und "Toleranzen und Verfleckungen" Werden verwendet, um die Ausführung eines Pod auf einem bestimmten oder bevorzugten Node einzuschränken. Verwenden von`ControllerPlugin` und `NodePlugin`, Sie können Bedingungen und Überschreibungen festlegen.

- Das Controller-Plug-in übernimmt Volume-Bereitstellung und -Management, beispielsweise Snapshots und Größenanpassungen.
- Das Node-Plug-in verarbeitet das Verbinden des Speichers mit dem Node.

Konfigurationsoptionen

> `spec.namespace` Ist in angegeben `TridentOrchestrator` Um den Namespace zu kennzeichnen, in dem Astra Trident installiert ist. Dieser Parameter **kann nicht aktualisiert werden, nachdem Astra Trident installiert wurde**. Der Versuch, dies zu tun, bewirkt das `TridentOrchestrator` Status zu ändern in `Failed`. Astra Trident ist nicht für die Migration auf Namespaces vorgesehen.

Diese Tabelle enthält Einzelheiten `TridentOrchestrator` Attribute.

Parameter	Beschreibung	Standard
`namespace`	Namespace für die Installation von Astra Trident in	„Standard“
`debug`	Aktivieren Sie das Debugging für Astra Trident	Falsch
`windows`	Einstellung auf `true` Ermöglicht die Installation auf Windows Worker-Knoten.	Falsch
`IPv6`	Installieren Sie Astra Trident über IPv6	Falsch
`k8sTimeout`	Zeitüberschreitung für Kubernetes-Betrieb	30 Sek.
`silenceAutosupport`	Schicken Sie AutoSupport Bundles nicht automatisch an NetApp	Falsch
`enableNodePrep`	Automatische Verwaltung der Abhängigkeiten von Workers Node (**BETA**)	Falsch
`autosupportImage`	Das Container-Image für AutoSupport Telemetrie	„netapp/Trident-AutoSupport:23.01“
`autosupportProxy`	Die Adresse/der Port eines Proxys zum Senden von AutoSupport Telemetrie	"http://proxy.example.com:8888"
`uninstall`	Eine Flagge, die zum Deinstallieren von Astra Trident verwendet wird	Falsch
`logFormat`	Astra Trident Protokollformat zur Verwendung [Text, json]	„Text“
`tridentImage`	Astra Trident-Image zu installieren	„netapp/Trident:21.04“
`imageRegistry`	Pfad zur internen Registrierung des Formats `<registry FQDN>[:port][/subpath]`	„K8s.gcr.io/sig-Speicherung (k8s 1.19+) oder quay.io/k8scsi“
`kubeletDir`	Pfad zum kubelet-Verzeichnis auf dem Host	„/var/lib/kubelet“
`wipeout`	Eine Liste mit zu löschenden Ressourcen, um Astra Trident vollständig zu entfernen	
`imagePullSecrets`	Secrets, um Bilder aus einer internen Registrierung zu ziehen	

Parameter	Beschreibung	Standard
`imagePullPolicy`	Legt die BildPull-Richtlinie für den Trident-Operator fest. Gültige Werte sind: `Always` Um immer das Bild zu ziehen. `IfNotPresent` Nur wenn das Image nicht auf dem Node vorhanden ist, soll das Image kopiert werden. `Never` Nie das Bild ziehen.	`IfNotPresent`
`controllerPluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Entspricht dem gleichen Format wie pod.spec.nodeSelector.	Kein Standard; optional
`controllerPluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Entspricht dem gleichen Format wie pod.spec.tolerations.	Kein Standard; optional
`nodePluginNodeSelector`	Zusätzliche Node-Auswahl für Pods Entspricht dem gleichen Format wie pod.spec.nodeSelector.	Kein Standard; optional
`nodePluginTolerations`	Überschreibt Kubernetes-Toleranzen für Pods. Entspricht dem gleichen Format wie pod.spec.tolerations.	Kein Standard; optional

Weitere Informationen zum Formatieren von Pod-Parametern finden Sie unter "Pods werden Nodes zugewiesen".

Beispielkonfigurationen

Sie können die oben genannten Attribute beim Definieren verwenden `TridentOrchestrator` Um die Installation anzupassen.

Beispiel 1: Grundlegende benutzerdefinierte Konfiguration

Dies ist ein Beispiel für eine benutzerdefinierte Grundkonfiguration.

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

Beispiel 2: Implementierung mit Node-Auswahl

Dieses Beispiel veranschaulicht die Implementierung von Trident mit Node-Selektoren:

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

Beispiel 3: Bereitstellung auf Windows Worker-Nodes

Dieses Beispiel zeigt die Bereitstellung auf einem Windows Worker-Knoten.

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

Installieren Sie mit tridentctl

Installieren Sie mit tridentctl

Sie können Astra Trident mithilfe von installieren `tridentctl`. Dieser Prozess gilt für Installationen, bei denen die von Astra Trident benötigten Container-Images entweder in einer privaten Registrierung gespeichert werden oder nicht. Um Ihre anzupassen `tridentctl` Die Bereitstellung finden Sie unter "Tridentctl-Implementierung anpassen".

Entscheidende Informationen zu Astra Trident 23.01

Sie müssen die folgenden wichtigen Informationen über Astra Trident lesen.

** Informationen über Astra TriperelT **

- Kubernetes 1.26 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Astra Trident setzt die Verwendung von Multipathing-Konfiguration in SAN-Umgebungen strikt um und empfiehlt den Nutzen von `find_multipaths: no` In Multipath.conf Datei.

  Verwendung einer Konfiguration ohne Multipathing oder Verwendung von `find_multipaths: yes` Oder `find_multipaths: smart` Der Wert in der Multipath.conf-Datei führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` Seit der Version 21.07.

Installieren Sie Astra Trident mit `tridentctl`

Prüfen "Die Übersicht über die Installation" Um sicherzustellen, dass Sie die Installationsvoraussetzungen erfüllt haben, und die richtige Installationsoption für Ihre Umgebung ausgewählt haben.

Bevor Sie beginnen

Melden Sie sich vor der Installation beim Linux-Host an, und überprüfen Sie, ob er einen funktionierenden "Unterstützter Kubernetes-Cluster" Und dass Sie die erforderlichen Berechtigungen haben.

Mit OpenShift, verwenden `oc` Statt `kubectl` In allen folgenden Beispielen, und melden Sie sich als **System:admin** zuerst mit dem Ausführen an `oc login -u system:admin` Oder `oc login -u kube-admin`.

1. Überprüfen Sie Ihre Kubernetes Version:

```
kubectl version
```

2. Überprüfung der Berechtigungen für Cluster-Administratoren:

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

3. Überprüfen Sie, ob Sie einen Pod starten können, der ein Image aus dem Docker Hub verwendet, und ob er das Storage-System über das POD-Netzwerk erreichen kann:

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

Schritt 1: Laden Sie das Trident Installer-Paket herunter

Das Installationspaket von Astra Trident erstellt einen Trident Pod, konfiguriert die CRD-Objekte, die zur Aufrechterhaltung des Zustands verwendet werden, und initialisiert die CSI-Sidecars, um Aktionen wie die Bereitstellung und das Anschließen von Volumes an Cluster-Hosts durchzuführen. Laden Sie die neueste Version des Trident Installationsprogramms herunter und extrahieren Sie sie aus "Die Sektion *Assets* auf GitHub". Aktualisieren Sie *<trident-installer-XX.XX.X.tar.gz>* im Beispiel mit Ihrer ausgewählten Astra Trident Version.

```
wget https://github.com/NetApp/trident/releases/download/v23.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
```

Schritt: Installieren Sie Astra Trident

Installieren Sie Astra Trident im gewünschten Namespace, indem Sie den ausführen `tridentctl install` Befehl. Sie können weitere Argumente hinzufügen, um den Speicherort der Bildregistrierung anzugeben.

Damit Astra Trident auf Windows-Knoten ausgeführt werden kann, fügen Sie die hinzu `--windows` Flag auf den Installationsbefehl: `$ ./tridentctl install --windows -n trident`.

Standardmodus

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

Bilder in einer Registrierung

```
./tridentctl install -n trident --image-registry <your-registry>
--autosupport-image <your-registry>/trident-autosupport:23.01 --trident
-image <your-registry>/trident:23.01.1
```

Bilder in verschiedenen Registern

Sie müssen anhängen `sig-storage` Bis zum `imageRegistry` Um unterschiedliche Registrierungsstandorte zu verwenden.

```
./tridentctl install -n trident --image-registry <your-registry>/sig-
storage --autosupport-image <your-registry>/netapp/trident-
autosupport:23.01 --trident-image <your-
registry>/netapp/trident:23.01.1
```

Ihr Installationsstatus sollte so aussehen.

```
....
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=23.01.1
INFO Trident installation succeeded.
....
```

Überprüfen Sie die Installation

Sie können Ihre Installation mithilfe des POD-Erstellungsstatus oder überprüfen `tridentctl`.

Den Status der Pod-Erstellung verwenden

Überprüfen Sie den Status der erstellten Pods, ob die Astra Trident-Installation abgeschlossen wurde:

```
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
```

Wenn das Installationsprogramm nicht erfolgreich abgeschlossen wurde, oder `trident-controller-<generated id>` (`trident-csi-<generated id>` In Versionen vor 23.01) hat keinen **laufenden** Status, die Plattform wurde nicht installiert. Nutzung `-d` Bis "Aktivieren Sie den Debug-Modus" Und das Problem beheben.

Wird Verwendet `tridentctl`

Verwenden Sie können `tridentctl` Um die installierte Version von Astra Trident zu überprüfen.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.01.1        | 23.01.1        |
+----------------+----------------+
```

Wie es weiter geht

Das ist jetzt möglich "Erstellen Sie ein Back-End und eine Storage-Klasse, stellen Sie ein Volume bereit und mounten Sie das Volume in einem POD".

Die tridentctl-Installation anpassen

Mit dem Astra Trident Installer können Sie die Installation anpassen.

Erfahren Sie mehr über das Installationsprogramm

Mit dem Astra Trident Installer können Sie Attribute anpassen. Wenn Sie beispielsweise das Trident-Image in ein privates Repository kopiert haben, können Sie den Bildnamen mithilfe von angeben `--trident-image`. Wenn Sie das Trident-Image sowie die erforderlichen CSI-Sidecar-Images in ein privates Repository kopiert haben, ist es möglicherweise besser, den Speicherort des Repository mithilfe von anzugeben `--image-registry` Schalter, der die Form nimmt `<registry FQDN>[:port]`.

Wenn Sie eine Distribution von Kubernetes verwenden, wo `kubelet` Speichert seine Daten auf einem anderen Pfad als den üblichen `/var/lib/kubelet`, Sie können den alternativen Pfad mit angeben `--kubelet-dir`.

Wenn Sie die Installation anpassen müssen, die über die Argumente des Installers hinausgeht, können Sie auch die Bereitstellungsdateien anpassen. Verwenden der `--generate-custom-yaml` Der Parameter erstellt die folgenden YAML-Dateien im Installationsprogramm `setup` Verzeichnis:

- `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`

Nachdem Sie diese Dateien erstellt haben, können Sie sie nach Ihren Bedürfnissen ändern und dann verwenden `--use-custom-yaml` Um Ihre benutzerdefinierte Bereitstellung zu installieren.

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

Was kommt als Nächstes?

Nach der Installation von Astra Trident können Sie mit dem Erstellen eines Backends fortfahren, eine Storage-Klasse erstellen, ein Volume bereitstellen und das Volume in einem Pod mounten.

Schritt 1: Erstellen Sie ein Backend

Jetzt können Sie mit Astra Trident ein Backend erstellen und Volumes bereitstellen. Erstellen Sie dazu ein `backend.json` Datei, die die erforderlichen Parameter enthält. Beispiele für Konfigurationsdateien für verschiedene Backend-Typen finden Sie im `sample-input` Verzeichnis.

Siehe "Hier" Weitere Informationen zum Konfigurieren der Datei für den Backend-Typ.

```
cp sample-input/<backend template>.json backend.json
vi backend.json
```

```
./tridentctl -n trident create backend -f backend.json
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+
```

Wenn die Erstellung fehlschlägt, ist mit der Back-End-Konfiguration ein Fehler aufgetreten. Sie können die Protokolle zur Bestimmung der Ursache anzeigen, indem Sie den folgenden Befehl ausführen:

```
./tridentctl -n trident logs
```

Nachdem Sie das Problem behoben haben, gehen Sie einfach zurück zum Anfang dieses Schritts und versuchen Sie es erneut. Weitere Tipps zur Fehlerbehebung finden Sie unter "Die Fehlerbehebung" Abschnitt.

Schritt 2: Erstellen Sie eine Storage-Klasse

Kubernetes Benutzer stellen Volumes mithilfe von persistenten Volume Claims (PVCs) bereit, die einen angeben "Storage-Klasse" Nach Name. Die Details sind für die Benutzer verborgen. In einer Storage-Klasse wird jedoch die provisionierung für die jeweilige Klasse (in diesem Fall Trident) angegeben, und die Bedeutung dieser Klasse für die provisionierung angegeben.

Kubernetes-Benutzer in Storage-Klasse erstellen geben an, wann sie ein Volume möchten. Die Konfiguration der Klasse muss das im vorherigen Schritt erstellte Backend modellieren, damit Astra Trident neue Volumes bereitstellen wird.

Die einfachste Storage-Klasse, mit der Sie beginnen können, basiert auf der `sample-input/storage-class-csi.yaml.templ` Datei, die mit dem Installationsprogramm geliefert wird, ersetzen *`BACKEND_TYPE`* Mit dem Namen des Speichertreibers.

```
./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
```

Dies ist ein Kubernetes-Objekt, die Storage-Verwendung ist `kubectl` Um sie in Kubernetes zu erstellen.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

Sie sollten jetzt in Kubernetes und Astra Trident eine **Basis-csi** Storage-Klasse sehen, und Astra Trident hätte die Pools auf dem Backend entdeckt haben sollen.

```
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"
        ]
      }
    }
  ]
}
```

Schritt 3: Stellen Sie Ihr erstes Volumen bereit

Nun können Sie das erste Volume dynamisch bereitstellen. Dazu wird ein Kubernetes erstellt "Persistent Volume Claim" (PVC) Objekt.

Erstellen Sie eine PVC für ein Volume, das die soeben erstellte Storage-Klasse verwendet.

Siehe `sample-input/pvc-basic-csi.yaml` Beispiel: Stellen Sie sicher, dass der Name der Speicherklasse mit dem übereinstimmt, den Sie erstellt haben.

```
kubectl create -f sample-input/pvc-basic-csi.yaml

kubectl get pvc --watch
NAME      STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS   AGE
basic     Pending
basic          1s
basic     Pending   pvc-3acb0d1c-b1ae-11e9-8d9f-5254004dfdb7   0
basic          5s
basic     Bound     pvc-3acb0d1c-b1ae-11e9-8d9f-5254004dfdb7   1Gi
RWO           basic          7s
```

Schritt 4: Mounten Sie die Volumes in einem POD

Nun lassen Sie uns den Datenträger einhängen. Wir werden einen nginx-Pod starten, der das PV unter einhängt `/usr/share/nginx/html`.

```
cat << EOF > task-pv-pod.yaml
kind: Pod
apiVersion: v1
metadata:
  name: task-pv-pod
spec:
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: task-pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
EOF
kubectl create -f task-pv-pod.yaml
```

```
# Wait for the pod to start
kubectl get pod --watch

# Verify that the volume is mounted on /usr/share/nginx/html
kubectl exec -it task-pv-pod -- df -h /usr/share/nginx/html

# Delete the pod
kubectl delete pod task-pv-pod
```

An diesem Punkt existiert der POD (Applikation) nicht mehr, das Volume ist jedoch weiterhin vorhanden. Sie können es von einem anderen POD nutzen, wenn Sie dies möchten.

Löschen Sie zum Löschen des Volumes die Forderung:

```
kubectl delete pvc basic
```

Sie können jetzt zusätzliche Aufgaben ausführen, wie z. B.:

- "Konfigurieren Sie zusätzliche Back-Ends."
- "Erstellen Sie zusätzliche Speicherklassen."