Realice la instalación mediante el operador Trident

Astra Trident

NetApp
March 05, 2026

This PDF was generated from https://docs.netapp.com/es-es/trident-2301/trident-get-started/kubernetes-deploy-operator.html on March 05, 2026. Always check docs.netapp.com for the latest.

Tabla de contenidos

Realice la instalación mediante el operador Trident . . . 1
Implemente manualmente el operador de Trident (modo estándar) . . . 1
Información crucial sobre Astra Trident 23.01 . . . 1
Implemente manualmente el operador de Trident e instale Trident . . . 1
Compruebe la instalación . . . 4
El futuro . . . 6
Implemente manualmente el operador Trident (modo sin conexión). . . . 6
Información crucial sobre Astra Trident 23.01 . . . 6
Implemente manualmente el operador de Trident e instale Trident . . . 6
Compruebe la instalación . . . 11
El futuro . . . 12
Puesta en marcha del operador de Trident con Helm (modo estándar) . . . 12
Información crucial sobre Astra Trident 23.01 . . . 12
Ponga en marcha el operador de Trident e instale Astra Trident con Helm . . . 13
Pasar los datos de configuración durante la instalación . . . 13
Opciones de configuración . . . 14
El futuro . . . 17
Implementar el operador de Trident con Helm (modo sin conexión) . . . 17
Información crucial sobre Astra Trident 23.01 . . . 17
Ponga en marcha el operador de Trident e instale Astra Trident con Helm . . . 17
Pasar los datos de configuración durante la instalación . . . 18
Opciones de configuración . . . 19
El futuro . . . 21
Personalice la instalación del operador de Trident . . . 22
Descripción de los pods de la controladora y los pods de nodo . . . 22
Opciones de configuración . . . 22
Configuraciones de ejemplo . . . 24

Realice la instalación mediante el operador Trident

Implemente manualmente el operador de Trident (modo estándar)

Es posible poner en marcha manualmente el operador de Trident para instalar Astra Trident. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident no se almacenan en un registro privado. Si dispone de un registro de imágenes privado, utilice "proceso de puesta en marcha sin conexión".

Información crucial sobre Astra Trident 23.01

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1.26 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Implemente manualmente el operador de Trident e instale Trident

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Antes de iniciar la instalación, inicie sesión en el host Linux y compruebe que esté gestionando un trabajo, "Clúster de Kubernetes compatible" y que tenga los privilegios necesarios.

Con OpenShift, utilícelo `oc` en lugar de `kubectl` en todos los ejemplos que siguen, e inicie sesión como **system:admin** primero ejecutando `oc login -u system:admin` o. `oc login -u kube-admin`.

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

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

3. Compruebe que puede iniciar un pod que utilice una imagen de Docker Hub para llegar al sistema de almacenamiento a través de la red de pod:

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

Paso 1: Descargue el paquete de instalación de Trident

El paquete de instalación de Astra Trident incluye todo lo necesario para poner en marcha al operador de Trident e instalar Astra Trident. Descargue y extraiga la versión más reciente del instalador de Trident "La sección *Assets* de 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
```

Paso 2: Cree la `TridentOrchestrator` CRD

Cree el `TridentOrchestrator` Definición de recurso personalizado (CRD). Creará una `TridentOrchestrator` Recursos personalizados más adelante. Use la versión adecuada de CRD YAML en `deploy/crds` para crear la `TridentOrchestrator` CRD.

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

Paso 3: Ponga en marcha el operador de Trident

El instalador de Astra Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo de paquete es una forma sencilla de poner en marcha el operador e instalar Astra Trident con una configuración predeterminada.

- Para los clústeres que ejecutan Kubernetes 1.24 o inferior, utilice `bundle_pre_1_25.yaml`.

- Utilice el para clústeres que ejecuten Kubernetes 1.25 o superior `bundle_post_1_25.yaml`.

El instalador de Trident pone en marcha el operador en la `trident` espacio de nombres. Si la `trident` no existe el espacio de nombres, utilice `kubectl apply -f deploy/namespace.yaml` para crearlo.

Pasos

1. Crear los recursos e implementar el operador:

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

> Para implementar el operador en un espacio de nombres distinto del `trident` espacio de nombres, actualización `serviceaccount.yaml`, `clusterrolebinding.yaml` y.. `operator.yaml` y genere el archivo del paquete con el `kustomization.yaml`:
>
> ```
> kubectl kustomize deploy/ > deploy/<bundle>.yaml
> ```

2. Compruebe que el operador se ha desplegado.

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

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

> Solo debe haber **una instancia** del operador en un clúster de Kubernetes. No cree varias implementaciones del operador Trident.

Paso 4: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear el `TridentOrchestrator` E instale Astra Trident. Opcionalmente, puede hacerlo "Personalice su instalación de Trident" uso de los atributos de la `TridentOrchestrator` espec.

```
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
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

El estado de `TridentOrchestrator` Indica si la instalación se realizó correctamente y muestra la versión de Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperar por sí solo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este método `TridentOrchestrator` CR.
Instalado	Astra Trident se ha instalado correctamente.
Desinstalando	El operador está desinstalando Astra Trident, porque `spec.uninstall=true`.
Desinstalado	Astra Trident se desinstala.
Error	El operador no pudo instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará recuperarse automáticamente de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	La `TridentOrchestrator` no se utiliza. Otro ya existe.

Uso del estado de creación de pod

Para confirmar si la instalación de Astra Trident ha finalizado, revise el estado de los pods creados:

```
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
```

Uso `tridentctl`

Puede utilizar `tridentctl` Para comprobar la versión de Astra Trident instalada.

```
./tridentctl -n trident version

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

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Implemente manualmente el operador Trident (modo sin conexión).

Es posible poner en marcha manualmente el operador de Trident para instalar Astra Trident. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident se almacenan en un registro privado. Si no dispone de un registro de imágenes privado, utilice "proceso de implementación estándar".

Información crucial sobre Astra Trident 23.01

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1.26 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Implemente manualmente el operador de Trident e instale Trident

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Inicie sesión en el host Linux y compruebe que está gestionando un funcionamiento y. "Clúster de Kubernetes

compatible" y que tenga los privilegios necesarios.

> Con OpenShift, utilícelo `oc` en lugar de `kubectl` en todos los ejemplos que siguen, e inicie sesión como **system:admin** primero ejecutando `oc login -u system:admin` o. `oc login -u kube-admin`.

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

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

3. Compruebe que puede iniciar un pod que utilice una imagen de Docker Hub para llegar al sistema de almacenamiento a través de la red de pod:

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

Paso 1: Descargue el paquete de instalación de Trident

El paquete de instalación de Astra Trident incluye todo lo necesario para poner en marcha al operador de Trident e instalar Astra Trident. Descargue y extraiga la versión más reciente del instalador de Trident "La sección *Assets* de 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
```

Paso 2: Cree la `TridentOrchestrator` CRD

Cree el `TridentOrchestrator` Definición de recurso personalizado (CRD). Creará una `TridentOrchestrator` Recursos personalizados más adelante. Use la versión adecuada de CRD YAML en `deploy/crds` para crear la `TridentOrchestrator` CRD:

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

Paso 3: Actualice la ubicación del registro en el operador

Pulg `/deploy/operator.yaml`, actualizar `image: docker.io/netapp/trident-operator:23.01.1` para reflejar la ubicación del registro de imágenes. Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Por ejemplo:

- `image: <your-registry>/trident-operator:23.01.1` si todas las imágenes están ubicadas en un registro.
- `image: <your-registry>/netapp/trident-operator:23.01.1` Si su imagen Trident se encuentra en un registro diferente de sus imágenes CSI.

Paso 4: Ponga en marcha el operador de Trident

El instalador de Trident pone en marcha el operador en la `trident` espacio de nombres. Si la `trident` no existe el espacio de nombres, utilice `kubectl apply -f deploy/namespace.yaml` para crearlo.

Para implementar el operador en un espacio de nombres distinto del `trident` espacio de nombres, actualización `serviceaccount.yaml`, `clusterrolebinding.yaml` y.. `operator.yaml` antes de desplegar el operador.

1. Crear los recursos e implementar el operador:

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

> El instalador de Astra Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo de paquete es una forma sencilla de poner en marcha el operador e instalar Astra Trident con una configuración predeterminada.
>
> - Para los clústeres que ejecutan Kubernetes 1.24 o inferior, utilice `bundle_pre_1_25.yaml`.
> - Utilice el para clústeres que ejecuten Kubernetes 1.25 o superior `bundle_post_1_25.yaml`.

2. Compruebe que el operador se ha desplegado.

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

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

> Solo debe haber **una instancia** del operador en un clúster de Kubernetes. No cree varias implementaciones del operador Trident.

Paso 5: Actualice la ubicación del registro de imágenes en el `TridentOrchestrator`

Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Actualizar `deploy/crds/tridentorchestrator_cr.yaml` para agregar las especificaciones de ubicación adicionales basadas en la configuración de su registro.

Imágenes en un registro

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

Imágenes en diferentes registros

Debe añadir `sig-storage` para la `imageRegistry` para usar diferentes ubicaciones de registro.

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

Paso 6: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear el `TridentOrchestrator` E instale Astra Trident. Si lo desea, puede ir más allá "Personalice su instalación de Trident" uso de los atributos de la `TridentOrchestrator` espec. En el siguiente ejemplo se muestra una instalación donde las imágenes Trident y CSI se encuentran en diferentes registros.

```
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
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

El estado de `TridentOrchestrator` Indica si la instalación se realizó correctamente y muestra la versión de Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperar por sí solo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este método `TridentOrchestrator` CR.
Instalado	Astra Trident se ha instalado correctamente.
Desinstalando	El operador está desinstalando Astra Trident, porque `spec.uninstall=true`.
Desinstalado	Astra Trident se desinstala.
Error	El operador no pudo instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará recuperarse automáticamente de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	La `TridentOrchestrator` no se utiliza. Otro ya existe.

Uso del estado de creación de pod

Para confirmar si la instalación de Astra Trident ha finalizado, revise el estado de los pods creados:

```
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
```

Uso `tridentctl`

Puede utilizar `tridentctl` Para comprobar la versión de Astra Trident instalada.

```
./tridentctl -n trident version

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

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Puesta en marcha del operador de Trident con Helm (modo estándar)

Puede poner en marcha el operador de Trident e instalar Astra Trident con Helm. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident no se almacenan en un registro privado. Si dispone de un registro de imágenes privado, utilice "proceso de puesta en marcha sin conexión".

Información crucial sobre Astra Trident 23.01

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1.26 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Ponga en marcha el operador de Trident e instale Astra Trident con Helm

Usar Trident "Carta del timón" Es posible poner en marcha el operador de Trident e instalar Trident en un paso.

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Además de la "requisitos previos a la implementación" que necesita "Versión timón 3".

Pasos

1. Añada el repositorio de Astra Trident Helm:

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

2. Uso `helm install` y especifique un nombre para la implementación como en el ejemplo siguiente donde `23.01.1` Es la versión de Astra Trident que está instalando.

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

> Si ya creó un espacio de nombres para Trident, el `--create-namespace` el parámetro no creará un espacio de nombres adicional.

Puede utilizar `helm list` para revisar detalles de la instalación como nombre, espacio de nombres, gráfico, estado, versión de la aplicación, y el número de revisión.

Pasar los datos de configuración durante la instalación

Existen dos formas de pasar los datos de configuración durante la instalación:

Opción	Descripción
`--values` (o. `-f`)	Especifique un archivo YAML con anulaciones. Esto se puede especificar varias veces y el archivo de la derecha tendrá prioridad.
`--set`	Especifique anulaciones en la línea de comandos.

Por ejemplo, para cambiar el valor predeterminado de `debug`, ejecute lo siguiente `--set` comando donde `23.01.1` Es la versión de Astra Trident que está instalando:

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

Opciones de configuración

Esta tabla y la `values.yaml` File, que forma parte del gráfico Helm, proporciona la lista de claves y sus valores predeterminados.

Opción	Descripción	Predeterminado
`nodeSelector`	Etiquetas de nodo para la asignación de pod	
`podAnnotations`	Anotaciones del pod	
`deploymentAnnotations`	Anotaciones de despliegue	
`tolerations`	Toleraciones para la asignación de POD	
`affinity`	Afinidad para la asignación de pod	
`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	

Opción	Descripción	Predeterminado
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`imageRegistry`	Identifica el registro del `trident-operator`, `trident`, y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	"/var/lib/kubelet"
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	""
`tridentIPv6`	Permite permitir que Astra Trident funcione en clústeres de IPv6.	`false`
`tridentK8sTimeout`	Anula el tiempo de espera predeterminado de 30 segundos para la mayoría de las operaciones de la API de Kubernetes (si no es cero, en segundos).	`0`
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Astra Trident.	`false`

Opción	Descripción	Predeterminado
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen del contenedor AutoSupport de Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permite que el contenedor Astra Trident AutoSupport telefonee a casa a través de un proxy HTTP.	""
`tridentLogFormat`	Establece el formato de registro de Astra Trident (`text` o. `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registro de auditorías de Astra Trident.	`true`
`tridentLogLevel`	Permite establecer el nivel de registro de Astra Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`tridentDebug`	Permite establecer el nivel de registro de Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo específicos de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentLogLayers`	Permite habilitar capas específicas de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentImage`	Permite anular por completo la imagen de Astra Trident.	""
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Astra Trident.	""
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	""
`windows`	Permite instalar Astra Trident en el nodo de trabajo de Windows.	`false`
`enableForceDetach`	Permite habilitar la función Forzar separación.	`false`
`excludePodSecurityPolicy`	Excluye la política de seguridad del pod del operador de la creación.	`false`

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Implementar el operador de Trident con Helm (modo sin conexión)

Puede poner en marcha el operador de Trident e instalar Astra Trident con Helm. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident se almacenan en un registro privado. Si no dispone de un registro de imágenes privado, utilice "proceso de implementación estándar".

Información crucial sobre Astra Trident 23.01

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1.26 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Ponga en marcha el operador de Trident e instale Astra Trident con Helm

Usar Trident "Carta del timón" Es posible poner en marcha el operador de Trident e instalar Trident en un paso.

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Además de la "requisitos previos a la implementación" que necesita "Versión timón 3".

Pasos

1. Añada el repositorio de Astra Trident Helm:

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

2. Uso `helm install` y especifique un nombre para su implementación y ubicación del registro de imágenes. Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. En los ejemplos: `23.01.1` Es la versión de Astra Trident que está instalando.

Imágenes en un registro

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

Imágenes en diferentes registros

Debe añadir `sig-storage` para la `imageRegistry` para usar diferentes ubicaciones de registro.

```
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>
```

Si ya creó un espacio de nombres para Trident, el `--create-namespace` el parámetro no creará un espacio de nombres adicional.

Puede utilizar `helm list` para revisar detalles de la instalación como nombre, espacio de nombres, gráfico, estado, versión de la aplicación, y el número de revisión.

Pasar los datos de configuración durante la instalación

Existen dos formas de pasar los datos de configuración durante la instalación:

Opción	Descripción
`--values` (o. `-f`)	Especifique un archivo YAML con anulaciones. Esto se puede especificar varias veces y el archivo de la derecha tendrá prioridad.
`--set`	Especifique anulaciones en la línea de comandos.

Por ejemplo, para cambiar el valor predeterminado de `debug`, ejecute lo siguiente `--set` comando donde

`23.01.1` Es la versión de Astra Trident que está instalando:

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

Opciones de configuración

Esta tabla y la `values.yaml` File, que forma parte del gráfico Helm, proporciona la lista de claves y sus valores predeterminados.

Opción	Descripción	Predeterminado
`nodeSelector`	Etiquetas de nodo para la asignación de pod	
`podAnnotations`	Anotaciones del pod	
`deploymentAnnotations`	Anotaciones de despliegue	
`tolerations`	Toleraciones para la asignación de POD	
`affinity`	Afinidad para la asignación de pod	
`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`imageRegistry`	Identifica el registro del `trident-operator`, `trident`, y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`

Opción	Descripción	Predeterminado
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	"/var/lib/kubelet"
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	""
`tridentIPv6`	Permite permitir que Astra Trident funcione en clústeres de IPv6.	`false`
`tridentK8sTimeout`	Anula el tiempo de espera predeterminado de 30 segundos para la mayoría de las operaciones de la API de Kubernetes (si no es cero, en segundos).	`0`
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Astra Trident.	`false`
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen del contenedor AutoSupport de Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permite que el contenedor Astra Trident AutoSupport telefonee a casa a través de un proxy HTTP.	""
`tridentLogFormat`	Establece el formato de registro de Astra Trident (`text` o. `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registro de auditorías de Astra Trident.	`true`

Opción	Descripción	Predeterminado
`tridentLogLevel`	Permite establecer el nivel de registro de Astra Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`tridentDebug`	Permite establecer el nivel de registro de Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo específicos de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentLogLayers`	Permite habilitar capas específicas de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentImage`	Permite anular por completo la imagen de Astra Trident.	""
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Astra Trident.	""
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	""
`windows`	Permite instalar Astra Trident en el nodo de trabajo de Windows.	`false`
`enableForceDetach`	Permite habilitar la función Forzar separación.	`false`
`excludePodSecurityPolicy`	Excluye la política de seguridad del pod del operador de la creación.	`false`

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen

en un pod".

Personalice la instalación del operador de Trident

El operador Trident le permite personalizar la instalación de Astra Trident con los atributos del `TridentOrchestrator` espec. Si desea personalizar la instalación más allá de qué `TridentOrchestrator` los argumentos lo permiten, considere usar `tridentctl` Para generar manifiestos YAML personalizados y modificarlos según sea necesario.

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

Opciones de configuración

`spec.namespace` se especifica en `TridentOrchestrator` Para indicar el espacio de nombres en el que está instalado Astra Trident. Este parámetro **no se puede actualizar después de instalar Astra Trident**. Al intentar hacerlo, se genera el `TridentOrchestrator` estado a cambiar a. `Failed`. Astra Trident no está pensado para la migración entre espacios de nombres.

Esta tabla detalla `TridentOrchestrator` atributos.

Parámetro	Descripción	Predeterminado
`namespace`	Espacio de nombres para instalar Astra Trident en	"predeterminado"
`debug`	Habilite la depuración para Astra Trident	falso
`windows`	Ajuste a. `true` Permite la instalación en nodos de trabajo de Windows.	falso
`IPv6`	Instale Astra Trident sobre IPv6	falso
`k8sTimeout`	Tiempo de espera para las operaciones de Kubernetes	30 seg
`silenceAutosupport`	No envíe paquetes AutoSupport a NetApp automáticamente	falso

Parámetro	Descripción	Predeterminado
`enableNodePrep`	Administrar automáticamente las dependencias del nodo de trabajo (**BETA**)	falso
`autosupportImage`	La imagen contenedora para telemetría AutoSupport	"netapp/trident-autosupport:23.01"
`autosupportProxy`	La dirección/puerto de un proxy para enviar telemetría AutoSupport	"http://proxy.example.com:8888"
`uninstall`	Una Marca utilizada para desinstalar Astra Trident	falso
`logFormat`	Formato de registro de Astra Trident para utilizar [text,json]	"texto"
`tridentImage`	Imagen de Astra Trident para instalar	"netapp/trident:21.04"
`imageRegistry`	Ruta de acceso al registro interno, del formato `<registry FQDN>[:port][/subpath]`	"k8s.gcr.io/sig-storage (k8s 1.19+) o quay.io/k8scsi"
`kubeletDir`	Ruta al directorio kubelet del host	“/var/lib/kubelet”
`wipeout`	Una lista de recursos para eliminar y realizar una eliminación completa de Astra Trident	
`imagePullSecrets`	Secretos para extraer imágenes de un registro interno	
`imagePullPolicy`	Establece la política de extracción de imágenes para el operador Trident. Valores válidos: `Always` para tirar siempre de la imagen. `IfNotPresent` para extraer la imagen solo si aún no existe en el nodo. `Never` para no tirar nunca de la imagen.	`IfNotPresent`
`controllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que pod.spec.nodeSelector.	Sin valores predeterminados; opcional
`controllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que el de pod.spec.tolerancias.	Sin valores predeterminados; opcional

Parámetro	Descripción	Predeterminado
`nodePluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que pod.spec.nodeSelector.	Sin valores predeterminados; opcional
`nodePluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que el de pod.spec.tolerancias.	Sin valores predeterminados; opcional

Para obtener más información sobre el formato de los parámetros del POD, consulte "Asignación de pods a nodos".

Configuraciones de ejemplo

Puede utilizar los atributos mencionados anteriormente al definir `TridentOrchestrator` para personalizar la instalación.

Ejemplo 1: Configuración personalizada básica

Este es un ejemplo de una configuración personalizada básica.

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

Ejemplo 2: Implementar con selectores de nodos

Este ejemplo ilustra cómo se puede implementar Trident con los selectores de nodos:

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

Ejemplo 3: Implementar en nodos de trabajo de Windows

Este ejemplo ilustra la implementación en un nodo de trabajo de Windows.

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

Información de copyright

Copyright © 2026 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en http://www.netapp.com/TM son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.