Realice la instalación mediante el operador Trident

Astra Trident

NetApp
June 28, 2024

Tabla de contenidos

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 vital sobre Astra Trident 24,02

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

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

- Kubernetes 1,27 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/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.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 una versión anterior, utilice `bundle_pre_1_25.yaml`.

- Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice `bundle_post_1_25.yaml`.

Antes de empezar

- De forma predeterminada, el instalador de Trident implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

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

  a. Cree el `kustomization.yaml` con el siguiente comando donde está *<bundle.yaml>* `bundle_pre_1_25.yaml` o. `bundle_post_1_25.yaml` Según su versión de Kubernetes.

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

  b. Compile el paquete con el siguiente comando donde está *<bundle.yaml>* `bundle_pre_1_25.yaml` o. `bundle_post_1_25.yaml` Según su versión de Kubernetes.

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

Pasos

1. Crear los recursos e implementar el operador:

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

2. Compruebe que se han creado el operador, el despliegue y los replicasets.

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

> 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:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:24.02.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

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 ha podido instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará automáticamente recuperarse 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 |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

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 vital sobre Astra Trident 24,02

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

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

- Kubernetes 1,27 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/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.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:24.02.0` 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:24.02.0` si todas las imágenes están ubicadas en un registro.

- `image: <your-registry>/netapp/trident-operator:24.02.0` Si su imagen Trident se encuentra en un registro diferente de sus imágenes CSI.

Paso 4: Despliegue el operador 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 una versión anterior, utilice `bundle_pre_1_25.yaml`.
- Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice `bundle_post_1_25.yaml`.

Antes de empezar

- De forma predeterminada, el instalador de Trident implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

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

  a. Cree el `kustomization.yaml` con el siguiente comando donde está *<bundle.yaml>* `bundle_pre_1_25.yaml` o. `bundle_post_1_25.yaml` Según su versión de Kubernetes.

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

  b. Compile el paquete con el siguiente comando donde está *<bundle.yaml>* `bundle_pre_1_25.yaml` o. `bundle_post_1_25.yaml` Según su versión de Kubernetes.

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

Pasos

1. Crear los recursos e implementar el operador:

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

2. Compruebe que se han creado el operador, el despliegue y los replicasets.

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

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

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:24.02"
tridentImage: "<your-registry>/netapp/trident:24.02.0"
```

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:24.02
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:24.02.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:24.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                      true
    Http Request Timeout:       90s
    Image Pull Secrets:
    Image Registry:       <your-registry>/sig-storage
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/netapp/trident:24.02.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v24.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

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 ha podido instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará automáticamente recuperarse 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 |
+----------------+----------------+
| 24.02.0        | 24.02.0        |
+----------------+----------------+
```

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 vital sobre Astra Trident 24,02

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

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

- Kubernetes 1,27 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 `100.2402.0` Es la versión de Astra Trident que está instalando.

```
helm install <name> netapp-trident/trident-operator --version 100.2402.0
--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 `100.2402.0` Es la versión de Astra Trident que está instalando:

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

Opción	Descripción	Predeterminado
`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`
`cloudProvider`	Establezca en "`Azure`" Cuando se utilizan identidades gestionadas o una identidad de nube en un clúster de AKS. Establecer en «AWS» cuando se utiliza una identidad de nube en un clúster de EKS.	""
`cloudIdentity`	Defina la identidad de carga de trabajo («azure.workload.identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx») cuando utilice la identidad de cloud en un clúster de AKS. Establezca el rol de AWS IAM ('eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role') cuando utilice la identidad de nube en un clúster de EKS.	""
`iscsiSelfHealingInterval`	El intervalo en el que se invoca la reparación automática de iSCSI.	`5m0s`
`iscsiSelfHealingWaitTime`	La duración después del cual la reparación automática de iSCSI inicia un intento de resolver una sesión obsoleta realizando un cierre de sesión y un inicio de sesión posterior.	`7m0s`

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.

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 vital sobre Astra Trident 24,02

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

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

- Kubernetes 1,27 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: `100.2402.0` Es la versión de Astra Trident que está instalando.

Imágenes en un registro

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

Si 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 `100.2402.0` Es la versión de Astra Trident que está instalando:

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

Opción	Descripción	Predeterminado
`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`
`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`

Opción	Descripción	Predeterminado
`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`

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	`"default"`
`debug`	Habilite la depuración para Astra Trident	`false`
`enableForceDetach`	`ontap-san` y.. `ontap-san-economy` solamente. Funciona con cierre de nodos no controlado (NGN) de Kubernetes para conceder a los administradores de clústeres la capacidad de migrar de forma segura cargas de trabajo con volúmenes montados a nodos nuevos en caso de que un nodo se vuelva en mal estado.	`false`
`windows`	Ajuste a. `true` Permite la instalación en nodos de trabajo de Windows.	`false`
`cloudProvider`	Establezca en `"Azure"` Cuando se utilizan identidades gestionadas o una identidad de nube en un clúster de AKS. Establecer en «AWS» cuando se utiliza una identidad de nube en un clúster de EKS.	`""`
`cloudIdentity`	Defina la identidad de carga de trabajo («azure.workload.identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx») cuando utilice la identidad de cloud en un clúster de AKS. Establezca el rol de AWS IAM ('eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role) cuando utilice la identidad de la nube en un clúster de EKS.	`""`
`IPv6`	Instale Astra Trident sobre IPv6	falso
`k8sTimeout`	Tiempo de espera para las operaciones de Kubernetes	`30sec`
`silenceAutosupport`	No envíe paquetes AutoSupport a NetApp automáticamente	`false`
`autosupportImage`	La imagen contenedora para telemetría AutoSupport	`"netapp/trident-autosupport:24.02"`
`autosupportProxy`	La dirección/puerto de un proxy para enviar AutoSupport Telemetría	`"http://proxy.example.com:8888"`
`uninstall`	Una Marca utilizada para desinstalar Astra Trident	`false`

Parámetro	Descripción	Predeterminado
`logFormat`	Formato de registro de Astra Trident para utilizar [text,json]	`"text"`
`tridentImage`	Imagen de Astra Trident para instalar	`"netapp/trident:24.02"`
`imageRegistry`	Ruta de acceso al registro interno, del formato `<registry FQDN>[:port][/subpath]`	`"k8s.gcr.io/sig-storage"` (Kubernetes 1,19 o posterior) o. `"quay.io/k8scsi"`
`kubeletDir`	Ruta al directorio kubelet del host	`"/var/lib/kubelet"`
`wipeout`	Lista de recursos que se van a suprimir para 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 `pod.spec.Tolerations`.	Sin valores predeterminados; opcional
`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 `pod.spec.Tolerations`.	Sin valores predeterminados; opcional

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

Detalles acerca de forzar separación

Forzar separación está disponible para `ontap-san` y.. `ontap-san-economy` solamente. Antes de habilitar la desconexión forzada, se debe habilitar el cierre de nodos (NGN) no controlado en el clúster de Kubernetes. Para obtener más información, consulte "Kubernetes: Cierre de nodo sin gracia".

Dado que Astra Trident se basa en NGN de Kubernetes, no lo elimine `out-of-service` mantiene un nodo en mal estado hasta que se reprograman todas las cargas de trabajo no tolerables. La aplicación o eliminación imprudente de la contaminación puede poner en peligro la protección de datos de back-end.

Cuando el administrador del clúster de Kubernetes ha aplicado el `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` mancha al nodo y. `enableForceDetach` se establece en `true`, Astra Trident determinará el estado del nodo y:

1. Cese el acceso de I/O back-end para los volúmenes montados en ese nodo.
2. Marque el objeto de nodo de Astra Trident como `dirty` (no es seguro para las nuevas publicaciones).

> La controladora Trident rechazará nuevas solicitudes de volumen de publicación hasta que el nodo se vuelva a calificar (después de haberse marcado como `dirty`) Por el pod del nodo de Trident. No se aceptarán todas las cargas de trabajo programadas con una RVP montada (incluso después de que el nodo del clúster esté en buen estado y listo) hasta que Astra Trident pueda verificar el nodo `clean` (seguro para nuevas publicaciones).

Cuando se restaure el estado del nodo y se elimine el tinte, Astra Trident:

1. Identifique y limpie las rutas publicadas obsoletas en el nodo.
2. Si el nodo está en `cleanable` estado (se ha eliminado la contaminación de fuera de servicio y el nodo está en `Ready` estatal) Y todas las rutas obsoletas publicadas están limpias, Astra Trident reenviará el nodo como `clean` y permitir nuevos volúmenes publicados al nodo.

Configuraciones de ejemplo

Puede utilizar los atributos en Opciones de configuración al definir `TridentOrchestrator` para personalizar la instalación.

Configuración personalizada básica

Este es un ejemplo de una instalación 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
```

Selectores de nodos

Este ejemplo instala Astra Trident con selectores de nodos.

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

Nodos de trabajo de Windows

En este ejemplo se instala Astra Trident 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
```

Identidades administradas en un cluster AKS

En este ejemplo se instala Astra Trident para habilitar identidades gestionadas en un clúster de AKS.

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

Identidad de nube en un clúster AKS

En este ejemplo se instala Astra Trident para usarlo con una identidad de cloud en un clúster de AKS.

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

Identidad de nube en un clúster de EKS

En este ejemplo se instala Astra Trident para usarlo con una identidad de cloud en un clúster de AKS.

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

Información de copyright

Información de la marca comercial