



Instale usando o operador Trident

Astra Trident

NetApp
January 31, 2025

Índice

- Instale usando o operador Trident 1
 - Implantar manualmente o operador Trident (modo padrão) 1
 - Implantar manualmente o operador Trident (modo off-line)..... 6
 - Implantar operador Trident usando Helm (modo padrão) 12
 - Implantar operador Trident usando Helm (modo off-line) 16
 - Personalizar a instalação do operador Trident..... 21

Instale usando o operador Trident

Implantar manualmente o operador Trident (modo padrão)

Você pode implantar manualmente o operador Trident para instalar o Astra Trident. Este processo aplica-se a instalações onde as imagens de contêiner exigidas pelo Astra Trident não são armazenadas em um Registro privado. Se tiver um registro de imagens privado, utilize o "[processo para implantação off-line](#)".

Informações críticas sobre o Astra Trident 23,04

Você deve ler as seguintes informações críticas sobre o Astra Trident.

** informações essenciais sobre o Astra Trident **

- O Kubernetes 1,27 agora é compatível com o Trident. Atualize o Trident antes de atualizar o Kubernetes.
- O Astra Trident reforça estritamente o uso de configuração multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo `multipath.conf`.

O uso de configuração não multipathing ou o uso `find_multipaths: yes` de ou `find_multipaths: smart` valor no arquivo `multipath.conf` resultará em falhas de montagem. A Trident recomenda o uso de `find_multipaths: no` desde a versão 21,07.

Implante manualmente o operador Trident e instale o Trident

Revise "[a visão geral da instalação](#)" para garantir que você atendeu aos pré-requisitos de instalação e selecionou a opção de instalação correta para o seu ambiente.

Antes de começar

Antes de iniciar a instalação, faça login no host Linux e verifique se ele está gerenciando um trabalho e "[Cluster compatível com Kubernetes](#)" se você tem o Privileges necessário.



Com OpenShift, use `oc` em vez de `kubectl` em todos os exemplos que se seguem, e faça login como **system:admin** primeiro executando `oc login -u system:admin` ou `oc login -u kube-admin`.

1. Verifique sua versão do Kubernetes:

```
kubectl version
```

2. Verifique o Privileges do administrador do cluster:

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

3. Verifique se você pode iniciar um pod que usa uma imagem do Docker Hub e alcançar seu sistema de armazenamento pela rede de pod:

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

Passo 1: Baixe o pacote de instalação do Trident

O pacote de instalação do Astra Trident contém tudo o que você precisa para implantar o operador Trident e instalar o Astra Trident. Baixe e extraia a versão mais recente do instalador do Trident a partir "[A seção assets no GitHub](#)"do .

```
wget https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

Passo 2: Crie o TridentOrchestrator CRD

Crie a TridentOrchestrator Definição de recurso personalizada (CRD). Você criará um TridentOrchestrator recurso personalizado mais tarde. Use a versão apropriada do CRD YAML em `deploy/crds` para criar o TridentOrchestrator CRD.

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

Etapa 3: Implante o operador Trident

O instalador do Astra Trident fornece um arquivo de pacote que pode ser usado para instalar o operador e criar objetos associados. O arquivo de pacote é uma maneira fácil de implantar o operador e instalar o Astra Trident usando uma configuração padrão.

- Para clusters que executam o Kubernetes 1,24 ou anterior, `bundle_pre_1_25.yaml` use o .

- Para clusters que executam o Kubernetes 1,25 ou posterior, `bundle_post_1_25.yaml` use o .

Antes de começar

- Por padrão, o instalador do Trident implanta o operador no `trident` namespace. Se o `trident` namespace não existir, crie-o usando:

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

- Para implantar o operador em um namespace diferente do `trident` namespace, atualizar `serviceaccount.yaml` `clusterrolebinding.yaml` `operator.yaml` e gerar o arquivo do pacote usando o `kustomization.yaml`.

- a. Crie o `kustomization.yaml` usando o seguinte comando onde `<bundle>` está `bundle_pre_1_25` ou `bundle_post_1_25` baseado na sua versão do Kubernetes.

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

- b. Compile o pacote usando o seguinte comando onde `<bundle>` está `bundle_pre_1_25` ou `bundle_post_1_25` baseado na sua versão do Kubernetes.

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

Passos

1. Crie os recursos e implante o operador:

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

2. Verifique se o operador, a implantação e as replicaset foram criados.

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



Deve haver apenas **uma instância** do operador em um cluster do Kubernetes. Não crie várias implantações do operador Trident.

Passo 4: Crie o `TridentOrchestrator` e instale o Trident

Agora você pode criar e instalar o `TridentOrchestrator` Astra Trident. Opcionalmente, você pode "[Personalize a instalação do Trident](#)" usar os atributos na `TridentOrchestrator` especificação.

```

kubect1 create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubect1 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:23.04
    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:23.04.0
  Message:              Trident installed Namespace:
trident
  Status:                Installed
  Version:                v23.04.0
Events:
  Type Reason Age From Message ---- -
Installing 74s trident-operator.netapp.io Installing Trident Normal
Installed 67s trident-operator.netapp.io Trident installed

```

Verifique a instalação

Existem várias maneiras de verificar sua instalação.

TridentOrchestrator Usando o status

O status de `TridentOrchestrator` indica se a instalação foi bem-sucedida e exibe a versão do Trident instalado. Durante a instalação, o status das `TridentOrchestrator` alterações de `Installing` para `Installed`. Se você observar o `Failed` status e o operador não conseguir recuperar sozinho, ["verifique os logs"](#).

| Estado | Descrição |
|---------------|---|
| A instalar | O operador está instalando o Astra Trident usando este <code>TridentOrchestrator</code> CR. |
| Instalado | O Astra Trident foi instalado com sucesso. |
| Desinstalação | O operador está desinstalando o Astra Trident, <code>spec.uninstall=true</code> porque . |
| Desinstalado | O Astra Trident foi desinstalado. |
| Falha | O operador não pôde instalar, corrigir, atualizar ou desinstalar o Astra Trident; o operador tentará recuperar automaticamente deste estado. Se este estado persistir, será necessário resolver o problema. |
| A atualizar | O operador está atualizando uma instalação existente. |
| Erro | O <code>TridentOrchestrator</code> não é utilizado. Outro já existe. |

Usando o status de criação do pod

Você pode confirmar se a instalação do Astra Trident foi concluída analisando o status dos pods criados:

```
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
```

Utilização `tridentctl`

Você pode usar `tridentctl` para verificar a versão do Astra Trident instalada.

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

```
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 23.04.0        | 23.04.0        |
+-----+-----+
```

O que vem a seguir

Agora você pode ["crie um back-end e uma classe de storage, provisione um volume e monte o volume em um pod"](#).

Implantar manualmente o operador Trident (modo off-line)

Você pode implantar manualmente o operador Trident para instalar o Astra Trident. Este processo aplica-se a instalações onde as imagens de contêiner exigidas pelo Astra Trident são armazenadas em um Registro privado. Se não tiver um registro de imagens privado, utilize o ["processo para implantação padrão"](#).

Informações críticas sobre o Astra Trident 23,04

Você deve ler as seguintes informações críticas sobre o Astra Trident.

** informações essenciais sobre o Astra Trident **

- O Kubernetes 1,27 agora é compatível com o Trident. Atualize o Trident antes de atualizar o Kubernetes.
- O Astra Trident reforça estritamente o uso de configuração multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo `multipath.conf`.

O uso de configuração não multipathing ou o uso `find_multipaths: yes` de ou `find_multipaths: smart` valor no arquivo `multipath.conf` resultará em falhas de montagem. A Trident recomenda o uso de `find_multipaths: no` desde a versão 21,07.

Implante manualmente o operador Trident e instale o Trident

Revise ["a visão geral da instalação"](#) para garantir que você atendeu aos pré-requisitos de instalação e selecionou a opção de instalação correta para o seu ambiente.

Antes de começar

Faça login no host Linux e verifique se ele está gerenciando um trabalho e ["Cluster compatível com Kubernetes"](#) que você tem o Privileges necessário.



Com OpenShift, use `oc` em vez de `kubectl` em todos os exemplos que se seguem, e faça login como **system:admin** primeiro executando `oc login -u system:admin` ou `oc login -u kube-admin`.

1. Verifique sua versão do Kubernetes:

```
kubectl version
```

2. Verifique o Privileges do administrador do cluster:

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

3. Verifique se você pode iniciar um pod que usa uma imagem do Docker Hub e alcançar seu sistema de armazenamento pela rede de pod:

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

Passo 1: Baixe o pacote de instalação do Trident

O pacote de instalação do Astra Trident contém tudo o que você precisa para implantar o operador Trident e instalar o Astra Trident. Baixe e extraia a versão mais recente do instalador do Trident a partir "[A seção assets no GitHub](#)" do .

```
wget https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

Passo 2: Crie o TridentOrchestrator CRD

Crie a TridentOrchestrator Definição de recurso personalizada (CRD). Você criará um TridentOrchestrator recurso personalizado mais tarde. Use a versão apropriada do CRD YAML em `deploy/crds` para criar o TridentOrchestrator CRD:

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

Etapa 3: Atualize a localização do Registro no operador

No `deploy/operator.yaml`, atualize `image: docker.io/netapp/trident-operator:23.04.0` para refletir a localização do registro de imagens. O "[Imagens de Trident e CSI](#)" pode estar localizado em um

Registro ou Registros diferentes, mas todas as imagens CSI devem estar localizadas no mesmo Registro. Por exemplo:

- `image: <your-registry>/trident-operator:23.04.0` se todas as suas imagens estiverem localizadas em um registro.
- `image: <your-registry>/netapp/trident-operator:23.04.0` Se a sua imagem Trident estiver localizada num registro diferente das suas imagens CSI.

Etapa 4: Implante o operador Trident

O instalador do Astra Trident fornece um arquivo de pacote que pode ser usado para instalar o operador e criar objetos associados. O arquivo de pacote é uma maneira fácil de implantar o operador e instalar o Astra Trident usando uma configuração padrão.

- Para clusters que executam o Kubernetes 1,24 ou anterior, `bundle_pre_1_25.yaml` use o .
- Para clusters que executam o Kubernetes 1,25 ou posterior, `bundle_post_1_25.yaml` use o .

Antes de começar

- Por padrão, o instalador do Trident implanta o operador no `trident` namespace. Se o `trident` namespace não existir, crie-o usando:

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

- Para implantar o operador em um namespace diferente do `trident` namespace, atualizar `serviceaccount.yaml` `clusterrolebinding.yaml` `operator.yaml` e gerar o arquivo do pacote usando o `kustomization.yaml`.
 - a. Crie o `kustomization.yaml` usando o seguinte comando onde `<bundle>` está `bundle_pre_1_25` ou `bundle_post_1_25` baseado na sua versão do Kubernetes.

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

- b. Compile o pacote usando o seguinte comando onde `<bundle>` está `bundle_pre_1_25` ou `bundle_post_1_25` baseado na sua versão do Kubernetes.

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

Passos

1. Crie os recursos e implante o operador:

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

2. Verifique se o operador, a implantação e as replicaset foram criados.

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



Deve haver apenas **uma instância** do operador em um cluster do Kubernetes. Não crie várias implantações do operador Trident.

Passo 5: Atualize a localização do registo de imagens no `TridentOrchestrator`

O "[Imagens de Trident e CSI](#)" pode estar localizado em um Registro ou Registros diferentes, mas todas as imagens CSI devem estar localizadas no mesmo Registro. Atualize `deploy/crds/tridentorchestrator_cr.yaml` para adicionar as especificações de localização adicionais com base na configuração do seu registo.

Imagens em um Registro

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:23.04"
tridentImage: "<your-registry>/trident:23.04.0"
```

Imagens em diferentes registos

Você deve anexar `sig-storage` ao `imageRegistry` para usar diferentes locais de Registro.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:23.04"
tridentImage: "<your-registry>/netapp/trident:23.04.0"
```

Passo 6: Crie o `TridentOrchestrator` e instale o Trident

Agora você pode criar e instalar o `TridentOrchestrator` Astra Trident. Opcionalmente, você pode usar ainda mais "[Personalize a instalação do Trident](#)" os atributos na `TridentOrchestrator` especificação. O exemplo a seguir mostra uma instalação onde as imagens Trident e CSI estão localizadas em diferentes Registros.

```

kubect1 create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubect1 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:23.04
  Debug:              true
  Image Registry:    <your-registry>/sig-storage
  Namespace:         trident
  Trident Image:     <your-registry>/netapp/trident:23.04.0
Status:
  Current Installation Params:
    IPv6:              false
    Autosupport Hostname:
    Autosupport Image: <your-registry>/netapp/trident-
autosupport:23.04
    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:23.04.0
  Message:            Trident installed
  Namespace:          trident
  Status:              Installed
  Version:             v23.04.0
Events:
  Type Reason Age From Message ---- -
-----
Normal
Installing 74s trident-operator.netapp.io Installing Trident Normal
Installed 67s trident-operator.netapp.io Trident installed

```

Verifique a instalação

Existem várias maneiras de verificar sua instalação.

`TridentOrchestrator` Usando o status

O status de `TridentOrchestrator` indica se a instalação foi bem-sucedida e exibe a versão do Trident instalado. Durante a instalação, o status das `TridentOrchestrator` alterações de `Installing` para `Installed`. Se você observar o `Failed` status e o operador não conseguir recuperar sozinho, "[verifique os logs](#)".

| Estado | Descrição |
|---------------|---|
| A instalar | O operador está instalando o Astra Trident usando este <code>TridentOrchestrator</code> CR. |
| Instalado | O Astra Trident foi instalado com sucesso. |
| Desinstalação | O operador está desinstalando o Astra Trident, <code>spec.uninstall=true</code> porque . |
| Desinstalado | O Astra Trident foi desinstalado. |
| Falha | O operador não pôde instalar, corrigir, atualizar ou desinstalar o Astra Trident; o operador tentará recuperar automaticamente deste estado. Se este estado persistir, será necessário resolver o problema. |
| A atualizar | O operador está atualizando uma instalação existente. |
| Erro | O <code>TridentOrchestrator</code> não é utilizado. Outro já existe. |

Usando o status de criação do pod

Você pode confirmar se a instalação do Astra Trident foi concluída analisando o status dos pods criados:

```
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
```

Utilização tridentctl

Você pode usar `tridentctl` para verificar a versão do Astra Trident instalada.

```
./tridentctl -n trident version

+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 23.04.0        | 23.04.0        |
+-----+-----+
```

O que vem a seguir

Agora você pode ["crie um back-end e uma classe de storage, provisione um volume e monte o volume em um pod"](#).

Implantar operador Trident usando Helm (modo padrão)

Você pode implantar o operador Trident e instalar o Astra Trident usando o Helm. Este processo aplica-se a instalações onde as imagens de contendor exigidas pelo Astra Trident não são armazenadas em um Registro privado. Se tiver um registro de imagens privado, utilize o ["processo para implantação off-line"](#).

Informações críticas sobre o Astra Trident 23,04

Você deve ler as seguintes informações críticas sobre o Astra Trident.

** informações essenciais sobre o Astra Trident **

- O Kubernetes 1,27 agora é compatível com o Trident. Atualize o Trident antes de atualizar o Kubernetes.
- O Astra Trident reforça estritamente o uso de configuração multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo `multipath.conf`.

O uso de configuração não multipathing ou o uso `find_multipaths: yes` de ou `find_multipaths: smart` valor no arquivo `multipath.conf` resultará em falhas de montagem. A Trident recomenda o uso de `find_multipaths: no` desde a versão 21,07.

Implante o operador Trident e instale o Astra Trident usando o Helm

Usando o Trident ["Carta do leme"](#), você pode implantar o operador Trident e instalar o Trident em uma etapa.

Revise ["a visão geral da instalação"](#) para garantir que você atendeu aos pré-requisitos de instalação e selecionou a opção de instalação correta para o seu ambiente.

Antes de começar

Além do "pré-requisitos de implantação" que você precisa "Helm versão 3".

Passos

1. Adicione o repositório Astra Trident Helm:

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

2. Use `helm install` e especifique um nome para sua implantação, como no exemplo a seguir, onde 23.04.0 está a versão do Astra Trident que você está instalando.

```
helm install <name> netapp-trident/trident-operator --version 23.04.0  
--create-namespace --namespace <trident-namespace>
```



Se você já criou um namespace para Trident, o `--create-namespace` parâmetro não criará um namespace adicional.

Você pode usar `helm list` para revisar detalhes de instalação, como nome, namespace, gráfico, status, versão do aplicativo e número de revisão.

Passe os dados de configuração durante a instalação

Há duas maneiras de passar dados de configuração durante a instalação:

| Opção | Descrição |
|---|---|
| <code>--values</code> (ou <code>-f</code>) | Especifique um arquivo YAML com substituições. Isso pode ser especificado várias vezes e o arquivo mais à direita terá precedência. |
| <code>--set</code> | Especifique substituições na linha de comando. |

Por exemplo, para alterar o valor padrão `debug` do , execute o seguinte `--set` comando onde 23.04.0 está a versão do Astra Trident que você está instalando:

```
helm install <name> netapp-trident/trident-operator --version 23.04.0  
--create-namespace --namespace --set tridentDebug=true
```

Opções de configuração

Esta tabela e o `values.yaml` arquivo, que faz parte do gráfico Helm, fornecem a lista de chaves e seus valores padrão.

| Opção | Descrição | Padrão |
|-------------------------------------|---|---------------------|
| nodeSelector | Etiquetas de nó para atribuição de pod | |
| podAnnotations | Anotações do pod | |
| deploymentAnnotations | Anotações de implantação | |
| tolerations | Tolerâncias para atribuição de pod | |
| affinity | Afinidade para atribuição de pod | |
| tridentControllerPluginNodeSelector | Seletores de nós adicionais para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| tridentControllerPluginTolerations | Substitui as tolerâncias do Kubernetes para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| tridentNodePluginNodeSelector | Seletores de nós adicionais para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| tridentNodePluginTolerations | Substitui as tolerâncias do Kubernetes para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| imageRegistry | Identifica o registo para as <code>trident-operator</code> , <code>`trident`</code> e outras imagens. Deixe vazio para aceitar o padrão. | "" |
| imagePullPolicy | Define a política de recebimento de imagens para o <code>trident-operator</code> . | IfNotPresent |
| imagePullSecrets | Define os segredos de extração da imagem para as <code>trident-operator</code> , <code>trident</code> e outras imagens. | |
| kubeletDir | Permite substituir a localização do host do estado interno do kubelet. | ""/var/lib/kubelet" |
| operatorLogLevel | Permite que o nível de log do operador Trident seja definido como: <code>trace</code> , <code>debug</code> , <code>info</code> , <code>warn</code> , <code>error</code> Ou <code>fatal</code> . | "info" |

| Opção | Descrição | Padrão |
|---|--|------------------------------|
| <code>operatorDebug</code> | Permite que o nível de log do operador Trident seja definido como debug. | <code>true</code> |
| <code>operatorImage</code> | Permite a substituição completa da imagem para <code>trident-operator</code> . | <code>""</code> |
| <code>operatorImageTag</code> | Permite substituir a etiqueta da <code>trident-operator</code> imagem. | <code>""</code> |
| <code>tridentIPv6</code> | Permite que o Astra Trident funcione em IPv6 clusters. | <code>false</code> |
| <code>tridentK8sTimeout</code> | Substitui o tempo limite padrão de 30 segundos para a maioria das operações da API do Kubernetes (se não for zero, em segundos). | <code>0</code> |
| <code>tridentHttpRequestTimeout</code> | Substitui o tempo limite padrão de 90 segundos para as solicitações HTTP, <code>0s</code> sendo uma duração infinita para o tempo limite. Valores negativos não são permitidos. | <code>"90s"</code> |
| <code>tridentSilenceAutosupport</code> | Permite desativar os relatórios periódicos do Astra Trident AutoSupport. | <code>false</code> |
| <code>tridentAutosupportImageTag</code> | Permite substituir a tag da imagem para o contentor Astra Trident AutoSupport. | <code><version></code> |
| <code>tridentAutosupportProxy</code> | Permite que o Astra Trident AutoSupport Container ligue para casa por meio de um proxy HTTP. | <code>""</code> |
| <code>tridentLogFormat</code> | Define o formato de registo Astra Trident (<code>text</code> ou <code>json</code>). | <code>"text"</code> |
| <code>tridentDisableAuditLog</code> | Desativa o registrator de auditoria Astra Trident. | <code>true</code> |
| <code>tridentLogLevel</code> | Permite que o nível de log do Astra Trident seja definido como: <code>trace</code> , <code>debug</code> , <code>info</code> , <code>warn</code> , <code>error</code> ou <code>fatal</code> . | <code>"info"</code> |
| <code>tridentDebug</code> | Permite que o nível de log do Astra Trident seja definido como debug. | <code>false</code> |
| <code>tridentLogWorkflows</code> | Permite que fluxos de trabalho específicos do Astra Trident sejam ativados para registo de rastreio ou supressão de registos. | <code>""</code> |

| Opção | Descrição | Padrão |
|--------------------------|--|--------|
| tridentLogLayers | Permite que camadas específicas do Astra Trident sejam ativadas para registo de rastreio ou supressão de registos. | "" |
| tridentImage | Permite a substituição completa da imagem para Astra Trident. | "" |
| tridentImageTag | Permite substituir a tag da imagem para Astra Trident. | "" |
| tridentProbePort | Permite substituir a porta padrão usada para sondas de disponibilidade/prontidão do Kubernetes. | "" |
| windows | Permite que o Astra Trident seja instalado no nó de trabalho do Windows. | false |
| enableForceDetach | Permite ativar a função forçar desanexar. | false |
| excludePodSecurityPolicy | Exclui a criação da diretiva de segurança do pod do operador. | false |

Compreensão dos pods dos nós e dos pods do controlador

O Astra Trident é executado como um único pod de controlador, além de um pod de nó em cada nó de trabalho no cluster. O pod de nó deve estar em execução em qualquer host onde você queira potencialmente montar um volume Astra Trident.

Kubernetes "[seletores de nós](#)" e "[tolerações e taints](#)" são usados para restringir um pod a ser executado em um nó específico ou preferencial. Usando o ControllerPlugin e `NodePlugin` o , você pode especificar restrições e substituições.

- O plugin controlador lida com o provisionamento e gerenciamento de volume, como snapshots e redimensionamento.
- O plug-in do nó manipula a conexão do armazenamento ao nó.

O que vem a seguir

Agora você pode "[crie um back-end e uma classe de storage, provisione um volume e monte o volume em um pod](#)".

Implantar operador Trident usando Helm (modo off-line)

Você pode implantar o operador Trident e instalar o Astra Trident usando o Helm. Este processo aplica-se a instalações onde as imagens de contêiner exigidas pelo Astra Trident são armazenadas em um Registro privado. Se não tiver um registro de imagens privado, utilize o "[processo para implantação padrão](#)".

Informações críticas sobre o Astra Trident 23,04

Você deve ler as seguintes informações críticas sobre o Astra Trident.

informações essenciais sobre o Astra Trident

- O Kubernetes 1,27 agora é compatível com o Trident. Atualize o Trident antes de atualizar o Kubernetes.
- O Astra Trident reforça estritamente o uso de configuração multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo `multipath.conf`.

O uso de configuração não multipathing ou o uso `find_multipaths: yes` de ou `find_multipaths: smart` valor no arquivo `multipath.conf` resultará em falhas de montagem. A Trident recomenda o uso de `find_multipaths: no` desde a versão 21,07.

Implante o operador Trident e instale o Astra Trident usando o Helm

Usando o Trident "[Carta do leme](#)", você pode implantar o operador Trident e instalar o Trident em uma etapa.

Revise "[a visão geral da instalação](#)" para garantir que você atendeu aos pré-requisitos de instalação e selecionou a opção de instalação correta para o seu ambiente.

Antes de começar

Além do "[pré-requisitos de implantação](#)" que você precisa "[Helm versão 3](#)".

Passos

1. Adicione o repositório Astra Trident Helm:

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

2. Use `helm install` e especifique um nome para a localização do Registro de imagens e implantação. O "[Imagens de Trident e CSI](#)" pode estar localizado em um Registro ou Registros diferentes, mas todas as imagens CSI devem estar localizadas no mesmo Registro. Nos exemplos 23.04.0, é a versão do Astra Trident que você está instalando.

Imagens em um Registro

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

Imagens em diferentes registros

Você deve anexar `sig-storage` ao `imageRegistry` para usar diferentes locais de Registro.

```
helm install <name> netapp-trident/trident-operator --version  
23.04.0 --set imageRegistry=<your-registry>/sig-storage --set  
operatorImage=<your-registry>/netapp/trident-operator:23.04.0 --set  
tridentAutosupportImage=<your-registry>/netapp/trident-  
autosupport:23.04 --set tridentImage=<your-  
registry>/netapp/trident:23.04.0 --create-namespace --namespace  
<trident-namespace>
```



Se você já criou um namespace para Trident, o `--create-namespace` parâmetro não criará um namespace adicional.

Você pode usar `helm list` para revisar detalhes de instalação, como nome, namespace, gráfico, status, versão do aplicativo e número de revisão.

Passe os dados de configuração durante a instalação

Há duas maneiras de passar dados de configuração durante a instalação:

| Opção | Descrição |
|---|---|
| <code>--values</code> (ou <code>-f</code>) | Especifique um arquivo YAML com substituições. Isso pode ser especificado várias vezes e o arquivo mais à direita terá precedência. |
| <code>--set</code> | Especifique substituições na linha de comando. |

Por exemplo, para alterar o valor padrão `debug` do , execute o seguinte `--set` comando onde `23.04.0` está a versão do Astra Trident que você está instalando:

```
helm install <name> netapp-trident/trident-operator --version 23.04.0  
--create-namespace --namespace --set tridentDebug=true
```

Opções de configuração

Esta tabela e o `values.yaml` arquivo, que faz parte do gráfico Helm, fornecem a lista de chaves e seus valores padrão.

| Opção | Descrição | Padrão |
|--|--|---------------------------|
| <code>nodeSelector</code> | Etiquetas de nó para atribuição de pod | |
| <code>podAnnotations</code> | Anotações do pod | |
| <code>deploymentAnnotations</code> | Anotações de implantação | |
| <code>tolerations</code> | Tolerâncias para atribuição de pod | |
| <code>affinity</code> | Afinidade para atribuição de pod | |
| <code>tridentControllerPluginNodeSelector</code> | Seletores de nós adicionais para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| <code>tridentControllerPluginTolerations</code> | Substitui as tolerâncias do Kubernetes para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| <code>tridentNodePluginNodeSelector</code> | Seletores de nós adicionais para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| <code>tridentNodePluginTolerations</code> | Substitui as tolerâncias do Kubernetes para pods. Compreensão dos pods dos nós e dos pods do controlador Consulte para obter detalhes. | |
| <code>imageRegistry</code> | Identifica o registo para as <code>trident-operator</code> , <code>trident</code> e outras imagens. Deixe vazio para aceitar o padrão. | "" |
| <code>imagePullPolicy</code> | Define a política de recebimento de imagens para o <code>trident-operator</code> . | <code>IfNotPresent</code> |
| <code>imagePullSecrets</code> | Define os segredos de extração da imagem para as <code>trident-operator</code> , <code>trident</code> e outras imagens. | |
| <code>kubeletDir</code> | Permite substituir a localização do host do estado interno do kubelet. | ""/var/lib/kubelet" |

| Opção | Descrição | Padrão |
|----------------------------|--|-----------|
| operatorLogLevel | Permite que o nível de log do operador Trident seja definido como: trace, , debug, info warn , , error Ou fatal. | "info" |
| operatorDebug | Permite que o nível de log do operador Trident seja definido como debug. | true |
| operatorImage | Permite a substituição completa da imagem para trident-operator. | "" |
| operatorImageTag | Permite substituir a etiqueta da trident-operator imagem. | "" |
| tridentIPv6 | Permite que o Astra Trident funcione em IPv6 clusters. | false |
| tridentK8sTimeout | Substitui o tempo limite padrão de 30 segundos para a maioria das operações da API do Kubernetes (se não for zero, em segundos). | 0 |
| tridentHttpRequestTimeout | Substitui o tempo limite padrão de 90 segundos para as solicitações HTTP, 0s sendo uma duração infinita para o tempo limite. Valores negativos não são permitidos. | "90s" |
| tridentSilenceAutosupport | Permite desativar os relatórios periódicos do Astra Trident AutoSupport. | false |
| tridentAutosupportImageTag | Permite substituir a tag da imagem para o contentor Astra Trident AutoSupport. | <version> |
| tridentAutosupportProxy | Permite que o Astra Trident AutoSupport Container ligue para casa por meio de um proxy HTTP. | "" |
| tridentLogFormat | Define o formato de registo Astra Trident (text`ou `json). | "text" |
| tridentDisableAuditLog | Desativa o registor de auditoria Astra Trident. | true |
| tridentLogLevel | Permite que o nível de log do Astra Trident seja definido como: trace, debug, , info warn , , error fatal Ou . | "info" |
| tridentDebug | Permite que o nível de log do Astra Trident seja definido como debug. | false |

| Opção | Descrição | Padrão |
|---------------------------------------|---|--------------------|
| <code>tridentLogWorkflows</code> | Permite que fluxos de trabalho específicos do Astra Trident sejam ativados para registo de rastreio ou supressão de registos. | "" |
| <code>tridentLogLayers</code> | Permite que camadas específicas do Astra Trident sejam ativadas para registo de rastreio ou supressão de registos. | "" |
| <code>tridentImage</code> | Permite a substituição completa da imagem para Astra Trident. | "" |
| <code>tridentImageTag</code> | Permite substituir a tag da imagem para Astra Trident. | "" |
| <code>tridentProbePort</code> | Permite substituir a porta padrão usada para sondas de disponibilidade/prontidão do Kubernetes. | "" |
| <code>windows</code> | Permite que o Astra Trident seja instalado no nó de trabalho do Windows. | <code>false</code> |
| <code>enableForceDetach</code> | Permite ativar a função forçar desanexar. | <code>false</code> |
| <code>excludePodSecurityPolicy</code> | Exclui a criação da diretiva de segurança do pod do operador. | <code>false</code> |

Compreensão dos pods dos nós e dos pods do controlador

O Astra Trident é executado como um único pod de controlador, além de um pod de nó em cada nó de trabalho no cluster. O pod de nó deve estar em execução em qualquer host onde você queira potencialmente montar um volume Astra Trident.

Kubernetes "[seletores de nós](#)" e "[tolerações e taints](#)" são usados para restringir um pod a ser executado em um nó específico ou preferencial. Usando o `ControllerPlugin` e `NodePlugin`, você pode especificar restrições e substituições.

- O plugin controlador lida com o provisionamento e gerenciamento de volume, como snapshots e redimensionamento.
- O plug-in do nó manipula a conexão do armazenamento ao nó.

O que vem a seguir

Agora você pode ["crie um back-end e uma classe de storage, provisione um volume e monte o volume em um pod"](#).

Personalizar a instalação do operador Trident

O operador Trident permite personalizar a instalação do Astra Trident usando os

atributos da `TridentOrchestrator` especificação. Se você quiser personalizar a instalação além do que `TridentOrchestrator` os argumentos permitem, considere usar `tridentctl` para gerar manifestos YAML personalizados para modificar conforme necessário.

Compreensão dos pods dos nós e dos pods do controlador

O Astra Trident é executado como um único pod de controlador, além de um pod de nó em cada nó de trabalho no cluster. O pod de nó deve estar em execução em qualquer host onde você queira potencialmente montar um volume Astra Trident.

Kubernetes "[seletores de nós](#)" e "[tolerações e taints](#)" são usados para restringir um pod a ser executado em um nó específico ou preferencial. Usando o `ControllerPlugin` e `NodePlugin`, você pode especificar restrições e substituições.

- O plugin controlador lida com o provisionamento e gerenciamento de volume, como snapshots e redimensionamento.
- O plug-in do nó manipula a conexão do armazenamento ao nó.

Opções de configuração



`spec.namespace` É especificado em `TridentOrchestrator` para indicar o namespace onde o Astra Trident está instalado. Este parâmetro **não pode ser atualizado após a instalação do Astra Trident**. Tentar fazê-lo faz com que o `TridentOrchestrator` status mude para `Failed`. O Astra Trident não se destina a ser migrado entre namespaces.

Esta tabela detalha `TridentOrchestrator` atributos.

| Parâmetro | Descrição | Padrão |
|--------------------------------|--|----------------|
| <code>namespace</code> | Namespace para instalar Astra Trident em | "predefinição" |
| <code>debug</code> | Habilite a depuração para o Astra Trident | falso |
| <code>enableForceDetach</code> | <code>ontap-san</code> e <code>ontap-san-economy</code> apenas. Funciona com o Kubernetes Non-Graceful Node Shutdown (NGNS) para conceder aos administradores de cluster a capacidade de migrar com segurança cargas de trabalho com volumes montados para novos nós caso um nó não seja saudável. Esta é uma característica experimental em 23,04. Detalhes sobre Force Detach Consulte para obter detalhes importantes. | false |
| <code>windows</code> | A configuração para <code>true</code> permite a instalação em nós de trabalho do Windows. | falso |

| Parâmetro | Descrição | Padrão |
|------------------------------|---|---|
| useIPv6 | Instalar o Astra Trident em IPv6 | falso |
| k8sTimeout | Tempo limite para operações do Kubernetes | 30sec |
| silenceAutosupport | Não envie pacotes AutoSupport para o NetApp automaticamente | falso |
| autosupportImage | A imagem do recipiente para a telemetria AutoSupport | "NetApp/Trident-AutoSupport:23,07" |
| autosupportProxy | O endereço/porta de um proxy para o envio de telemetria AutoSupport | "http://proxy.example.com:8888"" |
| uninstall | Um sinalizador usado para desinstalar o Astra Trident | falso |
| logFormat | Formato de log Astra Trident a ser usado [text,json] | "texto" |
| tridentImage | Imagem Astra Trident a instalar | "NetApp/Trident:23,07" |
| imageRegistry | Caminho para o Registro interno, do formato <registry FQDN>[:port] [/subpath] | "k8s.gcr.io/sig-storage" (mais de k8s 1,19 gb) ou "quay.io/k8scsi gb" |
| kubeletDir | Caminho para o diretório kubelet no host | "/var/lib/kubelet" |
| wipeout | Uma lista de recursos a serem excluídos para realizar uma remoção completa do Astra Trident | |
| imagePullSecrets | Segredos para extrair imagens de um Registro interno | |
| imagePullPolicy | Define a política de recebimento de imagens para o operador Trident. Os valores válidos são: Always Para sempre puxar a imagem. IfNotPresent para puxar a imagem apenas se ela ainda não existir no nó. Never para nunca puxar a imagem. | IfNotPresent |
| controllerPluginNodeSelector | Seletores de nós adicionais para pods. Segue o mesmo formato que pod.spec.nodeSelector. | Sem padrão; opcional |

| Parâmetro | Descrição | Padrão |
|--|---|----------------------|
| <code>controllerPluginTolerations</code> | Substitui as tolerâncias do Kubernetes para pods. Segue o mesmo formato que <code>pod.spec.Tolerations</code> . | Sem padrão; opcional |
| <code>nodePluginNodeSelector</code> | Seletores de nós adicionais para pods. Segue o mesmo formato que <code>pod.spec.nodeSelector</code> . | Sem padrão; opcional |
| <code>nodePluginTolerations</code> | Substitui as tolerâncias do Kubernetes para pods. Segue o mesmo formato que <code>pod.spec.Tolerations</code> . | Sem padrão; opcional |



Para obter mais informações sobre a formatação dos parâmetros do pod, "[Atribuindo pods a nós](#)" consulte .

Detalhes sobre Force Detach

O Force Detach está disponível apenas para `ontap-san` e `ontap-san-economy`. Antes de ativar a desconexão forçada, o desligamento do nó (NGNS) não gracioso deve ser ativado no cluster do Kubernetes. Para obter mais informações, "[Kubernetes: Desligamento do nó não gracioso](#)" consulte .



Como o Astra Trident conta COM NGNS do Kubernetes, não `out-of-service` remova as taints de um nó que não seja saudável até que todos os workloads não toleráveis sejam reprogramados. Aplicar ou remover a taint de forma imprudente pode comprometer a proteção de dados no back-end.

Quando o administrador do cluster do Kubernetes tiver aplicado a `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` alteração ao nó e `enableForceDetach` estiver definido como `true`, o Astra Trident determinará o status do nó e:

1. Cessar o acesso de e/S de back-end para volumes montados nesse nó.
2. Marque o objeto nó Astra Trident como `dirty` (não é seguro para novas publicações).



O controlador Trident rejeitará novas solicitações de volume de publicação até que o nó seja requalificado (depois de ter sido marcado como `dirty`) pelo pod de nó do Trident. Quaisquer cargas de trabalho agendadas com um PVC montado (mesmo depois que o nó do cluster estiver pronto e saudável) não serão aceitas até que o Astra Trident possa verificar o nó `clean` (seguro para novas publicações).

Quando a integridade do nó é restaurada e a taint é removida, o Astra Trident:

1. Identifique e limpe caminhos publicados obsoletos no nó.
2. Se o nó estiver em um `cleanable` estado (a alteração fora de serviço foi removida e o nó está `Ready` no estado) e todos os caminhos obsoletos e publicados estiverem limpos, o Astra Trident reajustará o nó como `clean` e permitirá novos volumes publicados no nó.

Exemplos de configurações

Você pode usar os atributos mencionados acima ao definir `TridentOrchestrator` para personalizar sua instalação.

Exemplo 1: Configuração personalizada básica

Este é um exemplo para uma configuração 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
```

Exemplo 2: Implante com seletores de nós

Este exemplo ilustra como o Trident pode ser implantado com seletores de nós:

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

Exemplo 3: Implantar em nós de trabalho do Windows

Este exemplo ilustra a implantação em um nó de trabalho do Windows.

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

Informações sobre direitos autorais

Copyright © 2025 NetApp, Inc. Todos os direitos reservados. Impresso nos EUA. Nenhuma parte deste documento protegida por direitos autorais pode ser reproduzida de qualquer forma ou por qualquer meio — gráfico, eletrônico ou mecânico, incluindo fotocópia, gravação, gravação em fita ou storage em um sistema de recuperação eletrônica — sem permissão prévia, por escrito, do proprietário dos direitos autorais.

O software derivado do material da NetApp protegido por direitos autorais está sujeito à seguinte licença e isenção de responsabilidade:

ESTE SOFTWARE É FORNECIDO PELA NETAPP "NO PRESENTE ESTADO" E SEM QUAISQUER GARANTIAS EXPRESSAS OU IMPLÍCITAS, INCLUINDO, SEM LIMITAÇÕES, GARANTIAS IMPLÍCITAS DE COMERCIALIZAÇÃO E ADEQUAÇÃO A UM DETERMINADO PROPÓSITO, CONFORME A ISENÇÃO DE RESPONSABILIDADE DESTES DOCUMENTOS. EM HIPÓTESE ALGUMA A NETAPP SERÁ RESPONSÁVEL POR QUALQUER DANO DIRETO, INDIRETO, INCIDENTAL, ESPECIAL, EXEMPLAR OU CONSEQUENCIAL (INCLUINDO, SEM LIMITAÇÕES, AQUISIÇÃO DE PRODUTOS OU SERVIÇOS SOBRESSALIENTES; PERDA DE USO, DADOS OU LUCROS; OU INTERRUPÇÃO DOS NEGÓCIOS), INDEPENDENTEMENTE DA CAUSA E DO PRINCÍPIO DE RESPONSABILIDADE, SEJA EM CONTRATO, POR RESPONSABILIDADE OBJETIVA OU PREJUÍZO (INCLUINDO NEGLIGÊNCIA OU DE OUTRO MODO), RESULTANTE DO USO DESTES SOFTWARES, MESMO SE ADVERTIDA DA RESPONSABILIDADE DE TAL DANO.

A NetApp reserva-se o direito de alterar quaisquer produtos descritos neste documento, a qualquer momento e sem aviso. A NetApp não assume nenhuma responsabilidade nem obrigação decorrentes do uso dos produtos descritos neste documento, exceto conforme expressamente acordado por escrito pela NetApp. O uso ou a compra deste produto não representam uma licença sob quaisquer direitos de patente, direitos de marca comercial ou quaisquer outros direitos de propriedade intelectual da NetApp.

O produto descrito neste manual pode estar protegido por uma ou mais patentes dos EUA, patentes estrangeiras ou pedidos pendentes.

LEGENDA DE DIREITOS LIMITADOS: o uso, a duplicação ou a divulgação pelo governo estão sujeitos a restrições conforme estabelecido no subparágrafo (b)(3) dos Direitos em Dados Técnicos - Itens Não Comerciais no DFARS 252.227-7013 (fevereiro de 2014) e no FAR 52.227- 19 (dezembro de 2007).

Os dados aqui contidos pertencem a um produto comercial e/ou serviço comercial (conforme definido no FAR 2.101) e são de propriedade da NetApp, Inc. Todos os dados técnicos e software de computador da NetApp fornecidos sob este Contrato são de natureza comercial e desenvolvidos exclusivamente com despesas privadas. O Governo dos EUA tem uma licença mundial limitada, irrevogável, não exclusiva, intransferível e não sublicenciável para usar os Dados que estão relacionados apenas com o suporte e para cumprir os contratos governamentais desse país que determinam o fornecimento de tais Dados. Salvo disposição em contrário no presente documento, não é permitido usar, divulgar, reproduzir, modificar, executar ou exibir os dados sem a aprovação prévia por escrito da NetApp, Inc. Os direitos de licença pertencentes ao governo dos Estados Unidos para o Departamento de Defesa estão limitados aos direitos identificados na cláusula 252.227-7015(b) (fevereiro de 2014) do DFARS.

Informações sobre marcas comerciais

NETAPP, o logotipo NETAPP e as marcas listadas em <http://www.netapp.com/TM> são marcas comerciais da NetApp, Inc. Outros nomes de produtos e empresas podem ser marcas comerciais de seus respectivos proprietários.