



# Implante com o operador Trident

## Astra Trident

NetApp  
February 05, 2025

# Índice

- Implante com o operador Trident ..... 1
  - Implante o operador Trident usando Helm ..... 1
  - Implante o operador Trident manualmente ..... 2
  - Personalizar a implantação do operador Trident ..... 7

# Implante com o operador Trident

Você pode implantar o Astra Trident com o operador Trident. Você pode implantar o operador Trident manualmente ou usando o Helm.



Se você ainda não se familiarizou com o "[conceitos básicos](#)", agora é um ótimo momento para fazer isso.

## O que você vai precisar

Para implantar o Astra Trident, os seguintes pré-requisitos devem ser atendidos:

- Você tem o Privileges completo para um cluster Kubernetes compatível com Kubernetes que executa o Kubernetes 1,17 e superior.
- Você tem acesso a um sistema de storage NetApp compatível.
- Você tem a capacidade de montar volumes de todos os nós de trabalho do Kubernetes.
- Você tem um host Linux com `kubectl` (ou `oc`, se estiver usando o OpenShift) instalado e configurado para gerenciar o cluster do Kubernetes que deseja usar.
- Você definiu a `KUBECONFIG` variável de ambiente para apontar para a configuração do cluster do Kubernetes.
- Você ativou o "[Portas de recurso exigidas pelo Astra Trident](#)".
- Se você estiver usando o Kubernetes com Docker Enterprise "[Siga os passos para ativar o acesso CLI](#)", .

Tem tudo isso? Ótimo! Vamos começar.

## Implante o operador Trident usando Helm

Execute as etapas listadas para implantar o operador Trident usando Helm.

### O que você vai precisar

Além dos pré-requisitos listados acima, para implantar o operador Trident usando o Helm, você precisa do seguinte:

- Kubernetes 1,17 e posterior
- Helm versão 3

### Passos

1. Adicionar o repositório Helm do Trident:

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

2. Use o `helm install` comando e especifique um nome para sua implantação. Veja o exemplo a seguir:

```
helm install <release-name> netapp-trident/trident-operator --version 22.1.0 --namespace <trident-namespace>
```



Se você ainda não criou um namespace para Trident, você pode adicionar o `--create-namespace` parâmetro ao `helm install` comando. Helm irá então criar automaticamente o namespace para você.

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

- `--values` (Ou `-f`): Especifique um arquivo YAML com substituições. Isso pode ser especificado várias vezes e o arquivo mais à direita terá precedência.
- `--set`: Especificar substituições na linha de comando.

Por exemplo, para alterar o valor padrão `debug` do , execute o seguinte `--set` comando:

```
$ helm install <name> netapp-trident/trident-operator --version 22.1.0  
--set tridentDebug=true
```

O `values.yaml` arquivo, que faz parte do gráfico Helm, fornece a lista de chaves e seus valores padrão.

`helm list` mostra detalhes sobre a instalação, como nome, namespace, gráfico, status, versão do aplicativo, número de revisão e assim por diante.

## Implante o operador Trident manualmente

Execute as etapas listadas para implantar manualmente o operador Trident.

### Etapa 1: Qualifique seu cluster Kubernetes

A primeira coisa que você precisa fazer é fazer login no host Linux e verificar se ele está gerenciando um *working*, "[Cluster compatível com Kubernetes](#)" que você tem o Privileges necessário para.



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`.

Para ver se sua versão do Kubernetes é posterior a 1,17, execute o seguinte comando:

```
kubectl version
```

Para ver se você tem o administrador do cluster do Kubernetes Privileges, execute o seguinte comando:

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

Para verificar se você pode iniciar um pod que usa uma imagem do Docker Hub e alcançar seu sistema de armazenamento pela rede de pod, execute o seguinte comando:

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

## Passo 2: Baixe e configure o operador



A partir de 21,01, o operador Trident tem o escopo do cluster. O uso do operador Trident para instalar o Trident requer a criação da `TridentOrchestrator` Definição de recursos personalizada (CRD) e a definição de outros recursos. Você deve executar estas etapas para configurar o operador antes de poder instalar o Astra Trident.

1. Baixe a versão mais recente da "[Pacote de instalação do Trident](#)" na seção *Downloads* e extraia-a.

```
wget https://github.com/NetApp/trident/releases/download/v21.04/trident-
installer-21.04.tar.gz
tar -xf trident-installer-21.04.tar.gz
cd trident-installer
```

2. Use o manifesto CRD apropriado para criar o `TridentOrchestrator` CRD. Em seguida, crie um `TridentOrchestrator` recurso personalizado mais tarde para instanciar uma instalação pelo operador.

Execute o seguinte comando:

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

3. Após a criação do `TridentOrchestrator` CRD, crie os seguintes recursos necessários para a implantação do operador:

- Um `ServiceAccount` para o operador
- Um `ClusterRole` e `ClusterRoleBinding` para o `ServiceAccount`
- Uma `PodSecurityPolicy` dedicada
- O próprio operador

O instalador do Trident contém manifestos para definir esses recursos. Por padrão, o operador é implantado no `trident` namespace. Se o `trident` namespace não existir, use o manifesto a seguir para criar um.

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

4. Para implantar o operador em um namespace diferente do namespace padrão `trident`, você deve atualizar o `serviceaccount.yaml`, `clusterrolebinding.yaml` e `operator.yaml` manifesta e gera o `bundle.yaml`.

Execute o seguinte comando para atualizar os manifestos YAML e gerar o `bundle.yaml` usando o

kustomization.yaml:

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

Execute o seguinte comando para criar os recursos e implantar o operador:

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

5. Para verificar o status do operador depois de ter implantado, faça o seguinte:

```
$ kubectl get deployment -n <operator-namespace>
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
trident-operator    1/1      1              1            3m

$ kubectl get pods -n <operator-namespace>
NAME                                READY    STATUS    RESTARTS
AGE
trident-operator-54cb664d-lnjxh    1/1      Running    0
3m
```

A implantação do operador cria com êxito um pod em execução em um dos nós de trabalho no cluster.



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

### Passo 3: Crie TridentOrchestrator e instale o Trident

Agora você está pronto para instalar o Astra Trident usando o operador! Isso exigirá a criação `TridentOrchestrator`do . O instalador do Trident vem com exemplos de definições para criar `TridentOrchestrator. Isso inicia uma instalação no trident namespace.`

```

$ 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:21.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                true
    Enable Node Prep:     false
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:            text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:21.04.0
  Message:              Trident installed Namespace:
trident
  Status:                Installed
  Version:                v21.04.0
Events:
  Type Reason Age From Message ---- -
  Installing 74s trident-operator.netapp.io Installing Trident Normal
  Installed 67s trident-operator.netapp.io Trident installed

```

O operador Trident permite personalizar a maneira como o Astra Trident é instalado usando os atributos na TridentOrchestrator especificação. ["Personalize a implantação do Trident"](#)Consulte .

O Status do TridentOrchestrator indica se a instalação foi bem-sucedida e exibe a versão do Trident instalado.

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.

Durante a instalação, o status das `TridentOrchestrator` alterações de `Installing` para `Installed`. Se observar o `Failed` estado e o operador não conseguir recuperar sozinho, deve verificar os registos do operador. Consulte "[solução de problemas](#)" a secção .

Você pode confirmar se a instalação do Astra Trident foi concluída dando uma olhada nos pods criados:

```
$ kubectl get pod -n trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-7d466bf5c7-v4cpw       5/5     Running   0           1m
trident-csi-mr6zc                    2/2     Running   0           1m
trident-csi-xrp7w                    2/2     Running   0           1m
trident-csi-zh2jt                    2/2     Running   0           1m
trident-operator-766f7b8658-ldzsv   1/1     Running   0           3m
```

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

```
$ ./tridentctl -n trident version
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 21.04.0        | 21.04.0        |
+-----+-----+
```

Agora você pode ir em frente e criar um backend. "[tarefas pós-implantação](#)" Consulte .



Para solucionar problemas durante a implantação, consulte "[solução de problemas](#)" a seção.



# Personalizar a implantação do operador Trident

O operador Trident permite personalizar a maneira como o Astra Trident é instalado usando os atributos na `TridentOrchestrator` especificação.

Consulte a tabela a seguir para obter a lista de 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>IPv6</code>	Instalar o Astra Trident em IPv6	falso
<code>k8sTimeout</code>	Tempo limite para operações do Kubernetes	30sec
<code>silenceAutosupport</code>	Não envie pacotes AutoSupport para o NetApp automaticamente	falso
<code>enableNodePrep</code>	Gerenciar dependências de nó de trabalho automaticamente ( <b>BETA</b> )	falso
<code>autosupportImage</code>	A imagem do recipiente para a telemetria AutoSupport	"NetApp/Trident-AutoSupport:21.04.0"
<code>autosupportProxy</code>	O endereço/porta de um proxy para o envio de telemetria AutoSupport	"<a href="http://proxy.example.com:8888" class="bare">http://proxy.example.com:8888"</a>"
<code>uninstall</code>	Um sinalizador usado para desinstalar o Astra Trident	falso
<code>logFormat</code>	Formato de log Astra Trident a ser usado [text,json]	"texto"
<code>tridentImage</code>	Imagem Astra Trident a instalar	"NetApp/Trident:21,04"
<code>imageRegistry</code>	Caminho para o Registro interno, do formato <registry FQDN>[:port] [/subpath]	"k8s.gcr.io/sig-storage (mais de k8s 1,17 gb) ou quay.io/k8scsi gb"
<code>kubeletDir</code>	Caminho para o diretório kubelet no host	"/var/lib/kubelet"
<code>wipeout</code>	Uma lista de recursos a serem excluídos para realizar uma remoção completa do Astra Trident	
<code>imagePullSecrets</code>	Segredos para extrair imagens de um Registro interno	

Parâmetro	Descrição	Padrão
<code>controllerPluginNodeSelector</code>	Seletores de nós adicionais para pods executando o plug-in CSI controlador Trident. Segue o mesmo formato que <code>pod.spec.nodeSelector</code> .	Sem padrão; opcional
<code>controllerPluginTolerations</code>	Substitui as tolerâncias para pods que executam o plug-in CSI controlador Trident. Segue o mesmo formato que <code>pod.spec.Tolerations</code> .	Sem padrão; opcional
<code>nodePluginNodeSelector</code>	Seletores de nós adicionais para pods executando o plug-in CSI nó Trident. Segue o mesmo formato que <code>pod.spec.nodeSelector</code> .	Sem padrão; opcional
<code>nodePluginTolerations</code>	Substitui as tolerâncias para pods que executam o plug-in CSI nó Trident. Segue o mesmo formato que <code>pod.spec.Tolerations</code> .	Sem padrão; opcional



`spec.namespace` É especificado em `TridentOrchestrator` para indicar em que namespace Astra Trident está instalado. Este parâmetro **não pode ser atualizado depois que o Astra Trident é instalado**. Tentar fazê-lo faz com que o estado de `TridentOrchestrator` mude para `Failed`. O Astra Trident não deve ser migrado entre namespaces.



A preparação automática de nó de trabalho é um recurso **beta** destinado a ser usado apenas em ambientes não produtivos.



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

Você pode usar os atributos mencionados acima ao definir `TridentOrchestrator` para personalizar sua instalação. Aqui está um exemplo:

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

Aqui está outro exemplo que mostra 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
```

Se você estiver procurando personalizar a instalação além do que `TridentOrchestrator` os argumentos permitem, considere usar `tridentctl` para gerar manifestos YAML personalizados que você pode modificar conforme necessário.

## 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 INTERRUPTÃO 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.