Referência

Astra Trident

NetApp
January 31, 2025

This PDF was generated from https://docs.netapp.com/pt-br/trident-2307/trident-reference/ports.html on January 31, 2025. Always check docs.netapp.com for the latest.

Índice

Referência

Portas Astra Trident

Saiba mais sobre as portas que o Astra Trident usa para comunicação.

Portas Astra Trident

O Astra Trident se comunica pelas seguintes portas:

Porta	Finalidade
8443	Backchannel HTTPS
8001	Endpoint de métricas Prometheus
8000	SERVIDOR REST do Trident
17546	Porta de sonda de disponibilidade/disponibilidade usada pelos pods daemonset Trident

A porta da sonda de disponibilidade/disponibilidade pode ser alterada durante a instalação utilizando o `--probe-port` sinalizador. É importante garantir que essa porta não esteja sendo usada por outro processo nos nós de trabalho.

API REST do Astra Trident

"comandos e opções tridentctl"Embora seja a maneira mais fácil de interagir com a API REST do Astra Trident, você pode usar o endpoint REST diretamente, se preferir.

Quando usar a API REST

A API REST é útil para instalações avançadas que usam o Astra Trident como um binário autônomo em implantações não Kubernetes.

Para uma melhor segurança, o Astra Trident `REST API` é restrito a localhost por padrão ao ser executado dentro de um pod. Para alterar esse comportamento, você precisa definir o argumento do Astra Trident `-address` na configuração do pod.

Usando a API REST

A API funciona da seguinte forma:

`GET`

- `GET <trident-address>/trident/v1/<object-type>`: Lista todos os objetos desse tipo.
- `GET <trident-address>/trident/v1/<object-type>/<object-name>`: Obtém os detalhes do objeto nomeado.

`POST`

`POST <trident-address>/trident/v1/<object-type>`: Cria um objeto do tipo especificado.

- Requer uma configuração JSON para o objeto a ser criado. Para a especificação de cada tipo de objeto, consulte `tridentctl` comandos e opções.
- Se o objeto já existir, o comportamento varia: Os backends atualizam o objeto existente, enquanto todos os outros tipos de objeto falharão a operação.

`DELETE`

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`: Exclui o recurso nomeado.

Os volumes associados a backends ou classes de armazenamento continuarão a existir; estes devem ser excluídos separadamente. Para obter mais informações, consulte o `tridentctl` comandos e opções.

Para exemplos de como essas APIs são chamadas, passe o (`-d`sinalizador debug). Para obter mais informações, consulte o `tridentctl` comandos e opções.

Opções de linha de comando

O Astra Trident expõe várias opções de linha de comando para o orquestrador Trident. Você pode usar essas opções para modificar sua implantação.

A registar

- `-debug`: Ativa a saída de depuração.
- `-loglevel <level>`: Define o nível de log (debug, info, warn, error, fatal). O padrão é INFO.

Kubernetes

- `-k8s_pod`: Use esta opção ou `-k8s_api_server` para ativar o suporte do Kubernetes. Isso faz com que o Trident use suas credenciais de conta de serviço do Kubernetes do pod que contêm para entrar em Contato com o servidor de API. Isso só funciona quando o Trident é executado como um pod em um cluster do Kubernetes com contas de serviço ativadas.
- `-k8s_api_server <insecure-address:insecure-port>`: Use esta opção ou `-k8s_pod` para ativar o suporte do Kubernetes. Quando especificado, o Trident se coneta ao servidor de API do Kubernetes usando o endereço e a porta inseguros fornecidos. Isso permite que o Trident seja implantado fora de um pod; no entanto, ele só suporta conexões inseguras com o servidor de API. Para se conetar com segurança, implante o Trident em um pod com a `-k8s_pod` opção.
- `-k8s_config_path <file>`: Obrigatório; você deve especificar esse caminho para um arquivo KubeConfig.

Docker

- `-volume_driver <name>`: Nome do driver usado ao Registrar o plug-in do Docker. O padrão é `netapp`.

- `-driver_port <port-number>`: Ouça nesta porta em vez de um soquete de domínio UNIX.
- `-config <file>`: Obrigatório; você deve especificar esse caminho para um arquivo de configuração de back-end.

DESCANSO

- `-address <ip-or-host>`: Especifica o endereço no qual o servidor REST do Trident deve escutar. O padrão é localhost. Ao ouvir no localhost e executar dentro de um pod Kubernetes, a interface REST não é diretamente acessível de fora do pod. `-address ""`Utilize para tornar a INTERFACE REST acessível a partir do endereço IP do pod.

> A interface REST DO Trident pode ser configurada para ouvir e servir apenas em 127.0.0.1 (para IPv4) ou [::1] (para IPv6).

- `-port <port-number>`: Especifica a porta na qual o servidor REST do Trident deve escutar. O padrão é 8000.
- `-rest`: Ativa a interface REST. O padrão é verdadeiro.

Produtos da NetApp integrados ao Kubernetes

O portfólio de produtos de storage do NetApp se integra a muitos aspectos diferentes de um cluster Kubernetes, fornecendo recursos avançados de gerenciamento de dados que melhoram o recurso, a funcionalidade, a performance e a disponibilidade da implantação do Kubernetes.

Astra

"Astra" Facilita o gerenciamento, a proteção e a migração de workloads em contêineres com muitos dados executados no Kubernetes dentro e entre nuvens públicas e no local. O Astra provisiona e fornece storage de contêiner persistente usando o Trident do portfólio de storage comprovado e expansivo da NetApp, na nuvem pública e no local. Ele também oferece um conjunto avançado de recursos avançados de gerenciamento de dados com reconhecimento de aplicações, como snapshot, backup e restauração, logs de atividade e clonagem ativa para proteção de dados, recuperação de desastres/dados, auditoria de dados e casos de uso de migração para workloads Kubernetes.

ONTAP

O ONTAP é o sistema operacional de storage unificado multiprotocolo da NetApp que oferece recursos avançados de gerenciamento de dados para qualquer aplicação. Os sistemas ONTAP têm configurações all-flash, híbridas ou totalmente HDD e oferecem muitos modelos de implantação diferentes, incluindo hardware projetado (FAS e AFF), white-box (ONTAP Select) e somente para nuvem (Cloud Volumes ONTAP).

> O Trident oferece suporte a todos os modelos de implantação do ONTAP mencionados acima.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP" É um dispositivo de storage somente de software que executa o software de gerenciamento de dados ONTAP na nuvem. Use o Cloud Volumes ONTAP para workloads de produção, recuperação de desastres, DevOps, compartilhamentos de arquivos e gerenciamento de banco de dados. Ele

amplia o storage empresarial para a nuvem oferecendo eficiência de storage, alta disponibilidade, replicação de dados, disposição em categorias e consistência de aplicações.

Amazon FSX para NetApp ONTAP

"Amazon FSX para NetApp ONTAP" É um serviço AWS totalmente gerenciado que permite iniciar e executar sistemas de arquivos equipados com o sistema operacional de storage NetApp ONTAP. O FSX para ONTAP permite que você aproveite os recursos do NetApp, o desempenho e os recursos administrativos com os quais você já conhece, ao mesmo tempo em que aproveita a simplicidade, a agilidade, a segurança e a escalabilidade do armazenamento de dados na AWS. O FSX para ONTAP suporta muitos dos recursos do sistema de arquivos ONTAP e APIs de administração.

Software Element

"Elemento" permite que o administrador de storage consolide workloads garantindo a performance e possibilitando um espaço físico do storage simplificado e otimizado. Acoplado a uma API que permite a automação de todos os aspectos do gerenciamento de storage, o Element permite que os administradores de storage façam mais com menos esforço.

NetApp HCI

"NetApp HCI" simplifica o gerenciamento e a escala do data center automatizando tarefas de rotina e permitindo que os administradores de infraestrutura se concentrem em funções mais importantes.

O NetApp HCI é totalmente suportado pelo Trident. O Trident pode provisionar e gerenciar dispositivos de storage para aplicações em contêiner diretamente na plataforma de storage subjacente da NetApp HCI.

Azure NetApp Files

"Azure NetApp Files" É um serviço de compartilhamento de arquivos do Azure de nível empresarial, desenvolvido pela NetApp. É possível executar os workloads mais exigentes baseados em arquivos no Azure de forma nativa, com a performance e o gerenciamento de rich data que você espera do NetApp.

Cloud Volumes Service para Google Cloud

"NetApp Cloud Volumes Service para Google Cloud" É um serviço de arquivos nativo da nuvem que fornece volumes nas em NFS e SMB com performance all-flash. Esse serviço permite que qualquer workload, incluindo aplicações legadas, seja executado na nuvem do GCP. Ele fornece um serviço totalmente gerenciado que oferece alta performance consistente, clonagem instantânea, proteção de dados e acesso seguro às instâncias do Google Compute Engine (GCE).

Objetos Kubernetes e Trident

É possível interagir com o Kubernetes e o Trident usando APIs REST lendo e escrevendo objetos de recursos. Há vários objetos de recursos que ditam a relação entre o Kubernetes e o Trident, o Trident e o storage, o Kubernetes e o storage. Alguns desses objetos são gerenciados pelo Kubernetes e os outros são gerenciados pelo Trident.

Como os objetos interagem uns com os outros?

Talvez a maneira mais fácil de entender os objetos, para que eles são e como eles interagem seja seguir uma única solicitação de armazenamento de um usuário do Kubernetes:

1. Um usuário cria `PersistentVolumeClaim` uma solicitação de um novo `PersistentVolume` de um tamanho específico de um Kubernetes `StorageClass` que foi configurado anteriormente pelo administrador.
2. O Kubernetes `StorageClass` identifica o Trident como seu provisionador e inclui parâmetros que informam ao Trident como provisionar um volume para a classe solicitada.
3. O Trident olha para si `StorageClass` mesmo com o mesmo nome que identifica a correspondência `Backends` e `StoragePools` que pode usar para provisionar volumes para a classe.
4. O Trident provisiona o storage em um back-end compatível e cria dois objetos: Um `PersistentVolume` no Kubernetes que diz ao Kubernetes como encontrar, montar e tratar o volume e um volume no Trident que mantém a relação entre o `PersistentVolume` e o storage real.
5. O Kubernetes vincula `PersistentVolumeClaim` o ao novo `PersistentVolume`. Pods que incluem a `PersistentVolumeClaim` montagem que Persistentvolume em qualquer host em que ele seja executado.
6. Um usuário cria um `VolumeSnapshot` de um PVC existente, usando um `VolumeSnapshotClass` que aponta para Trident.
7. O Trident identifica o volume que está associado ao PVC e cria um snapshot do volume em seu back-end. Ele também cria um `VolumeSnapshotContent` que instrui o Kubernetes sobre como identificar o snapshot.
8. Um usuário pode criar um `PersistentVolumeClaim` usando `VolumeSnapshot` como fonte.
9. O Trident identifica o instantâneo necessário e executa o mesmo conjunto de etapas envolvidas na criação de um `PersistentVolume` e um `Volume`.

Para ler mais sobre objetos do Kubernetes, é altamente recomendável que você leia a "Volumes persistentes" seção da documentação do Kubernetes.

Objetos do Kubernetes `PersistentVolumeClaim`

Um objeto Kubernetes `PersistentVolumeClaim` é uma solicitação de storage feita por um usuário de cluster do Kubernetes.

Além da especificação padrão, o Trident permite que os usuários especifiquem as seguintes anotações específicas de volume se quiserem substituir os padrões definidos na configuração de back-end:

Anotação	Opção de volume	Drivers suportados
Trident.NetApp.io/sistema de arquivos	Sistema de ficheiros	ONTAP-san, SolidFire-san, ONTAP-san-economy
Trident.NetApp.io/cloneFromPVC	CloneSourcevolume	ONTAP-nas, ONTAP-san, SolidFire-san, azure-NetApp-files, gcp-cvs, ONTAP-san-economy
Trident.NetApp.io/splitOnClone	SplitOnClone	ONTAP-nas, ONTAP-san
Trident.NetApp.io/protocolo	protocolo	qualquer
Trident.NetApp.io/exportPolicy	Política de exportação	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup

Anotação	Opção de volume	Drivers suportados
Trident.NetApp.io/snapshotPolicy	Política de SnapshotPolicy	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup, ONTAP-san
Trident.NetApp.io/snapshotServe	SnapshotServe	ONTAP-nas, ONTAP-nas-FlexGroup, ONTAP-san, gcp-cvs
Trident.NetApp.io/snapshotDirectory	SnapshotDirectory	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup
Trident.NetApp.io/unixPermissions	UnixPermissions	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup
Trident.NetApp.io/blocksize	Tamanho do bloco	SolidFire-san

Se o PV criado tiver a `Delete` política de recuperação, o Trident excluirá o PV e o volume de backup quando o PV for liberado (ou seja, quando o usuário exclui o PVC). Caso a ação de exclusão falhe, o Trident marca o PV como tal e periodicamente tenta novamente a operação até que seja bem-sucedida ou o PV seja excluído manualmente. Se o PV usar a `Retain` política, o Trident a ignora e assume que o administrador irá limpá-la do Kubernetes e do back-end, permitindo que o volume seja feito backup ou inspecionado antes de sua remoção. Observe que a exclusão do PV não faz com que o Trident exclua o volume de backup. Você deve removê-lo usando a API REST (`tridentctl`).

O Trident dá suporte à criação de snapshots de volume usando a especificação CSI: Você pode criar um instantâneo de volume e usá-lo como fonte de dados para clonar PVCs existentes. Dessa forma, cópias pontuais de PVS podem ser expostas ao Kubernetes na forma de snapshots. Os instantâneos podem então ser usados para criar novos PVS. Dê uma olhada `On-Demand Volume Snapshots` para ver como isso funcionaria.

O Trident também fornece as `cloneFromPVC` anotações e `splitOnClone` para a criação de clones. Você pode usar essas anotações para clonar um PVC sem precisar usar a implementação do CSI.

Aqui está um exemplo: Se um usuário já tem um PVC chamado `mysql`, o usuário pode criar um novo PVC chamado `mysqlclone` usando a anotação, como `trident.netapp.io/cloneFromPVC: mysql` . Com esse conjunto de anotações, o Trident clona o volume correspondente ao PVC mysql, em vez de provisionar um volume do zero.

Considere os seguintes pontos:

- Recomendamos clonar um volume ocioso.
- O PVC e seu clone devem estar no mesmo namespace do Kubernetes e ter a mesma classe de storage.
- Com os `ontap-nas` drivers e `ontap-san`, pode ser desejável definir a anotação de PVC `trident.netapp.io/splitOnClone` em conjunto `trident.netapp.io/cloneFromPVC` com o . Com `trident.netapp.io/splitOnClone` definido como `true`, o Trident divide o volume clonado do volume pai e, portanto, desacoplando completamente o ciclo de vida do volume clonado de seus pais às custas de perder alguma eficiência de storage. Não `trident.netapp.io/splitOnClone` configurá-lo ou configurá-lo para `false` resultar em consumo de espaço reduzido no back-end à custa de criar dependências entre os volumes pai e clone, de modo que o volume pai não possa ser excluído a menos que o clone seja excluído primeiro. Um cenário em que dividir o clone faz sentido é clonar um volume de banco de dados vazio, onde é esperado que o volume e seu clone diverjam muito e não se beneficiem das eficiências de armazenamento oferecidas pelo ONTAP.

O `sample-input` diretório contém exemplos de definições de PVC para uso com Trident. Consulte a para

obter uma descrição completa dos parâmetros e definições associados aos volumes Trident.

Objetos do Kubernetes `PersistentVolume`

Um objeto Kubernetes `PersistentVolume` representa um storage disponibilizado para o cluster do Kubernetes. Ele tem um ciclo de vida que é independente do pod que o usa.

> O Trident cria `PersistentVolume` objetos e os Registra no cluster do Kubernetes automaticamente com base nos volumes provisionados. Você não é esperado para gerenciá-los sozinho.

Quando você cria um PVC que se refere a um Trident-based `StorageClass`, o Trident provisiona um novo volume usando a classe de armazenamento correspondente e Registra um novo PV para esse volume. Ao configurar o volume provisionado e o PV correspondente, o Trident segue as seguintes regras:

- O Trident gera um nome PV para o Kubernetes e um nome interno que ele usa para provisionar o storage. Em ambos os casos, é garantir que os nomes são únicos em seu escopo.
- O tamanho do volume corresponde ao tamanho solicitado no PVC o mais próximo possível, embora possa ser arredondado para a quantidade alocável mais próxima, dependendo da plataforma.

Objetos do Kubernetes `StorageClass`

Os objetos Kubernetes `StorageClass` são especificados por nome em `PersistentVolumeClaims` para provisionar o storage com um conjunto de propriedades. A própria classe de storage identifica o provisionador a ser usado e define esse conjunto de propriedades em termos que o provisionador entende.

É um dos dois objetos básicos que precisam ser criados e gerenciados pelo administrador. O outro é o objeto backend do Trident.

Um objeto do Kubernetes `StorageClass` que usa o Trident é parecido com este:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters:
  <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Esses parâmetros são específicos do Trident e informam à Trident como provisionar volumes para a classe.

Os parâmetros da classe de armazenamento são:

Atributo	Tipo	Obrigatório	Descrição
atributos	map[string]string	não	Veja a seção atributos abaixo
StoragePools	MAP[string]StringList	não	Mapa de nomes de back-end para listas de pools de armazenamento dentro
Além disso, StoragePools	MAP[string]StringList	não	Mapa de nomes de back-end para listas de pools de armazenamento dentro
Excluir StoragePools	MAP[string]StringList	não	Mapa de nomes de back-end para listas de pools de armazenamento dentro

Os atributos de storage e seus possíveis valores podem ser classificados em atributos de seleção de pool de storage e atributos do Kubernetes.

Atributos de seleção do pool de armazenamento

Esses parâmetros determinam quais pools de storage gerenciado pelo Trident devem ser utilizados para provisionar volumes de um determinado tipo.

Atributo	Tipo	Valores	Oferta	Pedido	Suportado por
1	cadeia de carateres	hdd, híbrido, ssd	Pool contém Mídia desse tipo; híbrido significa ambos	Tipo de material especificado	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup, ONTAP-san, SolidFire-san
ProvisioningType	cadeia de carateres	fino, grosso	O pool é compatível com esse método de provisionamento	Método de provisionamento especificado	thick: all ONTAP; thin: all ONTAP & SolidFire-san
BackendType	cadeia de carateres	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup, ONTAP-san, SolidFire-san, gcp-cvs, azure-NetApp-files, ONTAP-san-economy	Pool pertence a este tipo de backend	Back-end especificado	Todos os drivers
instantâneos	bool	verdadeiro, falso	O pool é compatível com volumes com snapshots	Volume com instantâneos ativados	ONTAP-nas, ONTAP-san, SolidFire-san, gcp-cvs

Atributo	Tipo	Valores	Oferta	Pedido	Suportado por
clones	bool	verdadeiro, falso	O pool é compatível com volumes de clonagem	Volume com clones ativados	ONTAP-nas, ONTAP-san, SolidFire-san, gcp-cvs
criptografia	bool	verdadeiro, falso	O pool é compatível com volumes criptografados	Volume com encriptação ativada	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-flexgroups, ONTAP-san
IOPS	int	número inteiro positivo	O pool é capaz de garantir IOPS nessa faixa	Volume garantido estas operações de entrada/saída por segundo	SolidFire-san

1: Não suportado pelos sistemas ONTAP Select

Na maioria dos casos, os valores solicitados influenciam diretamente o provisionamento; por exemplo, a solicitação de provisionamento espesso resulta em um volume provisionado rapidamente. No entanto, um pool de storage de elemento usa o mínimo e o máximo de IOPS oferecidos para definir valores de QoS, em vez do valor solicitado. Nesse caso, o valor solicitado é usado apenas para selecionar o pool de armazenamento.

O ideal é usar `attributes` sozinho para modelar as qualidades do storage de que você precisa para atender às necessidades de uma classe específica. O Trident deteta e seleciona automaticamente pools de armazenamento que correspondem a *all* do `attributes` que você especificar.

Se você não conseguir usar `attributes` para selecionar automaticamente os pools certos para uma classe, use os `storagePools` parâmetros e `additionalStoragePools` para refinar ainda mais os pools ou até mesmo selecionar um conjunto específico de pools.

Você pode usar o `storagePools` parâmetro para restringir ainda mais o conjunto de pools que correspondem a qualquer `attributes` especificado . Em outras palavras, o Trident usa a interseção de pools identificados pelos `attributes` parâmetros e `storagePools` para o provisionamento. Você pode usar um parâmetro sozinho ou ambos juntos.

Você pode usar o `additionalStoragePools` parâmetro para estender o conjunto de pools que o Trident usa para provisionamento, independentemente de quaisquer pools selecionados pelos `attributes` parâmetros e. `storagePools`

Você pode usar o `excludeStoragePools` parâmetro para filtrar o conjunto de pools que o Trident usa para provisionar. O uso desse parâmetro remove todos os pools que correspondem.

`storagePools`Nos parâmetros e `additionalStoragePools`, cada entrada assume o formulário `<backend>:<storagePoolList>`, onde `<storagePoolList>` é uma lista separada por vírgulas de pools de armazenamento para o back-end especificado. Por exemplo, um valor para `additionalStoragePools` pode parecer como `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`. Essas listas aceitam valores de regex tanto para os valores de backend quanto de lista. Você pode usar `tridentctl get backend` para obter a lista de backends e suas piscinas.

Atributos do Kubernetes

Esses atributos não têm impacto na seleção de pools de storage/back-ends pelo Trident durante o provisionamento dinâmico. Em vez disso, esses atributos simplesmente fornecem parâmetros compatíveis com volumes persistentes do Kubernetes. Os nós de trabalho são responsáveis pelas operações de criação de sistema de arquivos e podem exigir utilitários de sistema de arquivos, como xfsprogs.

Atributo	Tipo	Valores	Descrição	Drivers relevantes	Versão do Kubernetes
FsType	cadeia de carateres	ext4, ext3, xfs, etc.	O tipo de sistema de arquivos para volumes de bloco	SolidFire-san, ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup, ONTAP-san, ONTAP-san-economy	Tudo
AllowVolumeExpansion	booleano	verdadeiro, falso	Ative ou desative o suporte para aumentar o tamanho do PVC	ONTAP-nas, ONTAP-nas-economy, ONTAP-nas-FlexGroup, ONTAP-san, ONTAP-san-economy, SolidFire-san, gcp-cvs, azure-NetApp-files	Mais de 1,11 anos
VolumeBindingMode	cadeia de carateres	Imediato, WaitForFirstConsumer	Escolha quando ocorre a vinculação de volume e o provisionamento dinâmico	Tudo	1,19 - 1,26

- O `fsType` parâmetro é usado para controlar o tipo de sistema de arquivos desejado para LUNs SAN. Além disso, o Kubernetes também usa a presença de `fsType` em uma classe de armazenamento para indicar que existe um sistema de arquivos. A propriedade do volume só pode ser controlada usando o `fsGroup` contexto de segurança de um pod se `fsType` estiver definido. Consulte "Kubernetes: Configurar um contexto de segurança para um pod ou contêiner" para obter uma visão geral sobre como definir a propriedade do volume usando o `fsGroup` contexto. O Kubernetes aplicará o `fsGroup` valor somente se:

  - `fsType` é definido na classe de armazenamento.
  - O modo de acesso de PVC é RWO.

  Para drivers de armazenamento NFS, já existe um sistema de arquivos como parte da exportação NFS. Para usar `fsGroup` a classe de armazenamento ainda precisa especificar um `fsType`. você pode configurá-lo como `nfs` ou qualquer valor não nulo.

- "Expanda volumes"Consulte para obter mais detalhes sobre a expansão de volume.
- O pacote de instalação do Trident fornece vários exemplos de definições de classe de armazenamento para uso com o Trident no `sample-input/storage-class-*.yaml`. A exclusão de uma classe de armazenamento Kubernetes faz com que a classe de armazenamento Trident correspondente também seja excluída.

Objetos do Kubernetes `VolumeSnapshotClass`

Os objetos do Kubernetes `VolumeSnapshotClass` são análogos ao `StorageClasses`. Eles ajudam a definir várias classes de armazenamento e são referenciados por instantâneos de volume para associar o snapshot à classe de snapshot necessária. Cada snapshot de volume é associado a uma classe de snapshot de volume único.

A `VolumeSnapshotClass` deve ser definida por um administrador para criar instantâneos. Uma classe de instantâneo de volume é criada com a seguinte definição:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

O `driver` especifica ao Kubernetes que as solicitações de snapshots de volume `csi-snapclass` da classe são tratadas pelo Trident. O `deletionPolicy` especifica a ação a ser tomada quando um instantâneo deve ser excluído. `deletionPolicy`Quando está definido como `Delete`, os objetos instantâneos de volume e o instantâneo subjacente no cluster de armazenamento são removidos quando um instantâneo é excluído. Alternativamente, configurá-lo para `Retain` significa que `VolumeSnapshotContent` e o instantâneo físico são retidos.

Objetos do Kubernetes `VolumeSnapshot`

Um objeto Kubernetes `VolumeSnapshot` é uma solicitação para criar um snapshot de um volume. Assim como um PVC representa uma solicitação feita por um usuário para um volume, um instantâneo de volume é

uma solicitação feita por um usuário para criar um instantâneo de um PVC existente.

Quando uma solicitação de snapshot de volume entra, o Trident gerencia automaticamente a criação do snapshot para o volume no back-end e expõe o snapshot criando um objeto exclusivo `VolumeSnapshotContent`. Você pode criar snapshots a partir de PVCs existentes e usar os snapshots como DataSource ao criar novos PVCs.

> A vida útil de um VolumeSnapshot é independente do PVC de origem: Um snapshot persiste mesmo depois que o PVC de origem é excluído. Ao excluir um PVC que tenha instantâneos associados, o Trident marca o volume de apoio para este PVC em um estado **Deletando**, mas não o remove completamente. O volume é removido quando todos os instantâneos associados são excluídos.

Objetos do Kubernetes `VolumeSnapshotContent`

Um objeto Kubernetes `VolumeSnapshotContent` representa um snapshot retirado de um volume já provisionado. Ele é análogo a `PersistentVolume` e significa um snapshot provisionado no cluster de storage. Semelhante aos `PersistentVolumeClaim` objetos e `PersistentVolume`, quando um snapshot é criado, o `VolumeSnapshotContent` objeto mantém um mapeamento um-para-um para o `VolumeSnapshot` objeto, que havia solicitado a criação do snapshot.

O `VolumeSnapshotContent` objeto contém detalhes que identificam exclusivamente o instantâneo, como o `snapshotHandle`. Esta `snapshotHandle` é uma combinação única do nome do PV e do nome do `VolumeSnapshotContent` objeto.

Quando uma solicitação de snapshot entra, o Trident cria o snapshot no back-end. Depois que o snapshot é criado, o Trident configura um `VolumeSnapshotContent` objeto e, portanto, expõe o snapshot à API do Kubernetes.

> Normalmente, você não precisa gerenciar o `VolumeSnapshotContent` objeto. Uma exceção a isso é quando você deseja "importar um instantâneo de volume"criar fora do Astra Trident.

Objetos do Kubernetes `CustomResourceDefinition`

Os recursos personalizados do Kubernetes são endpoints na API do Kubernetes que são definidos pelo administrador e são usados para agrupar objetos semelhantes. O Kubernetes dá suporte à criação de recursos personalizados para armazenar uma coleção de objetos. Você pode obter essas definições de recursos executando `kubectl get crds`o .

As definições personalizadas de recursos (CRDs) e os metadados de objetos associados são armazenados pelo Kubernetes em seu armazenamento de metadados. Isso elimina a necessidade de uma loja separada para o Trident.

O Astra Trident usa `CustomResourceDefinition` objetos para preservar a identidade de objetos Trident, como back-ends Trident, classes de storage Trident e volumes Trident. Esses objetos são gerenciados pelo Trident. Além disso, a estrutura de snapshot do volume CSI introduz algumas CRDs que são necessárias para definir snapshots de volume.

CRDs são uma construção do Kubernetes. Os objetos dos recursos definidos acima são criados pelo Trident. Como um exemplo simples, quando um back-end é criado usando `tridentctl`o , um objeto CRD correspondente `tridentbackends` é criado para consumo pelo Kubernetes.

Aqui estão alguns pontos a ter em mente sobre os CRDs do Trident:

- Quando o Trident é instalado, um conjunto de CRDs é criado e pode ser usado como qualquer outro tipo de recurso.
- Ao desinstalar o Trident usando o `tridentctl uninstall` comando, os pods Trident são excluídos, mas os CRDs criados não são limpos. Veja "Desinstale o Trident" para entender como o Trident pode ser completamente removido e reconfigurado do zero.

Objetos Astra Trident `StorageClass`

O Trident cria classes de storage correspondentes para objetos Kubernetes `StorageClass` que especificam `csi.trident.netapp.io/netapp.io/trident` no campo do provisionador. O nome da classe de storage corresponde ao do objeto Kubernetes `StorageClass` que ele representa.

> Com o Kubernetes, esses objetos são criados automaticamente quando um Kubernetes `StorageClass` que usa o Trident como provisionador é registrado.

As classes de armazenamento compreendem um conjunto de requisitos para volumes. O Trident atende a esses requisitos com os atributos presentes em cada pool de storage. Se forem correspondentes, esse pool de storage será um destino válido para volumes de provisionamento que usam essa classe de storage.

Você pode criar configurações de classe de armazenamento para definir diretamente classes de armazenamento usando a API REST. No entanto, para implantações do Kubernetes, esperamos que elas sejam criadas ao Registrar novos objetos do Kubernetes `StorageClass`.

Objetos de back-end do Astra Trident

Os backends representam os fornecedores de storage em cima dos quais o Trident provisiona volumes. Uma única instância do Trident pode gerenciar qualquer número de backends.

> Este é um dos dois tipos de objetos que você cria e gerencia a si mesmo. O outro é o objeto Kubernetes `StorageClass`.

Para obter mais informações sobre como construir esses objetos, "configurando backends"consulte .

Objetos Astra Trident `StoragePool`

Os pools de storage representam locais distintos disponíveis para provisionamento em cada back-end. Para ONTAP, eles correspondem a agregados em SVMs. Para NetApp HCI/SolidFire, estes correspondem a bandas de QoS especificadas pelo administrador. Para o Cloud Volumes Service, eles correspondem a regiões de provedores de nuvem. Cada pool de storage tem um conjunto de atributos de storage distintos, que definem suas características de performance e proteção de dados.

Ao contrário dos outros objetos aqui, os candidatos ao pool de armazenamento são sempre descobertos e gerenciados automaticamente.

Objetos Astra Trident `Volume`

Os volumes são a unidade básica de provisionamento, incluindo pontos de extremidade de back-end, como compartilhamentos NFS e iSCSI LUNs. No Kubernetes, eles correspondem diretamente `PersistentVolumes` ao . Ao criar um volume, certifique-se de que ele tenha uma classe de

armazenamento, que determina onde esse volume pode ser provisionado, juntamente com um tamanho.

- No Kubernetes, esses objetos são gerenciados automaticamente. Você pode visualizá-los para ver o que o Trident provisionou.
- Ao excluir um PV com instantâneos associados, o volume Trident correspondente é atualizado para um estado **Deletando**. Para que o volume Trident seja excluído, você deve remover os snapshots do volume.

Uma configuração de volume define as propriedades que um volume provisionado deve ter.

Atributo	Tipo	Obrigatório	Descrição
versão	cadeia de carateres	não	Versão da API Trident ("1")
nome	cadeia de carateres	sim	Nome do volume a criar
StorageClass	cadeia de carateres	sim	Classe de storage a ser usada ao provisionar o volume
tamanho	cadeia de carateres	sim	Tamanho do volume a provisionar em bytes
protocolo	cadeia de carateres	não	Tipo de protocolo a utilizar; "ficheiro" ou "bloco"
InternalName	cadeia de carateres	não	Nome do objeto no sistema de storage; gerado pelo Trident
CloneSourcevolume	cadeia de carateres	não	ONTAP (nas, san) & SolidFire-*: Nome do volume a partir do qual clonar
SplitOnClone	cadeia de carateres	não	ONTAP (nas, san): Divida o clone de seu pai
Política de SnapshotPolicy	cadeia de carateres	não	ONTAP-*: Política de snapshot a ser usada
SnapshotServe	cadeia de carateres	não	ONTAP-*: Porcentagem de volume reservado para snapshots
Política de exportação	cadeia de carateres	não	ONTAP-nas*: Política de exportação para usar
SnapshotDirectory	bool	não	ONTAP-nas*: Se o diretório snapshot está visível
UnixPermissions	cadeia de carateres	não	ONTAP-nas*: Permissões iniciais do UNIX
Tamanho do bloco	cadeia de carateres	não	SolidFire-*: Tamanho do bloco/setor

Atributo	Tipo	Obrigatório	Descrição
Sistema de ficheiros	cadeia de carateres	não	Tipo de sistema de ficheiros

O Trident gera `internalName` ao criar o volume. Isto consiste em duas etapas. Primeiro, ele prepende o prefixo de armazenamento (o padrão `trident` ou o prefixo na configuração de back-end) para o nome do volume, resultando em um nome do formulário `<prefix>-<volume-name>`. Em seguida, procede à higienização do nome, substituindo carateres não permitidos no backend. Para backends ONTAP, ele substitui hífens por sublinhados (assim, o nome interno se torna `<prefix>_<volume-name>`). Para backends de elemento, ele substitui sublinhados por hífens.

Você pode usar configurações de volume para provisionar volumes diretamente usando a API REST, mas nas implantações do Kubernetes, esperamos que a maioria dos usuários use o método padrão do Kubernetes `PersistentVolumeClaim`. O Trident cria esse objeto de volume automaticamente como parte do processo de provisionamento.

Objetos Astra Trident `Snapshot`

Os snapshots são uma cópia pontual de volumes, que pode ser usada para provisionar novos volumes ou restaurar o estado. No Kubernetes, eles correspondem diretamente a `VolumeSnapshotContent` objetos. Cada snapshot é associado a um volume, que é a origem dos dados do snapshot.

Cada `Snapshot` objeto inclui as propriedades listadas abaixo:

Atributo	Tipo	Obrigatório	Descrição
versão	Cadeia de carateres	Sim	Versão da API Trident ("1")
nome	Cadeia de carateres	Sim	Nome do objeto snapshot Trident
InternalName	Cadeia de carateres	Sim	Nome do objeto snapshot Trident no sistema de storage
Nome do volume	Cadeia de carateres	Sim	Nome do volume persistente para o qual o instantâneo é criado
VolumeInternalName	Cadeia de carateres	Sim	Nome do objeto de volume Trident associado no sistema de storage

No Kubernetes, esses objetos são gerenciados automaticamente. Você pode visualizá-los para ver o que o Trident provisionou.

Quando uma solicitação de objeto Kubernetes `VolumeSnapshot` é criada, o Trident funciona criando um objeto snapshot no sistema de storage de backup. `internalName`O deste objeto instantâneo é gerado combinando o prefixo `snapshot-` com o `UID` do `VolumeSnapshot` objeto (por exemplo, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` e `volumeInternalName` são preenchidos obtendo os detalhes do volume de apoio.

Objeto Astra Trident `ResourceQuota`

O deamonset do Trident consome uma `system-node-critical` classe de prioridade - a classe de prioridade mais alta disponível no Kubernetes - para garantir que o Astra Trident possa identificar e limpar volumes durante o desligamento gracioso do nó e permitir que os pods do Trident daemonset pré-empt cargas de trabalho com prioridade mais baixa em clusters onde há alta pressão de recursos.

Para conseguir isso, o Astra Trident emprega um `ResourceQuota` objeto para garantir que uma classe de prioridade "crítica do nó do sistema" no daemonset do Trident esteja satisfeita. Antes da implantação e criação do daemonset, o Astra Trident procura o `ResourceQuota` objeto e, se não for descoberto, o aplica.

Se você precisar de mais controle sobre a cota de recurso padrão e Classe de prioridade, você pode gerar um `custom.yaml` ou configurar o `ResourceQuota` objeto usando o gráfico de Helm.

O seguinte é um exemplo de um objeto 'ResourceQuota' priorizando o daemonset do Trident.

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
     matchExpressions:
       - operator : In
         scopeName: PriorityClass
         values: ["system-node-critical"]
```

Para obter mais informações sobre cotas de recursos, "Kubernetes: Cotas de recursos"consulte .

Limpe `ResourceQuota` se a instalação falhar

No caso raro em que a instalação falha depois que o `ResourceQuota` objeto é criado, primeiro "desinstalação"tente e depois reinstale.

Se isso não funcionar, remova manualmente o `ResourceQuota` objeto.

Retire `ResourceQuota`

Se você preferir controlar sua própria alocação de recursos, poderá remover o objeto Astra Trident `ResourceQuota` usando o comando:

```
kubectl delete quota trident-csi -n trident
```

comandos e opções tridentctl

O "Pacote de instalação do Trident" inclui um utilitário de linha de comando `tridentctl`, que fornece acesso simples ao Astra Trident. Usuários do Kubernetes com Privileges suficiente podem usá-lo para instalar o Astra Trident e interagir diretamente com ele para gerenciar o namespace que contém o pod Astra Trident.

Comandos e opções disponíveis

Para obter informações de uso, execute `tridentctl --help`o .

Os comandos disponíveis e as opções globais são:

```
Usage:
  tridentctl [command]
```

Comandos disponíveis:

- `create`: Adicionar um recurso ao Astra Trident.
- `delete`: Remova um ou mais recursos do Astra Trident.
- `get`: Obtenha um ou mais recursos do Astra Trident.
- `help`: Ajuda sobre qualquer comando.
- `images`: Imprimir uma tabela das imagens de contêiner que o Astra Trident precisa.
- `import`: Importar um recurso existente para o Astra Trident.
- `install`: Instalar o Astra Trident.
- `logs`: Imprima os logs do Astra Trident.
- `send`: Enviar um recurso de Astra Trident.
- `uninstall`: Desinstalar Astra Trident.
- `update`: Modificar um recurso no Astra Trident.
- `upgrade`: Atualizar um recurso no Astra Trident.
- `version`: Imprimir a versão do Astra Trident.

Bandeiras -

- `-d, --debug`: Saída de depuração.
- `-h, --help`: Ajuda para `tridentctl`.
- `-n, --namespace string`: Namespace da implantação do Astra Trident.
- `-o, --output string`: Formato de saída. Um de JSON|yaml|name|wide|ps (padrão).
- `-s, --server string`: Endereço/porta da interface REST do Astra Trident.

A interface REST DO Trident pode ser configurada para ouvir e servir apenas em 127.0.0.1 (para IPv4) ou [::1] (para IPv6).

A interface REST DO Trident pode ser configurada para ouvir e servir apenas em 127.0.0.1 (para IPv4) ou [::1] (para IPv6).

`create`

Você pode usar executar `create` o comando para adicionar um recurso ao Astra Trident.

```
Usage:
  tridentctl create [option]
```

Opção disponível
`backend`: : Adicionar um back-end ao Astra Trident.

`delete`

Você pode executar o `delete` comando para remover um ou mais recursos do Astra Trident.

```
Usage:
  tridentctl delete [option]
```

Opções disponíveis:

- `backend`: Excluir um ou mais back-ends de storage do Astra Trident.
- `snapshot`: Excluir um ou mais snapshots de volume do Astra Trident.
- `storageclass`: Excluir uma ou mais classes de storage do Astra Trident.
- `volume`: Excluir um ou mais volumes de storage do Astra Trident.

`get`

Você pode executar o `get` comando para obter um ou mais recursos do Astra Trident.

```
Usage:
  tridentctl get [option]
```

Opções disponíveis:

- `backend`: Obtenha um ou mais back-ends de storage do Astra Trident.
- `snapshot`: Obtenha um ou mais snapshots do Astra Trident.
- `storageclass`: Obtenha uma ou mais classes de storage do Astra Trident.

- `volume`: Obtenha um ou mais volumes do Astra Trident.

`volume` Bandeiras: * `-h, --help`: Ajuda para volumes. `--parentOfSubordinate string`* : Limitar consulta ao volume de origem subordinado. `--subordinateOf string`* : Limitar consulta a subordinados de volume.

images

Você pode executar `images` o sinalizador para imprimir uma tabela das imagens de contentor que o Astra Trident precisa.

```
Usage:
  tridentctl images [flags]
```

Flags: `-h, --help`: Help for images.
* * -v, --k8s-version string': Versão semântica do cluster do Kubernetes.

import volume

Você pode executar o `import volume` comando para importar um volume existente para o Astra Trident.

```
Usage:
  tridentctl import volume <backendName> <volumeName> [flags]
```

Alias:
`volume, v`

Bandeiras -

- `-f, --filename string`: Caminho para o arquivo PVC YAML ou JSON.
- `-h, --help`: Ajuda para volume.
- `--no-manage`: Criar apenas PV/PVC. Não assuma o gerenciamento do ciclo de vida do volume.

install

Você pode executar `install` os sinalizadores para instalar o Astra Trident.

```
Usage:
  tridentctl install [flags]
```

Bandeiras -

- `--autosupport-image string`: A imagem do contentor para telemetria AutoSupport (predefinição "NetApp/Trident AutoSupport:<current-version>").

- `--autosupport-proxy string`: O endereço/porta de um proxy para o envio de telemetria AutoSupport.
- `--enable-node-prep`: Tentativa de instalar os pacotes necessários nos nós.
- `--generate-custom-yaml`: Gere arquivos YAML sem instalar nada.
- `-h, --help`: Ajuda para instalar.
- `--http-request-timeout`: Substituir o tempo limite da solicitação HTTP para a API REST do controlador Trident (1m30s padrão).
- `--image-registry string`: O endereço/porta de um Registro de imagem interno.
- `--k8s-timeout duration`: O tempo limite para todas as operações do Kubernetes (3m0s padrão).
- `--kubelet-dir string`: A localização do host do estado interno do kubelet (padrão "/var/lib/kubelet").
- `--log-format string`: O formato de log Astra Trident (texto, json) (texto padrão).
- `--pv string`: O nome do PV legado usado pelo Astra Trident garante que isso não existe (padrão "Trident").
- `--pvc string`: O nome do PVC legado usado pelo Astra Trident garante que isso não existe (padrão "Trident").
- `--silence-autosupport`: Não envie pacotes AutoSupport automaticamente para o NetApp (padrão verdadeiro).
- `--silent`: Desativar a saída MOST durante a instalação.
- `--trident-image string`: A imagem Astra Trident a instalar.
- `--use-custom-yaml`: Use todos os arquivos YAML existentes que existem no diretório de configuração.
- `--use-ipv6`: Utilizar o IPv6 para a comunicação do Astra Trident.

`logs`

Você pode executar `logs` os sinalizadores para imprimir os logs do Astra Trident.

```
Usage:
  tridentctl logs [flags]
```

Bandeiras -

- `` `-a, --archive``: Crie um arquivo de suporte com todos os logs, a menos que especificado de outra forma.
- `` `-h, --help``: Ajuda para logs.
- `` `-l, --log string``: Log do Astra Trident para exibição. Um dos Trident|auto|Trident-operator|All (predefinição "auto").
- `` `--node string``: O nome do nó Kubernetes do qual você pode coletar logs do pod de nó.
- `` `-p, --previous``: Obter os logs para a instância de contentor anterior, se ela existir.
- `` `--sidecars``: Obter os logs para os recipientes sidecar.

send

Você pode executar o `send` comando para enviar um recurso do Astra Trident.

```
Usage:
  tridentctl send [option]
```

Opção disponível
`autosupport`: : Enviar um arquivo AutoSupport para o NetApp.

uninstall

Você pode executar `uninstall` os sinalizadores para desinstalar o Astra Trident.

```
Usage:
  tridentctl uninstall [flags]
```

Bandeiras: * `-h, --help`: Ajuda para desinstalar. * `--silent` : Desative a saída MOST durante a desinstalação.

update

Você pode executar os `update` comandos para modificar um recurso no Astra Trident.

```
Usage:
  tridentctl update [option]
```

Opções disponíveis
`backend`: : Atualize um back-end no Astra Trident.

version

Você pode executar `version` os sinalizadores para imprimir a versão do `tridentctl` e o serviço Trident em execução.

```
Usage:
  tridentctl version [flags]
```

Bandeiras: * `--client`: Apenas versão do cliente (nenhum servidor necessário). `-h, --help`* : Ajuda para a versão.

Padrões de segurança do pod (PSS) e restrições de contexto de segurança (SCC)

Os padrões de segurança do pod do Kubernetes (PSS) e as políticas de segurança do Pod (PSP) definem níveis de permissão e restringem o comportamento dos pods. As restrições de contexto de Segurança OpenShift (SCC) definem similarmente a restrição de pod específica ao OpenShift Kubernetes Engine. Para oferecer essa personalização, o Astra Trident habilita certas permissões durante a instalação. As seções a seguir detalham as permissões definidas pelo Astra Trident.

> O PSS substitui as políticas de segurança do Pod (PSP). A PSP foi obsoleta no Kubernetes v1,21 e será removida em v1,25. Para obter mais informações, "Kubernetes: Segurança"consulte .

Contexto de segurança do Kubernetes necessário e campos relacionados

Permissão	Descrição
Privilegiado	O CSI exige que os pontos de montagem sejam bidirecionais, o que significa que o pod de nó do Trident deve executar um contentor privilegiado. Para obter mais informações, "Kubernetes: Propagação de montagem"consulte .
Rede de host	Necessário para o daemon iSCSI. `iscsiadm` Gerencia montagens iSCSI e usa redes de host para se comunicar com o daemon iSCSI.
IPC do host	O NFS usa comunicação entre processos (IPC) para se comunicar com o NFSD.
PID do host	Necessário para iniciar `rpc-statd` o NFS. O Astra Trident consulta os processos de host para determinar se `rpc-statd` está em execução antes da montagem de volumes NFS.

Recursos	O `SYS_ADMIN` recurso é fornecido como parte dos recursos padrão para contentores privilegiados. Por exemplo, o Docker define esses recursos para contentores privilegiados:
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

Seccomp	O perfil Seccomp é sempre "unconfinado" em contentores privilegiados; portanto, não pode ser habilitado no Astra Trident.

Permissão	Descrição
SELinux	No OpenShift, os contentores privilegiados são executados no `spc_t` domínio ("contentor Super privilegiado") e os contentores sem privilégios são executados `container_t` no domínio. No `containerd`, com `container-selinux` instalado, todos os contentores são executados no `spc_t` domínio, o que desativa efetivamente o SELinux. Portanto, o Astra Trident não é adicionado `seLinuxOptions` a contêineres.
DAC	Os contentores privilegiados devem ser executados como root. Os contentores não privilegiados são executados como root para acessar os sockets unix exigidos pelo CSI.

Padrões de segurança do pod (PSS)

Etiqueta	Descrição	Padrão
`pod-security.kubernetes.io/enforce` `pod-security.kubernetes.io/enforce-version`	Permite que o controlador Trident e os nós sejam admitidos no namespace de instalação. Não altere a etiqueta do namespace.	`enforce: privileged enforce-version: <version of the current cluster or highest version of PSS tested.>`

Alterar os rótulos do namespace pode resultar em pods não sendo programados, um "erro ao criar: …" ou, "Aviso: Trident-csi-…". Se isso acontecer, verifique se a etiqueta do namespace para `privileged` foi alterada. Em caso afirmativo, reinstale o Trident.

Políticas de segurança do pod (PSP)

Campo	Descrição	Padrão
`allowPrivilegeEscalation`	Os contêineres privilegiados devem permitir o escalonamento de privilégios.	`true`
`allowedCSIDrivers`	O Trident não usa volumes efêmeros de CSI inline.	Vazio
`allowedCapabilities`	Os contêineres Trident não privilegiados não exigem mais recursos do que o conjunto padrão e os contentores privilegiados recebem todos os recursos possíveis.	Vazio
`allowedFlexVolumes`	O Trident não faz uso de um "Controlador Flexvolume", portanto, eles não estão incluídos na lista de volumes permitidos.	Vazio

Campo	Descrição	Padrão
`allowedHostPaths`	O pod de nó Trident monta o sistema de arquivos raiz do nó, portanto, não há benefício para definir esta lista.	Vazio
`allowedProcMountTypes`	O Trident não usa nenhum `ProcMountTypes`.	Vazio
`allowedUnsafeSysctls`	O Trident não requer nenhum inseguro `sysctls`.	Vazio
`defaultAddCapabilities`	Não são necessários recursos para serem adicionados a contentores privilegiados.	Vazio
`defaultAllowPrivilegeEscalation`	Permitir o escalonamento de privilégios é Tratado em cada pod Trident.	`false`
`forbiddenSysctls`	Não `sysctls` são permitidos.	Vazio
`fsGroup`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`hostIPC`	A montagem de volumes NFS requer que o IPC do host se comunique com `nfsd`	`true`
`hostNetwork`	O iscsiadm requer que a rede host se comunique com o daemon iSCSI.	`true`
`hostPID`	O PID do host é necessário para verificar se `rpc-statd` está sendo executado no nó.	`true`
`hostPorts`	O Trident não usa nenhuma porta de host.	Vazio
`privileged`	Os pods de nós do Trident devem executar um contêiner privilegiado para montar volumes.	`true`
`readOnlyRootFilesystem`	Os pods de nós do Trident devem gravar no sistema de arquivos do nó.	`false`
`requiredDropCapabilities`	Os pods de nós do Trident executam um contêiner privilegiado e não podem descartar recursos.	`none`
`runAsGroup`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`runAsUser`	Os contêineres do Trident são executados como raiz.	`runAsAny`
`runtimeClass`	O Trident não usa `RuntimeClasses`o .	Vazio

Campo	Descrição	Padrão
`seLinux`	O Trident não define `seLinuxOptions` porque atualmente existem diferenças em como os tempos de execução de contêineres e as distribuições do Kubernetes lidam com o SELinux.	Vazio
`supplementalGroups`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`volumes`	Os pods do Trident exigem esses plugins de volume.	`hostPath, projected, emptyDir`

Restrições de contexto de segurança (SCC)

Etiquetas	Descrição	Padrão
`allowHostDirVolumePlugin`	Os pods de nó Trident montam o sistema de arquivos raiz do nó.	`true`
`allowHostIPC`	A montagem de volumes NFS requer que o IPC do host se comunique com `nfsd`o .	`true`
`allowHostNetwork`	O iscsiadm requer que a rede host se comunique com o daemon iSCSI.	`true`
`allowHostPID`	O PID do host é necessário para verificar se `rpc-statd` está sendo executado no nó.	`true`
`allowHostPorts`	O Trident não usa nenhuma porta de host.	`false`
`allowPrivilegeEscalation`	Os contêineres privilegiados devem permitir o escalonamento de privilégios.	`true`
`allowPrivilegedContainer`	Os pods de nós do Trident devem executar um contêiner privilegiado para montar volumes.	`true`
`allowedUnsafeSysctls`	O Trident não requer nenhum inseguro `sysctls`.	`none`
`allowedCapabilities`	Os contêineres Trident não privilegiados não exigem mais recursos do que o conjunto padrão e os contentores privilegiados recebem todos os recursos possíveis.	Vazio
`defaultAddCapabilities`	Não são necessários recursos para serem adicionados a contentores privilegiados.	Vazio

Etiquetas	Descrição	Padrão
`fsGroup`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`groups`	Este SCC é específico do Trident e está vinculado ao seu usuário.	Vazio
`readOnlyRootFilesystem`	Os pods de nós do Trident devem gravar no sistema de arquivos do nó.	`false`
`requiredDropCapabilities`	Os pods de nós do Trident executam um contêiner privilegiado e não podem descartar recursos.	`none`
`runAsUser`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`seLinuxContext`	O Trident não define `seLinuxOptions` porque atualmente existem diferenças em como os tempos de execução de contêineres e as distribuições do Kubernetes lidam com o SELinux.	Vazio
`seccompProfiles`	Os contentores privilegiados funcionam sempre "sem confinamentos".	Vazio
`supplementalGroups`	Os contêineres do Trident são executados como raiz.	`RunAsAny`
`users`	Uma entrada é fornecida para vincular esse SCC ao usuário Trident no namespace Trident.	n/a.
`volumes`	Os pods do Trident exigem esses plugins de volume.	`hostPath, downwardAPI, projected, emptyDir`

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 DESTE DOCUMENTO. 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 SOBRESSALENTES; 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 DESTE SOFTWARE, 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.