Referência

Trident

NetApp
January 15, 2026

Índice

Referência

Portos Trident

Saiba mais sobre as portas que o Trident utiliza para comunicação.

Portos Trident

O Trident usa as seguintes portas para comunicação dentro do Kubernetes:

Porta	Propósito
8443	HTTPS de canal reverso
8001	endpoint de métricas do Prometheus
8000	Servidor REST Trident
17546	Porta de teste de atividade/prontidão usada pelos pods do daemonset Trident

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

API REST Trident

Enquanto"Comandos e opções do tridentctl" A maneira mais fácil de interagir com a API REST do Trident é por meio de métodos alternativos; você também pode usar o endpoint REST diretamente, se preferir.

Quando usar a API REST

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

Para maior segurança, o Trident `REST API` Por padrão, o acesso é restrito ao localhost quando executado dentro de um pod. Para alterar esse comportamento, você precisa configurar o Trident. `-address` argumento em sua configuração de pod.

Utilizando a API REST

Para exemplos de como essas APIs são chamadas, passe o parâmetro debug.(`-d`) bandeira. Para obter mais informações, consulte"Gerencie o Trident usando o tridentctl" .

A API funciona da seguinte forma:

PEGAR

`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.

PUBLICAR

`POST <trident-address>/trident/v1/<object-type>`

Cria um objeto do tipo especificado.

- Requer uma configuração JSON para que o objeto seja criado. Para obter a especificação de cada tipo de objeto, consulte"Gerencie o Trident usando o tridentctl" .
- Se o objeto já existir, o comportamento varia: os servidores atualizam o objeto existente, enquanto todos os outros tipos de objeto terão a operação falhada.

EXCLUIR

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`

Exclui o recurso especificado.

> Os volumes associados a backends ou classes de armazenamento continuarão a existir; estes devem ser excluídos separadamente. Para obter mais informações, consulte"Gerencie o Trident usando o tridentctl" .

Opções de linha de comando

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

Registro

`-debug`

Ativa a saída de depuração.

`-loglevel <level>`

Define o nível de registro (debug, info, warn, error, fatal). O valor padrão é info.

Kubernetes

`-k8s_pod`

Use esta opção ou `-k8s_api_server` Para habilitar o suporte ao Kubernetes. Ao configurar isso, o Trident usa as credenciais da conta de serviço do Kubernetes do pod que o contém para entrar em contato com o servidor da API. Isso só funciona quando o Trident é executado como um pod em um cluster Kubernetes com contas de serviço habilitadas.

`-k8s_api_server <insecure-address:insecure-port>`

Use esta opção ou `-k8s_pod` Para habilitar o suporte ao Kubernetes. Quando especificado, o Trident se conecta ao servidor da 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 suporta apenas conexões inseguras com o servidor da API. Para se conectar com segurança, implante o Trident em um pod com o `-k8s_pod` opção.

Docker

`-volume_driver <name>`

Nome do driver usado ao registrar o plugin do Docker. O padrão é `netapp` .

`-driver_port <port-number>`

Escute nesta porta em vez de um socket de domínio UNIX.

`-config <file>`

Obrigatório; você deve especificar este caminho para um arquivo de configuração do backend.

DESCANSAR

`-address <ip-or-host>`

Especifica o endereço no qual o servidor REST do Trident deve escutar. O padrão é localhost. Ao escutar em localhost e executar dentro de um pod do Kubernetes, a interface REST não é diretamente acessível de fora do pod. Usar `-address ""` Para tornar a interface REST acessível a partir do endereço IP do pod.

> A interface REST do Trident pode ser configurada para escutar 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 valor padrão é 8000.

`-rest`

Habilita a interface REST. O valor padrão é verdadeiro.

Objetos Kubernetes e Trident

Você pode interagir com o Kubernetes e o Trident usando APIs REST, lendo e gravando objetos de recursos. Existem diversos objetos de recursos que ditam a relação entre Kubernetes e Trident, Trident e armazenamento, e Kubernetes e armazenamento. Alguns desses objetos são gerenciados pelo Kubernetes e outros pelo Trident.

Como os objetos interagem entre si?

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

1. Um usuário cria um `PersistentVolumeClaim` solicitando um novo `PersistentVolume` de um tamanho específico de um Kubernetes `StorageClass` que foi previamente configurada pelo administrador.
2. O Kubernetes `StorageClass` Identifica o Trident como seu provedor e inclui parâmetros que informam ao Trident como provisionar um volume para a classe solicitada.
3. Trident analisa a si mesmo `StorageClass` com o mesmo nome que identifica a correspondência

`Backends` e `StoragePools` que pode ser usado para provisionar volumes para a classe.

4. O Trident provisiona armazenamento em um backend compatível e cria dois objetos: um `PersistentVolume` No Kubernetes, isso informa ao Kubernetes como encontrar, montar e tratar o volume, e um volume no Trident mantém a relação entre o `PersistentVolume` e o armazenamento propriamente dito.
5. O Kubernetes vincula o `PersistentVolumeClaim` para o novo `PersistentVolume` . Cápsulas que incluem o `PersistentVolumeClaim` Monte esse PersistentVolume em qualquer host em que ele esteja sendo executado.
6. Um usuário cria um `VolumeSnapshot` de um PVC existente, usando um `VolumeSnapshotClass` Isso aponta para Trident.
7. O Trident identifica o volume associado ao PVC e cria um instantâneo desse volume em seu servidor. Isso 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 obter mais informações sobre objetos do Kubernetes, recomendamos fortemente a leitura do seguinte: "Volumes Persistentes" seção da documentação do Kubernetes.

Kubernetes `PersistentVolumeClaim` objetos

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

Além da especificação padrão, o Trident permite que os usuários especifiquem as seguintes anotações específicas de volume, caso desejem substituir os valores padrão definidos na configuração do backend:

Anotação	Opção de volume	Drivers suportados
trident.netapp.io/fileSystem	sistema de arquivos	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
trident.netapp.io/snapshotPolicy	Política de instantâneo	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san
trident.netapp.io/snapshotReserve	snapshotReserve	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/diretóriodesnapshots	diretório de snapshots	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup

Anotação	Opção de volume	Drivers suportados
trident.netapp.io/unixPermissions	permissões do Unix	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/blockSize	tamanho do bloco	solidfire-san

Se o PV criado tiver o `Delete` Na política de recuperação, a Trident exclui tanto o PV quanto o volume de suporte quando o PV é liberado (ou seja, quando o usuário exclui o PVC). Caso a ação de exclusão falhe, o Trident marca o PV como tal e tenta novamente a operação periodicamente até que ela seja bem-sucedida ou o PV seja excluído manualmente. Se o PV usar o `Retain` Na política do Trident , ele ignora o volume e assume que o administrador o removerá do Kubernetes e do backend, permitindo que o volume seja copiado ou inspecionado antes de sua remoção. Note que excluir o PV não faz com que o Trident exclua o volume de backup. Você deve removê-lo usando a API REST.(`tridentctl`).

O Trident suporta a criação de Snapshots de Volume usando a especificação CSI: você pode criar um Snapshot de Volume e usá-lo como uma 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 em `On-Demand Volume Snapshots` para ver como isso funcionaria.

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

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

Considere os seguintes pontos:

- A NetApp recomenda clonar um volume ocioso.
- Um PVC e seu clone devem estar no mesmo namespace do Kubernetes e ter a mesma classe de armazenamento.
- Com o `ontap-nas` e `ontap-san` Para os motoristas, pode ser interessante definir a anotação PVC. `trident.netapp.io/splitOnClone` em conjunto com `trident.netapp.io/cloneFromPVC` . Com `trident.netapp.io/splitOnClone` definido para `true` O Trident separa o volume clonado do volume original, desacoplando completamente o ciclo de vida do volume clonado do seu original, à custa de alguma perda de eficiência de armazenamento. Não configurado `trident.netapp.io/splitOnClone` ou configurando-o para `false` Isso resulta em menor consumo de espaço no servidor, à custa da criação de dependências entre os volumes pai e clone, de forma que o volume pai não pode 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 se espera que o volume e seu clone divirjam bastante e não se beneficiem das eficiências de armazenamento oferecidas pelo ONTAP.

O `sample-input` O diretório contém exemplos de definições de PVC para uso com o Trident. Consulte Para obter uma descrição completa dos parâmetros e configurações associados aos volumes do Trident .

Kubernetes `PersistentVolume` objetos

Um Kubernetes `PersistentVolume` O objeto representa uma unidade de armazenamento que é disponibilizada para o cluster Kubernetes. Possui um ciclo de vida independente do pod que o utiliza.

Trident cria `PersistentVolume` objetos e os registra automaticamente no cluster Kubernetes com base nos volumes que provisiona. Não se espera que você os gerencie por conta própria.

Ao criar um PVC que se refere a um produto baseado no Trident `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 utiliza para provisionar o armazenamento. Em ambos os casos, é fundamental garantir que os nomes sejam únicos em seu escopo.
- O tamanho do volume corresponde o mais próximo possível ao tamanho solicitado no PVC, embora possa ser arredondado para a quantidade alocável mais próxima, dependendo da plataforma.

Kubernetes `StorageClass` objetos

Kubernetes `StorageClass` Os objetos são especificados por nome em `PersistentVolumeClaims` Provisionar armazenamento com um conjunto de propriedades. A própria classe de armazenamento 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 de backend Trident .

Um Kubernetes `StorageClass` Um objeto que utiliza o Trident tem a seguinte aparência:

```
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 ao Trident como provisionar volumes para a classe.

Os parâmetros da classe de armazenamento são:

Atributo	Tipo	Obrigatório	Descrição
atributos	map[string]string	não	Consulte a seção de atributos abaixo.
Piscinas de armazenamento	map[string]StringList	não	Mapeamento de nomes de backend para listas de pools de armazenamento dentro de

Atributo	Tipo	Obrigatório	Descrição
pools de armazenamento adicionais	map[string]StringList	não	Mapeamento de nomes de backend para listas de pools de armazenamento dentro de
excluirPoolsDeArmazenamento	map[string]StringList	não	Mapeamento de nomes de backend para listas de pools de armazenamento dentro de

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

Atributos de seleção do pool de armazenamento

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

Atributo	Tipo	Valores	Oferecer	Solicitar	Apoiado por
mídia[1]	corda	HDD, híbrido, SSD	A piscina contém mídias deste tipo; híbrido significa ambos	Tipo de mídia especificado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
tipo de provisionamento	corda	fino, grosso	O Pool suporta este método de provisionamento.	Método de provisionamento especificado	Espesso: tudo Ontap; fino: tudo Ontap e Solidfire-San
backendType	corda	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	Backend especificado	Todos os motoristas
instantâneos	bool	Verdadeiro, falso	O pool suporta volumes com snapshots.	Volume com snapshots ativados	ontap-nas, ontap-san, solidfire-san, gcp-cvs
clones	bool	Verdadeiro, falso	O Pool suporta a clonagem de volumes.	Volume com clones ativados	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Atributo	Tipo	Valores	Oferecer	Solicitar	Apoiado por
criptografia	bool	Verdadeiro, falso	O Pool suporta volumes criptografados	Volume com criptografia ativada	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	inteiro positivo	A Pool é capaz de garantir IOPS nessa faixa.	O volume garante esses IOPS.	solidfire-san

[1]: Não suportado pelos sistemas ONTAP Select

Na maioria dos casos, os valores solicitados influenciam diretamente o provisionamento; por exemplo, solicitar provisionamento espesso resulta em um volume provisionado de forma espessa. No entanto, um pool de armazenamento Element usa seus valores mínimo e máximo de IOPS oferecidos para definir os valores de QoS, em vez do valor solicitado. Neste caso, o valor solicitado é usado apenas para selecionar o conjunto de armazenamento.

Idealmente, você pode usar `attributes` sozinho para modelar as qualidades do armazenamento necessário para satisfazer as necessidades de uma classe específica. O Trident descobre e seleciona automaticamente os pools de armazenamento que correspondem a *todos* os requisitos. `attributes` que você especificar.

Se você se encontrar impossibilitado de usar `attributes` Para selecionar automaticamente as piscinas certas para uma turma, você pode usar o `storagePools` e `additionalStoragePools` parâmetros para refinar ainda mais os pools ou até mesmo para selecionar um conjunto específico de pools.

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

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

Você pode usar o `excludeStoragePools` parâmetro para filtrar o conjunto de pools que o Trident usa para provisionamento. Usar esse parâmetro remove todos os pools que correspondem.

No `storagePools` e `additionalStoragePools` parâmetros, cada entrada assume o formato `<backend>:<storagePoolList>` , onde `<storagePoolList>` é uma lista de pools de armazenamento separados por vírgulas para o backend especificado. Por exemplo, um valor para `additionalStoragePools` pode parecer `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze` . Essas listas aceitam valores regex tanto para o backend quanto para os valores da lista. Você pode usar `tridentctl get backend` Para obter a lista de backends e seus respectivos pools.

Atributos do Kubernetes

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

Atributo	Tipo	Valores	Descrição	Motoristas relevantes	Versão do Kubernetes
fsType	corda	ext4, ext3, xfs	O tipo de sistema de arquivos para volumes em bloco.	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy	Todos
permitirExpansãoDeVolume	booleano	Verdadeiro, falso	Ativar ou desativar 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	1.11+
modo de vinculação de volume	corda	Imediato, aguarde o primeiro consumidor	Escolha quando a vinculação de volume e o provisionamento dinâmico ocorrerão.	Todos	1,19 - 1,26

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

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

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

- Consulte"Expandir volumes" Para obter mais detalhes sobre a expansão do volume.
- O pacote de instalação do Trident fornece vários exemplos de definições de classes de armazenamento para uso com o Trident `.sample-input/storage-class-*.yaml` . A exclusão de uma classe de armazenamento do Kubernetes faz com que a classe de armazenamento correspondente do Trident também seja excluída.

Kubernetes `VolumeSnapshotClass` objetos

Kubernetes `VolumeSnapshotClass` Os objetos são análogos a `StorageClasses` . Eles ajudam a definir várias classes de armazenamento e são referenciados por snapshots de volume para associar o snapshot à classe de snapshot necessária. Cada instantâneo de volume está associado a uma única classe de instantâneo de volume.

UM `VolumeSnapshotClass` Deve ser definido por um administrador para que seja possível criar snapshots. Uma classe de snapshot 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 do `csi-snapclass` As aulas são gerenciadas pelo Trident. O `deletionPolicy` Especifica a ação a ser tomada quando um instantâneo precisa ser excluído. Quando `deletionPolicy` está definido para `Delete` Os objetos de snapshot de volume, bem como o snapshot subjacente no cluster de armazenamento, são removidos quando um snapshot é excluído. Alternativamente, você pode configurá-lo para `Retain` significa que `VolumeSnapshotContent` e a imagem física são preservadas.

Kubernetes `VolumeSnapshot` objetos

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

Quando uma solicitação de snapshot de volume é recebida, o Trident gerencia automaticamente a criação do snapshot para o volume no backend e o expõe criando um identificador único. `VolumeSnapshotContent` objeto. Você pode criar snapshots a partir de PVCs existentes e usar esses snapshots como fonte de dados ao criar novos PVCs.

> O ciclo de vida de um VolumeSnapshot é independente do PVC de origem: um snapshot persiste mesmo após a exclusão do PVC de origem. Ao excluir um PVC que possui snapshots associados, o Trident marca o volume de suporte desse PVC no estado **Excluindo**, mas não o remove completamente. O volume é removido quando todos os snapshots associados são excluídos.

Kubernetes `VolumeSnapshotContent` objetos

Um Kubernetes `VolumeSnapshotContent` O objeto representa um instantâneo tirado de um volume já provisionado. É análogo a um `PersistentVolume` e indica um snapshot provisionado no cluster de armazenamento. Semelhante a `PersistentVolumeClaim` e `PersistentVolume` objetos, quando um instantâneo é criado, o `VolumeSnapshotContent` o objeto mantém um mapeamento um-para-um com o `VolumeSnapshot` objeto, que havia solicitado a criação do instantâneo.

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

Quando uma solicitação de snapshot é recebida, o Trident cria o snapshot no servidor. Após a criação do snapshot, 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 ocorre quando você deseja"importar um snapshot de volume" Criado fora do Trident.

Kubernetes `VolumeGroupSnapshotClass` objetos

Kubernetes `VolumeGroupSnapshotClass` Os objetos são análogos a `VolumeSnapshotClass` . Eles ajudam a definir várias classes de armazenamento e são referenciados por snapshots de grupos de volumes para associar o snapshot à classe de snapshot necessária. Cada instantâneo de grupo de volumes está associado a uma única classe de instantâneo de grupo de volumes.

UM `VolumeGroupSnapshotClass` Deve ser definido por um administrador para criar um grupo de snapshots. Uma classe de snapshot de grupo de volumes é criada com a seguinte definição:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

O `driver` especifica ao Kubernetes que as solicitações de snapshots de grupos de volumes do `csi-group-snap-class` As aulas são gerenciadas pelo Trident. O `deletionPolicy` Especifica a ação a ser tomada quando um instantâneo de grupo precisa ser excluído. Quando `deletionPolicy` está definido para `Delete` Os objetos de snapshot do grupo de volumes, bem como o snapshot subjacente no cluster de armazenamento, são removidos quando um snapshot é excluído. Alternativamente, você pode configurá-lo para `Retain` significa que `VolumeGroupSnapshotContent` e a imagem física são preservadas.

Kubernetes `VolumeGroupSnapshot` objetos

Um Kubernetes `VolumeGroupSnapshot` O objeto é uma solicitação para criar um instantâneo de vários volumes. Assim como um PVC representa uma solicitação feita por um usuário para um volume, um snapshot de grupo de volumes é uma solicitação feita por um usuário para criar um snapshot de um PVC existente.

Quando uma solicitação de snapshot de grupo de volumes é recebida, o Trident gerencia automaticamente a criação do snapshot do grupo para os volumes no backend e expõe o snapshot criando um identificador único. `VolumeGroupSnapshotContent` objeto. Você pode criar snapshots a partir de PVCs existentes e usar esses snapshots como fonte de dados ao criar novos PVCs.

O ciclo de vida de um VolumeGroupSnapshot é independente do PVC de origem: um snapshot persiste mesmo após a exclusão do PVC de origem. Ao excluir um PVC que possui snapshots associados, o Trident marca o volume de suporte desse PVC no estado **Excluindo**, mas não o remove completamente. O snapshot do grupo de volumes é removido quando todos os snapshots associados são excluídos.

Kubernetes `VolumeGroupSnapshotContent` objetos

Um Kubernetes `VolumeGroupSnapshotContent` O objeto representa um instantâneo de grupo obtido de um volume já provisionado. É análogo a um `PersistentVolume` e indica um snapshot provisionado no cluster de armazenamento. Semelhante a `PersistentVolumeClaim` e `PersistentVolume` objetos, quando um instantâneo é criado, o `VolumeSnapshotContent` o objeto mantém um mapeamento um-para-um com o `VolumeSnapshot` objeto, que havia solicitado a criação do instantâneo.

O `VolumeGroupSnapshotContent` O objeto contém detalhes que identificam o grupo de instantâneos, como o `volumeGroupSnapshotHandle` e os identificadores de snapshot de volume individuais existentes no sistema de armazenamento.

Quando uma solicitação de snapshot é recebida, o Trident cria o snapshot do grupo de volumes no servidor. Após a criação do snapshot do grupo de volumes, o Trident configura um `VolumeGroupSnapshotContent` objeto e, portanto, expõe o snapshot à API do Kubernetes.

Kubernetes `CustomResourceDefinition` objetos

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

As definições de recursos personalizados (CRDs) e seus metadados de objeto associados são armazenados pelo Kubernetes em seu repositório de metadados. Isso elimina a necessidade de uma loja separada para o Trident.

Trident usa `CustomResourceDefinition` objetos para preservar a identidade de objetos Trident , como backends Trident , classes de armazenamento Trident e volumes Trident . Esses objetos são gerenciados pelo Trident. Além disso, a estrutura de snapshots de volume CSI introduz alguns CRDs que são necessários para definir snapshots de volume.

Os CRDs são uma construção do Kubernetes. Os objetos dos recursos definidos acima são criados pelo Trident. Como um exemplo simples, quando um backend é criado usando `tridentctl` , um correspondente `tridentbackends` O objeto CRD é criado para ser consumido pelo Kubernetes.

Aqui estão alguns pontos a serem lembrados sobre os CRDs da 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` O comando exclui os pods do Trident , mas os CRDs criados não são removidos. Consulte"Desinstale o Trident" Para entender como o Trident pode ser completamente removido e reconfigurado do zero.

Trident `StorageClass` objetos

O Trident cria classes de armazenamento correspondentes para o Kubernetes. `StorageClass` objetos que especificam `csi.trident.netapp.io` em seu campo de provisionamento. O nome da classe de armazenamento corresponde ao do Kubernetes. `StorageClass` objeto que representa.

> Com o Kubernetes, esses objetos são criados automaticamente quando um Kubernetes é iniciado. `StorageClass` que utiliza o Trident como provedor está registrado.

As classes de armazenamento compreendem um conjunto de requisitos para volumes. O Trident compara esses requisitos com os atributos presentes em cada pool de armazenamento; se houver correspondência, esse pool de armazenamento é um destino válido para o provisionamento de volumes usando essa classe de armazenamento.

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

Objetos de backend Trident

Os backends representam os provedores de armazenamento sobre os 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 por conta própria. O outro é o Kubernetes `StorageClass` objeto.

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

Trident `StoragePool` objetos

Os pools de armazenamento representam os locais distintos disponíveis para provisionamento em cada backend. Para o ONTAP, estes correspondem a agregados em SVMs. Para NetApp HCI/ SolidFire, estes correspondem às bandas de QoS especificadas pelo administrador. Para o Cloud Volumes Service, estes correspondem às regiões do provedor de nuvem. Cada pool de armazenamento possui um conjunto de atributos de armazenamento distintos, que definem suas características de desempenho e de proteção de dados.

Diferentemente dos outros objetos aqui presentes, os candidatos a pool de armazenamento são sempre descobertos e gerenciados automaticamente.

Trident `Volume` objetos

Os volumes são a unidade básica de provisionamento, compreendendo endpoints de back-end, como compartilhamentos NFS e LUNs iSCSI e FC. No Kubernetes, estes correspondem diretamente a `PersistentVolumes` . Ao criar um volume, certifique-se de que ele tenha uma classe de armazenamento, que determina onde esse volume pode ser provisionado, além de um tamanho.

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

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

Atributo	Tipo	Obrigatório	Descrição
versão	corda	não	Versão da API Trident ("1")
nome	corda	sim	Nome do volume a ser criado
classe de armazenamento	corda	sim	Classe de armazenamento a ser usada ao provisionar o volume
tamanho	corda	sim	Tamanho do volume a ser provisionado em bytes
protocolo	corda	não	Tipo de protocolo a ser usado: "arquivo" ou "bloco"
nome interno	corda	não	Nome do objeto no sistema de armazenamento; gerado pelo Trident.
cloneSourceVolume	corda	não	ontap (nas, san) e solidfire-*: Nome do volume a ser clonado
splitOnClone	corda	não	ontap (nas, san): Separar o clone de seu progenitor
Política de instantâneo	corda	não	ontap-*: Política de snapshot a ser usada
snapshotReserve	corda	não	ontap-*: Percentagem do volume reservada para snapshots
Política de exportação	corda	não	ontap-nas*: Política de exportação a ser usada
diretório de snapshots	bool	não	ontap-nas*: Indica se o diretório de snapshots está visível.
permissões do Unix	corda	não	ontap-nas*: Permissões iniciais do UNIX
tamanho do bloco	corda	não	solidfire-*: Tamanho do bloco/setor

Atributo	Tipo	Obrigatório	Descrição
sistema de arquivos	corda	não	Tipo de sistema de arquivos

Trident gera `internalName` ao criar o volume. Este processo consiste em duas etapas. Primeiro, ele adiciona o prefixo de armazenamento (seja o padrão). `trident` ou o prefixo na configuração do backend) para o nome do volume, resultando em um nome do tipo `<prefix>-<volume-name>` . Em seguida, procede à higienização do nome, substituindo caracteres não permitidos no sistema. Para backends ONTAP , ele substitui hífens por sublinhados (portanto, o nome interno se torna `<prefix>_<volume-name>`). Para backends Element, ele substitui sublinhados por hífenes.

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

Trident `Snapshot` objetos

Os snapshots são cópias pontuais de volumes, que podem ser usadas para provisionar novos volumes ou restaurar o estado atual. No Kubernetes, estes correspondem diretamente a `VolumeSnapshotContent` objetos. Cada instantâneo está associado a um volume, que é a fonte dos dados para o instantâneo.

Cada `Snapshot` O objeto inclui as propriedades listadas abaixo:

Atributo	Tipo	Obrigatório	Descrição
versão	Corda	Sim	Versão da API Trident ("1")
nome	Corda	Sim	Nome do objeto de instantâneo Trident
nome interno	Corda	Sim	Nome do objeto de snapshot do Trident no sistema de armazenamento
nomeDoVolume	Corda	Sim	Nome do Volume Persistente para o qual o snapshot foi criado.
volumeInternalName	Corda	Sim	Nome do objeto de volume Trident associado no sistema de armazenamento.

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

Quando um Kubernetes `VolumeSnapshot` Quando uma solicitação de objeto é criada, o Trident funciona criando um objeto de instantâneo no sistema de armazenamento subjacente. O `internalName` Este objeto de 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-se os detalhes do volume de suporte.

Trident `ResourceQuota` objeto

O conjunto demoníaco Trident consome um `system-node-critical` Classe de Prioridade — a classe de prioridade mais alta disponível no Kubernetes — para garantir que o Trident possa identificar e limpar volumes durante o desligamento correto do nó e permitir que os pods do daemonset do Trident preemptem cargas de trabalho com prioridade mais baixa em clusters com alta pressão de recursos.

Para isso, a Trident utiliza um `ResourceQuota` objeto para garantir que uma Classe de Prioridade "system-node-critical" no daemonset Trident seja atendida. Antes da implantação e da criação do DaemonSet, o Trident procura por… `ResourceQuota` objeto e, se não for descoberto, aplica-o.

Se você precisar de mais controle sobre a Cota de Recursos e a Classe de Prioridade padrão, poderá gerar um `custom.yaml` ou configurar o `ResourceQuota` objeto usando o gráfico Helm.

Segue abaixo um exemplo de um objeto `ResourceQuota` priorizando o daemonset 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, consulte"Kubernetes: Cotas de Recursos" .

Limpar `ResourceQuota` se a instalação falhar

Nos raros casos em que a instalação falha após o `ResourceQuota` O objeto foi criado, primeira tentativa"desinstalando" e depois reinstale.

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

Remover `ResourceQuota`

Se preferir controlar a alocação de recursos, você pode remover o Trident. `ResourceQuota` objeto usando o comando:

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

Padrões de Segurança de Pod (PSS) e Restrições de Contexto de Segurança (SCC)

Os padrões de segurança de pods (PSS) e as políticas de segurança de pods (PSP) do Kubernetes definem os níveis de permissão e restringem o comportamento dos pods. As Restrições de Contexto de Segurança (SCC) do OpenShift definem, de forma semelhante, restrições de pod específicas para o mecanismo Kubernetes do OpenShift. Para permitir essa personalização, o Trident habilita certas permissões durante a instalação. As seções a seguir detalham as permissões definidas pelo Trident.

O PSS substitui as Políticas de Segurança de Pod (PSP). O PSP foi descontinuado no Kubernetes v1.21 e será removido na v1.25. Para obter mais informações, consulte"Kubernetes: Segurança" .

Contexto de segurança obrigatório do Kubernetes e campos relacionados

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

Capacidades	O `SYS_ADMIN` Essa funcionalidade é fornecida como parte das funcionalidades padrão para contêineres privilegiados. Por exemplo, o Docker define essas capacidades para contêineres privilegiados:
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

Seccomp	O perfil Seccomp é sempre "Não confinado" em contêineres privilegiados; portanto, não pode ser ativado no Trident.

Permissão	Descrição
SELinux	No OpenShift, os contêineres privilegiados são executados no `spc_t` Domínio ("Contêiner com Super Privilégios") e contêineres sem privilégios são executados no `container_t` domínio. Sobre `containerd` , com `container-selinux` instalados, todos os contêineres são executados no `spc_t` domínio, o que efetivamente desativa o SELinux. Portanto, Trident não adiciona `seLinuxOptions` para contêineres.
DAC	Contêineres privilegiados devem ser executados como root. Contêineres sem privilégios são executados como root para acessar os sockets Unix necessários para o CSI.

Padrões de segurança de cápsulas (PSS)

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

Alterar os rótulos do namespace pode resultar na não programação dos pods, em um erro como "Erro ao criar: …" ou "Aviso: trident-csi-…". Caso isso aconteça, verifique se o rótulo do namespace para `privileged` foi alterado. Nesse caso, reinstale o Trident.

Políticas de segurança de pods (PSP)

Campo	Descrição	Padrão
`allowPrivilegeEscalation`	Contêineres privilegiados devem permitir a escalação de privilégios.	`true`
`allowedCSIDrivers`	O Trident não utiliza volumes efêmeros CSI embutidos.	Vazio
`allowedCapabilities`	Os contêineres Trident não privilegiados não exigem mais recursos do que o conjunto padrão, enquanto os contêineres privilegiados recebem todos os recursos possíveis.	Vazio
`allowedFlexVolumes`	Trident não utiliza um"Driver FlexVolume" , portanto, não estão incluídos na lista de volumes permitidos.	Vazio

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

Campo	Descrição	Padrão
`seLinux`	Trident não define `seLinuxOptions` Porque existem atualmente diferenças na forma como os ambientes de execução de contêineres e as distribuições do Kubernetes lidam com o SELinux.	Vazio
`supplementalGroups`	Os contêineres Trident são executados como root.	`RunAsAny`
`volumes`	Os módulos Trident requerem esses plugins de volume.	`hostPath, projected, emptyDir`

Restrições de Contexto de Segurança (SCC)

Etiquetas	Descrição	Padrão
`allowHostDirVolumePlugin`	Os pods do nó Trident montam o sistema de arquivos raiz do nó.	`true`
`allowHostIPC`	A montagem de volumes NFS requer que o host se comunique via IPC com o sistema. `nfsd` .	`true`
`allowHostNetwork`	O iscsiadm requer que a rede do host se comunique com o daemon iSCSI.	`true`
`allowHostPID`	É necessário o PID do host para verificar se `rpc-statd` está sendo executado no nó.	`true`
`allowHostPorts`	O Trident não utiliza nenhuma porta do host.	`false`
`allowPrivilegeEscalation`	Contêineres privilegiados devem permitir a escalação de privilégios.	`true`
`allowPrivilegedContainer`	Os pods do nó Trident devem executar um contêiner privilegiado para poderem montar volumes.	`true`
`allowedUnsafeSysctls`	Trident não exige nenhuma medida insegura. `sysctls` .	`none`
`allowedCapabilities`	Os contêineres Trident não privilegiados não exigem mais recursos do que o conjunto padrão, enquanto os contêineres privilegiados recebem todos os recursos possíveis.	Vazio
`defaultAddCapabilities`	Não é necessário adicionar nenhuma funcionalidade aos contêineres privilegiados.	Vazio

Etiquetas	Descrição	Padrão
`fsGroup`	Os contêineres Trident são executados como root.	`RunAsAny`
`groups`	Este SCC é específico para o Trident e está vinculado ao seu usuário.	Vazio
`readOnlyRootFilesystem`	Os pods do nó Trident devem gravar no sistema de arquivos do nó.	`false`
`requiredDropCapabilities`	Os pods do nó Trident executam um contêiner privilegiado e não podem descartar recursos.	`none`
`runAsUser`	Os contêineres Trident são executados como root.	`RunAsAny`
`seLinuxContext`	Trident não define `seLinuxOptions` Porque existem atualmente diferenças na forma como os ambientes de execução de contêineres e as distribuições do Kubernetes lidam com o SELinux.	Vazio
`seccompProfiles`	Contêineres privilegiados sempre são executados "Sem confinamento".	Vazio
`supplementalGroups`	Os contêineres Trident são executados como root.	`RunAsAny`
`users`	É fornecida uma entrada para vincular este SCC ao usuário Trident no namespace Trident .	n / D
`volumes`	Os módulos Trident requerem esses plugins de volume.	`hostPath, downwardAPI, projected, emptyDir`

Informações sobre direitos autorais

Informações sobre marcas comerciais