Gerenciar e monitorar o Trident

Trident

NetApp
January 14, 2026

Índice

Gerenciar e monitorar o Trident

Atualize o Trident

Atualize o Trident

Começando com o lançamento de 24,02, o Trident segue uma cadência de lançamento de quatro meses, oferecendo três grandes lançamentos a cada ano civil. Cada nova versão baseia-se nas versões anteriores e fornece novos recursos, melhorias de desempenho, correções de bugs e melhorias. Recomendamos que você atualize pelo menos uma vez por ano para aproveitar os novos recursos do Trident.

Considerações antes da atualização

Ao atualizar para a versão mais recente do Trident, considere o seguinte:

- Deve haver apenas uma instância do Trident instalada em todos os namespaces em um determinado cluster do Kubernetes.
- O Trident 23,07 e posterior requer instantâneos de volume v1 e não suporta mais instantâneos alfa ou beta.
- Se você criou o Cloud Volumes Service para o Google Cloud no "Tipo de serviço CVS", você deve atualizar a configuração de back-end para usar o `standardsw` nível de serviço ou `zoneredundantstandardsw` ao atualizar a partir do Trident 23,01. A falha ao atualizar o `serviceLevel` no back-end pode causar falha de volumes. "Amostras do tipo de serviço CVS"Consulte para obter detalhes.
- Ao atualizar, é importante que você forneça `parameter.fsType` em `StorageClasses` usado pelo Trident. Você pode excluir e recriar `StorageClasses` sem interromper volumes pré-existentes.
  - Este é um **requisito** para aplicação de "contextos de segurança" volumes SAN.
  - O diretório https://github.com/NetApp https://github.com/NetApp/Trident Trident/tree/master/Trident Trident-installer/sample-input[sample input] contém exemplos, como https://github.com/NetApp/Trident/blob/master/Trident-installer/sample-input/storage-class-samples/storage-class-[`storage-class-bronze-default.yaml`basic[`storage-class-basic.yaml.templ`
  - Para obter mais informações, "Problemas conhecidos"consulte .

Passo 1: Selecione uma versão

As versões do Trident seguem uma convenção de nomenclatura baseada em data `YY.MM`, onde "YY" é os últimos dois dígitos do ano e "MM" é o mês. Os lançamentos de ponto seguem uma `YY.MM.X` convenção, onde "X" é o nível de patch. Você selecionará a versão para a qual atualizar com base na versão da qual você está atualizando.

- Você pode fazer uma atualização direta para qualquer versão de destino que esteja dentro de uma janela de quatro versões da versão instalada. Por exemplo, você pode atualizar diretamente de 23,04 (ou qualquer lançamento de 23,04 pontos) para 24,06.
- Se você estiver atualizando de uma versão fora da janela de quatro versões, execute uma atualização em várias etapas. Use as instruções de atualização do "versão anterior"para atualizar para a versão mais recente que se encaixa na janela de quatro versões. Por exemplo, se você estiver executando o 22,01 e quiser atualizar para o 24,06:

a. Primeiro upgrade de 22,07 para 23,04.

b. Em seguida, atualize de 23,04 para 24,06.

> Ao atualizar usando o operador Trident na Plataforma de contêiner OpenShift, você deve atualizar para o Trident 21.01.1 ou posterior. O operador Trident lançado com 21.01.0 contém um problema conhecido que foi corrigido no 21.01.1. Para obter mais detalhes, consulte "Detalhes do problema no GitHub"a .

Passo 2: Determine o método de instalação original

Para determinar qual versão você usou para instalar o Trident originalmente:

1. Use `kubectl get pods -n trident` para examinar os pods.
   - Se não houver nenhum pod do operador, o Trident foi instalado usando `tridentctl`o .
   - Se houver um pod do operador, o Trident foi instalado usando o operador Trident manualmente ou usando o Helm.
2. Se houver um pod do operador, use `kubectl describe torc` para determinar se o Trident foi instalado usando o Helm.
   - Se houver uma etiqueta Helm, o Trident foi instalado usando Helm.
   - Se não houver nenhuma etiqueta Helm, o Trident foi instalado manualmente usando o operador Trident.

Passo 3: Selecione um método de atualização

Geralmente, você deve atualizar usando o mesmo método usado para a instalação inicial, no entanto, você pode "mova entre os métodos de instalação". Existem duas opções para atualizar o Trident.

- "Atualize usando o operador Trident"

> Sugerimos que você revise "Compreender o fluxo de trabalho de atualização do operador" antes de atualizar com o operador.

*

Atualize com o operador

Compreender o fluxo de trabalho de atualização do operador

Antes de usar o operador Trident para atualizar o Trident, você deve entender os processos em segundo plano que ocorrem durante a atualização. Isso inclui alterações no controlador Trident, no pod de nó e no pod de nó e no DaemonSet que permitem atualizações contínuas.

Manuseio de atualização do operador Trident

Um dos muitos "Benefícios de usar o operador Trident" que instalar e atualizar o Trident é o manuseio automático de objetos do Trident e Kubernetes sem interromper os volumes montados existentes. Dessa forma, o Trident pode oferecer suporte a atualizações sem inatividade ou "*atualizações contínuas*". Em particular, o operador do Trident se comunica com o cluster do Kubernetes para:

- Exclua e recrie a implantação do controlador Trident e o nó DaemonSet.
- Substitua o pod de nó Trident e o pod de nó Trident por novas versões.
  - Se um nó não for atualizado, ele não impedirá que os nós restantes sejam atualizados.
  - Somente os nós com um pod de nó Trident em execução podem montar volumes.

Para obter mais informações sobre a arquitetura do Trident no cluster do Kubernetes, "Arquitetura da Trident"consulte .

Fluxo de trabalho de atualização do operador

Quando você inicia uma atualização usando o operador Trident:

1. O operador **Trident**:
   a. Detecta a versão atualmente instalada do Trident (versão *n*).
   b. Atualiza todos os objetos Kubernetes, incluindo CRDs, RBAC e Trident SVC.
   c. Exclui a implantação do controlador Trident para a versão *n*.
   d. Cria a implantação do controlador Trident para a versão *n-1*.
2. O Kubernetes* cria o pod de controlador Trident para *n-1*.
3. O operador **Trident**:
   a. Exclui o nó Trident DaemonSet para *n*. O operador não espera o encerramento do Node Pod.
   b. Cria o nó Trident Daemonset para *n-1*.
4. **Kubernetes** cria pods de nós do Trident em nós que não executam o Pod de nó do Trident *n*. Isso garante que nunca mais de um pod de nó Trident, de qualquer versão, em um nó.

Atualize uma instalação do Trident usando o operador Trident ou Helm

Você pode atualizar o Trident usando o operador Trident manualmente ou usando o Helm. Você pode atualizar de uma instalação de operador Trident para outra instalação de operador Trident ou atualizar de uma `tridentctl` instalação para uma versão de operador Trident. Reveja "Selecione um método de atualização" antes de atualizar a instalação de um operador Trident.

Atualize uma instalação manual

Você pode atualizar de uma instalação de operador Trident com escopo de cluster para outra instalação de operador Trident com escopo de cluster. Todas as versões 21,01 e superiores do Trident usam um operador com escopo de cluster.

Para atualizar do Trident que foi instalado usando o operador com escopo de namespace (versões 20,07 a 20,10), use as instruções de atualização do "sua versão instalada"Trident.

Sobre esta tarefa

O Trident fornece um arquivo de pacote que você pode usar para instalar o operador e criar objetos associados para sua versão do Kubernetes.

- Para clusters que executam o Kubernetes 1,24, "bundle_pre_1_25.yaml"use o .

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

Antes de começar

Verifique se você está usando um cluster do Kubernetes executando "Uma versão compatível do Kubernetes"o .

Passos

1. Verifique sua versão do Trident:

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

2. Exclua o operador Trident que foi usado para instalar a instância atual do Trident. Por exemplo, se você estiver atualizando do 23,07, execute o seguinte comando:

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

3. Se você personalizou sua instalação inicial usando `TridentOrchestrator` atributos, você pode editar o `TridentOrchestrator` objeto para modificar os parâmetros de instalação. Isso pode incluir alterações feitas para especificar Registros de imagens Trident e CSI espelhados para o modo offline, habilitar logs de depuração ou especificar segredos de recebimento de imagens.
4. Instale o Trident usando o pacote correto do arquivo YAML para o seu ambiente, onde *<bundle.yaml>* é `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` baseado na sua versão do Kubernetes. Por exemplo, se você estiver instalando o Trident 24,10, execute o seguinte comando:

```
kubectl create -f 24.10.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

Atualize uma instalação do Helm

Você pode atualizar uma instalação do Trident Helm.

> Ao atualizar um cluster do Kubernetes do 1,24 para o 1,25 ou posterior que tenha o Trident instalado, você deve atualizar o Values.yaml para definir `excludePodSecurityPolicy` `true` ou adicionar `--set excludePodSecurityPolicy=true` `helm upgrade` ao comando antes de atualizar o cluster.

Se você já atualizou seu cluster do Kubernetes de 1,24 para 1,25 sem atualizar o leme do Trident, a atualização do leme falhará. Para que a atualização do leme passe, execute estes passos como pré-requisitos:

1. Instale o plugin Helm-mapkubeapis https://github.com/helm/helm-mapkubeapis do .
2. Execute uma execução a seco para a versão Trident no namespace onde o Trident está instalado. Isso lista os recursos, que serão limpos.

```
helm mapkubeapis --dry-run trident  --namespace trident
```

3. Execute uma corrida completa com o leme para fazer a limpeza.

```
helm mapkubeapis trident  --namespace trident
```

Passos

1. Se "Trident instalado usando Helm"você , você pode usar `helm upgrade trident netapp-trident/trident-operator --version 100.2410.0` o para atualizar em uma etapa. Se você não adicionou o repositório Helm ou não pode usá-lo para atualizar:
   a. Transfira a versão mais recente do Trident a partir de "A seção *assets* no GitHub".
   b. Use o `helm upgrade` comando onde `trident-operator-24.10.0.tgz` reflete a versão para a qual você deseja atualizar.

```
helm upgrade <name> trident-operator-24.10.0.tgz
```

> Se você definir opções personalizadas durante a instalação inicial (como especificar Registros privados e espelhados para imagens Trident e CSI), anexe o `helm upgrade` comando usando `--set` para garantir que essas opções estejam incluídas no comando upgrade, caso contrário, os valores serão redefinidos para padrão.

2. Execute `helm list` para verificar se o gráfico e a versão do aplicativo foram atualizados. Execute `tridentctl logs` para rever todas as mensagens de depuração.

Atualize de uma `tridentctl` instalação para o operador Trident

Pode atualizar para a versão mais recente do operador Trident a partir de uma `tridentctl` instalação. Os backends e PVCs existentes estarão automaticamente disponíveis.

> Antes de alternar entre os métodos de instalação, revise "Movendo-se entre os métodos de instalação"o .

Passos

1. Transfira a versão mais recente do Trident.

```
# Download the release required [24.10.0]
mkdir 24.10.0
cd 24.10.0
wget
https://github.com/NetApp/trident/releases/download/v24.10.0/trident-
installer-24.10.0.tar.gz
tar -xf trident-installer-24.10.0.tar.gz
cd trident-installer
```

2. Crie o `tridentorchestrator` CRD a partir do manifesto.

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

3. Implante o operador com escopo de cluster no mesmo namespace.

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

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-controller-79df798bdc-m79dc   6/6     Running   0          150d
trident-node-linux-xrst8              2/2     Running   0          150d
trident-operator-5574dbbc68-nthjv     1/1     Running   0          1m30s
```

4. Crie um `TridentOrchestrator` CR para instalar o Trident.

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

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-79df798bdc-m79dc        6/6     Running   0          1m
trident-csi-xrst8                   2/2     Running   0          1m
trident-operator-5574dbbc68-nthjv   1/1     Running   0          5m41s
```

5. Confirme se o Trident foi atualizado para a versão pretendida.

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v24.10.0
```

Atualize com o tridentctl

Você pode atualizar facilmente uma instalação existente do Trident usando `tridentctl`o .

Sobre esta tarefa

Desinstalar e reinstalar o Trident funciona como uma atualização. Quando você desinstalar o Trident, a reivindicação de volume persistente (PVC) e o volume persistente (PV) usados pela implantação do Trident não são excluídos. Os PVS que já foram provisionados permanecerão disponíveis enquanto o Trident estiver off-line, e o Trident provisionará volumes para quaisquer PVCs que forem criados no período intermediário depois que ele estiver novamente on-line.

Antes de começar

Revise "Selecione um método de atualização" antes de atualizar usando `tridentctl`o .

Passos

1. Execute o comando uninstall `tridentctl` para remover todos os recursos associados ao Trident, exceto para os CRDs e objetos relacionados.

```
./tridentctl uninstall -n <namespace>
```

2. Reinstale o Trident. "Instale o Trident usando o tridentctl"Consulte a .

> Não interrompa o processo de atualização. Certifique-se de que o instalador é executado até a conclusão.

Gerenciar o Trident usando o tridentctl

O "Pacote de instalação do Trident" inclui o `tridentctl` utilitário de linha de comando para fornecer acesso simples ao Trident. Os usuários do Kubernetes com Privileges suficientes podem usá-lo para instalar o Trident ou gerenciar o namespace que contém o pod Trident.

Comandos e sinalizadores globais

Você pode executar `tridentctl help` para obter uma lista de comandos disponíveis `tridentctl` ou anexar o `--help` sinalizador a qualquer comando para obter uma lista de opções e sinalizadores para esse comando específico.

```
tridentctl [command] [--optional-flag]
```

O utilitário Trident `tridentctl` suporta os seguintes comandos e sinalizadores globais.

Comandos

`create`
Adicione um recurso ao Trident.

`delete`
Remova um ou mais recursos do Trident.

`get`
Obtenha um ou mais recursos do Trident.

`help`
Ajuda sobre qualquer comando.

`images`
Imprima uma tabela das imagens de contentor que o Trident necessita.

`import`
Importar um recurso existente para o Trident.

`install`
Instale o Trident.

`logs`
Imprimir os registos a partir do Trident.

`send`
Enviar um recurso do Trident.

`uninstall`
Desinstale o Trident.

`update`
Modificar um recurso no Trident.

`update backend state`
Suspender temporariamente as operações de back-end.

`upgrade`
Atualizar um recurso no Trident.

`version`
Imprima a versão do Trident.

Bandeiras globais

`-d, --debug`
Saída de depuração.

`-h, --help`
Ajuda para `tridentctl`.

`-k, --kubeconfig string`
Especifique `KUBECONFIG` o caminho para executar comandos localmente ou de um cluster do Kubernetes para outro.

Como alternativa, você pode exportar a `KUBECONFIG` variável para apontar para um cluster Kubernetes específico e emitir `tridentctl` comandos para esse cluster.

`-n, --namespace string`
Namespace da implantação do Trident.

`-o, --output string`
Formato de saída. Um de JSON|yaml|name|wide|ps (padrão).

`-s, --server string`
Endereço/porta da interface REST do 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).

Opções de comando e sinalizadores

criar

Use o `create` comando para adicionar um recurso ao Trident.

```
tridentctl create [option]
```

Opções

`backend`: Adicione um backend ao Trident.

eliminar

Use o `delete` comando para remover um ou mais recursos do Trident.

```
tridentctl delete [option]
```

Opções

`backend`: Excluir um ou mais backends de armazenamento do Trident.
`snapshot`: Excluir um ou mais snapshots de volume do Trident.
`storageclass`: Excluir uma ou mais classes de armazenamento do Trident.

`volume`: Excluir um ou mais volumes de armazenamento do Trident.

obter

Use o `get` comando para obter um ou mais recursos do Trident.

```
tridentctl get [option]
```

Opções

`backend`: Obtenha um ou mais backends de armazenamento do Trident.
`snapshot`: Obter um ou mais snapshots do Trident.
`storageclass`: Obtenha uma ou mais classes de armazenamento do Trident.
`volume`: Obtenha um ou mais volumes do Trident.

Bandeiras

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

imagens

Use `images` sinalizadores para imprimir uma tabela das imagens de contentor que o Trident precisa.

```
tridentctl images [flags]
```

Bandeiras

`-h`, `--help`: Ajuda para imagens.
`-v --k8s-version string`, : Versão semântica do cluster do Kubernetes.

importar volume

Use o `import volume` comando para importar um volume existente para o Trident.

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

Aliases

`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.

instale

Use os `install` sinalizadores para instalar o Trident.

```
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 do Trident (texto, json) (texto padrão).
`--node-prep`: Permite que o Trident prepare os nós do cluster do Kubernetes para gerenciar volumes usando o protocolo de storage de dados especificado. **Atualmente, `iscsi` é o único valor suportado.**
`--pv string`: o nome do PV herdado usado pelo Trident, garante que isso não existe (padrão "Trident").
`--pvc string`: O nome do PVC legado usado pelo 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 Trident a instalar.
`--use-custom-yaml`: Use todos os arquivos YAML existentes que existem no diretório de configuração.
`--use-ipv6`: Utilizar IPv6 para a comunicação do Trident.

registos

Use `logs` sinalizadores para imprimir os logs do Trident.

```
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`, : Trident log a ser exibido. 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`, : Obtém os registos para a instância de contentor anterior, se existir.
`--sidecars`: Obter os logs para os recipientes sidecar.

enviar

Use o `send` comando para enviar um recurso do Trident.

```
tridentctl send [option]
```

Opções

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

desinstalar

Use `uninstall` sinalizadores para desinstalar o Trident.

```
tridentctl uninstall [flags]
```

Bandeiras

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

atualização

Use o `update` comando para modificar um recurso no Trident.

```
tridentctl update [option]
```

Opções

`backend`: Atualize um backend no Trident.

atualizar estado de back-end

Use o `update backend state` comando para suspender ou retomar as operações de back-end.

```
tridentctl update backend state <backend-name> [flag]
```

Pontos a considerar

- Se um back-end for criado usando um TridentBackendConfig (tbc), o back-end não poderá ser atualizado usando um `backend.json` arquivo.
- Se o `userState` foi definido em um tbc, ele não pode ser modificado usando o `tridentctl update backend state <backend-name> --user-state suspended/normal` comando.
- Para recuperar a capacidade de definir a `userState` via tridentctl depois de ter sido definida via tbc, o `userState` campo deve ser removido do tbc. Isso pode ser feito usando o `kubectl edit tbc` comando. Depois que o `userState` campo for removido, você pode usar o `tridentctl update backend state` comando para alterar o `userState` de um backend.
- Utilize os `tridentctl update backend state` para alterar o `userState`. Você também pode atualizar o `userState` usando `TridentBackendConfig` ou `backend.json` arquivo; isso aciona uma reinicialização completa do back-end e pode ser demorado.

  #### Bandeiras

  `-h --help`, : Ajuda para o estado de back-end.
  `--user-state`: Defina como `suspended` para pausar operações de back-end. Defina como `normal` para retomar as operações de back-end. Quando definido para `suspended`:

- `AddVolume` e `Import Volume` estão em pausa.
- `CloneVolume` `ResizeVolume`, , `PublishVolume` `UnPublishVolume` , `CreateSnapshot`, , , `GetSnapshot` `RestoreSnapshot`, , `DeleteSnapshot` `RemoveVolume` , `GetVolumeExternal`, , , `ReconcileNodeAccess` permanecer disponível.

Você também pode atualizar o estado de back-end usando `userState` o campo no arquivo de configuração de back-end `TridentBackendConfig` ou `backend.json`. Para obter mais informações, "Opções para

gerenciar backends"consulte e "Execute o gerenciamento de back-end com o kubectl".

Exemplo:

JSON

Siga estas etapas para atualizar o `userState` usando o `backend.json` arquivo:

1. Edite o `backend.json` arquivo para incluir o `userState` campo com o seu valor definido como 'uspended'.
2. Atualize o backend usando o `tridentctl backend update` comando e o caminho para o arquivo atualizado `backend.json`.

   Exemplo: `tridentctl backend update -f /<path to backend JSON file>/backend.json`

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "<redacted>",
    "svm": "nas-svm",
    "backendName": "customBackend",
    "username": "<redacted>",
    "password": "<redacted>",
    "userState": "suspended",
}
```

YAML

Você pode editar o tbc depois que ele foi aplicado usando o `kubectl edit <tbc-name> -n <namespace>` comando. O exemplo a seguir atualiza o estado de back-end para suspender usando a `userState: suspended` opção:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  backendName: customBackend
  storageDriverName: ontap-nas
  managementLIF: <redacted>
  svm: nas-svm
userState: suspended
  credentials:
    name: backend-tbc-ontap-nas-secret
```

versão

Use `version` sinalizadores para imprimir a versão do `tridentctl` e o serviço Trident em execução.

```
tridentctl version [flags]
```

Bandeiras

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

Suporte ao plugin

O Trigentctl suporta plugins semelhantes ao kubectl. O tridentctl deteta um plugin se o nome do arquivo binário do plugin seguir o esquema "tridentctl-<plugin>", e o binário está localizado em uma pasta listada a variável de ambiente PATH. Todos os plugins detetados estão listados na seção plugin da ajuda do tridentctl. Opcionalmente, você também pode limitar a pesquisa especificando uma pasta de plug-in na variável de enviornment TRIDENCTL_PLUGIN_PATH (exemplo: `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`). Se a variável for usada, tridenctl pesquisará somente na pasta especificada.

Monitore o Trident

O Trident fornece um conjunto de endpoints de métricas Prometheus que você pode usar para monitorar o desempenho do Trident.

Visão geral

As métricas fornecidas pelo Trident permitem que você faça o seguinte:

- Mantenha o controle sobre a integridade e a configuração do Trident. Você pode examinar como as operações são bem-sucedidas e se elas podem se comunicar com os backends como esperado.
- Examine as informações de uso do back-end e entenda quantos volumes são provisionados em um back-end e a quantidade de espaço consumido, etc.
- Mantenha um mapeamento da quantidade de volumes provisionados em backends disponíveis.
- Acompanhe o desempenho. Você pode dar uma olhada em quanto tempo leva para o Trident se comunicar com backends e realizar operações.

Por padrão, as métricas do Trident são expostas na porta de destino `8001` no `/metrics` endpoint. Essas métricas são **ativadas por padrão** quando o Trident está instalado.

O que você vai precisar

- Um cluster do Kubernetes com o Trident instalado.
- Uma instância Prometheus. Isso pode ser um "Implantação do Prometheus em contêiner" ou você pode optar por executar Prometheus como um "aplicação nativa".

Passo 1: Defina um alvo Prometheus

Você deve definir um alvo Prometheus para reunir as métricas e obter informações sobre os backends que o Trident gerencia, os volumes que ele cria e assim por diante. "blog"Isso explica como você pode usar Prometheus e Grafana com Trident para recuperar métricas. O blog explica como você pode executar o

Prometheus como um operador no cluster do Kubernetes e a criação de um ServiceMonitor para obter métricas do Trident.

Passo 2: Crie um Prometheus ServiceMonitor

Para consumir as métricas do Trident, você deve criar um Prometheus ServiceMonitor que vigia `trident-csi` o serviço e escuta na `metrics` porta. Um exemplo de ServiceMonitor se parece com isso:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
      release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
    - trident
  endpoints:
  - port: metrics
    interval: 15s
```

Essa definição do ServiceMonitor recupera as métricas retornadas pelo `trident-csi` serviço e procura especificamente o `metrics` ponto final do serviço. Como resultado, Prometheus agora está configurado para entender as métricas do Trident.

Além das métricas disponíveis diretamente do Trident, o kubelet expõe muitas `kubelet_volume_*` métricas por meio do seu próprio ponto de extremidade de métricas. O Kubelet pode fornecer informações sobre os volumes anexados e pods e outras operações internas que ele manipula. Consulte a "aqui".

Passo 3: Consultar métricas do Trident com PromQL

PromQL é bom para criar expressões que retornam dados de séries temporais ou tabulares.

Aqui estão algumas consultas PromQL que você pode usar:

Obtenha informações de saúde do Trident

- **Porcentagem de respostas HTTP 2XX do Trident**

```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- **Porcentagem de respostas REST do Trident via código de status**

```
(sum (trident_rest_ops_seconds_total_count) by (status_code)  / scalar
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- **Duração média em ms das operações realizadas pela Trident**

```
sum by (operation)
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by
(operation)
(trident_operation_duration_milliseconds_count{success="true"})
```

Obtenha informações de uso do Trident

- **Tamanho médio do volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Espaço total de volume provisionado por cada back-end**

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

Obtenha uso de volume individual

Isso é ativado somente se as métricas do kubelet também forem coletadas.

- **Porcentagem de espaço usado para cada volume**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Saiba mais sobre telemetria Trident AutoSupport

Por padrão, o Trident envia métricas de Prometheus e informações básicas de back-end para o NetApp em uma cadência diária.

- Para impedir que o Trident envie métricas do Prometheus e informações básicas de back-end para o NetApp, passe a `--silence-autosupport` bandeira durante a instalação do Trident.
- O Trident também pode enviar Registros de contentores para o suporte da NetApp sob demanda por meio `tridentctl send autosupport`do . Você precisará acionar o Trident para fazer o upload dos seus logs. Antes de enviar logs, você deve aceitar o NetApp'"política de privacidade"s .
- A menos que especificado, o Trident obtém os logs das últimas 24 horas.

- Você pode especificar o período de tempo de retenção do log com o `--since` sinalizador. Por exemplo `tridentctl send autosupport --since=1h`: . Essas informações são coletadas e enviadas por meio de um `trident-autosupport` contentor que é instalado ao lado do Trident. Pode obter a imagem do contentor em "Trident AutoSupport".
- A Trident AutoSupport não coleta nem transmite informações de identificação pessoal (PII) ou informações pessoais. Ele vem com um "EULA" que não é aplicável à própria imagem de contentor Trident. Você pode saber mais sobre o compromisso da NetApp com a segurança e a confiança dos dados "aqui" .

Um exemplo de payload enviado pela Trident é assim:

```
---
items:
- backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
  protocol: file
  config:
    version: 1
    storageDriverName: ontap-nas
    debug: false
    debugTraceFlags:
    disableDelete: false
    serialNumbers:
    - nwkvzfanek_SN
    limitVolumeSize: ''
  state: online
  online: true
```

- As mensagens do AutoSupport são enviadas para o ponto de extremidade do AutoSupport do NetApp. Se você estiver usando um Registro privado para armazenar imagens de contentor, você pode usar o `--image-registry` sinalizador.
- Você também pode configurar URLs de proxy gerando os arquivos YAML de instalação. Isso pode ser feito usando `tridentctl install --generate-custom-yaml` para criar os arquivos YAML e adicionar o `--proxy-url` argumento para o `trident-autosupport` contentor no `trident-deployment.yaml`.

Desativar métricas do Trident

Para **desabilitar métricas** de serem reportadas, você deve gerar YAMLs personalizados (usando o `--generate-custom-yaml` sinalizador) e editá-los para remover o `--metrics` sinalizador de ser invocado para o `trident-main` contentor.

Desinstale o Trident

Você deve usar o mesmo método para desinstalar o Trident que você usou para instalar o Trident.

Sobre esta tarefa

- Se você precisar de uma correção para bugs observados após uma atualização, problemas de dependência ou uma atualização mal sucedida ou incompleta, você deve desinstalar o Trident e reinstalar

a versão anterior usando as instruções específicas para esse "versão". Esta é a única maneira recomendada de *downgrade* para uma versão anterior.

- Para facilitar a atualização e reinstalação, desinstalar o Trident não remove os CRDs ou objetos relacionados criados pelo Trident. Se você precisar remover completamente o Trident e todos os seus dados, "Remova completamente Trident e CRDs"consulte .

Antes de começar

Se você estiver desativando clusters do Kubernetes, exclua todas as aplicações que usam volumes criados pelo Trident antes da desinstalação. Isso garante que os PVCs sejam inéditos nos nós do Kubernetes antes que sejam excluídos.

Determine o método de instalação original

Você deve usar o mesmo método para desinstalar o Trident que você usou para instalá-lo. Antes de desinstalar, verifique qual versão você usou para instalar o Trident originalmente.

1. Use `kubectl get pods -n trident` para examinar os pods.
   - Se não houver nenhum pod do operador, o Trident foi instalado usando `tridentctl`o .
   - Se houver um pod do operador, o Trident foi instalado usando o operador Trident manualmente ou usando o Helm.
2. Se houver um pod do operador, use `kubectl describe tproc trident` para determinar se o Trident foi instalado usando o Helm.
   - Se houver uma etiqueta Helm, o Trident foi instalado usando Helm.
   - Se não houver nenhuma etiqueta Helm, o Trident foi instalado manualmente usando o operador Trident.

Desinstale a instalação de um operador Trident

Você pode desinstalar manualmente uma instalação do operador do Trident ou usando o Helm.

Desinstalar a instalação manual

Se você instalou o Trident usando o operador, você pode desinstalá-lo fazendo um dos seguintes procedimentos:

1. **Editar `TridentOrchestrator` CR e definir o sinalizador de desinstalação**:

   ```
   kubectl patch torc <trident-orchestrator-name> --type=merge -p
   '{"spec":{"uninstall":true}}'
   ```

   Quando o `uninstall` sinalizador está definido como `true`, o operador Trident desinstala o Trident, mas não remove o próprio TridentOrchestrator. Você deve limpar o TridentOrchestrator e criar um novo se quiser instalar o Trident novamente.

2. **Excluir `TridentOrchestrator`**: Ao remover o `TridentOrchestrator` CR que foi usado para implantar o Trident, você instrui o operador a desinstalar o Trident. O operador processa a remoção `TridentOrchestrator` e remove a implantação do Trident e o daemonset, excluindo os pods do Trident que ele criou como parte da instalação.

```
kubectl delete -f deploy/<bundle.yaml> -n <namespace>
```

Desinstale a instalação do Helm

Se você instalou o Trident usando o Helm, você pode desinstalá-lo usando `helm uninstall`o .

```
#List the Helm release corresponding to the Trident install.
helm ls -n trident
NAME          NAMESPACE       REVISION        UPDATED
STATUS          CHART                           APP VERSION
trident       trident         1               2021-04-20
00:26:42.417764794 +0000 UTC deployed        trident-operator-21.07.1
21.07.1

#Uninstall Helm release to remove Trident
helm uninstall trident -n trident
release "trident" uninstalled
```

Desinstale uma `tridentctl` instalação

Use o `uninstall` comando in `tridentctl` para remover todos os recursos associados ao Trident, exceto para CRDs e objetos relacionados:

```
./tridentctl uninstall -n <namespace>
```

Informações sobre direitos autorais

Informações sobre marcas comerciais