Documentação do Trident 25.06

Trident

NetApp
January 15, 2026

This PDF was generated from https://docs.netapp.com/pt-br/trident-2506/index.html on January 15, 2026.
Always check docs.netapp.com for the latest.

Índice

Documentação do Trident 25.06 . . . 1
Notas de lançamento . . . 2
O que há de novo . . . 2
Novidades em 25.06.2 . . . 2
Mudanças em 25.06.1 . . . 2
Mudanças em 25.06 . . . 2
Mudanças em 25.02.1 . . . 5
Mudanças em 25.02 . . . 5
Mudanças em 24.10.1 . . . 7
Mudanças em 24.10 . . . 7
Mudanças em 24.06 . . . 9
Mudanças em 24.02 . . . 10
Mudanças em 23.10 . . . 10
Mudanças em 23.07.1 . . . 11
Mudanças em 23.07 . . . 11
Mudanças em 23.04 . . . 12
Mudanças em 23.01.1 . . . 13
Mudanças em 23.01 . . . 13
Mudanças em 22.10 . . . 14
Mudanças em 22.07 . . . 16
Mudanças em 22.04 . . . 17
Mudanças em 22.01.1 . . . 17
Mudanças em 22.01.0 . . . 18
Mudanças em 21.10.1 . . . 18
Mudanças em 21.10.0 . . . 19
Problemas conhecidos . . . 19
Encontre mais informações . . . 21
Versões anteriores da documentação . . . 21
Problemas conhecidos . . . 21
A restauração de backups Restic de arquivos grandes pode falhar. . . . 21
Começar . . . 22
Saiba mais sobre o Trident . . . 22
Saiba mais sobre o Trident . . . 22
Arquitetura Trident . . . 23
Conceitos . . . 26
Guia rápido para Trident . . . 30
O que vem a seguir? . . . 31
Requisitos . . . 31
Informações essenciais sobre o Trident . . . 32
Front-ends suportados (orquestradores) . . . 32
Backends suportados (armazenamento) . . . 32
Suporte do Trident para virtualização KubeVirt e OpenShift. . . . 33
Requisitos de funcionalidade . . . 33

Sistemas operacionais de host testados . . . 34
Configuração do host . . . 34
Configuração do sistema de armazenamento . . . 35
Portos Trident . . . 35
Imagens de contêiner e versões correspondentes do Kubernetes . . . 35
Instalar Trident . . . 37
Instale usando o operador Trident . . . 37
Instale usando o tridentctl . . . 37
Instale utilizando um operador certificado do OpenShift. . . . 37
Use o Trident . . . 38
Prepare o nó de trabalho . . . 38
Selecionar as ferramentas certas . . . 38
Descoberta de serviços de nó . . . 38
Volumes NFS . . . 39
volumes iSCSI . . . 39
Volumes NVMe/TCP . . . 43
SCSI sobre volumes FC . . . 44
Configurar e gerenciar backends . . . 47
Configurar backends . . . 47
Azure NetApp Files . . . 47
Google Cloud NetApp Volumes . . . 67
Configure um Cloud Volumes Service para o backend do Google Cloud. . . . 84
Configure um backend NetApp HCI ou SolidFire. . . . 96
Motoristas ONTAP SAN . . . 101
Drivers ONTAP NAS . . . 131
Amazon FSx for NetApp ONTAP . . . 167
Crie backends com kubectl . . . 203
Gerenciar back-ends . . . 210
Criar e gerenciar classes de armazenamento . . . 220
Criar uma classe de armazenamento . . . 220
Gerenciar classes de armazenamento . . . 223
Provisionar e gerenciar volumes . . . 225
Forneça um volume . . . 225
Expandir volumes . . . 229
Volumes de importação . . . 240
Personalize os nomes e rótulos dos volumes. . . . 248
Compartilhar um volume NFS entre namespaces . . . 251
Clonar volumes entre namespaces . . . 255
Replicar volumes usando o SnapMirror . . . 258
Utilizar a topologia CSI . . . 264
Trabalhar com instantâneos . . . 272
Trabalhar com snapshots de grupos de volumes . . . 280
Gerenciar e monitorar o Trident . . . 285
Trident aprimorado . . . 285
Trident aprimorado . . . 285

Atualize com a operadora . . . 286
Atualize com tridentctl . . . 291
Gerencie o Trident usando o tridentctl . . . 292
Comandos e bandeiras globais . . . 292
Opções e parâmetros de comando . . . 294
Suporte a plugins . . . 300
Monitor Trident . . . 300
Visão geral . . . 300
Etapa 1: Defina um alvo do Prometheus . . . 300
Etapa 2: Criar um ServiceMonitor do Prometheus . . . 301
Etapa 3: Consultar métricas do Trident com PromQL . . . 301
Saiba mais sobre a telemetria do Trident AutoSupport. . . . 302
Desativar métricas do Trident . . . 303
Desinstale o Trident . . . 303
Determine o método de instalação original. . . . 304
Desinstalar uma instalação do operador Trident . . . 304
Desinstale um `tridentctl` instalação . . . 305
Trident para Docker . . . 306
Pré-requisitos para implantação . . . 306
Verifique os requisitos . . . 306
Ferramentas NVMe . . . 308
Ferramentas FC . . . 309
Implantar Trident . . . 311
Método de plugin gerenciado pelo Docker (versão 1.13/17.03 e posterior) . . . 311
Método tradicional (versão 1.12 ou anterior) . . . 313
Inicie o Trident na inicialização do sistema. . . . 314
Atualize ou desinstale o Trident . . . 315
Atualizar . . . 316
Desinstalar . . . 317
Trabalhar com volumes . . . 317
Criar um volume . . . 317
Remover um volume . . . 318
Clonar um volume . . . 318
Acessar volumes criados externamente . . . 319
Opções de volume específicas do motorista . . . 320
Coletar registros . . . 324
Colete registros para solucionar problemas. . . . 325
Dicas gerais de resolução de problemas . . . 325
Gerenciar várias instâncias do Trident . . . 326
Etapas para o plugin gerenciado do Docker (versão 1.13/17.03 ou posterior) . . . 326
Passos para a versão tradicional (1.12 ou anterior) . . . 326
Opções de configuração de armazenamento . . . 327
Opções de configuração global . . . 327
Configuração ONTAP . . . 328
Configuração do software Element . . . 336

Problemas e limitações conhecidos . . . 338
A atualização do plugin Trident Docker Volume para a versão 20.10 ou posterior a partir de versões mais antigas resulta em falha na atualização com o erro "arquivo ou diretório inexistente". . . . 338
Os nomes dos volumes devem ter no mínimo 2 caracteres. . . . 339
O Docker Swarm possui certos comportamentos que impedem o Trident de oferecer suporte a todas as combinações de armazenamento e driver. . . . 339
Se um FlexGroup estiver sendo provisionado, o ONTAP não provisionará um segundo FlexGroup se este segundo tiver um ou mais FlexGroup em comum com o FlexGroup que está sendo provisionado. 339
Melhores práticas e recomendações . . . 340
Implantação . . . 340
Implantar em um namespace dedicado . . . 340
Utilize quotas e limites de intervalo para controlar o consumo de armazenamento. . . . 340
Configuração de armazenamento . . . 340
Visão geral da plataforma . . . 340
ONTAP e Cloud Volumes ONTAP: Melhores práticas . . . 340
Melhores práticas do SolidFire . . . 345
Onde posso encontrar mais informações? . . . 347
Integrar Trident . . . 347
Seleção e alocação de motoristas . . . 348
projeto de classe de armazenamento . . . 351
Projeto de piscina virtual . . . 352
Operações de volume . . . 353
Serviço de métricas . . . 357
Proteção de dados e recuperação de desastres . . . 358
Replicação e recuperação do Trident . . . 358
Replicação e recuperação de SVM . . . 359
Replicação e recuperação de volume . . . 360
Proteção de dados instantâneos . . . 360
Segurança . . . 360
Segurança . . . 360
Configuração Unificada de Chaves do Linux (LUKS) . . . 362
Criptografia Kerberos em voo . . . 368
Proteja seus aplicativos com o Trident Protect. . . . 376
Saiba mais sobre o Trident Protect. . . . 376
O que vem a seguir? . . . 376
Instale o Trident Protect . . . 376
Requisitos do Trident Protect . . . 376
Instale e configure o Trident Protect. . . . 380
Instale o plugin Trident Protect CLI . . . 383
Personalize a instalação do Trident Protect . . . 387
Gerenciar Trident Protect . . . 392
Gerencie a autorização e o controle de acesso do Trident Protect. . . . 392
Monitorar recursos do Trident Protect . . . 399
Gere um pacote de suporte Trident Protect . . . 404
Aprimore o Trident Protect . . . 406

Gerenciar e proteger aplicativos 407
Use objetos do Trident Protect AppVault para gerenciar buckets. 407
Defina uma aplicação para gestão com o Trident Protect. 421
Proteja aplicativos usando o Trident Protect. 425
Restaurar aplicativos 436
Replique aplicações usando NetApp SnapMirror e Trident Protect. 454
Migre aplicativos usando o Trident Protect. 470
Gerenciar os ganchos de execução do Trident Protect 474
Desinstale o Trident Protect. 486
Blogs Trident e Trident Protect 487
Blogs da Trident 487
Blogs da Trident Protect 487
Conhecimento e suporte 489
Perguntas frequentes 489
Perguntas gerais 489
Instale e utilize o Trident em um cluster Kubernetes. 489
Solução de problemas e suporte 490
Trident aprimorado 491
Gerenciar back-ends e volumes 492
Solução de problemas 496
Solução de problemas gerais 496
Implantação malsucedida do Trident usando o operador 498
Implantação malsucedida do Trident usando `tridentctl` 500
Remova completamente o Trident e os CRDs. 500
Falha na desempacotamento de nó NVMe com namespaces de bloco bruto RWX no Kubernetes 1.26 501
Os clientes NFSv4.2 relatam "argumento inválido" após atualizar o ONTAP quando esperavam que "v4.2-xattrs" estivesse habilitado 502
Apoiar 502
Suporte Trident 502
Autossuficiência 503
Apoio comunitário 503
Suporte técnico da NetApp 503
Para maiores informações 503
Referência 504
Portos Trident 504
Portos Trident 504
API REST Trident 504
Quando usar a API REST 504
Utilizando a API REST 504
Opções de linha de comando 505
Registro 505
Kubernetes 505
Docker 506
DESCANSAR 506
Objetos Kubernetes e Trident 506

Como os objetos interagem entre si? 506
Kubernetes `PersistentVolumeClaim` objetos 507
Kubernetes `PersistentVolume` objetos 508
Kubernetes `StorageClass` objetos 509
Kubernetes `VolumeSnapshotClass` objetos 513
Kubernetes `VolumeSnapshot` objetos 513
Kubernetes `VolumeSnapshotContent` objetos 513
Kubernetes `VolumeGroupSnapshotClass` objetos 514
Kubernetes `VolumeGroupSnapshot` objetos 514
Kubernetes `VolumeGroupSnapshotContent` objetos 515
Kubernetes `CustomResourceDefinition` objetos 515
Trident `StorageClass` objetos 516
Objetos de backend Trident 516
Trident `StoragePool` objetos 516
Trident `Volume` objetos 516
Trident `Snapshot` objetos 518
Trident `ResourceQuota` objeto 519
Padrões de Segurança de Pod (PSS) e Restrições de Contexto de Segurança (SCC) 520
Contexto de segurança obrigatório do Kubernetes e campos relacionados 520
Padrões de segurança de cápsulas (PSS) 521
Políticas de segurança de pods (PSP) 521
Restrições de Contexto de Segurança (SCC) 523
Avisos legais 525
Direitos autorais 525
Marcas Registradas 525
Patentes 525
Política de Privacidade 525
Código aberto 525

Documentação do Trident 25.06

Notas de lançamento

O que há de novo

As notas de versão fornecem informações sobre novos recursos, melhorias e correções de bugs na versão mais recente do NetApp Trident.

> O `tridentctl` O binário para Linux fornecido no arquivo zip do instalador é a versão testada e suportada. Esteja ciente de que `macos` binário fornecido no `/extras` Parte do arquivo zip não foi testada nem é suportada.

Novidades em 25.06.2

O resumo "Novidades" fornece detalhes sobre melhorias, correções e itens descontinuados para as versões do Trident e do Trident Protect.

Trident

Correções

- **Kubernetes**: Corrigido um problema crítico em que dispositivos iSCSI incorretos eram descobertos ao desanexar volumes de nós do Kubernetes.

Mudanças em 25.06.1

Trident

> Para clientes que usam o SolidFire, não atualizem para a versão 25.06.1 devido a um problema conhecido ao cancelar a publicação de volumes. A versão 25.06.2 será lançada em breve para resolver esse problema.

Correções

- **Kubernetes**:
  - Corrigido um problema em que os NQNs não eram verificados antes de serem desmapeados dos subsistemas.
  - Corrigido um problema em que várias tentativas de fechar um dispositivo LUKS levavam a falhas na desanexação de volumes.
  - Corrigido o problema de desempacotamento do volume iSCSI quando o caminho do dispositivo foi alterado desde sua criação.
  - Clonagem de blocos de volumes em classes de armazenamento.
- **OpenShift**: Corrigido um problema em que a preparação do nó iSCSI falhava com o OCP 4.19.
- Aumentamos o tempo limite ao clonar um volume usando backends SolidFire ("Edição nº 1008").

Mudanças em 25.06

Trident

Melhorias

- **Kubernetes**:
  - Adicionado suporte para snapshots de grupos de volumes CSI com `v1beta1` Instantâneo de grupo de volumes APIs do Kubernetes para ONTAP- driver iSCSI SAN. Ver"Trabalhar com snapshots de grupos de volumes" .

    > VolumeGroupSnapshot é um recurso beta do Kubernetes com APIs beta. O Kubernetes 1.32 é a versão mínima necessária para o VolumeGroupSnapshot.

  - Adicionado suporte para ONTAP ASA r2 para NVMe/TCP além de iSCSI. Verlink:"Opções e exemplos de configuração do ONTAP SAN" .
  - Adicionado suporte seguro para SMB em volumes ONTAP-NAS e ONTAP-NAS-Economy. Agora é possível usar usuários e grupos do Active Directory com volumes SMB para maior segurança. Ver"Ativar SMB seguro" .
  - Concorrência aprimorada de nós Trident para maior escalabilidade nas operações de nós para volumes iSCSI.
  - Adicionado `--allow-discards` Ao abrir volumes LUKS, é necessário permitir comandos de descarte/TRIM para recuperação de espaço.
  - Desempenho aprimorado na formatação de volumes criptografados com LUKS.
  - Limpeza aprimorada de dispositivos LUKS com falha, mas parcialmente formatados.
  - Idempotência aprimorada do nó Trident para conexão e desconexão de volumes NVMe.
  - Adicionado `internalID` campo para a configuração de volume Trident para o driver ONTAP-SAN-Economy.
  - Adicionado suporte para replicação de volumes com SnapMirror para backends NVMe. Ver"Replicar volumes usando o SnapMirror" .

Aprimoramentos experimentais

> Não deve ser utilizado em ambientes de produção.

- [Prévia Técnica] Habilitou operações simultâneas do controlador Trident através do `--enable-concurrency` sinalizador de recurso. Isso permite que as operações do controlador sejam executadas em paralelo, melhorando o desempenho em ambientes movimentados ou de grande porte.

  > Este recurso é experimental e atualmente suporta fluxos de trabalho paralelos limitados com o driver ONTAP-SAN (protocolos iSCSI e FCP).

- [Prévia Técnica] Adicionado suporte manual a QoS com o driver ANF.

Correções

- **Kubernetes**:
  - Corrigido um problema com o CSI NodeExpandVolume em que dispositivos multipath podiam ficar com tamanhos incongruentes quando os discos SCSI subjacentes não estavam disponíveis.
  - Corrigido o problema que impedia a limpeza de políticas de exportação duplicadas para os drivers

ONTAP-NAS e ONTAP-NAS-Economy.

  - Corrigido o problema em que os volumes GCNV usavam NFSv3 por padrão quando… `nfsMountOptions` está desativado; agora os protocolos NFSv3 e NFSv4 são suportados. Se `nfsMountOptions` Caso não seja fornecida uma versão específica do NFS, será utilizada a versão padrão do host (NFSv3 ou NFSv4).
  - Problema de implantação corrigido ao instalar o Trident usando o Kustomize ("Edição nº 831").
  - Corrigidas as políticas de exportação ausentes para PVCs criados a partir de snapshots ("Edição nº 1016").
  - Corrigido o problema em que os tamanhos de volume do ANF não eram alinhados automaticamente em incrementos de 1 GiB.
  - Problema resolvido ao usar NFSv3 com Bottlerocket.
- Tempo limite corrigido ao clonar um volume usando backends SolidFire ("Edição nº 1008").
- Corrigido problema com volumes ONTAP-NAS-Economy que expandiam até 300 TB apesar de falhas de redimensionamento.
- Corrigido um problema em que as operações de divisão de clones eram realizadas de forma síncrona ao usar a API REST do ONTAP .

Descontinuações:

- **Kubernetes**: Atualização da versão mínima do Kubernetes suportada para a v1.27.

Trident Protect

O NetApp Trident Protect oferece recursos avançados de gerenciamento de dados de aplicativos que aprimoram a funcionalidade e a disponibilidade de aplicativos Kubernetes com estado, com suporte dos sistemas de armazenamento NetApp ONTAP e do provisionador de armazenamento NetApp Trident CSI.

Melhorias

- Tempos de restauração aprimorados, oferecendo a opção de realizar backups completos com maior frequência.
- Melhoria na granularidade da definição de aplicativos e restauração seletiva com filtragem por Grupo-Versão-Tipo (GVK).
- Ressincronização e replicação reversa eficientes ao usar o AppMirrorRelationship (AMR) com o NetApp SnapMirror, para evitar a replicação completa do PVC.
- Adicionada a capacidade de usar a identidade do Pod do EKS para criar buckets do AppVault, eliminando a necessidade de especificar um segredo com as credenciais do bucket para clusters EKS.
- Adicionada a capacidade de ignorar a restauração de rótulos e anotações no namespace de restauração, se necessário.
- O AppMirrorRelationship (AMR) agora verificará a expansão do PVC de origem e realizará a expansão apropriada no PVC de destino, conforme necessário.

Correções

- Corrigido um bug em que os valores de anotação de snapshots anteriores estavam sendo aplicados a snapshots mais recentes. Todas as anotações de instantâneo agora estão aplicadas corretamente.
- Definida uma chave secreta para criptografia do movimentador de dados (Kopia/Restic) por padrão, caso não esteja definida.

- Adicionadas melhorias na validação e nas mensagens de erro para a criação do App Vault no S3.
- O AppMirrorRelationship (AMR) agora replica apenas os PVs no estado Bound, para evitar tentativas falhas.
- Corrigido um problema que exibia erros ao obter o conteúdo do AppVault em um AppVault com um grande número de backups.
- Os snapshots de máquinas virtuais (VMSnapshots) do KubeVirt são excluídos das operações de restauração e failover para evitar falhas.
- Corrigido um problema no Kopia em que os snapshots estavam sendo removidos prematuramente devido à programação de retenção padrão do Kopia estar sobrescrevendo a programação definida pelo usuário.

Mudanças em 25.02.1

Trident

Correções

- **Kubernetes**:
  - Corrigido um problema no trident-operator onde os nomes e versões das imagens sidecar eram preenchidos incorretamente ao usar um registro de imagens não padrão ("Edição nº 983").
  - Corrigido o problema em que as sessões multipath não conseguiam se recuperar durante uma devolução de failover do ONTAP ("Edição nº 961").

Mudanças em 25.02

A partir da versão 25.02 do Trident , o resumo "Novidades" fornece detalhes sobre melhorias, correções e itens descontinuados para as versões do Trident e do Trident Protect.

Trident

Melhorias

- **Kubernetes**:
  - Adicionado suporte para ONTAP ASA r2 para iSCSI.
  - Adicionado suporte para desanexação forçada de volumes ONTAP-NAS durante cenários de desligamento não controlado do nó. Os novos volumes ONTAP-NAS agora utilizarão políticas de exportação por volume gerenciadas pelo Trident. Foi fornecido um caminho de atualização para que os volumes existentes migrassem para o novo modelo de política de exportação ao cancelar a publicação, sem afetar as cargas de trabalho ativas.
  - Adicionada anotação cloneFromSnapshot.
  - Adicionado suporte para clonagem de volumes entre namespaces diferentes.
  - Recursos aprimorados de autorrecuperação em varreduras iSCSI permitem iniciar novas varreduras com base no host, canal, alvo e ID de LUN exatos.
  - Adicionado suporte para Kubernetes 1.32.
- **OpenShift**:
  - Adicionado suporte para preparação automática de nós iSCSI para RHCOS em clusters ROSA.
  - Adicionado suporte para drivers de virtualização OpenShift para ONTAP .

- Adicionado suporte a Fibre Channel no driver ONTAP-SAN.
- Adicionado suporte a NVMe LUKS.
- Alterei a imagem base para uma imagem provisória.
- Adicionada a descoberta e o registro do estado da conexão iSCSI quando as sessões iSCSI deveriam estar conectadas, mas não estão ("Edição nº 961").
- Adicionado suporte para volumes SMB com o driver google-cloud-netapp-volumes.
- Adicionada funcionalidade para permitir que volumes ONTAP ignorem a fila de recuperação ao serem excluídos.
- Adicionada a possibilidade de substituir as imagens padrão usando SHAs em vez de tags.
- Adicionada a flag image-pull-secrets ao instalador tridentctl.

Correções

- **Kubernetes**:
  - Corrigido o problema de endereços IP de nós ausentes nas políticas de exportação automática ("Edição nº 965").
  - Corrigido o problema de políticas de exportação automáticas que mudavam prematuramente para a política por volume no ONTAP-NAS-Economy.
  - Credenciais de configuração de backend corrigidas para suportar todas as partições ARN da AWS disponíveis ("Edição nº 913").
  - Adicionada opção para desativar a reconciliação automática do configurador no operador Trident ("Edição nº 924").
  - Adicionado securityContext para o contêiner csi-resizer ("Edição nº 976").

Trident Protect

O NetApp Trident Protect oferece recursos avançados de gerenciamento de dados de aplicativos que aprimoram a funcionalidade e a disponibilidade de aplicativos Kubernetes com estado, com suporte dos sistemas de armazenamento NetApp ONTAP e do provisionador de armazenamento NetApp Trident CSI.

Melhorias

- Adicionado suporte para backup e restauração de VMs de virtualização KubeVirt / OpenShift para armazenamento com volumeMode: File e volumeMode: Block (dispositivo bruto). Este suporte é compatível com todos os drivers Trident e aprimora os recursos de proteção existentes ao replicar o armazenamento usando o NetApp SnapMirror com Trident Protect.
- Adicionada a capacidade de controlar o comportamento de congelamento no nível do aplicativo para ambientes Kubevirt.
- Adicionada a possibilidade de configurar conexões proxy do AutoSupport .
- Adicionada a capacidade de definir um segredo para a criptografia do movimentador de dados (Kopia / Restic).
- Adicionada a capacidade de executar manualmente um gancho de execução.
- Adicionada a capacidade de configurar restrições de contexto de segurança (SCCs) durante a instalação do Trident Protect.
- Adicionada a possibilidade de configurar o nodeSelector durante a instalação do Trident Protect.

- Adicionado suporte para proxy de saída HTTP/HTTPS para objetos do AppVault.
- Filtro de recursos estendido para permitir a exclusão de recursos com escopo de cluster.
- Adicionada compatibilidade com o token de sessão da AWS nas credenciais do S3 AppVault.
- Adicionado suporte para coleta de recursos após a execução de hooks de pré-snapshot.

Correções

- Melhoramos a gestão de volumes temporários para que não precisem entrar na fila de recuperação de volumes do ONTAP .
- As anotações SCC foram restauradas aos seus valores originais.
- Eficiência de restauração aprimorada com suporte para operações paralelas.
- Suporte aprimorado para tempos limite de gancho de execução em aplicações maiores.

Mudanças em 24.10.1

Melhorias

- **Kubernetes**: Adicionado suporte para Kubernetes 1.32.
- Adicionada a descoberta e o registro do estado da conexão iSCSI quando as sessões iSCSI deveriam estar conectadas, mas não estão ("Edição nº 961").

Correções

- Corrigido o problema de endereços IP de nós ausentes nas políticas de exportação automática ("Edição nº 965").
- Corrigido o problema de políticas de exportação automáticas que mudavam prematuramente para a política por volume no ONTAP-NAS-Economy.
- Dependências do Trident e do Trident-ASUP atualizadas para corrigir as vulnerabilidades CVE-2024-45337 e CVE-2024-45310.
- Remoção de logouts para portais não-CHAP intermitentemente instáveis durante a autorrecuperação do iSCSI ("Edição nº 961").

Mudanças em 24.10

Melhorias

- O driver Google Cloud NetApp Volumes agora está disponível para o público em geral para volumes NFS e oferece suporte ao provisionamento com reconhecimento de zona.
- A identidade de carga de trabalho do GCP será usada como identidade de nuvem para Google Cloud NetApp Volumes com o GKE.
- Adicionado `formatOptions` Parâmetro de configuração para os drivers ONTAP-SAN e ONTAP-SAN-Economy que permite aos usuários especificar opções de formato LUN.
- Reduzimos o tamanho mínimo do volume do Azure NetApp Files para 50 GiB. O novo tamanho mínimo do Azure deverá estar disponível para o público em geral em novembro.
- Adicionado `denyNewVolumePools` Parâmetro de configuração para restringir os drivers ONTAP-NAS-Economy e ONTAP-SAN-Economy a pools Flexvol preexistentes.

- Adicionada detecção para a inclusão, remoção ou renomeação de agregados do SVM em todos os drivers ONTAP .
- Adicionou 18 MiB de sobrecarga aos LUNs LUKS para garantir que o tamanho de PVC relatado seja utilizável.
- Melhorias no tratamento de erros de estágio e desestágio de nós ONTAP-SAN e ONTAP-SAN-Economy para permitir a remoção de dispositivos após uma falha de estágio.
- Adicionamos um gerador de funções personalizadas que permite aos clientes criar uma função minimalista para o Trident no ONTAP.
- Adicionadas informações de registro para solução de problemas. `lsscsi` ("Edição nº 792").

Kubernetes

- Adicionadas novas funcionalidades do Trident para fluxos de trabalho nativos do Kubernetes:
  - Proteção de dados
  - Migração de dados
  - Recuperação de desastres
  - Mobilidade de aplicativos

  "Saiba mais sobre o Trident Protect.".
- Adicionada uma nova bandeira `--k8s-api-qps` Para os instaladores, é necessário definir o valor de QPS usado pelo Trident para se comunicar com o servidor da API do Kubernetes.
- Adicionado `--node-prep` Uma opção para os instaladores que permite o gerenciamento automático das dependências do protocolo de armazenamento nos nós do cluster Kubernetes. Compatibilidade testada e verificada com o protocolo de armazenamento iSCSI do Amazon Linux 2023.
- Adicionado suporte para desanexação forçada de volumes ONTAP-NAS-Economy durante cenários de desligamento não controlado do nó.
- Os novos volumes NFS ONTAP-NAS-Economy usarão políticas de exportação por qtree ao serem utilizados. `autoExportPolicy` opção de backend. As Qtrees serão mapeadas para políticas de exportação restritivas de nós apenas no momento da publicação, para melhorar o controle de acesso e a segurança. As qtrees existentes serão migradas para o novo modelo de política de exportação quando o Trident remover o volume de todos os nós, para que isso não afete as cargas de trabalho ativas.
- Adicionado suporte para Kubernetes 1.31.

Aprimoramentos experimentais

- Adicionada prévia técnica para suporte a Fibre Channel no driver ONTAP-SAN.

Correções

- **Kubernetes**:
  - Webhook de admissão do Rancher corrigido, impedindo instalações do Trident Helm ("Edição nº 839").
  - Chave de afinidade fixa nos valores do gráfico Helm ("Edição nº 898").
  - Corrigido: tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector não funcionará com o valor "true" ("Edição nº 899").
  - Instantâneos efêmeros excluídos criados durante a clonagem ("Edição nº 901").

- Adicionado suporte para Windows Server 2019.
- Corrigido o problema com o `go mod tidy` no repositório Trident ("Edição nº 767").

Depreciações

- **Kubernetes:**
  - A versão mínima do Kubernetes suportada foi atualizada para 1.25.
  - Suporte para a Política de Segurança POD removido.

Reformulação da marca do produto

A partir da versão 24.10, o Astra Trident passou a se chamar Trident (Netapp Trident). Essa mudança de marca não afeta nenhum recurso, plataforma suportada ou interoperabilidade do Trident.

Mudanças em 24.06

Melhorias

- **IMPORTANTE**: O `limitVolumeSize` O parâmetro agora limita os tamanhos de qtree/LUN nos drivers de economia do ONTAP . Use o novo `limitVolumePoolSize` parâmetro para controlar os tamanhos do Flexvol nesses drivers. ("Edição nº 341").
- Adicionada a capacidade de autorrecuperação do iSCSI para iniciar varreduras SCSI pelo ID exato do LUN caso grupos de interface obsoletos estejam em uso ("Edição nº 883").
- Adicionada a possibilidade de permitir operações de clonagem e redimensionamento de volumes mesmo quando o servidor estiver em modo de suspensão.
- Adicionada a capacidade de propagar as configurações de log definidas pelo usuário para o controlador Trident aos pods dos nós Trident .
- Adicionada a funcionalidade REST no Trident , em vez de ONTAPI (ZAPI), para versões do ONTAP 9.15.1 e posteriores.
- Adicionada a possibilidade de definir nomes de volumes e metadados personalizados nos backends de armazenamento ONTAP para novos volumes persistentes.
- Aprimorou o `azure-netapp-files` (ANF) driver para habilitar automaticamente o diretório de snapshots por padrão quando as opções de montagem NFS estiverem definidas para usar a versão 4.x do NFS.
- Adicionado suporte ao Bottlerocket para volumes NFS.
- Adicionada a pré-visualização técnica do Google Cloud NetApp Volumes.

Kubernetes

- Adicionado suporte para Kubernetes 1.30.
- Adicionada a capacidade do Trident DaemonSet de limpar montagens zumbis e arquivos residuais de rastreamento na inicialização ("Edição nº 883").
- Adicionada anotação em PVC `trident.netapp.io/luksEncryption` para importar dinamicamente volumes LUKS ("Edição nº 849").
- Adicionada a capacidade de reconhecer a topologia ao driver ANF.
- Adicionado suporte para nós do Windows Server 2022.

Correções

- Corrigido o problema de falhas na instalação do Trident devido a transações obsoletas.
- Corrigido o problema no tridentctl para ignorar mensagens de aviso do Kubernetes ("Edição nº 892").
- Controlador Trident alterado `SecurityContextConstraint` prioridade para `0` ("Edição nº 887").
- Os drivers ONTAP agora aceitam tamanhos de volume inferiores a 20 MiB ("Problema[#885").
- Correção no Trident para evitar a redução de volumes FlexVol durante a operação de redimensionamento do driver ONTAP-SAN.
- Corrigido o problema de falha na importação de volumes ANF com NFS v4.1.

Mudanças em 24.02

Melhorias

- Adicionado suporte para Identidade na Nuvem.
  - O AKS com ANF - Azure Workload Identity será usado como identidade na nuvem.
  - EKS com FSxN - A função AWS IAM será usada como identidade na nuvem.
- Adicionada a possibilidade de instalar o Trident como um complemento no cluster EKS a partir do console do EKS.
- Adicionada a capacidade de configurar e desativar a autorrecuperação do iSCSI ("Edição nº 864").
- Adicionada a funcionalidade Amazon FSx aos drivers ONTAP para permitir a integração com o AWS IAM e o SecretsManager, e para permitir que o Trident exclua volumes FSx com backups ("Edição nº 453").

Kubernetes

- Adicionado suporte para Kubernetes 1.29.

Correções

- Mensagens de aviso do ACP corrigidas quando o ACP não está ativado ("Edição nº 866").
- Adicionamos um atraso de 10 segundos antes de realizar a divisão de clones durante a exclusão de snapshots para drivers ONTAP , quando um clone está associado ao snapshot.

Depreciações

- Remoção da estrutura de atestações in-toto dos manifestos de imagem multiplataforma.

Mudanças em 23.10

Correções

- Expansão de volume fixa se um novo tamanho solicitado for menor que o tamanho total do volume para os drivers de armazenamento ontap-nas e ontap-nas-flexgroup ("Edição nº 834").
- Tamanho de volume fixo para exibir apenas o tamanho utilizável do volume durante a importação para drivers de armazenamento ontap-nas e ontap-nas-flexgroup ("Edição nº 722").
- Correção na conversão de nomes FlexVol para ONTAP-NAS-Economy.
- Corrigido problema de inicialização do Trident em um nó Windows quando o nó é reinicializado.

Melhorias

Kubernetes

Adicionado suporte para Kubernetes 1.28.

Trident

- Adicionada compatibilidade com o uso de Identidades Gerenciadas do Azure (AMI) com o driver de armazenamento azure-netapp-files.
- Adicionado suporte para NVMe sobre TCP para o driver ONTAP-SAN.
- Adicionada a capacidade de pausar o provisionamento de um volume quando o backend for definido como suspenso pelo usuário ("Edição nº 558").

Mudanças em 23.07.1

Kubernetes: Corrigida a exclusão de daemonset para suportar atualizações sem tempo de inatividade ("Edição nº 740").

Mudanças em 23.07

Correções

Kubernetes

- Correção na atualização do Trident para ignorar pods antigos presos no estado de término ("Edição nº 740").
- Adicionada tolerância à definição "transient-trident-version-pod" ("Edição nº 795").

Trident

- Corrigidas as solicitações ONTAPI (ZAPI) para garantir que os números de série do LUN sejam consultados ao obter os atributos do LUN, a fim de identificar e corrigir dispositivos iSCSI fantasmas durante as operações de preparação de nós.
- Correção no tratamento de erros no código do driver de armazenamento ("Edição nº 816").
- Correção no redimensionamento de cotas ao usar drivers ONTAP com use-rest=true.
- Correção na criação de clones de LUN em ontap-san-economy.
- Reverter o campo de informações de publicação de `rawDevicePath` para `devicePath` ; Adicionada lógica para preencher e recuperar (em alguns casos) `devicePath` campo.

Melhorias

Kubernetes

- Adicionado suporte para importação de snapshots pré-provisionados.
- Implantação minimizada e permissões do DaemonSet no Linux ("Edição nº 817").

Trident

- O campo de estado não será mais exibido para volumes e snapshots "online".

- Atualiza o estado do backend se o backend ONTAP estiver offline ("Edição nº 801" ,"#543").
- O número de série do LUN é sempre recuperado e publicado durante o fluxo de trabalho ControllerVolumePublish.
- Adicionada lógica adicional para verificar o número de série e o tamanho do dispositivo iSCSI multipath.
- Verificação adicional para volumes iSCSI para garantir que o dispositivo multipath correto esteja descompactado.

Aprimoramento Experimental

Adicionada a pré-visualização técnica do suporte a NVMe sobre TCP para o driver ONTAP-SAN.

Documentação

Foram feitas diversas melhorias de organização e formatação.

Depreciações

Kubernetes

- Suporte para snapshots da versão 1beta1 removido.
- Suporte removido para volumes e classes de armazenamento pré-CSI.
- A versão mínima do Kubernetes suportada foi atualizada para 1.22.

Mudanças em 23.04

> O recurso de desanexação forçada de volumes ONTAP-SAN-* é compatível apenas com versões do Kubernetes que possuem o recurso de desligamento não gradual do nó ativado. O recurso de desanexação forçada deve ser ativado durante a instalação usando o `--enable -force-detach` Bandeira do instalador Trident .

Correções

- Corrigido o problema com o Trident Operator, que agora usa IPv6 localhost para instalação quando especificado na especificação.
- As permissões da função de cluster do operador Trident foram corrigidas para estarem sincronizadas com as permissões do pacote ("Edição nº 799").
- Corrigido problema com a anexação de volume de bloco bruto em vários nós no modo RWX.
- Correção do suporte à clonagem FlexGroup e à importação de volumes SMB.
- Corrigido problema em que o controlador Trident não conseguia ser desligado imediatamente ("Edição nº 811").
- Adicionada correção para listar todos os nomes de igroup associados a um LUN específico provisionado com drivers ontap-san-*.
- Adicionada uma correção para permitir que processos externos sejam executados até a conclusão.
- Erro de compilação corrigido para a arquitetura s390 ("Edição nº 537").
- Corrigido o nível de registro incorreto durante as operações de montagem de volume ("Edição nº 781").
- Erro potencial de asserção de tipo corrigido ("Edição nº 802").

Melhorias

- Kubernetes:
  - Adicionado suporte para Kubernetes 1.27.
  - Adicionado suporte para importação de volumes LUKS.
  - Adicionado suporte para o modo de acesso PVC ReadWriteOncePod.
  - Adicionado suporte para desanexação forçada de volumes ONTAP-SAN-* durante cenários de desligamento não controlado do nó.
  - Todos os volumes ONTAP-SAN-* agora usarão igroups por nó. Para melhorar nossa postura de segurança, os LUNs serão mapeados para igroups somente enquanto estiverem ativamente publicados nesses nós. Os volumes existentes serão migrados para o novo esquema de igroup de forma oportunista, quando a Trident determinar que é seguro fazê-lo sem afetar as cargas de trabalho ativas ("Edição nº 758").
  - Segurança do Trident aprimorada através da limpeza de grupos de instâncias (igroups) gerenciados pelo Trident e não utilizados nos backends ONTAP-SAN-*.
- Adicionamos suporte para volumes SMB com Amazon FSx aos drivers de armazenamento ontap-nas-economy e ontap-nas-flexgroup.
- Adicionado suporte para compartilhamentos SMB com os drivers de armazenamento ontap-nas, ontap-nas-economy e ontap-nas-flexgroup.
- Adicionado suporte para nós arm64 ("Edição nº 732").
- Procedimento de desligamento do Trident aprimorado, desativando primeiro os servidores de API ("Edição nº 811").
- Adicionada compatibilidade com compilação multiplataforma para hosts Windows e arm64 no Makefile; consulte BUILD.md.

Depreciações

Kubernetes: Grupos de instâncias (igroups) com escopo de backend não serão mais criados ao configurar os drivers ontap-san e ontap-san-economy ("Edição nº 758").

Mudanças em 23.01.1

Correções

- Corrigido o problema com o Trident Operator, que agora usa IPv6 localhost para instalação quando especificado na especificação.
- As permissões da função de cluster do operador Trident foram corrigidas para estarem sincronizadas com as permissões do pacote."Edição nº 799" .
- Adicionada uma correção para permitir que processos externos sejam executados até a conclusão.
- Corrigido problema com a anexação de volume de bloco bruto em vários nós no modo RWX.
- Correção do suporte à clonagem FlexGroup e à importação de volumes SMB.

Mudanças em 23.01

O Kubernetes 1.27 agora é compatível com o Trident. Por favor, atualize o Trident antes de atualizar o Kubernetes.

Correções

- Kubernetes: Adicionadas opções para excluir a criação de Políticas de Segurança de Pod para corrigir instalações do Trident via Helm ("Edições nº 783, nº 794").

Melhorias

Kubernetes

- Adicionado suporte para Kubernetes 1.26.
- Melhoria na utilização geral dos recursos do Trident RBAC ("Edição nº 757").
- Adicionada automação para detectar e corrigir sessões iSCSI interrompidas ou obsoletas em nós de host.
- Adicionado suporte para expansão de volumes criptografados com LUKS.
- Kubernetes: Adicionado suporte para rotação de credenciais em volumes criptografados com LUKS.

Trident

- Adicionamos suporte para volumes SMB com Amazon FSx for NetApp ONTAP ao driver de armazenamento ontap-nas.
- Adicionada a compatibilidade com permissões NTFS ao usar volumes SMB.
- Adicionada a funcionalidade de pools de armazenamento para volumes do GCP com nível de serviço CVS.
- Adicionada a possibilidade de uso opcional do flexgroupAggregateList ao criar FlexGroups com o driver de armazenamento ontap-nas-flexgroup.
- Desempenho aprimorado para o driver de armazenamento ontap-nas-economy ao gerenciar vários volumes FlexVol.
- Ativadas as atualizações do dataLIF para todos os drivers de armazenamento ONTAP NAS.
- Atualizamos a convenção de nomenclatura do Trident Deployment e do DaemonSet para refletir o sistema operacional do nó host.

Depreciações

- Kubernetes: Atualizamos a versão mínima do Kubernetes suportada para 1.21.
- Não é mais necessário especificar DataLIFs ao configurar `ontap-san` ou `ontap-san-economy` motoristas.

Mudanças em 22.10

Você deve ler as seguintes informações críticas antes de atualizar para o Trident 22.10.

Informações críticas sobre o Trident 22.10

- O Kubernetes 1.25 agora é compatível com o Trident. Você precisa atualizar o Trident para a versão 22.10 antes de atualizar para o Kubernetes 1.25.
- O Trident agora impõe rigorosamente o uso da configuração de multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo multipath.conf.

  Utilização de configuração sem multipathing ou utilização de `find_multipaths: yes` ou `find_multipaths: smart` O valor no arquivo multipath.conf resultará em falhas de montagem. A Trident recomendou o uso de `find_multipaths: no` desde a versão de 21/07.

Correções

- Problema corrigido específico do backend ONTAP criado usando `credentials` O campo não ficou online durante a atualização para a versão 22.07.0 ("Edição nº 759").
- **Docker:** Corrigido um problema que impedia a inicialização do plugin de volume do Docker em alguns ambientes ("Edição nº 548" e"Edição nº 760").
- Corrigido problema de SLM específico para backends SAN ONTAP para garantir que apenas um subconjunto de dataLIFs pertencentes a nós de relatório sejam publicados.
- Corrigido um problema de desempenho que causava varreduras desnecessárias de LUNs iSCSI ao conectar um volume.
- Removemos as tentativas granulares no fluxo de trabalho iSCSI do Trident para que as falhas sejam detectadas rapidamente e os intervalos de repetição externa sejam reduzidos.
- Corrigido um problema em que era retornado um erro ao limpar um dispositivo iSCSI quando o dispositivo multipath correspondente já havia sido limpo.

Melhorias

- Kubernetes:
  - Adicionado suporte para Kubernetes 1.25. Você precisa atualizar o Trident para a versão 22.10 antes de atualizar para o Kubernetes 1.25.
  - Adicionamos uma ServiceAccount, ClusterRole e ClusterRoleBinding separadas para a implantação do Trident e o DaemonSet para permitir melhorias futuras nas permissões.
  - Adicionado suporte para"compartilhamento de volumes entre namespaces" .
- Todos os Trident `ontap-*` Os drivers de armazenamento agora funcionam com a API REST do ONTAP .
- Adicionado novo operador yaml(`bundle_post_1_25.yaml`) sem um `PodSecurityPolicy` Para dar suporte ao Kubernetes 1.25.
- Adicionado"suporte para volumes criptografados com LUKS" para `ontap-san` e `ontap-san-economy` drivers de armazenamento.
- Adicionado suporte para nós do Windows Server 2019.
- Adicionado"Suporte para volumes SMB em nós Windows" através do `azure-netapp-files` driver de armazenamento.
- A detecção automática de transição do MetroCluster para drivers ONTAP agora está disponível para o público em geral.

Depreciações

- **Kubernetes:** Atualizamos a versão mínima do Kubernetes suportada para 1.20.
- Driver Astra Data Store (ADS) removido.
- Suporte removido para `yes` e `smart` opções para `find_multipaths` Ao configurar o multipathing de nós de trabalho para iSCSI.

Mudanças em 22.07

Correções

Kubernetes

- Corrigido problema no tratamento de valores booleanos e numéricos para o seletor de nós ao configurar o Trident com o Helm ou o Trident Operator. ("Problema nº 700 do GitHub")
- Corrigido problema no tratamento de erros em caminhos não-CHAP, de forma que o kubelet tente novamente em caso de falha. "Problema nº 736 no GitHub")

Melhorias

- Transição de k8s.gcr.io para registry.k8s.io como registro padrão para imagens CSI.
- Os volumes ONTAP-SAN agora usarão igroups por nó e mapearão LUNs para igroups somente enquanto estiverem ativamente publicados nesses nós, a fim de melhorar nossa postura de segurança. Os volumes existentes serão migrados para o novo esquema de igroup de forma oportunista, quando a Trident determinar que é seguro fazê-lo sem afetar as cargas de trabalho ativas.
- Incluímos uma ResourceQuota nas instalações do Trident para garantir que o DaemonSet do Trident seja agendado quando o consumo de PriorityClass for limitado por padrão.
- Adicionada a funcionalidade de recursos de rede ao driver Azure NetApp Files . ("Problema nº 717 do GitHub")
- Adicionada detecção automática de transição para MetroCluster em versão prévia aos drivers ONTAP . ("Problema nº 228 no GitHub")

Depreciações

- **Kubernetes:** Atualizamos a versão mínima do Kubernetes suportada para 1.19.
- A configuração do backend não permite mais múltiplos tipos de autenticação em uma única configuração.

Remoções

- O driver AWS CVS (obsoleto desde a versão 22.04) foi removido.
- Kubernetes
  - Removida a capacidade SYS_ADMIN desnecessária dos pods dos nós.
  - Reduz o nodeprep a informações simples do host e descoberta ativa de serviços para realizar uma confirmação, da melhor forma possível, de que os serviços NFS/iSCSI estão disponíveis nos nós de trabalho.

Documentação

Um novo"Padrões de segurança de pods" A seção (PSS) foi adicionada, detalhando as permissões

habilitadas pelo Trident na instalação.

Mudanças em 22.04

A NetApp está continuamente melhorando e aprimorando seus produtos e serviços. Aqui estão algumas das funcionalidades mais recentes do Trident. Para versões anteriores, consulte "Versões anteriores da documentação" .

> Se você estiver atualizando de uma versão anterior do Trident e usar o Azure NetApp Files, `olocation` O parâmetro de configuração agora é um campo obrigatório e único.

Correções

- Análise aprimorada dos nomes dos iniciadores iSCSI. ("Problema nº 681 no GitHub")
- Corrigido problema que impedia a utilização de parâmetros da classe de armazenamento CSI. ("Problema nº 598 no GitHub")
- Correção de declaração de chave duplicada no CRD do Trident . ("Problema nº 671 no GitHub")
- Registros de snapshots CSI imprecisos foram corrigidos. ("Problema nº 629 no GitHub"))
- Problema resolvido ao cancelar a publicação de volumes em nós excluídos. ("Problema nº 691 no GitHub")
- Adicionada a funcionalidade de tratamento de inconsistências do sistema de arquivos em dispositivos de bloco. ("Problema nº 656 no GitHub")
- Corrigido problema ao extrair imagens de suporte automático ao configurar o `imageRegistry` sinalizador durante a instalação. ("Problema nº 715 no GitHub")
- Corrigido um problema em que o driver Azure NetApp Files não conseguia clonar um volume com várias regras de exportação.

Melhorias

- As conexões de entrada para os endpoints seguros da Trident agora exigem, no mínimo, TLS 1.3. ("Problema nº 698 no GitHub")
- A Trident agora adiciona cabeçalhos HSTS às respostas de seus endpoints seguros.
- O Trident agora tenta habilitar automaticamente o recurso de permissões Unix do Azure NetApp Files .
- **Kubernetes**: O daemonset Trident agora é executado na classe de prioridade system-node-critical. ("Problema nº 694 no GitHub")

Remoções

O driver da série E (desativado desde a versão 20.07) foi removido.

Mudanças em 22.01.1

Correções

- Problema resolvido ao cancelar a publicação de volumes em nós excluídos. ("Problema nº 691 no GitHub")
- Corrigido o erro que causava pânico ao acessar campos nulos para espaço agregado em respostas da API ONTAP .

Mudanças em 22.01.0

Correções

- **Kubernetes:** Aumentar o tempo limite de repetição do registro de nós para clusters grandes.
- Corrigido um problema em que o driver azure-netapp-files podia ser confundido por vários recursos com o mesmo nome.
- Agora, os DataLIFs IPv6 do SAN ONTAP funcionam se especificados entre colchetes.
- Corrigido um problema em que a tentativa de importar um volume já importado retornava EOF, deixando o PVC em estado pendente. ("Problema nº 489 no GitHub")
- Corrigido problema em que o desempenho do Trident ficava mais lento quando mais de 32 snapshots eram criados em um volume SolidFire .
- Substituí o SHA-1 pelo SHA-256 na criação do certificado SSL.
- Corrigido o driver Azure NetApp Files para permitir nomes de recursos duplicados e limitar as operações a um único local.
- Corrigido o driver Azure NetApp Files para permitir nomes de recursos duplicados e limitar as operações a um único local.

Melhorias

- Aprimoramentos do Kubernetes:
  - Adicionado suporte para Kubernetes 1.23.
  - Adicione opções de agendamento para pods do Trident quando instalados via Trident Operator ou Helm. ("Problema nº 651 no GitHub")
- Permitir volumes entre regiões no driver do GCP. ("Problema nº 633 no GitHub")
- Adicionada a opção 'unixPermissions' aos volumes do Azure NetApp Files . ("Problema nº 666 no GitHub")

Depreciações

A interface REST do Trident só pode escutar e atender em endereços 127.0.0.1 ou [::1].

Mudanças em 21.10.1

A versão v21.10.0 apresenta um problema que pode colocar o controlador Trident em um estado CrashLoopBackOff quando um nó é removido e, em seguida, adicionado novamente ao cluster Kubernetes. Este problema foi corrigido na versão 21.10.1 (problema nº 669 no GitHub).

Correções

- Corrigida uma possível condição de corrida que ocorria ao importar um volume em um backend CVS do GCP, resultando em falha na importação.
- Corrigido um problema que podia colocar o controlador Trident em estado CrashLoopBackOff quando um nó era removido e depois adicionado novamente ao cluster Kubernetes (problema nº 669 no GitHub).
- Corrigido o problema em que as SVMs não eram mais detectadas se nenhum nome de SVM fosse especificado (problema nº 612 do GitHub).

Mudanças em 21.10.0

Correções

- Corrigido o problema que impedia a montagem de clones de volumes XFS no mesmo nó que o volume de origem (problema nº 514 do GitHub).
- Corrigido o problema em que o Trident registrava um erro fatal ao ser desligado (problema nº 597 no GitHub).
- Correções relacionadas ao Kubernetes:
  - Retorna o espaço usado de um volume como o tamanho mínimo de restauração (restoreSize) ao criar snapshots com `ontap-nas` e `ontap-nas-flexgroup` drivers (problema 645 do GitHub).
  - Problema corrigido onde `Failed to expand filesystem` Ocorreu um erro após o redimensionamento do volume (problema nº 560 no GitHub).
  - Corrigido problema em que um pod podia ficar preso em `Terminating` estado (problema 572 do GitHub).
  - Corrigido o caso em que um `ontap-san-economy` O FlexVol pode estar cheio de LUNs de snapshot (problema nº 533 do GitHub).
  - Corrigido problema com o instalador YAML personalizado que utilizava uma imagem diferente (problema nº 613 no GitHub).
  - Correção no cálculo do tamanho do snapshot (problema nº 611 do GitHub).
  - Corrigido o problema em que todos os instaladores do Trident identificavam o Kubernetes puro como OpenShift (problema nº 639 no GitHub).
  - Corrigido o operador Trident para interromper a reconciliação caso o servidor da API do Kubernetes esteja inacessível (problema nº 599 do GitHub).

Melhorias

- Adicionado suporte para `unixPermissions` opção para volumes de desempenho GCP-CVS.
- Adicionado suporte para volumes CVS otimizados para escala no GCP, na faixa de 600 GiB a 1 TiB.
- Melhorias relacionadas ao Kubernetes:
  - Adicionado suporte para Kubernetes 1.22.
  - Habilitei o operador Trident e o gráfico Helm para funcionar com o Kubernetes 1.22 (problema nº 628 no GitHub).
  - Adicionada imagem do operador ao `tridentctl` comando images (problema nº 570 do GitHub).

Aprimoramentos experimentais

- Adicionado suporte para replicação de volume no `ontap-san` motorista.
- Adicionado suporte REST em **prévia técnica** para o `ontap-nas-flexgroup` , `ontap-san` , e `ontap-nas-economy` motoristas.

Problemas conhecidos

Os problemas conhecidos identificam problemas que podem impedi-lo de usar o produto com sucesso.

- Ao atualizar um cluster Kubernetes da versão 1.24 para a 1.25 ou posterior que tenha o Trident instalado, você deve atualizar o arquivo values.yaml para configurar `excludePodSecurityPolicy` para `true` ou adicione `--set excludePodSecurityPolicy=true` para o `helm upgrade` Execute o comando antes de atualizar o cluster.
- Trident agora impõe um espaço em branco `fsType` (`fsType=""`) para volumes que não possuem o `fsType` especificado em sua StorageClass. Ao trabalhar com o Kubernetes 1.17 ou posterior, o Trident oferece suporte ao fornecimento de um arquivo em branco. `fsType` para volumes NFS. Para volumes iSCSI, você precisa configurar o `fsType` na sua StorageClass ao impor um `fsGroup` Utilizando um contexto de segurança.
- Ao usar um backend em várias instâncias do Trident , cada arquivo de configuração do backend deve ter um conteúdo diferente. `storagePrefix` valor para backends ONTAP ou use um diferente `TenantName` para backends SolidFire . O Trident não consegue detectar volumes criados por outras instâncias do Trident . A tentativa de criar um volume existente em backends ONTAP ou SolidFire é bem-sucedida, porque o Trident trata a criação de volume como uma operação idempotente. Se `storagePrefix` ou `TenantName` Se não houver diferenças, pode haver conflitos de nomes para volumes criados no mesmo backend.
- Ao instalar o Trident (usando `tridentctl` ou o operador Trident) e usando `tridentctl` Para gerenciar o Trident, você deve garantir que `KUBECONFIG` A variável de ambiente está definida. Isso é necessário para indicar o cluster Kubernetes que `tridentctl` deve trabalhar contra. Ao trabalhar com vários ambientes Kubernetes, você deve garantir que o `KUBECONFIG` O arquivo possui fontes corretas.
- Para realizar a recuperação de espaço online para PVs iSCSI, o sistema operacional subjacente no nó de trabalho pode exigir que opções de montagem sejam passadas para o volume. Isso é válido para instâncias do RHEL/Red Hat Enterprise Linux CoreOS (RHCOS), que exigem o `discard` "opção de montagem" ; certifique-se de que a opção de descarte mountOption esteja incluída em seu[`StorageClass` ^] para dar suporte ao descarte de blocos online.
- Se você tiver mais de uma instância do Trident por cluster Kubernetes, o Trident não poderá se comunicar com outras instâncias nem descobrir outros volumes que elas criaram, o que leva a um comportamento inesperado e incorreto caso mais de uma instância esteja em execução no cluster. Deve haver apenas uma instância do Trident por cluster Kubernetes.
- Se baseado em Trident `StorageClass` Os objetos são excluídos do Kubernetes enquanto o Trident está offline, mas o Trident não remove as classes de armazenamento correspondentes do seu banco de dados quando volta a ficar online. Você deve excluir essas classes de armazenamento usando `tridentctl` ou a API REST.
- Se um usuário excluir um PV provisionado pelo Trident antes de excluir o PVC correspondente, o Trident não excluirá automaticamente o volume de suporte. Você deve remover o volume através de `tridentctl` ou a API REST.
- O ONTAP não pode provisionar mais de um FlexGroup simultaneamente, a menos que o conjunto de agregados seja exclusivo para cada solicitação de provisionamento.
- Ao usar o Trident sobre IPv6, você deve especificar `managementLIF` e `dataLIF` na definição do backend, entre colchetes. Por exemplo, `[fd20:8b1e:b258:2000:f816:3eff:feec:0]` .

  Você não pode especificar `dataLIF` em um backend ONTAP SAN. O Trident descobre todas as LIFs iSCSI disponíveis e as utiliza para estabelecer a sessão multipath.

- Se estiver usando o `solidfire-san` Ao configurar o driver com o OpenShift 4.5, certifique-se de que os nós de trabalho subjacentes utilizem MD5 como algoritmo de autenticação CHAP. Os algoritmos CHAP seguros e compatíveis com FIPS, SHA1, SHA-256 e SHA3-256, estão disponíveis no Element 12.7.

Encontre mais informações

- "Trident GitHub"
- "Blogs da Trident"

Versões anteriores da documentação

Se você não estiver executando o Trident 25.06, a documentação para versões anteriores está disponível com base no"Ciclo de vida de suporte do Trident" .

- "Trident 25.02"
- "Trident 24.10"
- "Trident 24.06"
- "Trident 24.02"
- "Trident 23.10"
- "Trident 23.07"
- "Trident 23.04"
- "Trident 23.01"
- "Trident 22.10"

Problemas conhecidos

Problemas conhecidos identificam problemas que podem impedir que você use esta versão do produto com sucesso.

Os seguintes problemas conhecidos afetam a versão atual:

A restauração de backups Restic de arquivos grandes pode falhar.

Ao restaurar arquivos de 30 GB ou maiores a partir de um backup do Amazon S3 feito com o Restic, a operação de restauração pode falhar. Como solução alternativa, faça backup dos dados usando o Kopia como gerenciador de dados (o Kopia é o gerenciador de dados padrão para backups). Consulte "Proteja aplicativos usando o Trident Protect." para obter instruções.

Começar

Saiba mais sobre o Trident

Saiba mais sobre o Trident

Trident é um projeto de código aberto com suporte completo, mantido pela NetApp. Foi projetado para ajudar você a atender às demandas de persistência do seu aplicativo em contêineres, usando interfaces padrão do setor, como a Interface de Armazenamento de Contêiner (CSI).

O que é Trident?

O NetApp Trident permite o consumo e o gerenciamento de recursos de armazenamento em todas as plataformas de armazenamento populares da NetApp , na nuvem pública ou em infraestruturas locais, incluindo clusters ONTAP locais (AFF, FAS e ASA), ONTAP Select, Cloud Volumes ONTAP, software Element (NetApp HCI, SolidFire), Azure NetApp Files, Amazon FSx for NetApp ONTAP e Cloud Volumes Service no Google Cloud.

Trident é um orquestrador de armazenamento dinâmico compatível com a Interface de Armazenamento de Contêineres (CSI) que se integra nativamente com"Kubernetes" . O Trident é executado como um único Pod Controlador e um Pod de Nó em cada nó de trabalho do cluster. Consulte "Arquitetura Trident" para mais detalhes.

O Trident também oferece integração direta com o ecossistema Docker para plataformas de armazenamento NetApp . O NetApp Docker Volume Plugin (nDVP) oferece suporte ao provisionamento e gerenciamento de recursos de armazenamento da plataforma de armazenamento para hosts Docker. Consulte "Implantar Trident para Docker" para mais detalhes.

Se esta for a sua primeira vez usando o Kubernetes, você deve se familiarizar com o"Conceitos e ferramentas do Kubernetes" .

Integração do Kubernetes com produtos NetApp

O portfólio de produtos de armazenamento da NetApp integra-se a diversos aspectos de um cluster Kubernetes, oferecendo recursos avançados de gerenciamento de dados que aprimoram a funcionalidade, a capacidade, o desempenho e a disponibilidade da implementação do Kubernetes.

Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"É um serviço AWS totalmente gerenciado que permite iniciar e executar sistemas de arquivos com tecnologia do sistema operacional de armazenamento NetApp ONTAP .

Azure NetApp Files

"Azure NetApp Files"É um serviço de compartilhamento de arquivos do Azure de nível empresarial, desenvolvido pela NetApp. Você pode executar suas cargas de trabalho baseadas em arquivos mais exigentes nativamente no Azure, com o desempenho e o gerenciamento de dados avançado que você espera da NetApp.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP"É um dispositivo de armazenamento exclusivamente de software que executa o software de gerenciamento de dados ONTAP na nuvem.

Google Cloud NetApp Volumes

"Google Cloud NetApp Volumes"É um serviço de armazenamento de arquivos totalmente gerenciado no Google Cloud que oferece armazenamento de arquivos de alto desempenho e nível empresarial.

Software Element

"Elemento"Permite ao administrador de armazenamento consolidar cargas de trabalho, garantindo o desempenho e possibilitando uma infraestrutura de armazenamento simplificada e otimizada.

NetApp HCI

"NetApp HCI"Simplifica a gestão e a escalabilidade do centro de dados, automatizando tarefas rotineiras e permitindo que os administradores de infraestrutura se concentrem em funções mais importantes.

O Trident pode provisionar e gerenciar dispositivos de armazenamento para aplicações em contêineres diretamente na plataforma de armazenamento NetApp HCI .

NetApp ONTAP

"NetApp ONTAP"O NetApp Multiprotocol, sistema operacional de armazenamento unificado, oferece recursos avançados de gerenciamento de dados para qualquer aplicação.

Os sistemas ONTAP possuem configurações totalmente em flash, híbridas ou totalmente em HDD e oferecem diversos modelos de implantação: clusters locais FAS, AFA e ASA , ONTAP Select e Cloud Volumes ONTAP. O Trident é compatível com esses modelos de implantação do ONTAP .

Arquitetura Trident

O Trident é executado como um único Pod Controlador e um Pod de Nó em cada nó de trabalho do cluster. O pod do nó deve estar em execução em qualquer host onde você queira potencialmente montar um volume Trident .

Entendendo os pods do controlador e os pods do nó

O Trident é implantado como uma única unidade.Cápsula de controle Trident e um ou maisCápsulas de nós Trident no cluster Kubernetes e utiliza contêineres sidecar CSI padrão do Kubernetes para simplificar a implantação de plugins CSI. "Contêineres Sidecar CSI do Kubernetes" são mantidos pela comunidade Kubernetes Storage.

Kubernetes"seletores de nós" e"tolerâncias e contaminações" São utilizadas para restringir a execução de um pod a um nó específico ou preferencial. Você pode configurar seletores de nós e tolerâncias para pods de controladores e nós durante a instalação do Trident .

- O plugin do controlador lida com o provisionamento e gerenciamento de volumes, como snapshots e redimensionamento.
- O plugin do nó gerencia a conexão do armazenamento ao nó.

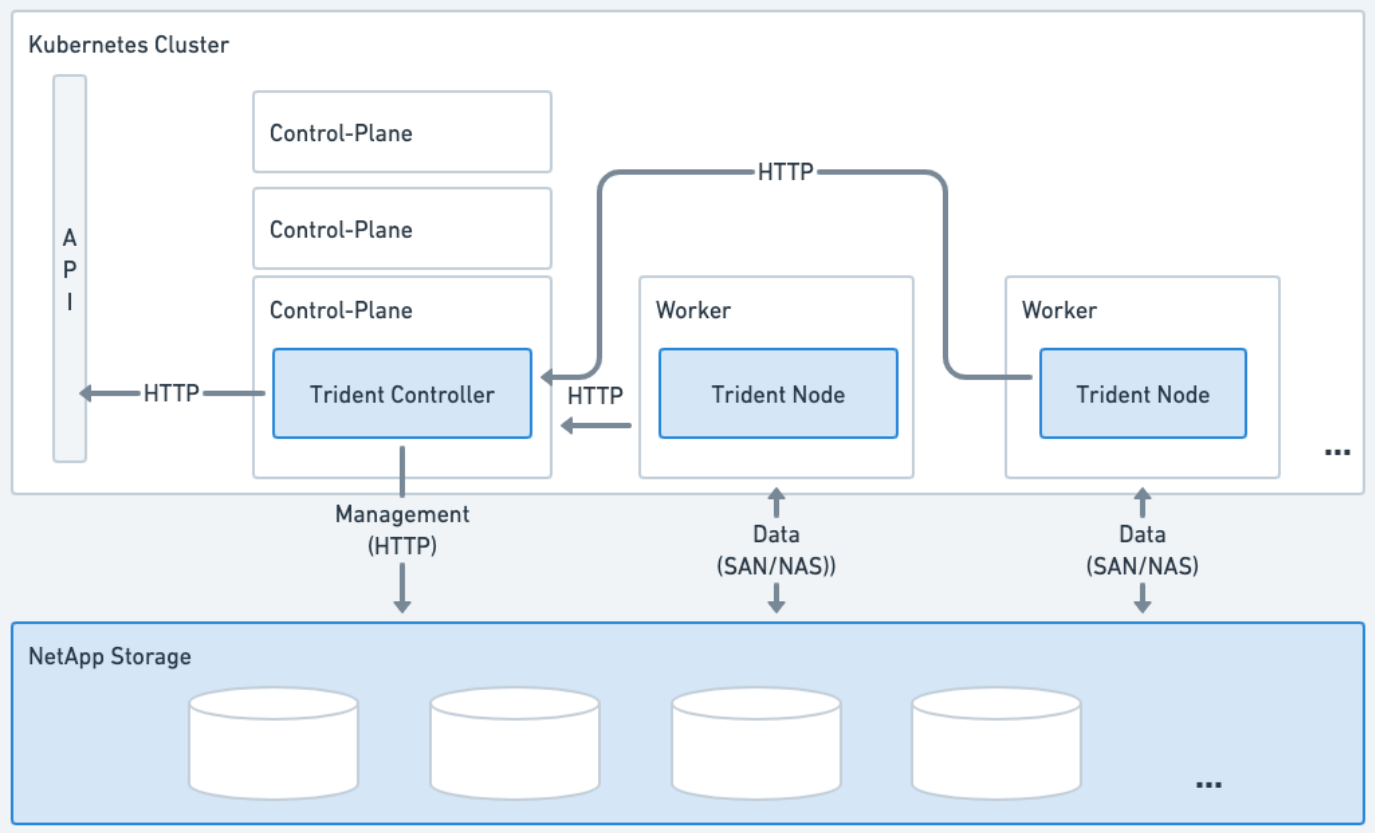


Figura 1. Trident implantado no cluster Kubernetes

Cápsula de controle Trident

O Trident Controller Pod é um único Pod que executa o plugin CSI Controller.

- Responsável pelo provisionamento e gerenciamento de volumes no armazenamento NetApp.
- Gerenciado por uma implantação do Kubernetes
- Pode ser executado no plano de controle ou nos nós de trabalho, dependendo dos parâmetros de instalação.

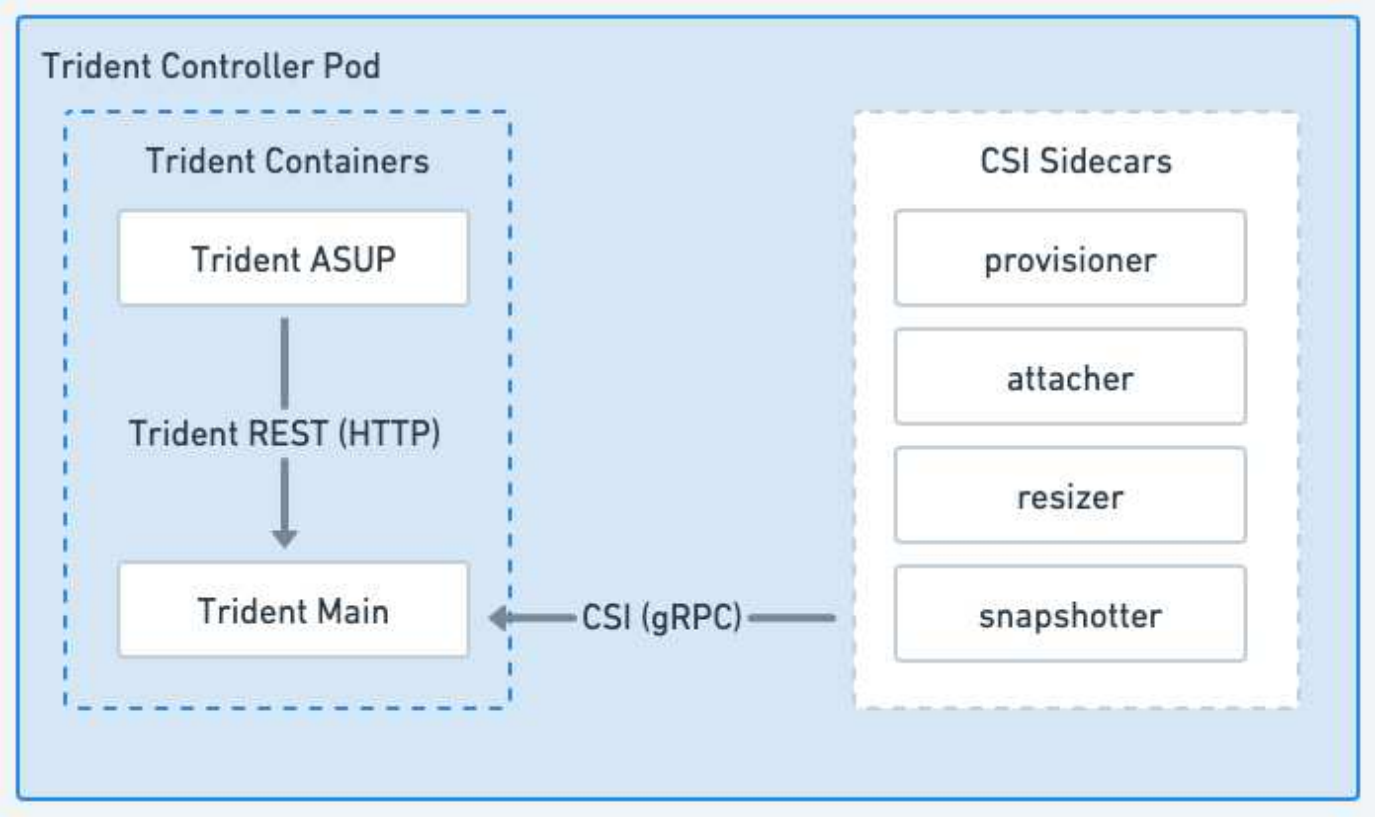


Figura 2. Diagrama do Módulo Controlador Trident

Cápsulas de nós Trident

Os Pods do Trident Node são Pods privilegiados que executam o plugin CSI Node.

- Responsável por montar e desmontar o armazenamento para os Pods em execução no host.
- Gerenciado por um DaemonSet do Kubernetes
- Deve ser executado em qualquer nó que possa montar armazenamento NetApp.

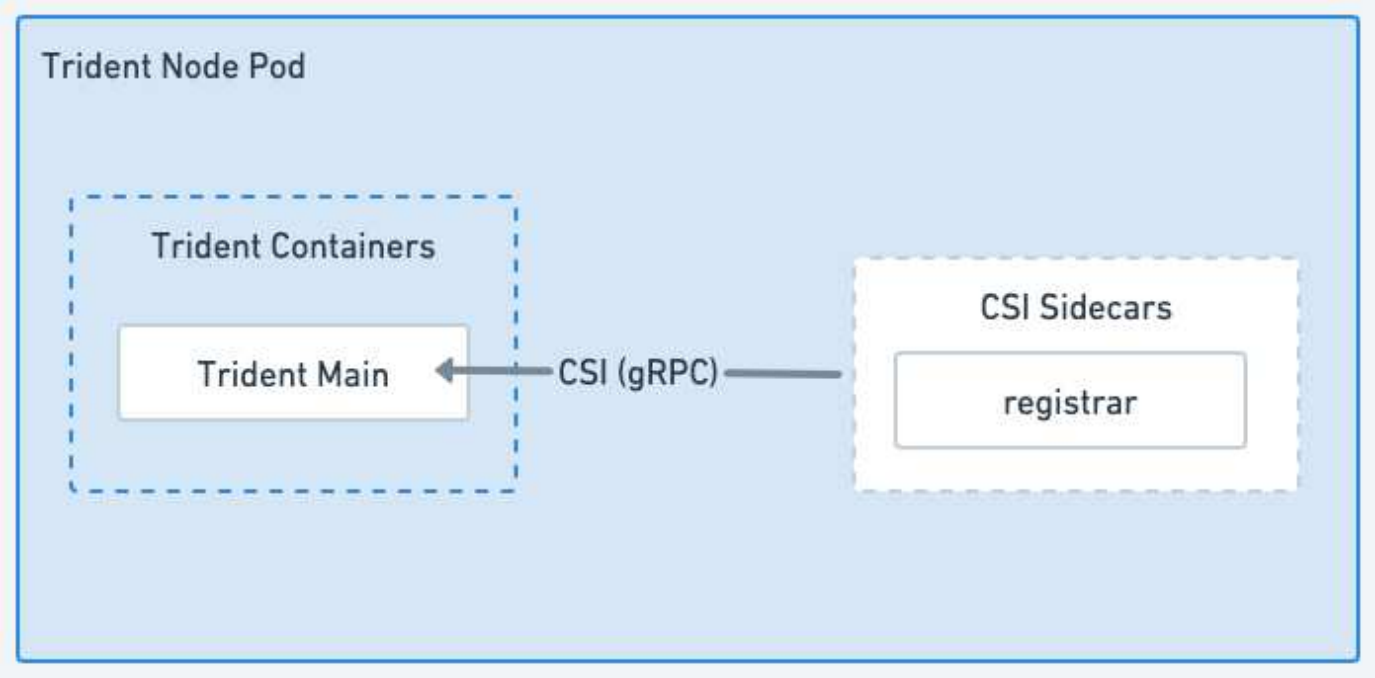


Figura 3. Diagrama do Trident Node Pod

Arquiteturas de cluster Kubernetes suportadas

O Trident é compatível com as seguintes arquiteturas do Kubernetes:

Arquiteturas de cluster Kubernetes	Suportado	Instalação padrão
Mestre único, computação	Sim	Sim
Múltiplos mestres, computação	Sim	Sim
Mestre, `etcd` , calcular	Sim	Sim
Mestre, infraestrutura, computação	Sim	Sim

Conceitos

Provisionamento

O provisionamento no Trident possui duas fases principais. A primeira fase associa uma classe de armazenamento ao conjunto de pools de armazenamento de backend adequados e ocorre como uma preparação necessária antes do provisionamento. A segunda fase inclui a criação do volume em si e requer a escolha de um conjunto de armazenamento dentre aqueles associados à classe de armazenamento do volume pendente.

Associação de classe de armazenamento

A associação de pools de armazenamento de backend a uma classe de armazenamento depende tanto dos atributos solicitados da classe de armazenamento quanto de seus atributos. `storagePools` ,

`additionalStoragePools` , e `excludeStoragePools` listas. Ao criar uma classe de armazenamento, o Trident compara os atributos e pools oferecidos por cada um de seus backends com aqueles solicitados pela classe de armazenamento. Se os atributos e o nome de um conjunto de armazenamento corresponderem a todos os atributos e nomes de conjunto solicitados, o Trident adicionará esse conjunto de armazenamento ao conjunto de conjuntos de armazenamento adequados para essa classe de armazenamento. Além disso, o Trident adiciona todos os pools de armazenamento listados no `additionalStoragePools` listar para esse conjunto, mesmo que seus atributos não atendam a todos ou a qualquer um dos atributos solicitados pela classe de armazenamento. Você deve usar o `excludeStoragePools` Lista para substituir e remover conjuntos de armazenamento do uso para uma classe de armazenamento. O Trident executa um processo semelhante sempre que você adiciona um novo backend, verificando se seus pools de armazenamento atendem aos das classes de armazenamento existentes e removendo quaisquer pools que tenham sido marcados como excluídos.

Criação de volume

O Trident utiliza então as associações entre classes de armazenamento e pools de armazenamento para determinar onde provisionar volumes. Ao criar um volume, o Trident primeiro obtém o conjunto de pools de armazenamento para a classe de armazenamento desse volume e, se você especificar um protocolo para o volume, o Trident remove os pools de armazenamento que não podem fornecer o protocolo solicitado (por exemplo, um backend NetApp HCI/ SolidFire não pode fornecer um volume baseado em arquivos, enquanto um backend ONTAP NAS não pode fornecer um volume baseado em blocos). O Trident randomiza a ordem desse conjunto resultante para facilitar uma distribuição uniforme dos volumes e, em seguida, itera sobre ele, tentando provisionar o volume em cada pool de armazenamento por sua vez. Se a operação for bem-sucedida em uma delas, ela retorna com êxito, registrando quaisquer falhas encontradas no processo. O Trident retorna uma falha **somente se** não conseguir provisionar em **todos** os pools de armazenamento disponíveis para a classe de armazenamento e o protocolo solicitados.

Instantâneos de volume

Saiba mais sobre como o Trident lida com a criação de snapshots de volume para seus drivers.

Saiba mais sobre a criação de snapshots de volume.

- Para o `ontap-nas` , `ontap-san` , `gcp-cvs` , e `azure-netapp-files` Nos drivers, cada Volume Persistente (PV) é mapeado para um FlexVol volume. Como resultado, os snapshots de volume são criados como snapshots NetApp . A tecnologia de snapshots da NetApp oferece mais estabilidade, escalabilidade, capacidade de recuperação e desempenho do que as tecnologias de snapshots concorrentes. Essas cópias instantâneas são extremamente eficientes, tanto em termos de tempo necessário para criá-las quanto em espaço de armazenamento.
- Para o `ontap-nas-flexgroup` No driver, cada Volume Persistente (PV) é mapeado para um FlexGroup. Como resultado, os snapshots de volume são criados como snapshots do NetApp FlexGroup . A tecnologia de snapshots da NetApp oferece mais estabilidade, escalabilidade, capacidade de recuperação e desempenho do que as tecnologias de snapshots concorrentes. Essas cópias instantâneas são extremamente eficientes, tanto em termos de tempo necessário para criá-las quanto em espaço de armazenamento.
- Para o `ontap-san-economy` O driver mapeia os PVs para LUNs criados em volumes FlexVol compartilhados. Os VolumeSnapshots dos PVs são obtidos através da execução de FlexClones do LUN associado. A tecnologia ONTAP FlexClone permite criar cópias até mesmo dos maiores conjuntos de dados quase instantaneamente. As cópias compartilham blocos de dados com seus arquivos originais, não consumindo espaço de armazenamento, exceto o necessário para os metadados.
- Para o `solidfire-san` No driver, cada PV corresponde a um LUN criado no software NetApp Element

/cluster NetApp HCI . Os VolumeSnapshots são representados por snapshots de Elementos do LUN subjacente. Esses instantâneos são cópias pontuais e ocupam apenas uma pequena quantidade de recursos e espaço do sistema.

- Ao trabalhar com o `ontap-nas` e `ontap-san` Os drivers e snapshots do ONTAP são cópias pontuais do FlexVol e consomem espaço no próprio FlexVol . Isso pode resultar na redução da quantidade de espaço gravável no volume ao longo do tempo, à medida que os snapshots são criados/agendados. Uma maneira simples de resolver isso é aumentar o volume redimensionando-o através do Kubernetes. Outra opção é excluir os snapshots que não são mais necessários. Quando um VolumeSnapshot criado por meio do Kubernetes é excluído, o Trident também exclui o snapshot ONTAP associado. Os snapshots do ONTAP que não foram criados por meio do Kubernetes também podem ser excluídos.

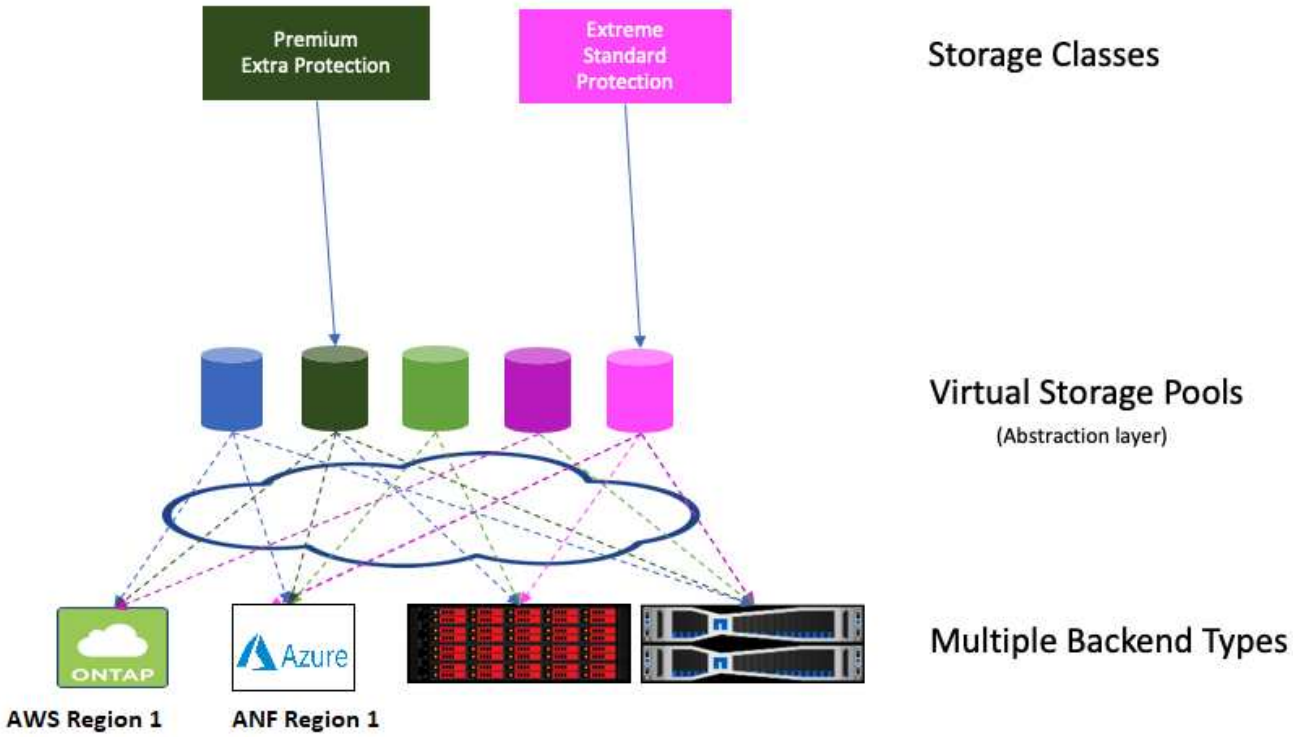
Com o Trident, você pode usar VolumeSnapshots para criar novos PVs a partir deles. A criação de PVs a partir desses snapshots é realizada utilizando a tecnologia FlexClone para backends ONTAP e CVS compatíveis. Ao criar um PV a partir de um snapshot, o volume de suporte é um FlexClone do volume pai do snapshot. O `solidfire-san` O driver utiliza clones de volume do software Element para criar PVs a partir de snapshots. Aqui, ele cria um clone a partir do instantâneo do elemento.

Piscinas virtuais

Os pools virtuais fornecem uma camada de abstração entre os backends de armazenamento do Trident e o Kubernetes. `StorageClasses` . Eles permitem que um administrador defina aspectos, como localização, desempenho e proteção para cada backend de uma maneira comum e independente do backend, sem criar um `StorageClass` Especifique qual backend físico, pool de backends ou tipo de backend usar para atender aos critérios desejados.

Saiba mais sobre piscinas virtuais

O administrador de armazenamento pode definir pools virtuais em qualquer um dos backends do Trident em um arquivo de definição JSON ou YAML.



Qualquer aspecto especificado fora da lista de pools virtuais é global para o backend e será aplicado a todos os pools virtuais, enquanto cada pool virtual pode especificar um ou mais aspectos individualmente (substituindo quaisquer aspectos globais do backend).

- Ao definir pools virtuais, não tente reorganizar a ordem dos pools virtuais existentes em uma definição de backend.
- Desaconselhamos a modificação dos atributos de um pool virtual existente. Você deve definir um novo pool virtual para efetuar as alterações.

A maioria dos aspectos é especificada em termos específicos do backend. Fundamentalmente, os valores dos aspectos não são expostos fora do driver do backend e não estão disponíveis para correspondência em `StorageClasses` Em vez disso, o administrador define um ou mais rótulos para cada pool virtual. Cada rótulo é um par chave:valor, e os rótulos podem ser comuns a diferentes sistemas de backend. Assim como os aspectos, os rótulos podem ser especificados por pool ou globalmente no backend. Diferentemente dos aspectos, que possuem nomes e valores predefinidos, o administrador tem total liberdade para definir as chaves e os valores dos rótulos conforme necessário. Para maior conveniência, os administradores de armazenamento podem definir rótulos por pool virtual e agrupar volumes por rótulo.

Os rótulos do pool virtual podem ser definidos usando estes caracteres:

- letras maiúsculas `A-Z`
- letras minúsculas `a-z`
- números `0-9`
- sublinhados _
- hífens -

UM `StorageClass` Identifica qual pool virtual usar, fazendo referência aos rótulos dentro de um parâmetro seletor. Os seletores de pool virtuais são compatíveis com as seguintes operadoras:

Operador	Exemplo	O valor do rótulo de um pool deve:
`=`	desempenho=premium	Corresponder
`!=`	desempenho!=extremo	Não corresponde
`in`	localização em (leste, oeste)	Esteja no conjunto de valores
`notin`	desempenho notin (prata, bronze)	Não estar no conjunto de valores
`<key>`	proteção	Existir com qualquer valor
`!<key>`	!proteção	Não existe

Grupos de acesso a volume

Saiba mais sobre como a Trident utiliza "grupos de acesso a volume" .

> Ignore esta seção se estiver usando o CHAP, que é recomendado para simplificar o gerenciamento e evitar o limite de escalabilidade descrito abaixo. Além disso, se você estiver usando o Trident no modo CSI, pode ignorar esta seção. O Trident utiliza o CHAP quando instalado como um provisionador CSI aprimorado.

Saiba mais sobre grupos de acesso por volume

O Trident pode usar grupos de acesso a volumes para controlar o acesso aos volumes que provisiona. Se o CHAP estiver desativado, espera-se encontrar um grupo de acesso chamado `trident` a menos que você especifique um ou mais IDs de grupo de acesso na configuração.

Embora o Trident associe novos volumes aos grupos de acesso configurados, ele não cria nem gerencia os próprios grupos de acesso. Os grupos de acesso devem existir antes da adição do backend de armazenamento ao Trident e precisam conter os IQNs iSCSI de todos os nós do cluster Kubernetes que possam potencialmente montar os volumes provisionados por esse backend. Na maioria das instalações, isso inclui todos os nós de trabalho do cluster.

Para clusters Kubernetes com mais de 64 nós, você deve usar vários grupos de acesso. Cada grupo de acesso pode conter até 64 IQNs, e cada volume pode pertencer a quatro grupos de acesso. Com o máximo de quatro grupos de acesso configurados, qualquer nó em um cluster de até 256 nós poderá acessar qualquer volume. Para obter os limites mais recentes para grupos de acesso por volume, consulte "aqui" .

Se você estiver modificando a configuração de uma que esteja usando a configuração padrão. `trident` Para acessar um grupo que também utiliza outros, inclua o ID do grupo. `trident` grupo de acesso na lista.

Guia rápido para Trident

Você pode instalar o Trident e começar a gerenciar recursos de armazenamento em poucos passos. Antes de começar, revise"Requisitos do Trident" .

> Para o Docker, consulte"Trident para Docker" .

1 Prepare o nó de trabalho

Todos os nós de trabalho no cluster Kubernetes devem ser capazes de montar os volumes que você provisionou para seus pods.

"Prepare o nó de trabalho"

2 Instalar Trident

A Trident oferece diversos métodos e modos de instalação otimizados para uma variedade de ambientes e organizações.

"Instalar Trident"

3 Crie um backend

Um backend define a relação entre o Trident e um sistema de armazenamento. Ele informa ao Trident como se comunicar com esse sistema de armazenamento e como o Trident deve provisionar volumes a partir dele.

"Configure um backend"para o seu sistema de armazenamento

4 Crie uma classe de armazenamento do Kubernetes.

O objeto StorageClass do Kubernetes especifica o Trident como provisionador e permite criar uma classe de armazenamento para provisionar volumes com atributos personalizáveis. O Trident cria uma classe de armazenamento correspondente para objetos do Kubernetes que especificam o provisionador Trident .

"Criar uma classe de armazenamento"

5 Forneça um volume

Um *PersistentVolume* (PV) é um recurso de armazenamento físico provisionado pelo administrador do cluster em um cluster Kubernetes. O *PersistentVolumeClaim* (PVC) é uma solicitação de acesso ao PersistentVolume no cluster.

Crie um PersistentVolume (PV) e um PersistentVolumeClaim (PVC) que utilize a StorageClass do Kubernetes configurada para solicitar acesso ao PV. Em seguida, você pode montar o painel fotovoltaico em um suporte.

"Forneça um volume"

O que vem a seguir?

Agora você pode adicionar backends adicionais, gerenciar classes de armazenamento, gerenciar backends e realizar operações de volume.

Requisitos

Antes de instalar o Trident, você deve revisar estes requisitos gerais do sistema. Alguns sistemas de backend específicos podem ter requisitos adicionais.

Informações essenciais sobre o Trident

Você deve ler as seguintes informações críticas sobre Trident.

Informações críticas sobre o Trident

- O Kubernetes 1.34 agora é compatível com o Trident. Atualize o Trident antes de atualizar o Kubernetes.
- A Trident impõe rigorosamente o uso da configuração de multipathing em ambientes SAN, com um valor recomendado de `find_multipaths: no` no arquivo multipath.conf.

  Utilização de configuração sem multipathing ou utilização de `find_multipaths: yes` ou `find_multipaths: smart` O valor no arquivo multipath.conf resultará em falhas de montagem. A Trident recomendou o uso de `find_multipaths: no` desde a versão de 21/07.

Front-ends suportados (orquestradores)

O Trident suporta vários mecanismos de contêineres e orquestradores, incluindo os seguintes:

- Anthos On-Prem (VMware) e Anthos em bare metal 1.16
- Kubernetes 1.27 - 1.34
- OpenShift 4.12, 4.14 - 4.19 (Se você planeja usar a preparação do nó iSCSI com o OpenShift 4.19, a versão mínima do Trident suportada é 25.06.1.)

> O Trident continua a oferecer suporte a versões mais antigas do OpenShift, em conformidade com o"Ciclo de vida da versão do Red Hat Extended Update Support (EUS)" , mesmo que dependam de versões do Kubernetes que já não são oficialmente suportadas pela parte superior da cadeia de suprimentos. Ao instalar o Trident nesses casos, você pode ignorar com segurança quaisquer mensagens de aviso sobre a versão do Kubernetes.

- Rancher Kubernetes Engine 2 (RKE2) v1.27.x - 1.34.x

> *Embora o Trident seja compatível com as versões 1.27.x a 1.34.x do Rancher Kubernetes Engine 2 (RKE2), o Trident foi qualificado atualmente apenas no RKE2 v1.28.5+rke2r1.*

O Trident também funciona com uma série de outras ofertas de Kubernetes totalmente gerenciadas e autogerenciadas, incluindo Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Mirantis Kubernetes Engine (MKE) e o portfólio VMware Tanzu.

Trident e ONTAP podem ser usados como provedores de armazenamento para"KubeVirt" .

> Antes de atualizar um cluster Kubernetes da versão 1.25 para a 1.26 ou posterior que tenha o Trident instalado, consulte o seguinte:"Atualize uma instalação do Helm" .

Backends suportados (armazenamento)

Para usar o Trident, você precisa de um ou mais dos seguintes backends suportados:

- Amazon FSx for NetApp ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP
- Google Cloud NetApp Volumes
- NetApp All SAN Array (ASA)
- Versões de cluster FAS, AFF, Select ou ASA r2 (iSCSI e NVMe/TCP) locais com suporte limitado da NetApp. Ver"Suporte à versão do software" .
- Software NetApp HCI/Element versão 11 ou superior

Suporte do Trident para virtualização KubeVirt e OpenShift.

Drivers de armazenamento suportados:

O Trident oferece suporte aos seguintes drivers ONTAP para virtualização KubeVirt e OpenShift:

- ontap-nas
- ontap-nas-economy
- ontap-san (iSCSI, FCP, NVMe sobre TCP)
- ontap-san-economy (somente iSCSI)

Pontos a considerar:

- Atualize a classe de armazenamento para ter o `fsType` parâmetro (por exemplo: `fsType: "ext4"`) em um ambiente de virtualização OpenShift. Se necessário, defina explicitamente o modo de volume para bloquear usando o `volumeMode=Block` parâmetro no `dataVolumeTemplates` Notificar a CDI para criar volumes de dados em bloco.
- *Modo de acesso RWX para drivers de armazenamento em bloco*: os drivers ontap-san (iSCSI, NVMe/TCP, FC) e ontap-san-economy (iSCSI) são suportados apenas com "volumeMode: Block" (dispositivo bruto). Para esses motoristas, o `fstype` O parâmetro não pode ser usado porque os volumes são fornecidos no modo de dispositivo bruto.
- Para fluxos de trabalho de migração em tempo real que exigem o modo de acesso RWX, estas combinações são suportadas:
  - NFS + `volumeMode=Filesystem`
  - iSCSI + `volumeMode=Block` (dispositivo bruto)
  - NVMe/TCP + `volumeMode=Block` (dispositivo bruto)
  - FC + `volumeMode=Block` (dispositivo bruto)

Requisitos de funcionalidade

A tabela abaixo resume os recursos disponíveis nesta versão do Trident e as versões do Kubernetes que ela suporta.

Recurso	Versão do Kubernetes	É necessário ter recursos de segurança?
Trident	1,27 - 1,34	Não

Recurso	Versão do Kubernetes	É necessário ter recursos de segurança?
Instantâneos de volume	1,27 - 1,34	Não
PVC de Instantâneos de Volume	1,27 - 1,34	Não
Redimensionamento de PV iSCSI	1,27 - 1,34	Não
ONTAP Bidirecional CHAP	1,27 - 1,34	Não
Políticas de Exportação Dinâmicas	1,27 - 1,34	Não
Operador Trident	1,27 - 1,34	Não
Topologia CSI	1,27 - 1,34	Não

Sistemas operacionais de host testados

Embora o Trident não ofereça suporte oficial a sistemas operacionais específicos, sabe-se que os seguintes funcionam:

- Versões do Red Hat Enterprise Linux CoreOS (RHCOS) suportadas pela plataforma OpenShift Container em AMD64 e ARM64
- Red Hat Enterprise Linux (RHEL) 8 ou posterior em AMD64 e ARM64

  NVMe/TCP requer RHEL 9 ou posterior.

- Ubuntu 22.04 LTS ou posterior em AMD64 e ARM64
- Servidor Windows 2022
- SUSE Linux Enterprise Server (SLES) 15 ou posterior

Por padrão, o Trident é executado em um contêiner e, portanto, funcionará em qualquer máquina virtual Linux. No entanto, esses trabalhadores precisam ser capazes de montar os volumes fornecidos Trident usando o cliente NFS padrão ou o iniciador iSCSI, dependendo dos backends que você estiver usando.

O `tridentctl` O utilitário também funciona em qualquer uma dessas distribuições Linux.

Configuração do host

Todos os nós de trabalho no cluster Kubernetes devem ser capazes de montar os volumes que você provisionou para seus pods. Para preparar os nós de trabalho, você deve instalar as ferramentas NFS, iSCSI ou NVMe, dependendo do driver selecionado.

"Prepare o nó de trabalho"

Configuração do sistema de armazenamento

O Trident pode exigir alterações no sistema de armazenamento antes que uma configuração de backend possa utilizá-lo.

"Configurar backends"

Portos Trident

O Trident requer acesso a portas específicas para comunicação.

"Portos Trident"

Imagens de contêiner e versões correspondentes do Kubernetes

Para instalações isoladas da rede (air-gapped), a lista a seguir serve como referência para as imagens de contêiner necessárias para instalar o Trident. Use o `tridentctl images` Comando para verificar a lista de imagens de contêiner necessárias.

Imagens de contêiner necessárias para o Trident 25.06.2

Versões do Kubernetes	Imagem do contêiner
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.2 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.2 (opcional)

Imagens de contêiner necessárias para o Trident 25.06

Versões do Kubernetes	Imagem do contêiner
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.0 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.0 (opcional)

Instalar Trident

Instale usando o operador Trident .

Instale usando o tridentctl

Instale utilizando um operador certificado do OpenShift.

Use o Trident

Prepare o nó de trabalho

Todos os nós de trabalho no cluster Kubernetes devem ser capazes de montar os volumes que você provisionou para seus pods. Para preparar os nós de trabalho, você deve instalar as ferramentas NFS, iSCSI, NVMe/TCP ou FC, dependendo da sua seleção de driver.

Selecionar as ferramentas certas

Se você estiver usando uma combinação de drivers, deverá instalar todas as ferramentas necessárias para seus drivers. As versões recentes do Red Hat Enterprise Linux CoreOS (RHCOS) vêm com essas ferramentas instaladas por padrão.

Ferramentas NFS

"Instale as ferramentas NFS"Se você estiver usando: `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` .

Ferramentas iSCSI

"Instale as ferramentas iSCSI"Se você estiver usando: `ontap-san` , `ontap-san-economy` , `solidfire-san` .

Ferramentas NVMe

"Instale as ferramentas NVMe"se você estiver usando `ontap-san` para o protocolo de memória não volátil expressa (NVMe) sobre TCP (NVMe/TCP).

> A NetApp recomenda o ONTAP 9.12 ou posterior para NVMe/TCP.

Ferramentas SCSI sobre FC

Consulte"Formas de configurar hosts SAN FC e FC-NVMe" Para obter mais informações sobre como configurar seus hosts SAN FC e FC-NVMe.

"Instale as ferramentas FC"se você estiver usando `ontap-san` com sanType `fcp` (SCSI sobre FC).

Pontos a considerar: * SCSI sobre FC é suportado em ambientes OpenShift e KubeVirt. * SCSI sobre FC não é suportado no Docker. * A autorrecuperação do iSCSI não se aplica ao SCSI sobre FC.

Descoberta de serviços de nó

O Trident tenta detectar automaticamente se o nó pode executar serviços iSCSI ou NFS.

> A descoberta de serviços de nós identifica os serviços descobertos, mas não garante que estejam configurados corretamente. Por outro lado, a ausência de um serviço detectado não garante que a montagem do volume falhará.

Revisar eventos

O Trident cria eventos para o nó identificar os serviços descobertos. Para rever esses eventos, execute:

```
kubectl get event -A --field-selector involvedObject.name=<Kubernetes node name>
```

Análise dos serviços descobertos

O Trident identifica os serviços habilitados para cada nó no CR do nó Trident . Para visualizar os serviços detectados, execute:

```
tridentctl get node -o wide -n <Trident namespace>
```

Volumes NFS

Instale as ferramentas NFS usando os comandos para o seu sistema operacional. Certifique-se de que o serviço NFS seja iniciado durante a inicialização do sistema.

RHEL 8+

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```

Reinicie os nós de trabalho após instalar as ferramentas NFS para evitar falhas ao conectar volumes aos contêineres.

volumes iSCSI

O Trident pode estabelecer automaticamente uma sessão iSCSI, analisar LUNs, descobrir dispositivos multipath, formatá-los e montá-los em um pod.

Capacidades de autorrecuperação do iSCSI

Para sistemas ONTAP , o Trident executa a autorrecuperação iSCSI a cada cinco minutos para:

1. **Identifique** o estado desejado da sessão iSCSI e o estado atual da sessão iSCSI.
2. **Compare** o estado desejado com o estado atual para identificar os reparos necessários. A Trident determina as prioridades de reparo e quando antecipar os reparos.
3. **Executar os reparos** necessários para retornar o estado atual da sessão iSCSI ao estado desejado.

Os registros da atividade de autocura estão localizados em `trident-main` contêiner no respectivo pod do Daemonset. Para visualizar os registros, você precisa ter configurado `debug` para "verdadeiro" durante a instalação do Trident .

Os recursos de autorrecuperação do Trident iSCSI podem ajudar a prevenir:

- Sessões iSCSI obsoletas ou com problemas de funcionamento que podem ocorrer após uma falha de conectividade de rede. No caso de uma sessão inativa, o Trident aguarda sete minutos antes de encerrar a sessão para restabelecer a conexão com um portal.

> Por exemplo, se os segredos CHAP fossem rotacionados no controlador de armazenamento e a rede perdesse a conectividade, os segredos CHAP antigos (obsoletos) poderiam persistir. A função de autorrecuperação consegue reconhecer isso e restabelecer automaticamente a sessão para aplicar os segredos CHAP atualizados.

- Sessões iSCSI ausentes
- LUNs ausentes

Pontos a considerar antes de atualizar o Trident

- Se apenas os grupos de interfaces (igroups) por nó (introduzidos na versão 23.04+) estiverem em uso, a autorrecuperação do iSCSI iniciará novas varreduras SCSI para todos os dispositivos no barramento SCSI.
- Se apenas grupos de interface (igroups) com escopo de backend (obsoletos a partir da versão 23.04) estiverem em uso, a autorrecuperação do iSCSI iniciará novas varreduras SCSI para obter os IDs de LUN exatos no barramento SCSI.
- Se uma combinação de igroups por nó e igroups com escopo de backend estiver em uso, a autorrecuperação do iSCSI iniciará novas varreduras SCSI para obter os IDs de LUN exatos no barramento SCSI.

Instale as ferramentas iSCSI

Instale as ferramentas iSCSI usando os comandos para o seu sistema operacional.

Antes de começar

- Cada nó no cluster Kubernetes deve ter um IQN único. **Este é um pré-requisito necessário**.
- Se estiver usando o RHCOS versão 4.5 ou posterior, ou outra distribuição Linux compatível com RHEL, com o `solidfire-san` Para drivers e Element OS 12.5 ou anterior, certifique-se de que o algoritmo de autenticação CHAP esteja definido como MD5 em `/etc/iscsi/iscsid.conf` Os algoritmos CHAP seguros e compatíveis com FIPS, SHA1, SHA-256 e SHA3-256, estão disponíveis no Element 12.7.

```
sudo sed -i 's/^\(node.session.auth.chap_algs\).*/\1 = MD5/'
/etc/iscsi/iscsid.conf
```

- Ao usar nós de trabalho que executam RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) com PVs iSCSI, especifique o `discard` A opção mountOption na StorageClass permite realizar a recuperação de espaço em linha. Consulte "Documentação da Red Hat" .
- Certifique-se de ter atualizado para a versão mais recente do `multipath-tools` .

RHEL 8+

1. Instale os seguintes pacotes do sistema:

```
sudo yum install -y lsscsi iscsi-initiator-utils device-mapper-
multipath
```

2. Verifique se a versão do iscsi-initiator-utils é 6.2.0.874-2.el7 ou posterior:

```
rpm -q iscsi-initiator-utils
```

3. Configure a digitalização para manual:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Ativar multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

Garantir `/etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

5. Certifique-se de que `iscsid` e `multipathd` estão em execução:

```
sudo systemctl enable --now iscsid multipathd
```

6. Ative e inicie `iscsi` :

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. Instale os seguintes pacotes do sistema:

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools
scsitools
```

2. Verifique se a versão do open-iscsi é 2.0.874-5ubuntu2.10 ou posterior (para bionic) ou 2.0.874-7.1ubuntu6.1 ou posterior (para focal):

```
dpkg -l open-iscsi
```

3. Configure a digitalização para manual:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Ativar multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

> Garantir `/etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

5. Certifique-se de que `open-iscsi` e `multipath-tools` estão habilitados e em execução:

```
sudo systemctl status multipath-tools
sudo systemctl enable --now open-iscsi.service
sudo systemctl status open-iscsi
```

> Para o Ubuntu 18.04, você deve descobrir as portas de destino com `iscsiadm` antes de começar `open-iscsi` para que o daemon iSCSI seja iniciado. Você também pode modificar o `iscsi` serviço para iniciar `iscsid` automaticamente.

Configure ou desative a autorrecuperação do iSCSI.

Você pode configurar as seguintes opções de autorrecuperação do Trident iSCSI para corrigir sessões obsoletas:

- **Intervalo de autorrecuperação do iSCSI**: Determina a frequência com que a autorrecuperação do iSCSI é acionada (padrão: 5 minutos). Você pode configurá-lo para ser executado com mais frequência, definindo um número menor, ou com menos frequência, definindo um número maior.

Definir o intervalo de autorrecuperação do iSCSI para 0 interrompe completamente a autorrecuperação do iSCSI. Não recomendamos desativar a autorrecuperação do iSCSI; ela só deve ser desativada em cenários específicos, quando a autorrecuperação do iSCSI não estiver funcionando conforme o esperado ou para fins de depuração.

- **Tempo de espera de auto-recuperação do iSCSI**: Determina a duração que a auto-recuperação do iSCSI aguarda antes de encerrar uma sessão com problemas e tentar entrar novamente (padrão: 7 minutos). Você pode configurá-lo para um número maior, de forma que as sessões identificadas como não íntegras tenham que esperar mais tempo antes de serem desconectadas e, em seguida, uma nova tentativa de login seja feita, ou para um número menor, para desconectar e entrar mais cedo.

Leme

Para configurar ou alterar as configurações de autorrecuperação do iSCSI, passe o `iscsiSelfHealingInterval` e `iscsiSelfHealingWaitTime` parâmetros durante a instalação ou atualização do Helm.

O exemplo a seguir define o intervalo de autorrecuperação do iSCSI para 3 minutos e o tempo de espera para autorrecuperação para 6 minutos:

```
helm install trident trident-operator-100.2506.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

Para configurar ou alterar as configurações de autorrecuperação do iSCSI, passe o `iscsi-self-healing-interval` e `iscsi-self-healing-wait-time` parâmetros durante a instalação ou atualização do tridentctl.

O exemplo a seguir define o intervalo de autorrecuperação do iSCSI para 3 minutos e o tempo de espera para autorrecuperação para 6 minutos:

```
tridentctl install --iscsi-self-healing-interval=3m0s --iscsi-self
-healing-wait-time=6m0s -n trident
```

Volumes NVMe/TCP

Instale as ferramentas NVMe usando os comandos específicos do seu sistema operacional.

- O NVMe requer RHEL 9 ou posterior.
- Se a versão do kernel do seu nó Kubernetes for muito antiga ou se o pacote NVMe não estiver disponível para a sua versão do kernel, talvez seja necessário atualizar a versão do kernel do seu nó para uma que inclua o pacote NVMe.

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Verifique a instalação

Após a instalação, verifique se cada nó no cluster Kubernetes possui um NQN exclusivo usando o comando:

```
cat /etc/nvme/hostnqn
```

Trident modifica o `ctrl_device_tmo` Valor para garantir que o NVMe não abandone o caminho caso ele falhe. Não altere esta configuração.

SCSI sobre volumes FC

Agora você pode usar o protocolo Fibre Channel (FC) com o Trident para provisionar e gerenciar recursos de armazenamento no sistema ONTAP .

Pré-requisitos

Configure as definições de rede e de nó necessárias para a FC.

Configurações de rede

1. Obtenha o WWPN das interfaces de destino. Consulte "exibição de interface de rede" para mais informações.
2. Obtenha o WWPN para as interfaces no iniciador (Host).

   Consulte os utilitários correspondentes do sistema operacional do host.
3. Configure o zoneamento no switch FC usando os WWPNs do host e do destino.

   Consulte a documentação do respectivo fornecedor do switch para obter informações.

   Consulte a seguinte documentação do ONTAP para obter detalhes:

   - "Visão geral do zoneamento Fibre Channel e FCoE"
   - "Formas de configurar hosts SAN FC e FC-NVMe"

Instale as ferramentas FC

Instale as ferramentas FC usando os comandos para o seu sistema operacional.

- Ao usar nós de trabalho que executam RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) com PVs FC, especifique o `discard` A opção mountOption na StorageClass permite realizar a recuperação de espaço em linha. Consulte "Documentação da Red Hat" .

RHEL 8+

1. Instale os seguintes pacotes do sistema:

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. Ativar multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

> Garantir `/etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

3. Certifique-se de que `multipathd` está em execução:

```
sudo systemctl enable --now multipathd
```

Ubuntu

1. Instale os seguintes pacotes do sistema:

```
sudo apt-get install -y lsscsi sg3-utils multipath-tools scsitools
```

2. Ativar multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

> Garantir `/etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

3. Certifique-se de que `multipath-tools` Está habilitado e em execução:

```
sudo systemctl status multipath-tools
```

Configurar e gerenciar backends

Configurar backends

Um backend define a relação entre o Trident e um sistema de armazenamento. Ele informa ao Trident como se comunicar com esse sistema de armazenamento e como o Trident deve provisionar volumes a partir dele.

O Trident oferece automaticamente pools de armazenamento de backends que correspondem aos requisitos definidos por uma classe de armazenamento. Aprenda como configurar o backend do seu sistema de armazenamento.

- "Configure um backend do Azure NetApp Files"
- "Configure um backend do Google Cloud NetApp Volumes"
- "Configure um Cloud Volumes Service para o backend do Google Cloud Platform."
- "Configure um backend NetApp HCI ou SolidFire."
- "Configure um backend com drivers ONTAP ou Cloud Volumes ONTAP NAS."
- "Configure um backend com drivers ONTAP SAN ou Cloud Volumes ONTAP."
- "Use o Trident com o Amazon FSx for NetApp ONTAP"

Azure NetApp Files

Configure um backend do Azure NetApp Files

Você pode configurar o Azure NetApp Files como backend para o Trident. Você pode conectar volumes NFS e SMB usando um backend do Azure NetApp Files . O Trident também oferece suporte ao gerenciamento de credenciais usando identidades gerenciadas para clusters do Azure Kubernetes Services (AKS).

Detalhes do driver Azure NetApp Files

O Trident fornece os seguintes drivers de armazenamento do Azure NetApp Files para comunicação com o cluster. Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`azure-netapp-files`	NFS SMB	Sistema de arquivos	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Considerações

- O serviço Azure NetApp Files não suporta volumes menores que 50 GiB. O Trident cria automaticamente volumes de 50 GiB se um volume menor for solicitado.
- O Trident suporta volumes SMB montados em pods executados apenas em nós Windows.

Identidades gerenciadas para AKS

Suporte Trident"identidades gerenciadas" para clusters do Azure Kubernetes Services. Para aproveitar o gerenciamento simplificado de credenciais oferecido pelas identidades gerenciadas, você precisa ter:

- Um cluster Kubernetes implantado usando o AKS.
- Identidades gerenciadas configuradas no cluster Kubernetes do AKS
- Trident instalado que inclui o `cloudProvider` para especificar `"Azure"` .

Operador do Trident

Para instalar o Trident usando o operador Trident , edite `tridentorchestrator_cr.yaml` para definir `cloudProvider` para `"Azure"` . Por exemplo:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
```

Leme

O exemplo a seguir instala conjuntos Trident. `cloudProvider` para o Azure usando a variável de ambiente `$CP` :

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

O exemplo a seguir instala o Trident e configura o `cloudProvider` bandeira para `Azure` :

```
tridentctl install --cloud-provider="Azure" -n trident
```

Identidade na nuvem para AKS

A identidade na nuvem permite que os pods do Kubernetes acessem recursos do Azure autenticando-se como uma identidade de carga de trabalho, em vez de fornecer credenciais explícitas do Azure.

Para aproveitar as vantagens da identidade na nuvem no Azure, você precisa ter:

- Um cluster Kubernetes implantado usando o AKS.

- Identidade de carga de trabalho e emissor OIDC configurados no cluster Kubernetes do AKS
- Trident instalado que inclui o `cloudProvider` para especificar `"Azure"` e `cloudIdentity` especificando a identidade da carga de trabalho

Operador do Trident

Para instalar o Trident usando o operador Trident , edite `tridentorchestrator_cr.yaml` para definir `cloudProvider` para `"Azure"` e definir `cloudIdentity` para `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Por exemplo:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxx' # Edit
```

Leme

Defina os valores para os parâmetros **cloud-provider (CP)** e **cloud-identity (CI)** usando as seguintes variáveis de ambiente:

```
export CP="Azure"
export CI="'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx'"
```

O exemplo a seguir instala o Trident e configura `cloudProvider` para o Azure usando a variável de ambiente `$CP` e define o `cloudIdentity` usando a variável de ambiente `$CI` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>tridentctl</code>

Defina os valores para os parâmetros **provedor de nuvem** e **identidade de nuvem** usando as seguintes variáveis de ambiente:

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx"
```

O exemplo a seguir instala o Trident e configura o `cloud-provider` bandeira para `$CP` , e `cloud-identity` para `$CI` :

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Prepare-se para configurar um backend do Azure NetApp Files.

Antes de configurar o backend do Azure NetApp Files , você precisa garantir que os seguintes requisitos sejam atendidos.

Pré-requisitos para volumes NFS e SMB

Se você estiver usando o Azure NetApp Files pela primeira vez ou em um novo local, será necessária alguma configuração inicial para configurar o Azure NetApp Files e criar um volume NFS. Consulte "Azure: Configure o Azure NetApp Files e crie um volume NFS." .

Para configurar e usar um "Azure NetApp Files" Para o backend, você precisa do seguinte:

- `subscriptionID`, `tenantID` , `clientID` , `location` , e `clientSecret` São opcionais ao usar identidades gerenciadas em um cluster AKS.
- `tenantID`, `clientID` , e `clientSecret` São opcionais ao usar uma identidade de nuvem em um cluster AKS.

- Uma piscina de capacidade máxima. Consulte"Microsoft: Criar um pool de capacidade para o Azure NetApp Files" .
- Uma sub-rede delegada ao Azure NetApp Files. Consulte"Microsoft: Delegar uma sub-rede ao Azure NetApp Files" .
- `subscriptionID`de uma assinatura do Azure com o Azure NetApp Files ativado.
- `tenantID`, `clientID` , e `clientSecret` de um"Registro do aplicativo" no Azure Active Directory com permissões suficientes para o serviço Azure NetApp Files . O cadastro no aplicativo deve usar um dos seguintes métodos:
  - O papel de Proprietário ou Colaborador"predefinido pelo Azure" .
  - UM"Função de Colaborador personalizada" no nível de assinatura(`assignableScopes`) com as seguintes permissões, que se limitam apenas ao que o Trident exige. Após criar a função personalizada,"Atribua a função usando o portal do Azure." .

Função de colaborador personalizado

```
{
  "id": "/subscriptions/<subscription-
id>/providers/Microsoft.Authorization/roleDefinitions/<role-
definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [
          "Microsoft.NetApp/netAppAccounts/capacityPools/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/delete",
```

```
            "Microsoft.Features/features/read",
            "Microsoft.Features/operations/read",
            "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
          ],
          "notActions": [],
          "dataActions": [],
          "notDataActions": []
        }
      ]
    }
}
```

- O Azure `location` que contém pelo menos um "sub-rede delegada" . A partir da versão 22.01 do Trident , o `location` O parâmetro é um campo obrigatório no nível superior do arquivo de configuração do backend. Os valores de localização especificados em pools virtuais são ignorados.
- Para usar `Cloud Identity` , pegue o `client ID` de um "identidade gerenciada atribuída pelo usuário" e especifique esse ID em `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Requisitos adicionais para volumes de PME

Para criar um volume SMB, você precisa ter:

- Active Directory configurado e conectado ao Azure NetApp Files. Consulte"Microsoft: Criar e gerenciar conexões do Active Directory para o Azure NetApp Files" .
- Um cluster Kubernetes com um nó controlador Linux e pelo menos um nó de trabalho Windows executando o Windows Server 2022. O Trident suporta volumes SMB montados em pods executados apenas em nós Windows.
- É necessário pelo menos um segredo Trident contendo suas credenciais do Active Directory para que o Azure NetApp Files possa se autenticar no Active Directory. Para gerar segredos `smbcreds` :

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Um proxy CSI configurado como um serviço do Windows. Para configurar um `csi-proxy` , consulte"GitHub: Proxy CSI" ou"GitHub: CSI Proxy para Windows" para nós do Kubernetes executados no Windows.

Opções e exemplos de configuração do backend do Azure NetApp Files

Saiba mais sobre as opções de configuração de back-end NFS e SMB para o Azure NetApp Files e veja exemplos de configuração.

Opções de configuração do backend

O Trident usa sua configuração de back-end (sub-rede, rede virtual, nível de serviço e localização) para criar volumes do Azure NetApp Files em pools de capacidade disponíveis na localização solicitada e que correspondam ao nível de serviço e à sub-rede solicitados.

* A partir da versão NetApp Trident 25.06, pools de capacidade de QoS manuais são suportados como uma prévia técnica.*

Os back-ends do Azure NetApp Files oferecem essas opções de configuração.

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	"azure-netapp-files"
`backendName`	Nome personalizado ou o backend de armazenamento	Nome do motorista + "_" + caracteres aleatórios
`subscriptionID`	O ID da assinatura da sua assinatura do Azure. Opcional quando as identidades gerenciadas estão habilitadas em um cluster do AKS.	
`tenantID`	O ID do locatário de um registro de aplicativo é opcional quando identidades gerenciadas ou identidade na nuvem são usadas em um cluster AKS.	
`clientID`	O ID do cliente de um registro de aplicativo é opcional quando identidades gerenciadas ou identidade na nuvem são usadas em um cluster AKS.	
`clientSecret`	O segredo do cliente de um registro de aplicativo é opcional quando identidades gerenciadas ou identidade na nuvem são usadas em um cluster AKS.	
`serviceLevel`	Um de `Standard` , `Premium` , ou `Ultra`	"" (aleatório)
`location`	Nome da localização do Azure onde os novos volumes serão criados. Opcional quando as identidades gerenciadas estão habilitadas em um cluster AKS.	

Parâmetro	Descrição	Padrão
`resourceGroups`	Lista de grupos de recursos para filtrar recursos descobertos	"[]" (sem filtro)
`netappAccounts`	Lista de contas NetApp para filtrar recursos descobertos	"[]" (sem filtro)
`capacityPools`	Lista de pools de capacidade para filtrar recursos descobertos	"[]" (sem filtro, aleatório)
`virtualNetwork`	Nome de uma rede virtual com uma sub-rede delegada	""
`subnet`	Nome de uma sub-rede delegada a `Microsoft.Netapp/volumes`	""
`networkFeatures`	Conjunto de recursos de VNet para um volume, pode ser `Basic` ou `Standard` . O recurso de Recursos de Rede não está disponível em todas as regiões e pode ser necessário ativá-lo por meio de uma assinatura. Especificando `networkFeatures` Quando a funcionalidade não está habilitada, o provisionamento de volumes falha.	""
`nfsMountOptions`	Controle preciso das opções de montagem NFS. Ignorado para volumes SMB. Para montar volumes usando NFS versão 4.1, inclua `nfsvers=4` Na lista de opções de montagem separadas por vírgulas, escolha NFS v4.1. As opções de montagem definidas na definição de uma classe de armazenamento substituem as opções de montagem definidas na configuração do backend.	"nfsvers=3"
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a este valor.	"" (não aplicado por padrão)
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo, `\{"api": false, "method": true, "discovery": true}` . Não utilize esta opção a menos que esteja solucionando problemas e precise de um despejo de logs detalhado.	nulo

Parâmetro	Descrição	Padrão
`nasType`	Configure a criação de volumes NFS ou SMB. As opções são `nfs` , `smb` ou nulo. Definir como nulo utiliza, por padrão, volumes NFS.	`nfs`
`supportedTopologies`	Representa uma lista de regiões e zonas que são suportadas por este sistema. Para obter mais informações, consulte"Utilizar a topologia CSI" .	
`qosType`	Representa o tipo de QoS: Automático ou Manual. **Prévia técnica para Trident 25.06**	Auto
`maxThroughput`	Define a taxa de transferência máxima permitida em MiB/s. Suportado apenas para pools de capacidade de QoS manual. **Prévia técnica para Trident 25.06**	`4 MiB/sec`

Para obter mais informações sobre os recursos de rede, consulte"Configure os recursos de rede para um volume do Azure NetApp Files." .

Permissões e recursos necessários

Se você receber um erro "Nenhum pool de capacidade encontrado" ao criar um PVC, é provável que o registro do seu aplicativo não tenha as permissões e os recursos necessários (sub-rede, rede virtual, pool de capacidade) associados. Se o modo de depuração estiver ativado, o Trident registrará os recursos do Azure descobertos quando o backend for criado. Verifique se está sendo utilizada a função apropriada.

Os valores para `resourceGroups` , `netappAccounts` , `capacityPools` , `virtualNetwork` , e `subnet` Podem ser especificados usando nomes curtos ou nomes totalmente qualificados. Na maioria das situações, recomenda-se o uso de nomes completos, pois nomes curtos podem corresponder a vários recursos com o mesmo nome.

O `resourceGroups` , `netappAccounts` , e `capacityPools` Os valores são filtros que restringem o conjunto de recursos descobertos àqueles disponíveis para este backend de armazenamento e podem ser especificados em qualquer combinação. Os nomes completos seguem este formato:

Tipo	Formatar
Grupo de recursos	<grupo de recursos>
Conta NetApp	<grupo de recursos>/<conta NetApp>
piscina de capacidade	<grupo de recursos>/<conta NetApp>/<pool de capacidade>
Rede virtual	<grupo de recursos>/<rede virtual>
Sub-rede	<grupo de recursos>/<rede virtual>/<sub-rede>

Provisionamento de volume

Você pode controlar o provisionamento de volumes padrão especificando as seguintes opções em uma seção específica do arquivo de configuração. Consulte Configurações de exemplo para mais detalhes.

Parâmetro	Descrição	Padrão
`exportRule`	Regras de exportação para novos volumes. `exportRule` Deve ser uma lista separada por vírgulas de qualquer combinação de endereços IPv4 ou sub-redes IPv4 na notação CIDR. Ignorado para volumes SMB.	"0.0.0.0/0"
`snapshotDir`	Controla a visibilidade do diretório .snapshot.	"verdadeiro" para NFSv4 "falso" para NFSv3
`size`	O tamanho padrão de novos volumes	"100G"
`unixPermissions`	Permissões Unix de novos volumes (4 dígitos octais). Ignorado para volumes SMB.	"" (Recurso em pré-visualização, requer inclusão na lista de permissões na assinatura)

Configurações de exemplo

Os exemplos a seguir mostram configurações básicas que deixam a maioria dos parâmetros com os valores padrão. Esta é a maneira mais fácil de definir um backend.

Configuração mínima

Esta é a configuração mínima absoluta do backend. Com essa configuração, o Trident descobre todas as suas contas NetApp , pools de capacidade e sub-redes delegadas ao Azure NetApp Files no local configurado e coloca novos volumes em um desses pools e sub-redes aleatoriamente. Porque `nasType` é omitido, o `nfs` A configuração padrão será aplicada e o servidor provisionará volumes NFS.

Essa configuração é ideal para quem está começando a usar o Azure NetApp Files e experimentando novos recursos, mas, na prática, você provavelmente desejará fornecer um escopo adicional para os volumes provisionados.

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
  tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
  clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
  clientSecret: SECRET
  location: eastus
```

Identidades gerenciadas para AKS

Esta configuração de backend omite `subscriptionID`, `tenantID`, `clientID`, e `clientSecret`, que são opcionais ao usar identidades gerenciadas.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
```

Identidade na nuvem para AKS

Esta configuração de backend omite `tenantID`, `clientID`, e `clientSecret`, que são opcionais ao usar uma identidade na nuvem.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

Configuração específica de nível de serviço com filtros de capacidade

Essa configuração de backend coloca volumes no Azure. `eastus` localização em um `Ultra` pool de capacidade. O Trident descobre automaticamente todas as sub-redes delegadas ao Azure NetApp Files nesse local e coloca um novo volume em uma delas aleatoriamente.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
```

Exemplo de backend com pools de capacidade de QoS manuais

Essa configuração de backend coloca volumes no Azure. `eastus` Localização com pools de capacidade QoS manuais. **Prévia técnica no NetApp Trident 25.06**.

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

Configuração avançada

Essa configuração de backend reduz ainda mais o escopo do posicionamento de volumes para uma única sub-rede e também modifica algumas configurações padrão de provisionamento de volumes.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
virtualNetwork: my-virtual-network
subnet: my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: "true"
  size: 200Gi
  unixPermissions: "0777"
```

Configuração de pool virtual

Essa configuração de backend define vários pools de armazenamento em um único arquivo. Isso é útil quando você tem vários pools de capacidade que suportam diferentes níveis de serviço e deseja criar classes de armazenamento no Kubernetes que os representem. Rótulos de piscinas virtuais foram usados para diferenciar as piscinas com base em `performance` .

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
  - application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
  - labels:
      performance: gold
    serviceLevel: Ultra
    capacityPools:
      - ultra-1
      - ultra-2
    networkFeatures: Standard
  - labels:
      performance: silver
    serviceLevel: Premium
    capacityPools:
      - premium-1
  - labels:
      performance: bronze
    serviceLevel: Standard
    capacityPools:
      - standard-1
      - standard-2
```

Configuração de topologias suportadas

O Trident facilita o provisionamento de volumes para cargas de trabalho com base em regiões e zonas de disponibilidade. O `supportedTopologies` O bloco nesta configuração de backend é usado para fornecer uma lista de regiões e zonas por backend. Os valores de região e zona especificados aqui devem corresponder aos valores de região e zona dos rótulos em cada nó do cluster Kubernetes. Essas regiões e zonas representam a lista de valores permitidos que podem ser fornecidos em uma classe de armazenamento. Para classes de armazenamento que contêm um subconjunto das regiões e zonas fornecidas em um backend, o Trident cria volumes na região e zona mencionadas. Para obter mais informações, consulte"Utilizar a topologia CSI" .

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

Definições de classe de armazenamento

A seguir `StorageClass` As definições referem-se aos conjuntos de armazenamento acima.

Definições de exemplo usando `parameter.selector` campo

Usando `parameter.selector` Você pode especificar para cada um `StorageClass` O pool virtual que é usado para hospedar um volume. O volume terá os aspectos definidos na piscina escolhida.

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze
allowVolumeExpansion: true
```

Exemplos de definições para volumes SMB

Usando `nasType` , `node-stage-secret-name` , e `node-stage-secret-namespace` Você pode especificar um volume SMB e fornecer as credenciais necessárias do Active Directory.

Configuração básica no namespace padrão

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Utilizando segredos diferentes por namespace

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Utilizando segredos diferentes em cada volume.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb`Filtros para pools que suportam volumes SMB. `nasType: nfs` ou `nasType: null` Filtros para pools NFS.

Crie o backend

Após criar o arquivo de configuração do backend, execute o seguinte comando:

```
tridentctl create backend -f <backend-file>
```

Se a criação do backend falhar, há algo errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs
```

Após identificar e corrigir o problema com o arquivo de configuração, você poderá executar o comando de criação novamente.

Google Cloud NetApp Volumes

Configure um backend do Google Cloud NetApp Volumes

Agora você pode configurar o Google Cloud NetApp Volumes como backend para o Trident. Você pode conectar volumes NFS e SMB usando um backend do Google Cloud NetApp Volumes .

Detalhes do driver do Google Cloud NetApp Volumes

A Trident fornece o `google-cloud-netapp-volumes` O driver deve se comunicar com o cluster. Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`google-cloud-netapp-volumes`	NFS SMB	Sistema de arquivos	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Identidade na nuvem para GKE

A identidade na nuvem permite que os pods do Kubernetes acessem recursos do Google Cloud autenticando-se como uma identidade de carga de trabalho, em vez de fornecer credenciais explícitas do Google Cloud.

Para aproveitar as vantagens da identidade na nuvem do Google Cloud, você precisa ter:

- Um cluster Kubernetes implantado usando o GKE.
- Identidade de carga de trabalho configurada no cluster GKE e servidor de metadados GKE configurado nos pools de nós.

- Uma conta de serviço do GCP com a função de administrador de Google Cloud NetApp Volumes (roles/netapp.admin) ou uma função personalizada.
- O Trident instalado inclui o cloudProvider para especificar "GCP" e o cloudIdentity especificando a nova conta de serviço do GCP. Segue abaixo um exemplo.

Operador do Trident

Para instalar o Trident usando o operador Trident , edite `tridentorchestrator_cr.yaml` para definir `cloudProvider` para `"GCP"` e definir `cloudIdentity` para `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com` .

Por exemplo:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com'
```

Leme

Defina os valores para os parâmetros **cloud-provider (CP)** e **cloud-identity (CI)** usando as seguintes variáveis de ambiente:

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

O exemplo a seguir instala o Trident e configura `cloudProvider` para o GCP usando a variável de ambiente `$CP` e define o `cloudIdentity` usando a variável de ambiente `$ANNOTATION` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>tridentctl</code>

Defina os valores para os parâmetros **provedor de nuvem** e **identidade de nuvem** usando as seguintes variáveis de ambiente:

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

O exemplo a seguir instala o Trident e configura o `cloud-provider` bandeira para `$CP` , e `cloud-identity` para `$ANNOTATION` :

```
tridentctl install --cloud-provider=$CP --cloud
-identity="$ANNOTATION" -n trident
```

Prepare-se para configurar um backend do Google Cloud NetApp Volumes.

Antes de configurar o backend do Google Cloud NetApp Volumes , você precisa garantir que os seguintes requisitos sejam atendidos.

Pré-requisitos para volumes NFS

Se você estiver usando o Google Cloud NetApp Volumes pela primeira vez ou em um novo local, será necessária alguma configuração inicial para configurar o Google Cloud NetApp Volumes e criar um volume NFS. Consulte"Antes de começar" .

Certifique-se de ter o seguinte antes de configurar o backend do Google Cloud NetApp Volumes :

- Uma conta do Google Cloud configurada com o serviço Google Cloud NetApp Volumes . Consulte"Google Cloud NetApp Volumes" .
- Número do projeto da sua conta do Google Cloud. Consulte"Identificação de projetos" .
- Uma conta de serviço do Google Cloud com o administrador de volumes da NetApp.(`roles/netapp.admin`) papel. Consulte"Funções e permissões de Gestão de Identidade e Acesso" .
- Arquivo de chave API para sua conta GCNV. Consulte"Criar uma chave de conta de serviço"
- Uma piscina de armazenamento. Consulte"Visão geral dos pools de armazenamento" .

Para obter mais informações sobre como configurar o acesso aos Google Cloud NetApp Volumes, consulte:"Configure o acesso aos Google Cloud NetApp Volumes." .

Opções e exemplos de configuração do backend do Google Cloud NetApp Volumes

Saiba mais sobre as opções de configuração de back-end para o Google Cloud NetApp Volumes e veja exemplos de configuração.

Opções de configuração do backend

Cada backend provisiona volumes em uma única região do Google Cloud. Para criar volumes em outras regiões, você pode definir backends adicionais.

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	O valor de `storageDriverName` deve ser especificado como "google-cloud-netapp-volumes".

Parâmetro	Descrição	Padrão
`backendName`	(Opcional) Nome personalizado do backend de armazenamento	Nome do driver + "_" + parte da chave da API
`storagePools`	Parâmetro opcional usado para especificar os conjuntos de armazenamento para a criação de volumes.	
`projectNumber`	Número do projeto da conta do Google Cloud. O valor pode ser encontrado na página inicial do portal do Google Cloud.	
`location`	Localização no Google Cloud onde o Trident cria volumes GCNV. Ao criar clusters Kubernetes entre regiões, os volumes criados em uma região são mantidos. `location` Pode ser usado em cargas de trabalho agendadas em nós em várias regiões do Google Cloud. O tráfego entre regiões diferentes acarreta um custo adicional.	
`apiKey`	Chave de API para a conta de serviço do Google Cloud com o `netapp.admin` papel. Inclui o conteúdo formatado em JSON do arquivo de chave privada de uma conta de serviço do Google Cloud (copiado integralmente para o arquivo de configuração do backend). O `apiKey` Deve incluir pares de chave-valor para as seguintes chaves: `type` , `project_id` , `client_email` , `client_id` , `auth_uri` , `token_uri` , `auth_provider_x509_cert_url` , e `client_x509_cert_url` .	
`nfsMountOptions`	Controle preciso das opções de montagem NFS.	"nfsvers=3"
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a esse valor.	"" (não aplicado por padrão)
`serviceLevel`	O nível de serviço de um pool de armazenamento e seus volumes. Os valores são `flex` , `standard` , `premium` , ou `extreme` .	
`labels`	Conjunto de rótulos arbitrários formatados em JSON para aplicar aos volumes	""
`network`	A rede Google Cloud é usada para os volumes GCNV.	
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo, `{"api":false, "method":true}` . Não utilize esta opção a menos que esteja solucionando problemas e precise de um despejo de logs detalhado.	nulo
`nasType`	Configure a criação de volumes NFS ou SMB. As opções são `nfs` , `smb` ou nulo. Definir como nulo utiliza, por padrão, volumes NFS.	`nfs`

Parâmetro	Descrição	Padrão

`supportedTopologies`	Representa uma lista de regiões e zonas que são suportadas por este sistema. Para obter mais informações, consulte"Utilizar a topologia CSI" . Por exemplo:
	`supportedTopologies:`
	`- topology.kubernetes.io/region: asia-east1`
	`topology.kubernetes.io/zone: asia-east1-a`	

Opções de provisionamento de volume

Você pode controlar o provisionamento de volume padrão em `defaults` seção do arquivo de configuração.

Parâmetro	Descrição	Padrão
`exportRule`	Regras de exportação para novos volumes. Deve ser uma lista de endereços IPv4, separados por vírgulas, contendo qualquer combinação de endereços IPv4.	"0.0.0.0/0"
`snapshotDir`	Acesso ao `.snapshot` diretório	"verdadeiro" para NFSv4 "falso" para NFSv3
`snapshotReserve`	Percentagem do volume reservada para instantâneos	"" (aceitar o valor padrão de 0)
`unixPermissions`	Permissões Unix de novos volumes (4 dígitos octais).	""

Configurações de exemplo

Os exemplos a seguir mostram configurações básicas que deixam a maioria dos parâmetros com os valores padrão. Esta é a maneira mais fácil de definir um backend.

Configuração mínima

Esta é a configuração mínima absoluta do backend. Com essa configuração, o Trident descobre todos os seus pools de armazenamento delegados ao Google Cloud NetApp Volumes no local configurado e coloca novos volumes em um desses pools aleatoriamente. Porque `nasType` é omitido, o `nfs` A configuração padrão será aplicada e o servidor provisionará volumes NFS.

Essa configuração é ideal para quem está começando a usar o Google Cloud NetApp Volumes e a experimentar, mas na prática, provavelmente será necessário definir um escopo adicional para os volumes provisionados.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    XsYg6gyxy4zq7OlwWgLwGa==\n
    -----END PRIVATE KEY-----\n

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuração para volumes SMB

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuração com filtro StoragePools

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuração de pool virtual

Essa configuração de backend define vários pools virtuais em um único arquivo. Os pools virtuais são definidos em `storage` seção. São úteis quando você tem vários pools de armazenamento que suportam diferentes níveis de serviço e deseja criar classes de armazenamento no Kubernetes que os representem. Etiquetas de piscinas virtuais são usadas para diferenciar as piscinas. Por exemplo, no exemplo abaixo. `performance` rótulo e `serviceLevel` O tipo é usado para diferenciar pools virtuais.

Você também pode definir alguns valores padrão que serão aplicáveis a todos os pools virtuais e sobrescrever os valores padrão para pools virtuais individuais. No exemplo a seguir, `snapshotReserve` e `exportRule` servem como valores padrão para todos os pools virtuais.

Para obter mais informações, consulte"Piscinas virtuais" .

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
```

```
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Identidade na nuvem para GKE

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

Configuração de topologias suportadas

O Trident facilita o provisionamento de volumes para cargas de trabalho com base em regiões e zonas de disponibilidade. O `supportedTopologies` O bloco nesta configuração de backend é usado para fornecer uma lista de regiões e zonas por backend. Os valores de região e zona especificados aqui devem corresponder aos valores de região e zona dos rótulos em cada nó do cluster Kubernetes. Essas regiões e zonas representam a lista de valores permitidos que podem ser fornecidos em uma classe de armazenamento. Para classes de armazenamento que contêm um subconjunto das regiões e zonas fornecidas em um backend, o Trident cria volumes na região e zona mencionadas. Para obter mais informações, consulte"Utilizar a topologia CSI" .

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

O que vem a seguir?

Após criar o arquivo de configuração do backend, execute o seguinte comando:

```
kubectl create -f <backend-file>
```

Para verificar se o backend foi criado com sucesso, execute o seguinte comando:

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE   STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound   Success
```

Se a criação do backend falhar, há algo errado com a configuração do backend. Você pode descrever o backend usando o `kubectl get tridentbackendconfig <backend-name>` Execute o comando ou visualize os registros para determinar a causa, executando o seguinte comando:

```
tridentctl logs
```

Após identificar e corrigir o problema com o arquivo de configuração, você pode excluir o backend e executar o comando de criação novamente.

Definições de classe de armazenamento

A seguir, apresentamos um básico. `StorageClass` definição que se refere ao backend acima.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

Exemplos de definições usando o `parameter.selector` campo:

Usando `parameter.selector` Você pode especificar para cada um `StorageClass` o"piscina virtual" que é usado para hospedar um volume. O volume terá os aspectos definidos na piscina escolhida.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

Para obter mais detalhes sobre classes de armazenamento, consulte"Criar uma classe de armazenamento" .

Exemplos de definições para volumes SMB

Usando `nasType` , `node-stage-secret-name` , e `node-stage-secret-namespace` Você pode especificar um volume SMB e fornecer as credenciais necessárias do Active Directory. Qualquer usuário/senha do Active Directory, com quaisquer permissões ou sem permissões, pode ser usado como segredo de estágio do nó.

Configuração básica no namespace padrão

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Utilizando segredos diferentes por namespace

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Utilizando segredos diferentes em cada volume.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb`Filtros para pools que suportam volumes SMB. `nasType: nfs` ou `nasType: null` Filtros para pools NFS.

Exemplo de definição de PVC

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

Para verificar se o PVC está vinculado, execute o seguinte comando:

```
kubectl get pvc gcnv-nfs-pvc

NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc  Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213  100Gi
RWX            gcnv-nfs-sc  1m
```

Configure um Cloud Volumes Service para o backend do Google Cloud.

Aprenda como configurar o NetApp Cloud Volumes Service para Google Cloud como backend para sua instalação do Trident usando as configurações de exemplo fornecidas.

Detalhes do driver do Google Cloud

A Trident fornece o `gcp-cvs` O driver deve se comunicar com o cluster. Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`gcp-cvs`	NFS	Sistema de arquivos	RWO, ROX, RWX, RWOP	`nfs`

Saiba mais sobre o suporte do Trident para o Cloud Volumes Service do Google Cloud.

O Trident pode criar volumes do Cloud Volumes Service de duas maneiras diferentes."tipos de serviço" :

- **CVS-Performance**: O tipo de serviço Trident padrão. Este tipo de serviço otimizado para desempenho é mais adequado para cargas de trabalho de produção que valorizam o desempenho. O tipo de serviço CVS-Performance é uma opção de hardware que suporta volumes com tamanho mínimo de 100 GiB. Você pode escolher um dos seguintes:"três níveis de serviço" :
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: O tipo de serviço CVS oferece alta disponibilidade zonal com níveis de desempenho limitados a moderados. O tipo de serviço CVS é uma opção de software que utiliza pools de armazenamento para suportar volumes tão pequenos quanto 1 GiB. O pool de armazenamento pode conter até 50 volumes, onde todos os volumes compartilham a capacidade e o desempenho do pool. Você pode escolher um dos seguintes:"dois níveis de serviço" :
  - `standardsw`
  - `zoneredundantstandardsw`

O que você vai precisar

Para configurar e usar o "Cloud Volumes Service para Google Cloud" Para o backend, você precisa do seguinte:

- Uma conta do Google Cloud configurada com o serviço NetApp Cloud Volumes Service.
- Número do projeto da sua conta do Google Cloud
- conta de serviço do Google Cloud com o `netappcloudvolumes.admin` papel
- Arquivo de chave de API para sua conta do Cloud Volumes Service .

Opções de configuração do backend

Cada backend provisiona volumes em uma única região do Google Cloud. Para criar volumes em outras regiões, você pode definir backends adicionais.

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	"gcp-cvs"
`backendName`	Nome personalizado ou o backend de armazenamento	Nome do driver + "_" + parte da chave da API
`storageClass`	Parâmetro opcional usado para especificar o tipo de serviço CVS. Usar `software` Para selecionar o tipo de serviço CVS. Caso contrário, o Trident assume o tipo de serviço CVS-Performance.(`hardware`).	
`storagePools`	Somente o tipo de serviço CVS. Parâmetro opcional usado para especificar os conjuntos de armazenamento para a criação de volumes.	
`projectNumber`	Número do projeto da conta do Google Cloud. O valor pode ser encontrado na página inicial do portal do Google Cloud.	

Parâmetro	Descrição	Padrão
`hostProjectNumber`	Obrigatório se estiver usando uma rede VPC compartilhada. Neste cenário, `projectNumber` é o projeto de serviço, e `hostProjectNumber` é o projeto anfitrião.	
`apiRegion`	A região do Google Cloud onde o Trident cria volumes do Cloud Volumes Service . Ao criar clusters Kubernetes entre regiões, os volumes criados em uma região são mantidos. `apiRegion` Pode ser usado em cargas de trabalho agendadas em nós em várias regiões do Google Cloud. O tráfego entre regiões diferentes acarreta um custo adicional.	
`apiKey`	Chave de API para a conta de serviço do Google Cloud com o `netappcloudvolumes.admin` papel. Inclui o conteúdo formatado em JSON do arquivo de chave privada de uma conta de serviço do Google Cloud (copiado integralmente para o arquivo de configuração do backend).	
`proxyURL`	URL do proxy, caso seja necessário um servidor proxy para conectar-se à conta CVS. O servidor proxy pode ser um proxy HTTP ou um proxy HTTPS. Para um proxy HTTPS, a validação do certificado é ignorada para permitir o uso de certificados autoassinados no servidor proxy. Servidores proxy com autenticação ativada não são suportados.	
`nfsMountOptions`	Controle preciso das opções de montagem NFS.	"nfsvers=3"
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a esse valor.	"" (não aplicado por padrão)
`serviceLevel`	O nível de desempenho ou serviço CVS para novos volumes. Os valores de desempenho do CVS são `standard` , `premium` , ou `extreme` . Os valores CVS são `standardsw` ou `zoneredundantstandardsw` .	A configuração padrão do CVS-Performance é "standard". O padrão do CVS é "standardsw".
`network`	A rede do Google Cloud é usada para os volumes do Cloud Volumes Service .	"padrão"
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo, `\{"api":false, "method":true}` . Não utilize esta opção a menos que esteja solucionando problemas e precise de um despejo de logs detalhado.	nulo

`allowedTopologies`	Para habilitar o acesso entre regiões, sua definição de StorageClass para `allowedTopologies` Deve incluir todas as regiões. Por exemplo:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Opções de provisionamento de volume

Você pode controlar o provisionamento de volume padrão em `defaults` seção do arquivo de configuração.

Parâmetro	Descrição	Padrão
`exportRule`	Regras de exportação para novos volumes. Deve ser uma lista separada por vírgulas de qualquer combinação de endereços IPv4 ou sub-redes IPv4 na notação CIDR.	"0.0.0.0/0"
`snapshotDir`	Acesso ao `.snapshot` diretório	"falso"
`snapshotReserve`	Percentagem do volume reservada para instantâneos	"" (aceitar o valor padrão CVS de 0)
`size`	O tamanho dos novos volumes. O requisito mínimo de desempenho do CVS é de 100 GiB. O tamanho mínimo exigido pelo CVS é 1 GiB.	O tipo de serviço CVS-Performance tem como padrão "100GiB". O tipo de serviço CVS não define um valor padrão, mas requer um mínimo de 1 GiB.

Exemplos de tipos de serviço CVS-Performance

Os exemplos a seguir fornecem configurações de amostra para o tipo de serviço CVS-Performance.

Exemplo 1: Configuração mínima

Esta é a configuração mínima de backend usando o tipo de serviço CVS-Performance padrão com o nível de serviço "standard" padrão.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: "012345678901"
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: <id_value>
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: "123456789012345678901"
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```

Exemplo 2: Configuração do nível de serviço

Este exemplo ilustra as opções de configuração do backend, incluindo o nível de serviço e os valores padrão de volume.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```

Exemplo 3: Configuração de pool virtual

Este exemplo usa `storage` para configurar pools virtuais e o `StorageClasses` que se referem a eles. ConsulteDefinições de classe de armazenamento para ver como as classes de armazenamento foram definidas.

Aqui, são definidos valores padrão específicos para todos os pools virtuais, que definem o `snapshotReserve` a 5% e o `exportRule` para 0.0.0.0/0. Os pools virtuais são definidos em `storage` seção. Cada piscina virtual individual define a sua própria. `serviceLevel` E algumas pools sobrescrevem os valores padrão. Rótulos de piscinas virtuais foram usados para diferenciar as piscinas com base em `performance` e `protection` .

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
    performance: extreme
    protection: extra
  serviceLevel: extreme
```

```
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
    performance: extreme
    protection: standard
  serviceLevel: extreme
- labels:
    performance: premium
    protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
    performance: premium
    protection: standard
  serviceLevel: premium
- labels:
    performance: standard
  serviceLevel: standard
```

Definições de classe de armazenamento

As seguintes definições de StorageClass aplicam-se ao exemplo de configuração de pool virtual. Usando `parameters.selector` Você pode especificar para cada StorageClass o pool virtual usado para hospedar um volume. O volume terá os aspectos definidos na piscina escolhida.

Exemplo de classe de armazenamento

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme; protection=extra
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=standard
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=extra
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=standard
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
```

```
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: protection=extra
allowVolumeExpansion: true
```

- A primeira StorageClass(`cvs-extreme-extra-protection`) mapeia para a primeira piscina virtual. Esta é a única piscina que oferece desempenho extremo com uma reserva instantânea de 10%.
- A última StorageClass(`cvs-extra-protection`) menciona qualquer pool de armazenamento que forneça uma reserva de snapshot de 10%. O Trident decide qual pool virtual será selecionado e garante que o requisito de reserva de snapshots seja atendido.

Exemplos de tipos de serviço CVS

Os exemplos a seguir fornecem configurações de amostra para o tipo de serviço CVS.

Exemplo 1: Configuração mínima

Esta é a configuração mínima de backend usando `storageClass` para especificar o tipo de serviço CVS e o padrão `standardsw` nível de serviço.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
serviceLevel: standardsw
```

Exemplo 2: Configuração do pool de armazenamento

Esta configuração de backend de exemplo usa `storagePools` Para configurar um pool de armazenamento.

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw
```

O que vem a seguir?

Após criar o arquivo de configuração do backend, execute o seguinte comando:

```
tridentctl create backend -f <backend-file>
```

Se a criação do backend falhar, há algo errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs
```

Após identificar e corrigir o problema com o arquivo de configuração, você poderá executar o comando de criação novamente.

Configure um backend NetApp HCI ou SolidFire.

Aprenda como criar e usar um backend Element com sua instalação do Trident .

Detalhes do driver do elemento

A Trident fornece o `solidfire-san` Driver de armazenamento para comunicação com o cluster. Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

O `solidfire-san` O driver de armazenamento suporta os modos de volume *arquivo* e *bloco*. Para o `Filesystem` Com o comando `volumeMode`, o Trident cria um volume e um sistema de arquivos. O tipo de sistema de arquivos é especificado pela StorageClass.

Motorista	Protocolo	Modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`solidfire-san`	iSCSI	Bloquear	RWO, ROX, RWX, RWOP	Sem sistema de arquivos. Dispositivo de bloco bruto.
`solidfire-san`	iSCSI	Sistema de arquivos	RWO, RWOP	`xfs`, `ext3` , `ext4`

Antes de começar

Você precisará do seguinte antes de criar um backend Element.

- Um sistema de armazenamento compatível que execute o software Element.
- Credenciais de administrador de cluster NetApp HCI/ SolidFire ou de usuário locatário que possa gerenciar volumes.
- Todos os seus nós de trabalho do Kubernetes devem ter as ferramentas iSCSI apropriadas instaladas. Consulte"Informações sobre a preparação do nó de trabalho" .

Opções de configuração do backend

Consulte a tabela a seguir para obter as opções de configuração do backend:

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	Sempre "solidfire-san"

Parâmetro	Descrição	Padrão
`backendName`	Nome personalizado ou o backend de armazenamento	"solidfire_" + endereço IP de armazenamento (iSCSI)
`Endpoint`	MVIP para o cluster SolidFire com credenciais de locatário	
`SVIP`	Endereço IP e porta de armazenamento (iSCSI)	
`labels`	Conjunto de rótulos arbitrários formatados em JSON para aplicar aos volumes.	""
`TenantName`	Nome do locatário a ser usado (criado se não for encontrado)	
`InitiatorIFace`	Restringir o tráfego iSCSI a uma interface de host específica	"padrão"
`UseCHAP`	Use CHAP para autenticar iSCSI. Trident usa CHAP.	verdadeiro
`AccessGroups`	Lista de IDs de grupos de acesso a serem usados	Encontra o ID de um grupo de acesso chamado "trident".
`Types`	Especificações de QoS	
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a este valor.	"" (não aplicado por padrão)
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo: {"api":false, "method":true}	nulo

Não use `debugTraceFlags` a menos que você esteja solucionando problemas e precise de um despejo de logs detalhado.

Exemplo 1: Configuração de backend para `solidfire-san` driver com três tipos de volume

Este exemplo mostra um arquivo de backend que utiliza autenticação CHAP e modela três tipos de volume com garantias de QoS específicas. Muito provavelmente, você definiria classes de armazenamento para consumir cada um deles usando o `IOPS` parâmetro de classe de armazenamento.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
TenantName: <tenant>
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
```

Exemplo 2: Configuração de backend e classe de armazenamento para `solidfire-san` motorista com piscinas virtuais

Este exemplo mostra o arquivo de definição de backend configurado com pools virtuais, juntamente com StorageClasses que fazem referência a eles.

O Trident copia os rótulos presentes em um pool de armazenamento para o LUN de armazenamento de backend durante o provisionamento. Para maior conveniência, os administradores de armazenamento podem definir rótulos por pool virtual e agrupar volumes por rótulo.

No arquivo de definição de backend de exemplo mostrado abaixo, valores padrão específicos são definidos para todos os pools de armazenamento, que definem o `type` em Silver. Os pools virtuais são definidos em `storage` seção. Neste exemplo, alguns dos pools de armazenamento definem seu próprio tipo, e alguns pools substituem os valores padrão definidos acima.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
```

```
SVIP: <svip>:3260
TenantName: <tenant>
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
  - labels:
      performance: gold
      cost: "4"
    zone: us-east-1a
    type: Gold
  - labels:
      performance: silver
      cost: "3"
    zone: us-east-1b
    type: Silver
  - labels:
      performance: bronze
      cost: "2"
    zone: us-east-1c
    type: Bronze
  - labels:
      performance: silver
      cost: "1"
    zone: us-east-1d
```

As definições de StorageClass a seguir referem-se aos pools virtuais acima. Usando o

`parameters.selector` No campo `StorageClass`, cada `StorageClass` especifica qual(is) pool(s) virtual(is) pode(m) ser usado(s) para hospedar um volume. O volume terá os aspectos definidos na piscina virtual escolhida.

A primeira StorageClass(`solidfire-gold-four`) será mapeado para o primeiro pool virtual. Esta é a única piscina que oferece desempenho de ouro com um `Volume Type QoS` de ouro. A última StorageClass(`solidfire-silver`) menciona qualquer pool de armazenamento que ofereça um desempenho prata. A Trident decidirá qual pool virtual será selecionado e garantirá que o requisito de armazenamento seja atendido.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold; cost=4
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=3
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze; cost=2
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=1
```

```
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
  fsType: ext4
```

Encontre mais informações

- "Grupos de acesso a volume"

Motoristas ONTAP SAN

Visão geral do driver ONTAP SAN

Aprenda a configurar um backend ONTAP com os drivers ONTAP SAN do Cloud Volumes ONTAP .

Detalhes do driver ONTAP SAN

A Trident fornece os seguintes drivers de armazenamento SAN para comunicação com o cluster ONTAP . Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`ontap-san`	iSCSI SCSI sobre FC	Bloquear	RWO, ROX, RWX, RWOP	Sem sistema de arquivos; dispositivo de bloco bruto
`ontap-san`	iSCSI SCSI sobre FC	Sistema de arquivos	RWO, RWOP ROX e RWX não estão disponíveis no modo de volume do sistema de arquivos.	`xfs`, `ext3` , `ext4`

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`ontap-san`	NVMe/TCP ConsulteConsiderações adicionais para NVMe/TCP.	Bloquear	RWO, ROX, RWX, RWOP	Sem sistema de arquivos; dispositivo de bloco bruto
`ontap-san`	NVMe/TCP ConsulteConsiderações adicionais para NVMe/TCP.	Sistema de arquivos	RWO, RWOP ROX e RWX não estão disponíveis no modo de volume do sistema de arquivos.	`xfs`, `ext3` , `ext4`
`ontap-san-economy`	iSCSI	Bloquear	RWO, ROX, RWX, RWOP	Sem sistema de arquivos; dispositivo de bloco bruto
`ontap-san-economy`	iSCSI	Sistema de arquivos	RWO, RWOP ROX e RWX não estão disponíveis no modo de volume do sistema de arquivos.	`xfs`, `ext3` , `ext4`

- Usar `ontap-san-economy` somente se a contagem de uso de volume persistente prevista for maior que"limites de volume ONTAP suportados" .
- Usar `ontap-nas-economy` somente se a contagem de uso de volume persistente prevista for maior que"limites de volume ONTAP suportados" e o `ontap-san-economy` O driver não pode ser usado.
- Não use `ontap-nas-economy` Se você prevê a necessidade de proteção de dados, recuperação de desastres ou mobilidade.
- A NetApp não recomenda o uso do Flexvol autogrow em todos os drivers ONTAP , exceto no ontap-san. Como solução alternativa, o Trident suporta o uso de reserva de snapshots e dimensiona os volumes Flexvol de acordo.

Permissões do usuário

O Trident espera ser executado como administrador ONTAP ou SVM, normalmente usando o `admin` usuário do cluster ou um `vsadmin` Usuário SVM, ou um usuário com um nome diferente que tenha a mesma função. Para implementações do Amazon FSx for NetApp ONTAP , o Trident espera ser executado como administrador do ONTAP ou do SVM, usando o cluster. `fsxadmin` usuário ou um `vsadmin` Usuário SVM, ou um usuário com um nome diferente que tenha a mesma função. O `fsxadmin` O usuário é um substituto limitado para o usuário administrador do cluster.

Se você usar o `limitAggregateUsage` Para configurar o parâmetro, são necessárias permissões de administrador do cluster. Ao usar o Amazon FSx for NetApp ONTAP com Trident, o `limitAggregateUsage` O parâmetro não funcionará com o `vsadmin` e `fsxadmin` contas de usuário. A operação de configuração falhará se você especificar este parâmetro.

Embora seja possível criar uma função mais restritiva dentro do ONTAP que um driver Trident possa usar, não recomendamos isso. A maioria das novas versões do Trident chamará APIs adicionais que precisarão ser consideradas, tornando as atualizações difíceis e propensas a erros.

Considerações adicionais para NVMe/TCP

O Trident suporta o protocolo de memória não volátil expressa (NVMe) usando o `ontap-san` motorista incluindo:

- IPv6
- Instantâneos e clones de volumes NVMe
- Redimensionar um volume NVMe
- Importar um volume NVMe criado fora do Trident para que seu ciclo de vida possa ser gerenciado pelo Trident.
- Multicaminhamento nativo NVMe
- Encerramento correto ou incorreto dos nós K8s (24.06)

O Trident não suporta:

- DH-HMAC-CHAP que é suportado nativamente por NVMe
- Mapeamento de dispositivos (DM) com múltiplos caminhos
- Criptografia LUKS

O NVMe é compatível apenas com APIs REST ONTAP e não com ONTAPI (ZAPI).

Prepare-se para configurar o backend com os drivers ONTAP SAN.

Compreenda os requisitos e as opções de autenticação para configurar um backend ONTAP com drivers ONTAP SAN.

Requisitos

Para todos os backends ONTAP , o Trident exige que pelo menos um agregado seja atribuído à SVM.

"Sistemas ASA r2"diferem de outros sistemas ONTAP (ASA, AFF e FAS) na implementação de sua camada de armazenamento. Nos sistemas ASA r2, as zonas de disponibilidade de armazenamento são usadas em vez de agregados. Consulte"esse" Artigo da Base de Conhecimento sobre como atribuir agregados a SVMs em sistemas ASA r2.

Lembre-se de que você também pode executar mais de um driver e criar classes de armazenamento que apontem para um ou outro. Por exemplo, você poderia configurar um `san-dev` classe que usa o `ontap-san` motorista e um `san-default` classe que usa o `ontap-san-economy` um.

Todos os seus nós de trabalho do Kubernetes devem ter as ferramentas iSCSI apropriadas instaladas.

Consulte "Prepare o nó de trabalho" para mais detalhes.

Autenticar o backend ONTAP

O Trident oferece dois modos de autenticação de um backend ONTAP .

- Com base em credenciais: o nome de usuário e a senha de um usuário do ONTAP com as permissões necessárias. Recomenda-se o uso de uma função de login de segurança predefinida, como `admin` ou `vsadmin` Para garantir a máxima compatibilidade com as versões do ONTAP .
- Com base em certificado: o Trident também pode se comunicar com um cluster ONTAP usando um certificado instalado no backend. Aqui, a definição do backend deve conter os valores codificados em Base64 do certificado do cliente, da chave e do certificado da CA confiável, se utilizado (recomendado).

Você pode atualizar os sistemas de backend existentes para alternar entre métodos baseados em credenciais e métodos baseados em certificados. No entanto, apenas um método de autenticação é suportado por vez. Para mudar para um método de autenticação diferente, você deve remover o método existente da configuração do backend.

> Se você tentar fornecer **tanto credenciais quanto certificados**, a criação do backend falhará com um erro informando que mais de um método de autenticação foi fornecido no arquivo de configuração.

Ativar autenticação baseada em credenciais

O Trident requer as credenciais de um administrador com escopo de SVM/cluster para se comunicar com o backend do ONTAP . Recomenda-se o uso de funções padrão predefinidas, como: `admin` ou `vsadmin` . Isso garante a compatibilidade futura com versões futuras do ONTAP que possam expor APIs de recursos a serem usadas por versões futuras do Trident . É possível criar e usar uma função de login de segurança personalizada com o Trident, mas isso não é recomendado.

Uma definição de backend de exemplo terá a seguinte aparência:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Lembre-se de que a definição do backend é o único lugar onde as credenciais são armazenadas em texto simples. Após a criação do backend, os nomes de usuário/senhas são codificados em Base64 e armazenados como segredos do Kubernetes. A criação ou atualização de um backend é a única etapa que exige conhecimento das credenciais. Sendo assim, trata-se de uma operação exclusiva para administradores, a ser realizada pelo administrador do Kubernetes/armazenamento.

Habilitar autenticação baseada em certificado

Novos e existentes sistemas de backend podem usar um certificado e se comunicar com o backend ONTAP . São necessários três parâmetros na definição do backend.

- clientCertificate: Valor do certificado do cliente codificado em Base64.
- clientPrivateKey: Valor codificado em Base64 da chave privada associada.
- trustedCACertificate: Valor codificado em Base64 do certificado da Autoridade Certificadora (CA) confiável. Caso esteja utilizando uma Autoridade Certificadora (CA) confiável, este parâmetro deve ser fornecido. Isso pode ser ignorado se nenhuma Autoridade Certificadora (CA) confiável for utilizada.

Um fluxo de trabalho típico envolve as seguintes etapas.

Passos

1. Gere um certificado e uma chave de cliente. Ao gerar o código, defina o Nome Comum (CN) para o usuário ONTAP que será usado para autenticação.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. Adicione um certificado CA confiável ao cluster ONTAP . Isso pode já estar sendo tratado pelo administrador de armazenamento. Ignore se nenhuma Autoridade Certificadora (CA) confiável for utilizada.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Instale o certificado e a chave do cliente (do passo 1) no cluster ONTAP .

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Confirme se a função de login de segurança do ONTAP é compatível. `cert` método de autenticação.

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

5. Teste a autenticação usando o certificado gerado. Substitua < ONTAP Management LIF> e <vserver name> pelo endereço IP do Management LIF e pelo nome do SVM.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifique o certificado, a chave e o certificado da CA confiável em Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Crie o backend usando os valores obtidos na etapa anterior.

```
cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"trustedCACertificate": "QNFinfO...SiqOyN",
"storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       0 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Atualize os métodos de autenticação ou altere as credenciais.

Você pode atualizar um backend existente para usar um método de autenticação diferente ou para rotacionar suas credenciais. Isso funciona nos dois sentidos: os sistemas internos que utilizam nome de usuário/senha podem ser atualizados para usar certificados; os sistemas internos que utilizam certificados podem ser atualizados para usar nome de usuário/senha. Para fazer isso, você deve remover o método de autenticação existente e adicionar o novo método de autenticação. Em seguida, utilize o arquivo backend.json atualizado, que contém os parâmetros necessários, para executar o comando. `tridentctl backend update` .

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Ao rotacionar senhas, o administrador de armazenamento deve primeiro atualizar a senha do usuário no ONTAP. Em seguida, é realizada uma atualização do sistema interno. Ao rotacionar certificados, vários certificados podem ser adicionados ao usuário. Em seguida, o sistema de backend é atualizado para usar o novo certificado, após o que o certificado antigo pode ser excluído do cluster ONTAP .

A atualização de um backend não interrompe o acesso a volumes já criados, nem afeta as conexões de volume feitas posteriormente. Uma atualização bem-sucedida do backend indica que o Trident pode se comunicar com o backend ONTAP e lidar com futuras operações em grande volume.

Criar função ONTAP personalizada para Trident

Você pode criar uma função de cluster ONTAP com privilégios mínimos para que não precise usar a função de administrador do ONTAP para executar operações no Trident. Ao incluir o nome de usuário em uma configuração de backend do Trident , o Trident usa a função de cluster ONTAP que você criou para executar as operações.

Consulte"Gerador de funções personalizadas Trident" Para obter mais informações sobre como criar funções personalizadas do Trident .

Utilizando a CLI do ONTAP

1. Crie uma nova função usando o seguinte comando:

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Crie um nome de usuário para o usuário do Trident :

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. Atribua a função ao usuário:

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilizando o Gerenciador de Sistemas

Execute as seguintes etapas no ONTAP System Manager:

1. **Criar uma função personalizada**:
   a. Para criar uma função personalizada no nível do cluster, selecione **Cluster > Configurações**.

      (Ou) Para criar uma função personalizada no nível da SVM, selecione **Armazenamento > VMs de armazenamento > `required SVM` > Configurações > Usuários e funções**.

   b. Selecione o ícone de seta (→) ao lado de **Usuários e Funções**.
   c. Selecione **+Adicionar** em **Funções**.
   d. Defina as regras para a função e clique em **Salvar**.
2. **Atribua a função ao usuário do Trident *: + Execute as seguintes etapas na página *Usuários e Funções**:
   a. Selecione o ícone Adicionar * em **Usuários**.
   b. Selecione o nome de usuário desejado e, em seguida, selecione uma função no menu suspenso **Função**.
   c. Clique em **Salvar**.

Consulte as páginas seguintes para obter mais informações:

- "Funções personalizadas para administração do ONTAP"ou"Defina funções personalizadas"
- "Trabalhar com funções e usuários"

Autenticar conexões com CHAP bidirecional

O Trident pode autenticar sessões iSCSI com CHAP bidirecional para o `ontap-san` e `ontap-san-economy` motoristas. Isso requer a ativação do `useCHAP` opção na sua definição de backend. Quando definido para `true` O Trident configura a segurança do iniciador padrão da SVM para CHAP bidirecional e define o nome de usuário e os segredos a partir do arquivo de backend. A NetApp recomenda o uso do protocolo CHAP

bidirecional para autenticar conexões. Veja a seguinte configuração de exemplo:

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

> O `useCHAP` O parâmetro é uma opção booleana que pode ser configurada apenas uma vez. Por padrão, está definido como falso. Depois de definir como verdadeiro, você não poderá definir como falso.

Além de `useCHAP=true` , o `chapInitiatorSecret` , `chapTargetInitiatorSecret` , `chapTargetUsername` , e `chapUsername` Os campos devem ser incluídos na definição do backend. Os segredos podem ser alterados após a criação de um backend executando o seguinte comando: `tridentctl update` .

Como funciona

Ao configurar `useCHAP` Para confirmar, o administrador de armazenamento instrui o Trident a configurar o CHAP no backend de armazenamento. Isso inclui o seguinte:

- Configurando o CHAP no SVM:
  - Se o tipo de segurança do iniciador padrão da SVM for "nenhum" (definido por padrão) **e** não houver LUNs preexistentes no volume, o Trident definirá o tipo de segurança padrão como `CHAP` e prossiga para a configuração do iniciador CHAP e do nome de usuário e segredos de destino.
  - Se a SVM contiver LUNs, o Trident não habilitará o CHAP na SVM. Isso garante que o acesso aos LUNs já presentes na SVM não seja restringido.
- Configurar o nome de usuário e os segredos do iniciador e do alvo CHAP; essas opções devem ser especificadas na configuração do backend (como mostrado acima).

Após a criação do backend, o Trident cria um correspondente. `tridentbackend` O CRD armazena os segredos CHAP e os nomes de usuário como segredos do Kubernetes. Todos os PVs criados pelo Trident neste backend serão montados e conectados via CHAP.

Gire as credenciais e atualize os backends

Você pode atualizar as credenciais CHAP atualizando os parâmetros CHAP em `backend.json` arquivo. Isso exigirá a atualização dos segredos CHAP e o uso do `tridentctl update` comando para refletir essas mudanças.

Ao atualizar os segredos CHAP de um backend, você deve usar `tridentctl` para atualizar o backend. Não atualize as credenciais no cluster de armazenamento usando a CLI do ONTAP ou o ONTAP System Manager, pois o Trident não conseguirá detectar essas alterações.

```
cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "password",
    "chapInitiatorSecret": "cl9qxUpDaTeD",
    "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+----------------+----------------+--------------------------------------
+--------+---------+
|   NAME         | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+----------------+----------------+--------------------------------------
+--------+---------+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |       7 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

As conexões existentes permanecerão inalteradas; elas continuarão ativas se as credenciais forem atualizadas pelo Trident no SVM. Novas conexões utilizam as credenciais atualizadas e as conexões existentes permanecem ativas. Desconectar e reconectar os sistemas fotovoltaicos antigos fará com que eles passem a usar as credenciais atualizadas.

Opções e exemplos de configuração do ONTAP SAN

Aprenda como criar e usar drivers ONTAP SAN com sua instalação do Trident . Esta seção fornece exemplos de configuração de backend e detalhes para mapear backends para StorageClasses.

"Sistemas ASA r2"Diferem de outros sistemas ONTAP (ASA, AFF e FAS) na implementação de sua camada de armazenamento. Essas variações afetam o uso de certos parâmetros, conforme indicado. "Saiba mais sobre as diferenças entre os sistemas ASA r2 e outros sistemas ONTAP.".

Somente o `ontap-san` O driver (com protocolos iSCSI e NVMe/TCP) é compatível com sistemas ASA r2.

Na configuração do backend Trident , não é necessário especificar que seu sistema é um ASA r2. Ao selecionar `ontap-san` como o `storageDriverName` O Trident detecta automaticamente o ASA r2 ou o sistema ONTAP tradicional. Alguns parâmetros de configuração de backend não se aplicam aos sistemas ASA r2, conforme indicado na tabela abaixo.

Opções de configuração do backend

Consulte a tabela a seguir para obter as opções de configuração do backend:

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	`ontap-san`ou `ontap-san-economy`
`backendName`	Nome personalizado ou o backend de armazenamento	Nome do motorista + "_" + dataLIF
`managementLIF`	Endereço IP de um cluster ou LIF de gerenciamento de SVM. É possível especificar um nome de domínio totalmente qualificado (FQDN). Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como por exemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Para uma transição perfeita para o MetroCluster , consulte oExemplo MetroCluster . Se você estiver usando as credenciais "vsadmin", `managementLIF` deve ser a do SVM; se estiver usando credenciais de "administrador", `managementLIF` deve ser o do cluster.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parâmetro	Descrição	Padrão
`dataLIF`	Endereço IP do protocolo LIF. Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como por exemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Não especifique para iSCSI.** Trident usa"Mapa LUN Seletivo ONTAP" para descobrir as LIFs iSCSI necessárias para estabelecer uma sessão de múltiplos caminhos. Um aviso é gerado se `dataLIF` está explicitamente definido. **Omitir para Metrocluster.** Veja oExemplo MetroCluster .	Derivado pelo SVM
`svm`	Máquina virtual de armazenamento a ser usada **Omitir para Metrocluster.** Veja oExemplo MetroCluster .	Derivado de uma SVM `managementLIF` é especificado
`useCHAP`	Usar CHAP para autenticar iSCSI para drivers ONTAP SAN [Booleano]. Definir para `true` Para que o Trident configure e utilize o CHAP bidirecional como autenticação padrão para a SVM fornecida no backend. Consulte "Prepare-se para configurar o backend com os drivers ONTAP SAN." para mais detalhes. **Não compatível com FCP ou NVMe/TCP.**	`false`
`chapInitiatorSecret`	Segredo do iniciador CHAP. Obrigatório se `useCHAP=true`	""
`labels`	Conjunto de rótulos arbitrários formatados em JSON para aplicar aos volumes	""
`chapTargetInitiatorSecret`	Segredo iniciador do alvo CHAP. Obrigatório se `useCHAP=true`	""
`chapUsername`	Nome de usuário de entrada. Obrigatório se `useCHAP=true`	""
`chapTargetUsername`	Nome de usuário alvo. Obrigatório se `useCHAP=true`	""
`clientCertificate`	Valor do certificado do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`clientPrivateKey`	Valor da chave privada do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`trustedCACertificate`	Valor codificado em Base64 do certificado da Autoridade Certificadora (CA) confiável. Opcional. Utilizado para autenticação baseada em certificado.	""
`username`	Nome de usuário necessário para se comunicar com o cluster ONTAP . Usado para autenticação baseada em credenciais. Para autenticação do Active Directory, consulte "Autenticar o Trident em um SVM de backend usando credenciais do Active Directory".	""

Parâmetro	Descrição	Padrão
`password`	Senha necessária para se comunicar com o cluster ONTAP . Usado para autenticação baseada em credenciais. Para autenticação do Active Directory, consulte "Autenticar o Trident em um SVM de backend usando credenciais do Active Directory".	""
`svm`	Máquina virtual de armazenamento para usar	Derivado de uma SVM `managementLIF` é especificado
`storagePrefix`	Prefixo usado ao provisionar novos volumes no SVM. Não pode ser modificado posteriormente. Para atualizar esse parâmetro, você precisará criar um novo backend.	`trident`
`aggregate`	Agregado para provisionamento (opcional; se definido, deve ser atribuído à SVM). Para o `ontap-nas-flexgroup` motorista, esta opção é ignorada. Caso não esteja atribuído, qualquer um dos agregados disponíveis pode ser usado para provisionar um volume FlexGroup . Quando o agregado é atualizado no SVM, ele é atualizado automaticamente no Trident por meio de polling no SVM, sem a necessidade de reiniciar o Controlador Trident . Quando você configura um agregado específico no Trident para provisionar volumes, se o agregado for renomeado ou movido para fora do SVM, o backend entrará em estado de falha no Trident durante a consulta ao agregado do SVM. Você deve alterar o agregado para um que esteja presente na SVM ou removê-lo completamente para que o backend volte a ficar online. **Não especificar para sistemas ASA r2**.	""
`limitAggregateUsage`	O provisionamento falhará se a utilização for superior a esta percentagem. Se você estiver usando um backend Amazon FSx for NetApp ONTAP , não especifique. `limitAggregateUsage` . O fornecido `fsxadmin` e `vsadmin` Não possuem as permissões necessárias para recuperar o uso agregado e limitá-lo usando o Trident. **Não especificar para sistemas ASA r2**.	"" (não aplicado por padrão)
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a este valor. Também restringe o tamanho máximo dos volumes que gerencia para LUNs.	"" (não aplicado por padrão)

Parâmetro	Descrição	Padrão
`lunsPerFlexvol`	Número máximo de LUNs por Flexvol, deve estar no intervalo [50, 200]	`100`
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo: {"api":false, "method":true} Não utilize a menos que esteja solucionando problemas e precise de um despejo de log detalhado.	`null`
`useREST`	Parâmetro booleano para usar APIs REST do ONTAP . `useREST`Quando definido para `true` O Trident usa APIs REST do ONTAP para se comunicar com o backend; quando configurado para `false` O Trident utiliza chamadas ONTAPI (ZAPI) para se comunicar com o backend. Este recurso requer o ONTAP 9.11.1 e posterior. Além disso, a função de login do ONTAP utilizada deve ter acesso ao `ontapi` aplicativo. Isso é satisfeito pelo predefinido `vsadmin` e `cluster-admin` papéis. A partir da versão Trident 24.06 e do ONTAP 9.15.1 ou posterior, `useREST` está definido para `true` por padrão; alterar `useREST` para `false` para usar chamadas ONTAPI (ZAPI). `useREST`Está totalmente qualificado para NVMe/TCP. O NVMe é compatível apenas com APIs REST ONTAP e não com ONTAPI (ZAPI). **Se especificado, sempre defina como `true` para sistemas ASA r2**.	`true`para ONTAP 9.15.1 ou posterior, caso contrário `false`.
`sanType`	Use para selecionar `iscsi` para iSCSI, `nvme` para NVMe/TCP ou `fcp` para SCSI sobre Fibre Channel (FC).	`iscsi`se estiver em branco

Parâmetro	Descrição	Padrão
`formatOptions`	Usar `formatOptions` para especificar argumentos de linha de comando para o `mkfs` comando, que será aplicado sempre que um volume for formatado. Isso permite formatar o volume de acordo com suas preferências. Certifique-se de especificar as opções de formatação semelhantes às opções do comando mkfs, excluindo o caminho do dispositivo. Exemplo: "-E nodiscard" **Compatível com `ontap-san` e `ontap-san-economy` drivers com protocolo iSCSI. Além disso, é compatível com sistemas ASA r2 ao usar os protocolos iSCSI e NVMe/TCP.**	
`limitVolumePoolSize`	Tamanho máximo de FlexVol solicitável ao usar LUNs no backend ontap-san-economy.	"" (não aplicado por padrão)
`denyNewVolumePools`	Restringe `ontap-san-economy` backends da criação de novos volumes FlexVol para conter seus LUNs. Apenas os Flexvols preexistentes são usados para provisionar novos PVs.	

Recomendações para usar formatOptions

A Trident recomenda a seguinte opção para agilizar o processo de formatação:

-E nodiscard:

- Mantenha os blocos salvos e não tente descartá-los durante a criação do sistema de arquivos (o descarte inicial de blocos é útil em dispositivos de estado sólido e em armazenamento com provisionamento esparso/dinâmico). Esta opção substitui a opção obsoleta "-K" e é aplicável a todos os sistemas de arquivos (xfs, ext3 e ext4).

Autenticar o Trident em um SVM de backend usando credenciais do Active Directory

Você pode configurar o Trident para autenticar em um SVM de backend usando credenciais do Active Directory (AD). Antes que uma conta do AD possa acessar o SVM, você deve configurar o acesso do controlador de domínio do AD ao cluster ou SVM. Para administração de cluster com uma conta do AD, você deve criar um túnel de domínio. Consulte "Configurar o acesso do controlador de domínio do Active Directory no ONTAP" para mais detalhes.

passos

1. Configurar as definições do Sistema de Nomes de Domínio (DNS) para um SVM de backend:

```
vserver services dns create -vserver <svm_name> -dns-servers
<dns_server_ip1>,<dns_server_ip2>
```

2. Execute o seguinte comando para criar uma conta de computador para o SVM no Active Directory:

```
vserver active-directory create -vserver DataSVM -account-name ADSERVER1
-domain demo.netapp.com
```

3. Use este comando para criar um usuário ou grupo do AD para gerenciar o cluster ou SVM

```
security login create -vserver <svm_name> -user-or-group-name
<ad_user_or_group> -application <application> -authentication-method domain
-role vsadmin
```

4. No arquivo de configuração do backend do Trident , defina o `username` e `password` parâmetros para o nome do usuário ou grupo do AD e senha, respectivamente.

Opções de configuração de backend para provisionamento de volumes

Você pode controlar o provisionamento padrão usando essas opções em `defaults` seção da configuração. Para ver um exemplo, consulte os exemplos de configuração abaixo.

Parâmetro	Descrição	Padrão
`spaceAllocation`	Alocação de espaço para LUNs	"verdadeiro" **Se especificado, defina como `true` para sistemas ASA r2**.
`spaceReserve`	Modo de reserva de espaço: "nenhum" (fino) ou "volume" (grosso). **Definir para `none` para sistemas ASA r2**.	"nenhum"
`snapshotPolicy`	Política de instantâneo a ser utilizada. **Definir para `none` para sistemas ASA r2**.	"nenhum"
`qosPolicy`	Grupo de políticas de QoS a ser atribuído aos volumes criados. Escolha uma das opções qosPolicy ou adaptiveQosPolicy para cada pool de armazenamento/backend. A utilização de grupos de políticas de QoS com o Trident requer o ONTAP 9.8 ou posterior. Você deve usar um grupo de políticas de QoS não compartilhado e garantir que o grupo de políticas seja aplicado a cada componente individualmente. Um grupo de políticas de QoS compartilhado impõe o limite máximo para a taxa de transferência total de todas as cargas de trabalho.	""
`adaptiveQosPolicy`	Grupo de políticas de QoS adaptativas a serem atribuídas aos volumes criados. Escolha qosPolicy ou adaptiveQosPolicy por pool de armazenamento/backend.	""
`snapshotReserve`	Percentagem do volume reservada para instantâneos. **Não especificar para sistemas ASA r2**.	"0" se `snapshotPolicy` é "nenhum", caso contrário ""
`splitOnClone`	Separar um clone de seu progenitor no momento da criação.	"falso"
`encryption`	Ative a Criptografia de Volume NetApp (NVE) no novo volume; o padrão é `false` . Para usar esta opção, o NVE precisa estar licenciado e habilitado no cluster. Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE" .	"falso" **Se especificado, defina como `true` para sistemas ASA r2**.

Parâmetro	Descrição	Padrão
`luksEncryption`	Ative a criptografia LUKS. Consulte"Use o Linux Unified Key Setup (LUKS)" .	"" **Definido para `false` para sistemas ASA r2**.
`tieringPolicy`	Política de escalonamento para usar "nenhum" **Não especificar para sistemas ASA r2**.	
`nameTemplate`	Modelo para criar nomes de volume personalizados.	""

Exemplos de provisionamento em volume

Aqui está um exemplo com valores padrão definidos:

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

> Para todos os volumes criados usando o `ontap-san` O driver Trident adiciona 10% a mais de capacidade ao FlexVol para acomodar os metadados do LUN. O LUN será provisionado com o tamanho exato que o usuário solicitar no PVC. O Trident adiciona 10% ao FlexVol (mostrado como tamanho disponível no ONTAP). Os usuários agora receberão a quantidade de capacidade utilizável que solicitaram. Essa alteração também impede que as LUNs se tornem somente leitura, a menos que o espaço disponível esteja totalmente utilizado. Isso não se aplica a ontap-san-economy.

Para back-ends que definem `snapshotReserve` O Trident calcula o tamanho dos volumes da seguinte forma:

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

O 1.1 representa os 10% extras que a Trident adiciona ao FlexVol para acomodar os metadados do LUN Para `snapshotReserve` = 5%, e solicitação de PVC = 5 GiB, o tamanho total do volume é 5,79 GiB e o tamanho disponível é 5,5 GiB. O `volume show` O comando deve exibir resultados semelhantes a este exemplo:

Atualmente, o redimensionamento é a única maneira de usar o novo cálculo para um volume existente.

Exemplos de configuração mínima

Os exemplos a seguir mostram configurações básicas que deixam a maioria dos parâmetros com os valores padrão. Esta é a maneira mais fácil de definir um backend.

> Se você estiver usando o Amazon FSx no NetApp ONTAP com Trident, a NetApp recomenda que você especifique nomes DNS para LIFs em vez de endereços IP.

Exemplo de SAN ONTAP

Esta é uma configuração básica usando o `ontap-san` motorista.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

Exemplo MetroCluster

Você pode configurar o backend para evitar a necessidade de atualizar manualmente a definição do backend após a troca de modo (switchover) e o retorno ao modo anterior (switchback)."Replicação e recuperação de SVM" .

Para uma transição perfeita e um retorno perfeito, especifique a SVM usando `managementLIF` e omitir o `svm` parâmetros. Por exemplo:

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Exemplo de economia ONTAP SAN

```
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

Exemplo de autenticação baseada em certificado

Neste exemplo de configuração básica `clientCertificate` , `clientPrivateKey` , e `trustedCACertificate` (opcional, se estiver usando uma CA confiável) são preenchidos em `backend.json` e extraem os valores codificados em base64 do certificado do cliente, da chave privada e do certificado da CA confiável, respectivamente.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Exemplos CHAP bidirecionais

Esses exemplos criam um backend com `useCHAP` definido para `true` .

Exemplo ONTAP SAN CHAP

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

Exemplo de economia ONTAP SAN CHAP

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

Exemplo NVMe/TCP

Você precisa ter uma SVM configurada com NVMe no seu backend ONTAP . Esta é uma configuração básica de backend para NVMe/TCP.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

Exemplo de SCSI sobre FC (FCP)

Você precisa ter uma SVM configurada com FC em seu backend ONTAP . Esta é uma configuração básica de backend para FC.

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

Exemplo de configuração de backend com nameTemplate

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Exemplo de formatOptions para o driver ontap-san-economy

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

Exemplos de backends com pools virtuais

Nesses arquivos de definição de backend de exemplo, valores padrão específicos são definidos para todos os pools de armazenamento, como: `spaceReserve` em nenhum, `spaceAllocation` em falso, e `encryption` falso. Os pools virtuais são definidos na seção de armazenamento.

O Trident define os rótulos de provisionamento no campo "Comentários". Os comentários são definidos no FlexVol volume. O Trident copia todos os rótulos presentes em um pool virtual para o volume de armazenamento durante o provisionamento. Para maior conveniência, os administradores de armazenamento podem definir rótulos por pool virtual e agrupar volumes por rótulo.

Nesses exemplos, alguns dos pools de armazenamento definem seus próprios limites. `spaceReserve` , `spaceAllocation` , e `encryption` valores, e alguns pools substituem os valores padrão.

Exemplo de SAN ONTAP

```yaml
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "40000"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
      adaptiveQosPolicy: adaptive-extreme
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
      qosPolicy: premium
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

Exemplo de economia ONTAP SAN

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
labels:
  store: san_economy_store
region: us_east_1
storage:
  - labels:
      app: oracledb
      cost: "30"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
  - labels:
      app: postgresdb
      cost: "20"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
  - labels:
      app: mysqldb
      cost: "10"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1c
```

```
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

Exemplo NVMe/TCP

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: "false"
  encryption: "true"
storage:
  - labels:
      app: testApp
      cost: "20"
    defaults:
      spaceAllocation: "false"
      encryption: "false"
```

Mapear backends para StorageClasses

As seguintes definições de StorageClass referem-se a:Exemplos de backends com pools virtuais . Usando o `parameters.selector` No campo `StorageClass`, cada `StorageClass` especifica quais pools virtuais podem ser usados para hospedar um volume. O volume terá os aspectos definidos na piscina virtual escolhida.

- O `protection-gold` A StorageClass será mapeada para o primeiro pool virtual no `ontap-san` backend. Esta é a única piscina que oferece proteção de nível ouro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- O `protection-not-gold` A StorageClass será mapeada para o segundo e terceiro pool virtual em `ontap-san` backend. Essas são as únicas pools que oferecem um nível de proteção diferente do ouro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- O `app-mysqldb` A StorageClass será mapeada para o terceiro pool virtual em `ontap-san-economy` backend. Este é o único pool que oferece configuração de pool de armazenamento para aplicativos do tipo mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- O `protection-silver-creditpoints-20k` A StorageClass será mapeada para o segundo pool virtual em `ontap-san` backend. Este é o único pool que oferece proteção de nível prata e 20.000 pontos de crédito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- O `creditpoints-5k` A StorageClass será mapeada para o terceiro pool virtual em `ontap-san` backend e o quarto pool virtual no `ontap-san-economy` backend. Essas são as únicas ofertas de piscina com 5000 pontos de crédito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

- O `my-test-app-sc` A classe de armazenamento será mapeada para o `testAPP` piscina virtual no `ontap-san` motorista com `sanType: nvme` . Esta é a única piscina que oferece `testApp` .

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-test-app-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=testApp"
  fsType: "ext4"
```

A Trident decidirá qual pool virtual será selecionado e garantirá que o requisito de armazenamento seja atendido.

Drivers ONTAP NAS

Visão geral do driver ONTAP NAS

Aprenda a configurar um backend ONTAP com os drivers ONTAP NAS do ONTAP e do Cloud Volumes ONTAP .

Detalhes do driver ONTAP NAS

A Trident fornece os seguintes drivers de armazenamento NAS para comunicação com o cluster ONTAP . Os modos de acesso suportados são: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Motorista	Protocolo	modo de volume	Modos de acesso suportados	Sistemas de arquivos suportados
`ontap-nas`	NFS SMB	Sistema de arquivos	RWO, ROX, RWX, RWOP	"," `nfs` , `smb`
`ontap-nas-economy`	NFS SMB	Sistema de arquivos	RWO, ROX, RWX, RWOP	"," `nfs` , `smb`
`ontap-nas-flexgroup`	NFS SMB	Sistema de arquivos	RWO, ROX, RWX, RWOP	"," `nfs` , `smb`

- Usar `ontap-san-economy` somente se a contagem de uso de volume persistente prevista for maior que"limites de volume ONTAP suportados" .
- Usar `ontap-nas-economy` somente se a contagem de uso de volume persistente prevista for maior que"limites de volume ONTAP suportados" e o `ontap-san-economy` O driver não pode ser usado.
- Não use `ontap-nas-economy` Se você prevê a necessidade de proteção de dados, recuperação de desastres ou mobilidade.
- A NetApp não recomenda o uso do Flexvol autogrow em todos os drivers ONTAP , exceto no ontap-san. Como solução alternativa, o Trident suporta o uso de reserva de snapshots e dimensiona os volumes Flexvol de acordo.

Permissões do usuário

O Trident espera ser executado como administrador ONTAP ou SVM, normalmente usando o `admin` usuário do cluster ou um `vsadmin` Usuário SVM, ou um usuário com um nome diferente que tenha a mesma função.

Para implementações do Amazon FSx for NetApp ONTAP , o Trident espera ser executado como administrador do ONTAP ou do SVM, usando o cluster. `fsxadmin` usuário ou um `vsadmin` Usuário SVM, ou um usuário com um nome diferente que tenha a mesma função. O `fsxadmin` O usuário é um substituto limitado para o usuário administrador do cluster.

Se você usar o `limitAggregateUsage` Para configurar o parâmetro, são necessárias permissões de administrador do cluster. Ao usar o Amazon FSx for NetApp ONTAP com Trident, o `limitAggregateUsage` O parâmetro não funcionará com o `vsadmin` e `fsxadmin` contas de usuário. A operação de configuração falhará se você especificar este parâmetro.

Embora seja possível criar uma função mais restritiva dentro do ONTAP que um driver Trident possa usar, não recomendamos isso. A maioria das novas versões do Trident chamará APIs adicionais que precisarão ser consideradas, tornando as atualizações difíceis e propensas a erros.

Prepare-se para configurar um backend com drivers ONTAP NAS.

Compreenda os requisitos, as opções de autenticação e as políticas de exportação para configurar um backend ONTAP com drivers ONTAP NAS.

Requisitos

- Para todos os backends ONTAP , o Trident exige que pelo menos um agregado seja atribuído à SVM.
- Você pode executar mais de um driver e criar classes de armazenamento que apontem para um ou outro. Por exemplo, você poderia configurar uma classe Gold que usa o `ontap-nas` motorista e uma classe Bronze que usa o `ontap-nas-economy` um.
- Todos os seus nós de trabalho do Kubernetes devem ter as ferramentas NFS apropriadas instaladas. Consulte"aqui" para mais detalhes.
- O Trident suporta volumes SMB montados em pods executados apenas em nós Windows. Consulte Prepare-se para provisionar volumes SMB para mais detalhes.

Autenticar o backend ONTAP

O Trident oferece dois modos de autenticação de um backend ONTAP .

- Baseado em credenciais: Este modo requer permissões suficientes no backend do ONTAP . Recomenda-se usar uma conta associada a uma função de login de segurança predefinida, como: `admin` ou `vsadmin` Para garantir a máxima compatibilidade com as versões do ONTAP .
- Baseado em certificado: Este modo requer um certificado instalado no servidor para que o Trident se comunique com um cluster ONTAP . Aqui, a definição do backend deve conter os valores codificados em Base64 do certificado do cliente, da chave e do certificado da CA confiável, se utilizado (recomendado).

Você pode atualizar os sistemas de backend existentes para alternar entre métodos baseados em credenciais e métodos baseados em certificados. No entanto, apenas um método de autenticação é suportado por vez. Para mudar para um método de autenticação diferente, você deve remover o método existente da configuração do backend.

> Se você tentar fornecer **tanto credenciais quanto certificados**, a criação do backend falhará com um erro informando que mais de um método de autenticação foi fornecido no arquivo de configuração.

Ativar autenticação baseada em credenciais

O Trident requer as credenciais de um administrador com escopo de SVM/cluster para se comunicar com o backend do ONTAP . Recomenda-se o uso de funções padrão predefinidas, como: `admin` ou `vsadmin` . Isso garante a compatibilidade futura com versões futuras do ONTAP que possam expor APIs de recursos a serem usadas por versões futuras do Trident . É possível criar e usar uma função de login de segurança personalizada com o Trident, mas isso não é recomendado.

Uma definição de backend de exemplo terá a seguinte aparência:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

Lembre-se de que a definição do backend é o único lugar onde as credenciais são armazenadas em texto simples. Após a criação do backend, os nomes de usuário/senhas são codificados em Base64 e armazenados como segredos do Kubernetes. A criação/atualização de um backend é a única etapa que exige conhecimento das credenciais. Sendo assim, trata-se de uma operação exclusiva para administradores, a ser realizada pelo administrador do Kubernetes/armazenamento.

Habilitar autenticação baseada em certificado

Novos e existentes sistemas de backend podem usar um certificado e se comunicar com o backend ONTAP . São necessários três parâmetros na definição do backend.

- clientCertificate: Valor do certificado do cliente codificado em Base64.
- clientPrivateKey: Valor codificado em Base64 da chave privada associada.
- trustedCACertificate: Valor codificado em Base64 do certificado da Autoridade Certificadora (CA) confiável. Caso esteja utilizando uma Autoridade Certificadora (CA) confiável, este parâmetro deve ser fornecido. Isso pode ser ignorado se nenhuma Autoridade Certificadora (CA) confiável for utilizada.

Um fluxo de trabalho típico envolve as seguintes etapas.

Passos

1. Gere um certificado e uma chave de cliente. Ao gerar o código, defina o Nome Comum (CN) para o usuário ONTAP que será usado para autenticação.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Adicione um certificado CA confiável ao cluster ONTAP . Isso pode já estar sendo tratado pelo administrador de armazenamento. Ignore se nenhuma Autoridade Certificadora (CA) confiável for utilizada.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Instale o certificado e a chave do cliente (do passo 1) no cluster ONTAP .

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Confirme se a função de login de segurança do ONTAP é compatível. `cert` método de autenticação.

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

5. Teste a autenticação usando o certificado gerado. Substitua < ONTAP Management LIF> e <vserver name> pelo endereço IP do Management LIF e pelo nome do SVM. Você deve garantir que a política de serviço do LIF esteja definida como `default-data-management` .

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifique o certificado, a chave e o certificado da CA confiável em Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Crie o backend usando os valores obtidos na etapa anterior.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Atualize os métodos de autenticação ou altere as credenciais.

Você pode atualizar um backend existente para usar um método de autenticação diferente ou para rotacionar suas credenciais. Isso funciona nos dois sentidos: os sistemas internos que utilizam nome de usuário/senha podem ser atualizados para usar certificados; os sistemas internos que utilizam certificados podem ser atualizados para usar nome de usuário/senha. Para isso, você deve remover o método de autenticação existente e adicionar o novo método de autenticação. Em seguida, utilize o arquivo backend.json atualizado, que contém os parâmetros necessários, para executar o comando. `tridentctl update backend`.

```
cat cert-backend-updated.json
```

```
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}
```

```
#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Ao rotacionar senhas, o administrador de armazenamento deve primeiro atualizar a senha do usuário no ONTAP. Em seguida, é realizada uma atualização do sistema interno. Ao rotacionar certificados, vários certificados podem ser adicionados ao usuário. Em seguida, o sistema de backend é atualizado para usar o novo certificado, após o que o certificado antigo pode ser excluído do cluster ONTAP .

A atualização de um backend não interrompe o acesso a volumes já criados, nem afeta as conexões de volume feitas posteriormente. Uma atualização bem-sucedida do backend indica que o Trident pode se comunicar com o backend ONTAP e lidar com futuras operações em grande volume.

Criar função ONTAP personalizada para Trident

Você pode criar uma função de cluster ONTAP com privilégios mínimos para que não precise usar a função de administrador do ONTAP para executar operações no Trident. Ao incluir o nome de usuário em uma configuração de backend do Trident , o Trident usa a função de cluster ONTAP que você criou para executar as operações.

Consulte"Gerador de funções personalizadas Trident" Para obter mais informações sobre como criar funções personalizadas do Trident .

Utilizando a CLI do ONTAP

1. Crie uma nova função usando o seguinte comando:

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Crie um nome de usuário para o usuário do Trident :

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Atribua a função ao usuário:

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilizando o Gerenciador de Sistemas

Execute as seguintes etapas no ONTAP System Manager:

1. **Criar uma função personalizada**:
   a. Para criar uma função personalizada no nível do cluster, selecione **Cluster > Configurações**.

      (Ou) Para criar uma função personalizada no nível da SVM, selecione **Armazenamento > VMs de armazenamento > `required SVM` > Configurações > Usuários e funções**.

   b. Selecione o ícone de seta (→) ao lado de **Usuários e Funções**.
   c. Selecione **+Adicionar** em **Funções**.
   d. Defina as regras para a função e clique em **Salvar**.
2. **Atribua a função ao usuário do Trident *: + Execute as seguintes etapas na página *Usuários e Funções**:
   a. Selecione o ícone Adicionar * em **Usuários**.
   b. Selecione o nome de usuário desejado e, em seguida, selecione uma função no menu suspenso **Função**.
   c. Clique em **Salvar**.

Consulte as páginas seguintes para obter mais informações:

- "Funções personalizadas para administração do ONTAP"ou"Defina funções personalizadas"
- "Trabalhar com funções e usuários"

Gerenciar políticas de exportação NFS

O Trident utiliza políticas de exportação NFS para controlar o acesso aos volumes que provisiona.

A Trident oferece duas opções para trabalhar com políticas de exportação:

- O Trident pode gerenciar dinamicamente a própria política de exportação; nesse modo de operação, o administrador de armazenamento especifica uma lista de blocos CIDR que representam endereços IP admissíveis. O Trident adiciona automaticamente à política de exportação, no momento da publicação, os endereços IP dos nós aplicáveis que se enquadram nesses intervalos. Alternativamente, quando nenhum CIDR for especificado, todos os IPs unicast de escopo global encontrados no nó para o qual o volume está sendo publicado serão adicionados à política de exportação.
- Os administradores de armazenamento podem criar uma política de exportação e adicionar regras manualmente. O Trident utiliza a política de exportação padrão, a menos que um nome de política de exportação diferente seja especificado na configuração.

Gerenciar políticas de exportação dinamicamente

O Trident oferece a capacidade de gerenciar dinamicamente as políticas de exportação para backends ONTAP . Isso permite ao administrador de armazenamento especificar um espaço de endereços permitido para os IPs dos nós de trabalho, em vez de definir regras explícitas manualmente. Isso simplifica bastante a gestão da política de exportação; as alterações na política de exportação não exigem mais intervenção manual no cluster de armazenamento. Além disso, isso ajuda a restringir o acesso ao cluster de armazenamento apenas aos nós de trabalho que estão montando volumes e possuem endereços IP no intervalo especificado, permitindo um gerenciamento preciso e automatizado.

> Não utilize Network Address Translation (NAT) ao usar políticas de exportação dinâmicas. Com NAT, o controlador de armazenamento vê o endereço NAT de front-end e não o endereço IP real do host; portanto, o acesso será negado quando nenhuma correspondência for encontrada nas regras de exportação.

Exemplo

Existem duas opções de configuração que devem ser utilizadas. Aqui está um exemplo de definição de backend:

```
---
version: 1
storageDriverName: ontap-nas-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true
```

> Ao utilizar este recurso, você deve garantir que a junção raiz em sua SVM tenha uma política de exportação previamente criada com uma regra de exportação que permita o bloco CIDR do nó (como a política de exportação padrão). Siga sempre as melhores práticas recomendadas pela NetApp para dedicar uma SVM ao Trident.

Segue abaixo uma explicação de como essa funcionalidade opera, utilizando o exemplo acima:

- `autoExportPolicy`está definido para `true`. Isso indica que o Trident cria uma política de exportação para cada volume provisionado com esse backend para o `svm1` SVM e lidar com a adição e exclusão de regras usando `autoexportCIDRs` blocos de endereço. Até que um volume seja anexado a um nó, ele utiliza uma política de exportação vazia, sem regras para impedir o acesso indesejado a esse volume. Quando um volume é publicado em um nó, o Trident cria uma política de exportação com o mesmo nome da qtree subjacente, contendo o endereço IP do nó dentro do bloco CIDR especificado. Esses IPs também serão adicionados à política de exportação usada pelo FlexVol volume pai.
  - Por exemplo:
    - UUID do backend 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy`definido para `true`
    - prefixo de armazenamento `trident`
    - UUID do PVC a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - qtree chamado trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c cria uma política de exportação para o FlexVol chamado `trident-403b5326-8482-40db96d0-d83fb3f4daec`, uma política de exportação para a qtree chamada `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` e uma política de exportação vazia chamada `trident_empty` na SVM. As regras para a política de exportação FlexVol serão um superconjunto de quaisquer regras contidas nas políticas de exportação qtree. A política de exportação vazia será reutilizada por quaisquer volumes que não estejam anexados.
- `autoExportCIDRs`Contém uma lista de blocos de endereços. Este campo é opcional e o valor padrão é ["0.0.0.0/0", "::/0"]. Caso não esteja definido, o Trident adiciona todos os endereços unicast de escopo global encontrados nos nós de trabalho com publicações.

Neste exemplo, o `192.168.0.0/24` O espaço de endereçamento é fornecido. Isso indica que os endereços IP dos nós do Kubernetes que se enquadram nesse intervalo de endereços com publicações serão adicionados à política de exportação criada Trident . Quando o Trident registra um nó no qual está sendo executado, ele recupera os endereços IP do nó e os verifica em relação aos blocos de endereços fornecidos em `autoExportCIDRs` No momento da publicação, após filtrar os IPs, o Trident cria as regras de política de exportação para os IPs do cliente para o nó no qual está publicando.

Você pode atualizar `autoExportPolicy` e `autoExportCIDRs` para os backends depois de criá-los. Você pode adicionar novos CIDRs para um backend que é gerenciado automaticamente ou excluir CIDRs existentes. Tenha cuidado ao excluir CIDRs para garantir que as conexões existentes não sejam interrompidas. Você também pode optar por desativar. `autoExportPolicy` para um backend e recorrer a uma política de exportação criada manualmente como alternativa. Isso exigirá a configuração do `exportPolicy` parâmetro na sua configuração de backend.

Após o Trident criar ou atualizar um backend, você pode verificar o backend usando `tridentctl` ou o correspondente `tridentbackend` CRD:

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

Quando um nó é removido, o Trident verifica todas as políticas de exportação para remover as regras de acesso correspondentes ao nó. Ao remover o endereço IP deste nó das políticas de exportação dos backends gerenciados, o Trident impede montagens não autorizadas, a menos que este endereço IP seja reutilizado por um novo nó no cluster.

Para backends já existentes, atualize o backend com `tridentctl update backend` Garante que o Trident gerencie as políticas de exportação automaticamente. Isso cria duas novas políticas de exportação com os nomes do UUID e da árvore de consulta (qtree) do backend, quando necessárias. Os volumes presentes no servidor usarão as políticas de exportação recém-criadas após serem desmontados e montados novamente.

> Excluir um backend com políticas de exportação gerenciadas automaticamente excluirá a política de exportação criada dinamicamente. Se o backend for recriado, ele será tratado como um novo backend e resultará na criação de uma nova política de exportação.

Se o endereço IP de um nó ativo for atualizado, você deverá reiniciar o pod do Trident nesse nó. A Trident atualizará então a política de exportação dos servidores que gerencia para refletir essa alteração de IP.

Prepare-se para provisionar volumes SMB

Com um pouco de preparação adicional, você pode provisionar volumes SMB usando `ontap-nas` motoristas.

> Você deve configurar os protocolos NFS e SMB/CIFS na SVM para criar um `ontap-nas-economy` Volume SMB para clusters ONTAP locais. A falha na configuração de qualquer um desses protocolos fará com que a criação do volume SMB falhe.

> `autoExportPolicy`Não é compatível com volumes SMB.

Antes de começar

Antes de poder provisionar volumes SMB, você precisa ter o seguinte.

- Um cluster Kubernetes com um nó controlador Linux e pelo menos um nó de trabalho Windows executando o Windows Server 2022. O Trident suporta volumes SMB montados em pods executados apenas em nós Windows.
- Pelo menos um segredo Trident contendo suas credenciais do Active Directory. Para gerar segredos `smbcreds` :

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Um proxy CSI configurado como um serviço do Windows. Para configurar um `csi-proxy` , consulte"GitHub: Proxy CSI" ou"GitHub: CSI Proxy para Windows" para nós do Kubernetes executados no Windows.

Passos

1. Para o ONTAP local, você pode opcionalmente criar um compartilhamento SMB ou a Trident pode criar um para você.

   Os compartilhamentos SMB são necessários para o Amazon FSx para ONTAP.

   Você pode criar os compartilhamentos administrativos SMB de duas maneiras: usando o"Console de gerenciamento da Microsoft" Acesse as Pastas Compartilhadas pelo snap-in ou usando a CLI do ONTAP . Para criar compartilhamentos SMB usando a CLI do ONTAP :

   a. Se necessário, crie a estrutura de diretórios para o compartilhamento.

      O `vserver cifs share create` O comando verifica o caminho especificado na opção -path durante a criação do compartilhamento. Se o caminho especificado não existir, o comando falhará.

   b. Crie um compartilhamento SMB associado à SVM especificada:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

   c. Verifique se o compartilhamento foi criado:

```
vserver cifs share show -share-name share_name
```

      Consulte"Criar um compartilhamento SMB" Para obter detalhes completos.

2. Ao criar o backend, você deve configurar o seguinte para especificar os volumes SMB. Para todas as opções de configuração do backend FSx para ONTAP , consulte"Opções e exemplos de configuração do FSx para ONTAP" .

Parâmetro	Descrição	Exemplo
`smbShare`	Você pode especificar uma das seguintes opções: o nome de um compartilhamento SMB criado usando o Console de Gerenciamento da Microsoft ou a CLI do ONTAP ; um nome para permitir que o Trident crie o compartilhamento SMB; ou você pode deixar o parâmetro em branco para impedir o acesso comum aos volumes compartilhados. Este parâmetro é opcional para o ONTAP local. Este parâmetro é obrigatório para backends do Amazon FSx para ONTAP e não pode estar em branco.	`smb-share`
`nasType`	**Deve ser configurado para `smb` .** Se for nulo, o valor padrão é `nfs` .	`smb`
`securityStyle`	Estilo de segurança para novos volumes. **Deve ser configurado para `ntfs` ou `mixed` para volumes SMB.**	`ntfs`ou `mixed` para volumes SMB
`unixPermissions`	Modo para novos volumes. **Deve ficar vazio para volumes SMB.**	""

Ativar SMB seguro

A partir da versão 25.06, o NetApp Trident oferece suporte ao provisionamento seguro de volumes SMB criados usando `ontap-nas` e `ontap-nas-economy` back-ends. Quando o SMB seguro está habilitado, você pode fornecer acesso controlado aos compartilhamentos SMB para usuários e grupos de usuários do Active Directory (AD) usando Listas de Controle de Acesso (ACLs).

Pontos a serem lembrados

- Importando `ontap-nas-economy` O volume não é suportado.
- Somente clones somente leitura são suportados para `ontap-nas-economy` volumes.
- Se o SMB seguro estiver ativado, o Trident ignorará o compartilhamento SMB mencionado no backend.
- A atualização da anotação PVC, da anotação da classe de armazenamento e do campo de backend não atualiza a ACL de compartilhamento SMB.
- A ACL de compartilhamento SMB especificada na anotação do PVC clonado terá precedência sobre as do PVC de origem.
- Certifique-se de fornecer usuários válidos do Active Directory ao habilitar o SMB seguro. Usuários inválidos não serão adicionados à ACL.
- Se você fornecer o mesmo usuário do Active Directory no backend, na classe de armazenamento e no PVC com permissões diferentes, a prioridade de permissão será: PVC, classe de armazenamento e, por último, backend.
- O Secure SMB é compatível com `ontap-nas` Importações de volume gerenciadas e não aplicáveis a importações de volume não gerenciadas.

Passos

1. Especifique o usuário adAdminUser no TridentBackendConfig conforme mostrado no exemplo a seguir:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Adicione a anotação na classe de armazenamento.

   Adicione o `trident.netapp.io/smbShareAdUser` Anotação na classe de armazenamento para habilitar o SMB seguro sem falhas. O valor do usuário especificado para a anotação `trident.netapp.io/smbShareAdUser` deve ser o mesmo que o nome de usuário especificado no `smbcreds` segredo. Você pode escolher uma das seguintes opções para `smbShareAdUserPermission` : `full_control` , `change` , ou `read` . A permissão padrão é `full_control` .

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. Criar um PVC.

   O exemplo a seguir cria um PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Opções e exemplos de configuração do ONTAP NAS

Aprenda a criar e usar drivers ONTAP NAS com sua instalação do Trident . Esta seção fornece exemplos de configuração de backend e detalhes para mapear backends para StorageClasses.

Opções de configuração do backend

Consulte a tabela a seguir para obter as opções de configuração do backend:

Parâmetro	Descrição	Padrão
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	`ontap-nas`, `ontap-nas-economy` , ou `ontap-nas-flexgroup`
`backendName`	Nome personalizado ou o backend de armazenamento	Nome do motorista + "_" + dataLIF
`managementLIF`	Endereço IP de um cluster ou LIF de gerenciamento de SVM. Um nome de domínio totalmente qualificado (FQDN) pode ser especificado. Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como por exemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Para uma transição perfeita para o MetroCluster , consulte oExemplo MetroCluster .	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parâmetro	Descrição	Padrão
`dataLIF`	Endereço IP do protocolo LIF. A NetApp recomenda especificar `dataLIF` . Caso não sejam fornecidos, o Trident obtém os dataLIFs da SVM. Você pode especificar um nome de domínio totalmente qualificado (FQDN) para ser usado nas operações de montagem NFS, permitindo criar um DNS round-robin para balancear a carga entre várias dataLIFs. Pode ser alterado após a configuração inicial. Consulte . Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como por exemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Omitir para Metrocluster.** Veja oExemplo MetroCluster .	Endereço especificado ou derivado de SVM, caso não seja especificado (não recomendado).
`svm`	Máquina virtual de armazenamento a ser usada **Omitir para Metrocluster.** Veja oExemplo MetroCluster .	Derivado de uma SVM `managementLIF` é especificado
`autoExportPolicy`	Ativar a criação e atualização automática da política de exportação [Booleano]. Usando o `autoExportPolicy` e `autoExportCIDRs` O Trident pode gerenciar políticas de exportação automaticamente, dependendo das opções disponíveis.	falso
`autoExportCIDRs`	Lista de CIDRs para filtrar os IPs dos nós do Kubernetes quando `autoExportPolicy` está habilitado. Usando o `autoExportPolicy` e `autoExportCIDRs` O Trident pode gerenciar políticas de exportação automaticamente, dependendo das opções disponíveis.	["0.0.0.0/0", "::/0"]`
`labels`	Conjunto de rótulos arbitrários formatados em JSON para aplicar aos volumes	""
`clientCertificate`	Valor do certificado do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`clientPrivateKey`	Valor da chave privada do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`trustedCACertificate`	Valor codificado em Base64 do certificado da Autoridade Certificadora (CA) confiável. Opcional. Utilizado para autenticação baseada em certificado.	""
`username`	Nome de usuário para conectar ao cluster/SVM. Usado para autenticação baseada em credenciais. Para autenticação do Active Directory, consulte "Autenticar o Trident em um SVM de backend usando credenciais do Active Directory".	

Parâmetro	Descrição	Padrão
`password`	Senha para conectar ao cluster/SVM. Usado para autenticação baseada em credenciais. Para autenticação do Active Directory, consulte "Autenticar o Trident em um SVM de backend usando credenciais do Active Directory".	
`storagePrefix`	Prefixo usado ao provisionar novos volumes no SVM. Não pode ser atualizado depois de configurado. Ao usar o ontap-nas-economy e um storagePrefix com 24 ou mais caracteres, as qtrees não terão o prefixo de armazenamento incorporado, embora ele esteja presente no nome do volume.	"tridente"
`aggregate`	Agregado para provisionamento (opcional; se definido, deve ser atribuído à SVM). Para o `ontap-nas-flexgroup` motorista, esta opção é ignorada. Caso não esteja atribuído, qualquer um dos agregados disponíveis pode ser usado para provisionar um volume FlexGroup . Quando o agregado é atualizado no SVM, ele é atualizado automaticamente no Trident por meio de polling no SVM, sem a necessidade de reiniciar o Controlador Trident . Quando você configura um agregado específico no Trident para provisionar volumes, se o agregado for renomeado ou movido para fora do SVM, o backend entrará em estado de falha no Trident durante a consulta ao agregado do SVM. Você deve alterar o agregado para um que esteja presente na SVM ou removê-lo completamente para que o backend volte a ficar online.	""
`limitAggregateUsage`	O provisionamento falhará se a utilização for superior a esta percentagem. **Não se aplica ao Amazon FSx para ONTAP**.	"" (não aplicado por padrão)

Parâmetro	Descrição	Padrão
flexgroupAggregateList	Lista de agregados para provisionamento (opcional; se definida, deve ser atribuída à SVM). Todos os agregados atribuídos à SVM são usados para provisionar um volume FlexGroup . Compatível com o driver de armazenamento **ontap-nas-flexgroup**. Quando a lista agregada é atualizada no SVM, a lista é atualizada automaticamente no Trident por meio de polling no SVM, sem a necessidade de reiniciar o Controlador Trident . Quando você configura uma lista de agregação específica no Trident para provisionar volumes, se a lista de agregação for renomeada ou movida para fora do SVM, o backend entrará em estado de falha no Trident durante a consulta da lista de agregação do SVM. Você deve alterar a lista agregada para uma que esteja presente na SVM ou removê-la completamente para que o backend volte a funcionar.	""
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a este valor. Também restringe o tamanho máximo dos volumes que gerencia para qtrees, e o `qtreesPerFlexvol` Essa opção permite personalizar o número máximo de qtrees por FlexVol volume.	"" (não aplicado por padrão)
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo: {"api":false, "method":true} Não use `debugTraceFlags` a menos que você esteja solucionando problemas e precise de um despejo de logs detalhado.	nulo
`nasType`	Configure a criação de volumes NFS ou SMB. As opções são `nfs` , `smb` ou nulo. Definir como nulo utiliza, por padrão, volumes NFS.	`nfs`

Parâmetro	Descrição	Padrão
`nfsMountOptions`	Lista de opções de montagem NFS separadas por vírgulas. As opções de montagem para volumes persistentes do Kubernetes são normalmente especificadas nas classes de armazenamento, mas se nenhuma opção de montagem for especificada em uma classe de armazenamento, o Trident usará as opções de montagem especificadas no arquivo de configuração do backend de armazenamento. Se nenhuma opção de montagem for especificada na classe de armazenamento ou no arquivo de configuração, o Trident não definirá nenhuma opção de montagem em um volume persistente associado.	""
`qtreesPerFlexvol`	Número máximo de Qtrees por FlexVol, deve estar no intervalo [50, 300]	"200"
`smbShare`	Você pode especificar uma das seguintes opções: o nome de um compartilhamento SMB criado usando o Console de Gerenciamento da Microsoft ou a CLI do ONTAP ; um nome para permitir que o Trident crie o compartilhamento SMB; ou você pode deixar o parâmetro em branco para impedir o acesso comum aos volumes compartilhados. Este parâmetro é opcional para o ONTAP local. Este parâmetro é obrigatório para backends do Amazon FSx para ONTAP e não pode estar em branco.	`smb-share`
`useREST`	Parâmetro booleano para usar APIs REST do ONTAP . `useREST` `Quando definido para `true` O Trident usa APIs REST do ONTAP para se comunicar com o backend; quando configurado para `false` O Trident utiliza chamadas ONTAPI (ZAPI) para se comunicar com o backend. Este recurso requer o ONTAP 9.11.1 e posterior. Além disso, a função de login do ONTAP utilizada deve ter acesso ao `ontapi` aplicativo. Isso é satisfeito pelo predefinido `vsadmin` e `cluster-admin` papéis. A partir da versão Trident 24.06 e do ONTAP 9.15.1 ou posterior, `useREST` está definido para `true` por padrão; alterar `useREST` para `false` para usar chamadas ONTAPI (ZAPI).	`true` `para ONTAP 9.15.1 ou posterior, caso contrário` `false` .
`limitVolumePoolSize`	Tamanho máximo de FlexVol solicitável ao usar Qtrees no backend ontap-nas-economy.	"" (não aplicado por padrão)
`denyNewVolumePools`	Restringe `ontap-nas-economy` backends da criação de novos volumes FlexVol para conter suas Qtrees. Apenas os Flexvols preexistentes são usados para provisionar novos PVs.	
`adAdminUser`	Usuário ou grupo de usuários administradores do Active Directory com acesso total aos compartilhamentos SMB. Use este parâmetro para conceder direitos de administrador ao compartilhamento SMB com controle total.	

Opções de configuração de backend para provisionamento de volumes

Você pode controlar o provisionamento padrão usando essas opções em `defaults` seção da configuração. Para ver um exemplo, consulte os exemplos de configuração abaixo.

Parâmetro	Descrição	Padrão
`spaceAllocation`	Alocação de espaço para Qtrees	"verdadeiro"
`spaceReserve`	Modo de reserva de espaço; "nenhum" (fino) ou "volume" (grosso)	"nenhum"
`snapshotPolicy`	Política de instantâneo a ser usada	"nenhum"
`qosPolicy`	Grupo de políticas de QoS a ser atribuído aos volumes criados. Escolha qosPolicy ou adaptiveQosPolicy por pool de armazenamento/backend.	""
`adaptiveQosPolicy`	Grupo de políticas de QoS adaptativas a serem atribuídas aos volumes criados. Escolha uma das opções qosPolicy ou adaptiveQosPolicy para cada pool de armazenamento/backend. Não suportado por ontap-nas-economy.	""
`snapshotReserve`	Percentagem do volume reservada para instantâneos	"0" se `snapshotPolicy` é "nenhum", caso contrário ""
`splitOnClone`	Separar um clone de seu progenitor no momento da criação.	"falso"
`encryption`	Ative a Criptografia de Volume NetApp (NVE) no novo volume; o padrão é `false` . Para usar esta opção, o NVE precisa estar licenciado e habilitado no cluster. Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE" .	"falso"
`tieringPolicy`	Política de níveis para usar "nenhum"	
`unixPermissions`	Modo para novos volumes	"777" para volumes NFS; vazio (não aplicável) para volumes SMB.
`snapshotDir`	Controla o acesso ao `.snapshot` diretório	"verdadeiro" para NFSv4 "falso" para NFSv3
`exportPolicy`	Política de exportação a ser utilizada	"padrão"
`securityStyle`	Estilo de segurança para novos volumes. Suporte a NFS `mixed` e `unix` estilos de segurança. Suporte para PMEs `mixed` e `ntfs` estilos de segurança.	O padrão do NFS é `unix` . O padrão SMB é `ntfs` .
`nameTemplate`	Modelo para criar nomes de volume personalizados.	""

A utilização de grupos de políticas de QoS com o Trident requer o ONTAP 9.8 ou posterior. Você deve usar um grupo de políticas de QoS não compartilhado e garantir que o grupo de políticas seja aplicado a cada componente individualmente. Um grupo de políticas de QoS compartilhado impõe o limite máximo para a taxa de transferência total de todas as cargas de trabalho.

Exemplos de provisionamento em volume

Aqui está um exemplo com valores padrão definidos:

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: "10"
```

Para `ontap-nas` e `ontap-nas-flexgroups` O Trident agora usa um novo cálculo para garantir que o FlexVol seja dimensionado corretamente com a porcentagem snapshotReserve e o PVC. Quando o usuário solicita um PVC, o Trident cria o FlexVol original com mais espaço usando o novo cálculo. Esse cálculo garante que o usuário receba o espaço gravável solicitado no PVC, e não menos espaço do que o solicitado. Antes da versão 21.07, quando o usuário solicita um PVC (por exemplo, 5 GiB), com o snapshotReserve em 50%, ele obtém apenas 2,5 GiB de espaço gravável. Isso ocorre porque o que o usuário solicitou foi o volume completo e `snapshotReserve` é uma porcentagem disso. Com o Trident 21.07, o que o usuário solicita é o espaço gravável e o Trident define o `snapshotReserve` número como porcentagem do volume total. Isso não se aplica a `ontap-nas-economy` . Veja o exemplo a seguir para ver como isso funciona:

O cálculo é o seguinte:

```
Total volume size = (PVC requested size) / (1 - (snapshotReserve
percentage) / 100)
```

Para snapshotReserve = 50% e solicitação de PVC = 5 GiB, o tamanho total do volume é 5/.5 = 10 GiB e o tamanho disponível é 5 GiB, que é o que o usuário solicitou na solicitação de PVC O `volume show` O comando deve exibir resultados semelhantes a este exemplo:

Os backends existentes de instalações anteriores provisionarão volumes conforme explicado acima ao atualizar o Trident. Para volumes criados antes da atualização, você deve redimensioná-los para que a alteração seja observada. Por exemplo, um PVC de 2 GiB com `snapshotReserve=50` O resultado anterior era um volume que fornecia 1 GiB de espaço gravável. Redimensionar o volume para 3 GiB, por exemplo, fornece ao aplicativo 3 GiB de espaço gravável em um volume de 6 GiB.

Exemplos de configuração mínima

Os exemplos a seguir mostram configurações básicas que deixam a maioria dos parâmetros com os valores padrão. Esta é a maneira mais fácil de definir um backend.

> Se você estiver usando o Amazon FSx no NetApp ONTAP com Trident, a recomendação é especificar nomes DNS para LIFs em vez de endereços IP.

Exemplo de economia ONTAP NAS

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Exemplo de grupo flexível ONTAP NAS

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Exemplo MetroCluster

Você pode configurar o backend para evitar a necessidade de atualizar manualmente a definição do backend após a troca de modo (switchover) e o retorno ao modo anterior (switchback)."Replicação e recuperação de SVM" .

Para uma transição perfeita e um retorno perfeito, especifique a SVM usando `managementLIF` e omitir o `dataLIF` e `svm` parâmetros. Por exemplo:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Exemplo de volumes SMB

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Exemplo de autenticação baseada em certificado

Este é um exemplo mínimo de configuração de backend. `clientCertificate` , `clientPrivateKey` , e `trustedCACertificate` (opcional, se estiver usando uma CA confiável) são preenchidos em `backend.json` e extraem os valores codificados em base64 do certificado do cliente, da chave privada e do certificado da CA confiável, respectivamente.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Exemplo de política de exportação automática

Este exemplo mostra como você pode instruir o Trident a usar políticas de exportação dinâmicas para criar e gerenciar a política de exportação automaticamente. Isso funciona da mesma forma para o `ontap-nas-economy` e `ontap-nas-flexgroup` motoristas.

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

Exemplo de endereço IPv6

Este exemplo mostra `managementLIF` usando um endereço IPv6.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

Exemplo de uso de volumes SMB no Amazon FSx para ONTAP

O `smbShare` Este parâmetro é necessário para o FSx para ONTAP usando volumes SMB.

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Exemplo de configuração de backend com nameTemplate

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Exemplos de backends com pools virtuais

Nos arquivos de definição de backend de exemplo mostrados abaixo, valores padrão específicos são definidos para todos os pools de armazenamento, como: `spaceReserve` em nenhum, `spaceAllocation` em falso, e `encryption` falso. Os pools virtuais são definidos na seção de armazenamento.

O Trident define os rótulos de provisionamento no campo "Comentários". Os comentários estão definidos no FlexVol para `ontap-nas` ou FlexGroup para `ontap-nas-flexgroup` . O Trident copia todos os rótulos presentes em um pool virtual para o volume de armazenamento durante o provisionamento. Para maior conveniência, os administradores de armazenamento podem definir rótulos por pool virtual e agrupar volumes por rótulo.

Nesses exemplos, alguns dos pools de armazenamento definem seus próprios limites. `spaceReserve` , `spaceAllocation` , e `encryption` valores, e alguns pools substituem os valores padrão.

Exemplo de ONTAP NAS

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: "false"
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      app: msoffice
      cost: "100"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
      adaptiveQosPolicy: adaptive-premium
  - labels:
      app: slack
      cost: "75"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      app: wordpress
```

```
      cost: "50"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      app: mysqldb
      cost: "25"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

Exemplo de FlexGroup NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "50000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: gold
      creditpoints: "30000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      protection: bronze
      creditpoints: "10000"
    zone: us_east_1d
    defaults:
```

```
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

Exemplo de economia ONTAP NAS

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: nas_economy_store
region: us_east_1
storage:
  - labels:
      department: finance
      creditpoints: "6000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: engineering
      creditpoints: "3000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      department: humanresource
      creditpoints: "2000"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
```

```
      encryption: "false"
      unixPermissions: "0775"
```

Mapear backends para StorageClasses

As seguintes definições de StorageClass referem-se aExemplos de backends com pools virtuais . Usando o `parameters.selector` No campo `StorageClass`, cada `StorageClass` especifica quais pools virtuais podem ser usados para hospedar um volume. O volume terá os aspectos definidos na piscina virtual escolhida.

- O `protection-gold` A StorageClass será mapeada para o primeiro e o segundo pool virtual no `ontap-nas-flexgroup` backend. Essas são as únicas piscinas que oferecem proteção de nível ouro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- O `protection-not-gold` A StorageClass será mapeada para o terceiro e quarto pool virtual no `ontap-nas-flexgroup` backend. Essas são as únicas pools que oferecem um nível de proteção diferente do ouro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- O `app-mysqldb` A classe de armazenamento será mapeada para o quarto pool virtual no `ontap-nas` backend. Este é o único pool que oferece configuração de pool de armazenamento para aplicativos do tipo mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- O `protection-silver-creditpoints-20k` A StorageClass será mapeada para o terceiro pool virtual no `ontap-nas-flexgroup` backend. Este é o único pool que oferece proteção de nível prata e 20.000 pontos de crédito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- O `creditpoints-5k` A StorageClass será mapeada para o terceiro pool virtual no `ontap-nas` backend e o segundo pool virtual no `ontap-nas-economy` backend. Essas são as únicas ofertas de piscina com 5000 pontos de crédito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

A Trident decidirá qual pool virtual será selecionado e garantirá que o requisito de armazenamento seja atendido.

Atualizar `dataLIF` após a configuração inicial

Você pode alterar o dataLIF após a configuração inicial executando o seguinte comando para fornecer ao novo arquivo JSON de backend o dataLIF atualizado.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```

Se os PVCs estiverem conectados a um ou mais pods, você deve desligar todos os pods correspondentes e, em seguida, ligá-los novamente para que o novo dataLIF entre em vigor.

Exemplos de segurança SMB

Configuração de backend com o driver ontap-nas

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuração de backend com o driver ontap-nas-economy

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuração de backend com pool de armazenamento

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Exemplo de classe de armazenamento com o driver ontap-nas

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Certifique-se de adicionar `annotations` Para habilitar o SMB seguro. O SMB seguro não funciona sem as anotações, independentemente das configurações definidas no Backend ou no PVC.

Exemplo de classe de armazenamento com o driver ontap-nas-economy

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Exemplo de PVC com um único usuário AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Exemplo de PVC com vários usuários de AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Amazon FSx for NetApp ONTAP

Use o Trident com o Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"É um serviço AWS totalmente gerenciado que permite aos clientes iniciar e executar sistemas de arquivos com tecnologia do sistema operacional de armazenamento NetApp ONTAP . O FSx para ONTAP permite que você aproveite os recursos, o desempenho e as capacidades administrativas da NetApp com os quais você já está familiarizado, ao mesmo tempo que desfruta da simplicidade, agilidade, segurança e escalabilidade do armazenamento de dados na AWS. O FSx para ONTAP oferece suporte aos recursos do sistema de arquivos ONTAP e às APIs de administração.

Você pode integrar seu sistema de arquivos Amazon FSx for NetApp ONTAP com o Trident para garantir que os clusters Kubernetes em execução no Amazon Elastic Kubernetes Service (EKS) possam provisionar volumes persistentes de bloco e arquivo com suporte do ONTAP.

Um sistema de arquivos é o recurso principal no Amazon FSx, análogo a um cluster ONTAP em um ambiente

local. Dentro de cada SVM, você pode criar um ou vários volumes, que são contêineres de dados que armazenam os arquivos e pastas do seu sistema de arquivos. Com o Amazon FSx for NetApp ONTAP, este será fornecido como um sistema de arquivos gerenciado na nuvem. O novo tipo de sistema de arquivos é chamado * NetApp ONTAP*.

Ao usar o Trident com o Amazon FSx for NetApp ONTAP, você pode garantir que os clusters Kubernetes em execução no Amazon Elastic Kubernetes Service (EKS) possam provisionar volumes persistentes em bloco e em arquivo com suporte do ONTAP.

Requisitos

Além de"Requisitos do Trident" Para integrar o FSx para ONTAP com o Trident, você precisa de:

- Um cluster Amazon EKS existente ou um cluster Kubernetes autogerenciado com `kubectl` instalado.
- Um sistema de arquivos e uma máquina virtual de armazenamento (SVM) Amazon FSx for NetApp ONTAP existentes e acessíveis a partir dos nós de trabalho do seu cluster.
- Nós de trabalho que estão preparados para"NFS ou iSCSI" .

> Certifique-se de seguir os passos de preparação do nó necessários para o Amazon Linux e o Ubuntu. "Imagens de máquinas da Amazon" (AMIs) dependendo do seu tipo de AMI EKS.

Considerações

- Volumes SMB:
  - Os volumes SMB são suportados usando o `ontap-nas` Somente o motorista.
  - Volumes SMB não são suportados com o complemento Trident EKS.
  - O Trident suporta volumes SMB montados em pods executados apenas em nós Windows. Consulte "Prepare-se para provisionar volumes SMB" para mais detalhes.
- Antes da versão 24.02 do Trident , os volumes criados em sistemas de arquivos Amazon FSx com backups automáticos ativados não podiam ser excluídos pelo Trident. Para evitar esse problema no Trident 24.02 ou posterior, especifique o `fsxFilesystemID` , AWS `apiRegion` , AWS `apikey` e AWS `secretKey` no arquivo de configuração de backend para AWS FSx para ONTAP.

> Se você estiver especificando uma função do IAM para o Trident, poderá omitir a especificação do… `apiRegion` , `apiKey` , e `secretKey` campos para Trident explicitamente. Para obter mais informações, consulte"Opções e exemplos de configuração do FSx para ONTAP" .

Uso simultâneo do driver Trident SAN/iSCSI e EBS-CSI

Se você planeja usar drivers ontap-san (por exemplo, iSCSI) com AWS (EKS, ROSA, EC2 ou qualquer outra instância), a configuração multipath necessária nos nós pode entrar em conflito com o driver CSI do Amazon Elastic Block Store (EBS). Para garantir que o multipathing funcione sem interferir nos discos EBS no mesmo nó, você precisa excluir o EBS da sua configuração de multipathing. Este exemplo mostra um `multipath.conf` Arquivo que inclui as configurações Trident necessárias, excluindo os discos EBS do multipathing:

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

Autenticação

O Trident oferece dois modos de autenticação.

- Baseado em credenciais (recomendado): Armazena as credenciais com segurança no AWS Secrets Manager. Você pode usar o `fsxadmin` usuário para o seu sistema de arquivos ou o `vsadmin` Usuário configurou sua SVM.

> A Trident espera ser administrada como uma `vsadmin` Usuário SVM ou como um usuário com um nome diferente que tenha a mesma função. O Amazon FSx for NetApp ONTAP possui um `fsxadmin` usuário que é um substituto limitado do ONTAP `admin` usuário do cluster. Recomendamos vivamente a utilização de `vsadmin` com Trident.

- Com base em certificado: o Trident se comunicará com a SVM no seu sistema de arquivos FSx usando um certificado instalado na sua SVM.

Para obter detalhes sobre como ativar a autenticação, consulte a documentação de autenticação para o seu tipo de driver:

- "Autenticação ONTAP NAS"
- "Autenticação ONTAP SAN"

Imagens de Máquina da Amazon (AMIs) testadas

O cluster EKS suporta diversos sistemas operacionais, mas a AWS otimizou determinadas Amazon Machine Images (AMIs) para contêineres e EKS. As seguintes AMIs foram testadas com o NetApp Trident 25.02.

AMI	NAS	NAS-economia	iSCSI	iSCSI-economia
AL2023_x86_64_STANDARD	Sim	Sim	Sim	Sim
AL2_x86_64	Sim	Sim	Sim*	Sim*
BOTTLEROCKET_x86_64	Sim**	Sim	N / D	N / D
AL2023_ARM_64_STANDARD	Sim	Sim	Sim	Sim
AL2_ARM_64	Sim	Sim	Sim*	Sim*

BOTTLEROCKET_ARM_64	Sim**	Sim	N / D	N / D

- * Não é possível excluir o PV sem reiniciar o nó
- ** Não funciona com NFSv3 com Trident versão 25.02.

> Se a AMI desejada não estiver listada aqui, isso não significa que ela não seja compatível; significa simplesmente que ela não foi testada. Esta lista serve como um guia para AMIs que comprovadamente funcionam.

Testes realizados com:

- Versão EKS: 1.32
- Método de instalação: Helm 25.06 e como um complemento AWS 25.06
- Para NAS, foram testados tanto o NFSv3 quanto o NFSv4.1.
- Para SAN, apenas o iSCSI foi testado, não o NVMe-oF.

Testes realizados:

- Criar: Classe de armazenamento, PVC, cápsula
- Excluir: pod, pvc (regular, qtree/lun – econômico, NAS com backup AWS)

Encontre mais informações

- "Documentação do Amazon FSx for NetApp ONTAP"
- "Postagem no blog sobre Amazon FSx for NetApp ONTAP"

Crie uma função do IAM e um segredo da AWS.

Você pode configurar pods do Kubernetes para acessar recursos da AWS autenticando-se como uma função do AWS IAM em vez de fornecer credenciais explícitas da AWS.

> Para autenticar usando uma função do AWS IAM, você precisa ter um cluster Kubernetes implantado usando o EKS.

Criar segredo do AWS Secrets Manager

Como o Trident emitirá APIs contra um servidor virtual FSx para gerenciar o armazenamento para você, ele precisará de credenciais para fazer isso. A forma segura de transmitir essas credenciais é por meio de um segredo do AWS Secrets Manager. Portanto, se você ainda não possui uma, precisará criar um segredo do AWS Secrets Manager que contenha as credenciais da conta vsadmin.

Este exemplo cria um segredo do AWS Secrets Manager para armazenar credenciais do Trident CSI:

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\":\"vsadmin\",\"password\":\"<svmpassword>\"}"
```

Criar política de IAM

O Trident também precisa de permissões da AWS para funcionar corretamente. Portanto, você precisa criar uma política que conceda ao Trident as permissões necessárias.

Os exemplos a seguir criam uma política do IAM usando a AWS CLI:

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

Exemplo de JSON de política:

```
{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx:DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}
```

Criar identidade de Pod ou função IAM para associação de conta de serviço (IRSA)

Você pode configurar uma conta de serviço do Kubernetes para assumir uma função do AWS Identity and Access Management (IAM) com a Identidade do Pod do EKS ou a função do IAM para associação de conta de serviço (IRSA). Qualquer Pod configurado para usar a conta de serviço poderá acessar qualquer serviço

da AWS para o qual a função tenha permissões de acesso.

Identidade do Pod

As associações de identidade de pods do Amazon EKS permitem gerenciar credenciais para seus aplicativos, de forma semelhante à maneira como os perfis de instância do Amazon EC2 fornecem credenciais para instâncias do Amazon EC2.

Instale o Pod Identity no seu cluster EKS:

Você pode criar uma identidade de Pod através do console da AWS ou usando o seguinte comando da AWS CLI:

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

Para obter mais informações, consulte"Configure o agente de identidade do Amazon EKS Pod." .

Criar trust-relationship.json:

Crie o arquivo trust-relationship.json para permitir que o Service Principal do EKS assuma essa função para a identidade do Pod. Em seguida, crie uma função com esta política de confiança:

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

Arquivo trust-relationship.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Anexe a política de função à função do IAM:

Anexe a política de função da etapa anterior à função IAM que foi criada:

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \
  --role-name fsxn-csi-role
```

Criar uma associação de identidade de pod:

Crie uma associação de identidade de pod entre a função IAM e a conta de serviço Trident (trident-controller).

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

Função IAM para associação de conta de serviço (IRSA)

Utilizando a AWS CLI:

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \
  --assume-role-policy-document file://trust-relationship.json
```

Arquivo trust-relationship.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

Atualize os seguintes valores em `trust-relationship.json` arquivo:

- **<account_id>** - Seu ID de conta da AWS
- **<oidc_provider>** - O OIDC do seu cluster EKS. Você pode obter o provedor oidc executando o seguinte comando:

```
aws eks describe-cluster --name my-cluster --query
"cluster.identity.oidc.issuer"\
  --output text | sed -e "s/^https:\/\///"
```

Associe a função IAM à política IAM:

Após a criação da função, associe a política (criada na etapa anterior) à função usando este comando:

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

Verifique se o provedor do OICD está associado:

Verifique se o seu provedor OIDC está associado ao seu cluster. Você pode verificar isso usando este comando:

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Se a saída estiver vazia, use o seguinte comando para associar o IAM OIDC ao seu cluster:

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

Se você estiver usando o eksctl, utilize o exemplo a seguir para criar uma função IAM para a conta de serviço no EKS:

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Instalar Trident

O Trident simplifica o gerenciamento de armazenamento do Amazon FSx for NetApp ONTAP no Kubernetes, permitindo que seus desenvolvedores e administradores se concentrem na implantação de aplicativos.

Você pode instalar o Trident usando um dos seguintes métodos:

- Leme
- Complemento EKS

Se você deseja utilizar a funcionalidade de instantâneo, instale o complemento CSI snapshot controller. Consulte"Ative a funcionalidade de instantâneo para volumes CSI." para mais informações.

Instale o Trident via Helm.

Identidade do Pod

1. Adicione o repositório Trident Helm:

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

2. Instale o Trident usando o seguinte exemplo:

```
helm install trident-operator netapp-trident/trident-operator --version 100.2502.1 --namespace trident --create-namespace
```

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

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                           APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2502.0    25.02.0
```

Associação de conta de serviço (IRSA)

1. Adicione o repositório Trident Helm:

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

2. Defina os valores para **provedor de nuvem** e **identidade de nuvem**:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

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

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2506.0    25.06.0
```

Se você pretende usar iSCSI, certifique-se de que o iSCSI esteja habilitado em sua máquina cliente. Se você estiver usando o sistema operacional de nó de trabalho AL2023, poderá automatizar a instalação do cliente iSCSI adicionando o parâmetro `node prep` na instalação do Helm:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

Instale o Trident através do complemento EKS.

O complemento Trident EKS inclui os patches de segurança e correções de bugs mais recentes, e é validado pela AWS para funcionar com o Amazon EKS. O complemento EKS permite garantir de forma consistente que seus clusters Amazon EKS estejam seguros e estáveis, além de reduzir o trabalho necessário para instalar, configurar e atualizar complementos.

Pré-requisitos

Certifique-se de ter o seguinte antes de configurar o complemento Trident para AWS EKS:

- Uma conta de cluster Amazon EKS com assinatura adicional.
- Permissões da AWS para o marketplace da AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo de AMI: Amazon Linux 2 (AL2_x86_64) ou Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo de nó: AMD ou ARM
- Um sistema de arquivos Amazon FSx for NetApp ONTAP

Habilite o complemento Trident para AWS.

Console de gerenciamento

1. Abra o console do Amazon EKS em https://console.aws.amazon.com/eks/home#/clusters .
2. No painel de navegação à esquerda, selecione **Clusters**.
3. Selecione o nome do cluster para o qual deseja configurar o complemento NetApp Trident CSI.
4. Selecione **Complementos** e depois selecione **Obter mais complementos**.
5. Siga estes passos para selecionar o complemento:
   a. Desça a página até a seção **Complementos do AWS Marketplace** e digite **"Trident"** na caixa de pesquisa.
   b. Selecione a caixa de seleção no canto superior direito da caixa Trident by NetApp .
   c. Selecione **Avançar**.
6. Na página de configurações **Configurar complementos selecionados**, faça o seguinte:

   Ignore estas etapas se estiver usando a associação de identidade do Pod.

   a. Selecione a **Versão** que deseja usar.
   b. Se você estiver usando a autenticação IRSA, certifique-se de definir os valores de configuração disponíveis nas Configurações opcionais:
      - Selecione a **Versão** que deseja usar.
      - Siga o **esquema de configuração do complemento** e defina o parâmetro **configurationValues** na seção **Valores de configuração** com o role-arn que você criou na etapa anterior (o valor deve estar no seguinte formato):

```
{

  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
  "cloudProvider": "AWS"

}
```

   +
   Se você selecionar "Substituir" como método de resolução de conflitos, uma ou mais configurações do complemento existente poderão ser substituídas pelas configurações do complemento do Amazon EKS. Se você não ativar essa opção e houver um conflito com suas configurações existentes, a operação falhará. Você pode usar a mensagem de erro resultante para solucionar o conflito. Antes de selecionar esta opção, certifique-se de que o complemento Amazon EKS não gerencie configurações que você precise gerenciar manualmente.

7. Selecione **Próximo**.
8. Na página **Revisar e adicionar**, escolha **Criar**.

   Após a conclusão da instalação do complemento, você verá o complemento instalado.

CLI da AWS

1. Crie o `add-on.json` arquivo:

Para a identidade do Pod, utilize o seguinte formato:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

Para autenticação IRSA, utilize o seguinte formato:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

Substituir `<role ARN>` com o ARN da função que foi criada na etapa anterior.

2. Instale o complemento Trident EKS.

```
aws eks create-addon --cli-input-json file://add-on.json
```

eksctl

O comando de exemplo a seguir instala o complemento Trident EKS:

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

Atualize o complemento Trident EKS.

Console de gerenciamento

1. Abra o console do Amazon EKS https://console.aws.amazon.com/eks/home#/clusters .
2. No painel de navegação à esquerda, selecione **Clusters**.
3. Selecione o nome do cluster para o qual deseja atualizar o complemento NetApp Trident CSI.
4. Selecione a aba **Complementos**.
5. Selecione * Trident by NetApp* e depois selecione **Editar**.
6. Na página **Configurar Trident by NetApp**, faça o seguinte:
   a. Selecione a **Versão** que deseja usar.
   b. Expanda as **Configurações opcionais** e modifique conforme necessário.
   c. Selecione **Salvar alterações**.

CLI da AWS

O exemplo a seguir atualiza o complemento EKS:

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- Verifique a versão atual do seu complemento FSxN Trident CSI. Substituir `my-cluster` com o nome do seu cluster.

```
eksctl get addon --name netapp_trident-operator --cluster my-cluster
```

Exemplo de saída:

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.6.0-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- Atualize o complemento para a versão retornada em ATUALIZAÇÃO DISPONÍVEL na saída da etapa anterior.

```
eksctl update addon --name netapp_trident-operator --version
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

Se você remover o `--force` Se alguma das opções e configurações do complemento do Amazon EKS entrarem em conflito com suas configurações existentes, a atualização do complemento do Amazon EKS falhará e você receberá uma mensagem de erro para ajudá-lo a resolver o conflito. Antes de especificar esta opção, certifique-se de que o complemento Amazon EKS não gerencie configurações que você precisa gerenciar, pois essas configurações serão sobrescritas com esta opção. Para obter mais informações sobre outras opções para esta configuração, consulte"Complementos" . Para obter mais informações sobre o gerenciamento de campos do Kubernetes no Amazon EKS, consulte"gerenciamento de campos do Kubernetes" .

Desinstale/remova o complemento Trident EKS.

Você tem duas opções para remover um complemento do Amazon EKS:

- **Preservar software adicional no seu cluster** – Esta opção remove o gerenciamento de quaisquer configurações pelo Amazon EKS. Isso também remove a capacidade do Amazon EKS de notificá-lo sobre atualizações e de atualizar automaticamente o complemento do Amazon EKS após você iniciar uma atualização. No entanto, isso preserva o software adicional no seu cluster. Essa opção transforma o complemento em uma instalação autogerenciada, em vez de um complemento do Amazon EKS. Com essa opção, não há tempo de inatividade para o complemento. Mantenha o `--preserve` opção no comando para preservar o complemento.
- **Remova completamente o software complementar do seu cluster** – A NetApp recomenda que você remova o complemento Amazon EKS do seu cluster somente se não houver recursos no seu cluster que dependam dele. Remova o `--preserve` opção da `delete` comando para remover o complemento.

Se o complemento tiver uma conta IAM associada a ele, essa conta IAM não será removida.

Console de gerenciamento

1. Abra o console do Amazon EKS em https://console.aws.amazon.com/eks/home#/clusters .
2. No painel de navegação à esquerda, selecione **Clusters**.
3. Selecione o nome do cluster para o qual deseja remover o complemento NetApp Trident CSI.
4. Selecione a guia **Complementos** e, em seguida, selecione * Trident by NetApp*.
5. Selecione **Remover**.
6. Na caixa de diálogo **Remover confirmação do netapp_trident-operator**, faça o seguinte:
   a. Se você deseja que o Amazon EKS pare de gerenciar as configurações do complemento, selecione **Preservar no cluster**. Faça isso se quiser manter o software complementar no seu cluster para poder gerenciar todas as configurações do complemento por conta própria.
   b. Digite **netapp_trident-operator**.
   c. Selecione **Remover**.

CLI da AWS

Substituir `my-cluster` com o nome do seu cluster e, em seguida, execute o seguinte comando.

```
aws eks delete-addon --cluster-name my-cluster --addon-name
netapp_trident-operator --preserve
```

eksctl

O comando a seguir desinstala o complemento Trident EKS:

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Configure o backend de armazenamento

Integração de drivers ONTAP SAN e NAS

Para criar um backend de armazenamento, você precisa criar um arquivo de configuração no formato JSON ou YAML. O arquivo precisa especificar o tipo de armazenamento desejado (NAS ou SAN), o sistema de arquivos, a SVM de onde os dados serão obtidos e como autenticar com ela. O exemplo a seguir mostra como definir o armazenamento baseado em NAS e usar um segredo da AWS para armazenar as credenciais da SVM que você deseja usar:

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFilesystemID: fs-xxxxxxxxxx
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFilesystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

Execute os seguintes comandos para criar e validar a Configuração de Backend do Trident (TBC):

- Crie a configuração de backend do Trident (TBC) a partir do arquivo YAML e execute o seguinte comando:

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- Verifique se a configuração do backend Trident (TBC) foi criada com sucesso:

```
Kubectl get tbc -n trident
```

```
NAME                           BACKEND NAME          BACKEND UUID
PHASE   STATUS

backend-tbc-ontap-nas          tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9   Bound   Success
```

Detalhes do driver FSx para ONTAP

Você pode integrar o Trident com o Amazon FSx for NetApp ONTAP usando os seguintes drivers:

- `ontap-san`Cada PV provisionado é um LUN dentro de seu próprio volume Amazon FSx for NetApp ONTAP . Recomendado para armazenamento em bloco.
- `ontap-nas`Cada PV provisionado é um volume completo do Amazon FSx for NetApp ONTAP . Recomendado para NFS e SMB.
- `ontap-san-economy`Cada PV provisionado é um LUN com um número configurável de LUNs por volume do Amazon FSx for NetApp ONTAP .
- `ontap-nas-economy`Cada PV provisionado é uma qtree, com um número configurável de qtrees por volume do Amazon FSx for NetApp ONTAP .
- `ontap-nas-flexgroup`Cada PV provisionado é um volume FlexGroup completo do Amazon FSx for NetApp ONTAP .

Para obter detalhes sobre o motorista, consulte"drivers NAS" e"Drivers SAN" .

Após a criação do arquivo de configuração, execute este comando para criá-lo no seu EKS:

```
kubectl create -f configuration_file
```

Para verificar o status, execute este comando:

```
kubectl get tbc -n trident
```

```
NAME                     BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

Configuração avançada do backend e exemplos

Consulte a tabela a seguir para obter as opções de configuração do backend:

Parâmetro	Descrição	Exemplo
`version`		Sempre 1
`storageDriverName`	Nome do driver de armazenamento	`ontap-nas`, `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy`
`backendName`	Nome personalizado ou o backend de armazenamento	Nome do motorista + "_" + dataLIF
`managementLIF`	Endereço IP de um cluster ou LIF de gerenciamento de SVM. Um nome de domínio totalmente qualificado (FQDN) pode ser especificado. Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Se você fornecer o `fsxFilesystemID` sob o `aws` campo, você não precisa fornecer o `managementLIF` porque o Trident recupera a SVM `managementLIF` Informações da AWS. Portanto, você deve fornecer as credenciais de um usuário no SVM (por exemplo: vsadmin) e o usuário deve ter a seguinte permissão: `vsadmin` papel.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parâmetro	Descrição	Exemplo
`dataLIF`	Endereço IP do protocolo LIF. * Drivers ONTAP NAS **: A NetApp recomenda especificar dataLIF. Caso não sejam fornecidos, o Trident obtém os dataLIFs da SVM. Você pode especificar um nome de domínio totalmente qualificado (FQDN) para ser usado nas operações de montagem NFS, permitindo criar um DNS round-robin para balancear a carga entre várias dataLIFs. Pode ser alterado após a configuração inicial. Consulte . * Drivers ONTAP SAN**: Não especifique para iSCSI. O Trident utiliza o ONTAP Selective LUN Map para descobrir os LIFs iSCI necessários para estabelecer uma sessão de múltiplos caminhos. Um aviso é gerado se dataLIF for definido explicitamente. Pode ser configurado para usar endereços IPv6 se o Trident foi instalado usando a opção IPv6. Os endereços IPv6 devem ser definidos entre colchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Ativar a criação e atualização automática da política de exportação [Booleano]. Usando o `autoExportPolicy` e `autoExportCIDRs` O Trident pode gerenciar políticas de exportação automaticamente, dependendo das opções disponíveis.	`false`
`autoExportCIDRs`	Lista de CIDRs para filtrar os IPs dos nós do Kubernetes quando `autoExportPolicy` está habilitado. Usando o `autoExportPolicy` e `autoExportCIDRs` O Trident pode gerenciar políticas de exportação automaticamente, dependendo das opções disponíveis.	"["0.0.0.0/0", "::/0"]"

Parâmetro	Descrição	Exemplo
`labels`	Conjunto de rótulos arbitrários formatados em JSON para aplicar aos volumes	""
`clientCertificate`	Valor do certificado do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`clientPrivateKey`	Valor da chave privada do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.	""
`trustedCACertificate`	Valor codificado em Base64 do certificado da Autoridade Certificadora (CA) confiável. Opcional. Utilizado para autenticação baseada em certificado.	""
`username`	Nome de usuário para conectar-se ao cluster ou SVM. Usado para autenticação baseada em credenciais. Por exemplo, vsadmin.	
`password`	Senha para conectar-se ao cluster ou SVM. Usado para autenticação baseada em credenciais.	
`svm`	Máquina virtual de armazenamento para usar	Derivado se um managementLIF de SVM for especificado.
`storagePrefix`	Prefixo usado ao provisionar novos volumes no SVM. Não pode ser modificado após a criação. Para atualizar esse parâmetro, você precisará criar um novo backend.	`trident`
`limitAggregateUsage`	**Não especifique para Amazon FSx for NetApp ONTAP.** O fornecido `fsxadmin` e `vsadmin` Não possuem as permissões necessárias para recuperar o uso agregado e limitá-lo usando o Trident.	Não utilize.
`limitVolumeSize`	O provisionamento falhará se o tamanho do volume solicitado for superior a este valor. Também restringe o tamanho máximo dos volumes que gerencia para qtrees e LUNs, e o `qtreesPerFlexvol` Essa opção permite personalizar o número máximo de qtrees por FlexVol volume.	"" (não aplicado por padrão)

Parâmetro	Descrição	Exemplo
`lunsPerFlexvol`	O número máximo de LUNs por volume Flexvol deve estar no intervalo [50, 200]. Somente SAN.	“100”
`debugTraceFlags`	Sinalizadores de depuração a serem usados na resolução de problemas. Exemplo: {"api":false, "method":true} Não use `debugTraceFlags` a menos que você esteja solucionando problemas e precise de um despejo de logs detalhado.	nulo
`nfsMountOptions`	Lista de opções de montagem NFS separadas por vírgulas. As opções de montagem para volumes persistentes do Kubernetes são normalmente especificadas nas classes de armazenamento, mas se nenhuma opção de montagem for especificada em uma classe de armazenamento, o Trident usará as opções de montagem especificadas no arquivo de configuração do backend de armazenamento. Se nenhuma opção de montagem for especificada na classe de armazenamento ou no arquivo de configuração, o Trident não definirá nenhuma opção de montagem em um volume persistente associado.	""
`nasType`	Configure a criação de volumes NFS ou SMB. As opções são `nfs` , `smb` , ou nulo. **Deve ser configurado para `smb` para volumes SMB.** Definir como nulo utiliza, por padrão, volumes NFS.	`nfs`
`qtreesPerFlexvol`	Número máximo de Qtrees por FlexVol volume, deve estar no intervalo [50, 300]	`"200"`

Parâmetro	Descrição	Exemplo
`smbShare`	Você pode especificar uma das seguintes opções: o nome de um compartilhamento SMB criado usando o Console de Gerenciamento da Microsoft ou a CLI do ONTAP , ou um nome para permitir que o Trident crie o compartilhamento SMB. Este parâmetro é necessário para os backends do Amazon FSx para ONTAP .	`smb-share`
`useREST`	Parâmetro booleano para usar APIs REST do ONTAP . Quando definido para `true` A Trident utilizará APIs REST do ONTAP para se comunicar com o backend. Este recurso requer o ONTAP 9.11.1 e posterior. Além disso, a função de login do ONTAP utilizada deve ter acesso ao `ontap` aplicativo. Isso é satisfeito pelo predefinido `vsadmin` e `cluster-admin` papéis.	`false`
`aws`	Você pode especificar o seguinte no arquivo de configuração do AWS FSx para ONTAP: - `fsxFilesystemID` Especifique o ID do sistema de arquivos AWS FSx. - `apiRegion` : Nome da região da API da AWS. - `apikey` Chave da API da AWS. - `secretKey` Chave secreta da AWS.	"" "" ""
`credentials`	Especifique as credenciais do FSx SVM a serem armazenadas no AWS Secrets Manager. - `name` : Nome de recurso da Amazon (ARN) do segredo, que contém as credenciais do SVM. - `type` : Definir para `awsarn` . Consulte"Criar um segredo do AWS Secrets Manager" para mais informações.	

Opções de configuração de backend para provisionamento de volumes

Você pode controlar o provisionamento padrão usando essas opções em `defaults` seção da configuração. Para ver um exemplo, consulte os exemplos de configuração abaixo.

Parâmetro	Descrição	Padrão
`spaceAllocation`	Alocação de espaço para LUNs	`true`
`spaceReserve`	Modo de reserva de espaço; "nenhum" (fino) ou "volume" (grosso)	`none`
`snapshotPolicy`	Política de instantâneo a ser usada	`none`
`qosPolicy`	Grupo de políticas de QoS a ser atribuído aos volumes criados. Escolha uma das opções qosPolicy ou adaptiveQosPolicy por pool de armazenamento ou backend. A utilização de grupos de políticas de QoS com o Trident requer o ONTAP 9.8 ou posterior. Você deve usar um grupo de políticas de QoS não compartilhado e garantir que o grupo de políticas seja aplicado a cada componente individualmente. Um grupo de políticas de QoS compartilhado impõe o limite máximo para a taxa de transferência total de todas as cargas de trabalho.	""
`adaptiveQosPolicy`	Grupo de políticas de QoS adaptativas a serem atribuídas aos volumes criados. Escolha uma das opções qosPolicy ou adaptiveQosPolicy por pool de armazenamento ou backend. Não suportado por ontap-nas-economy.	""
`snapshotReserve`	Percentagem do volume reservada para instantâneos "0"	Se `snapshotPolicy` é `none` , `else` ""
`splitOnClone`	Separar um clone de seu progenitor no momento da criação.	`false`
`encryption`	Ative a Criptografia de Volume NetApp (NVE) no novo volume; o padrão é `false` . Para usar esta opção, o NVE precisa estar licenciado e habilitado no cluster. Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE" .	`false`
`luksEncryption`	Ative a criptografia LUKS. Consulte"Use o Linux Unified Key Setup (LUKS)" . Somente SAN.	""

Parâmetro	Descrição	Padrão
`tieringPolicy`	Política de escalonamento a ser usada `none`	
`unixPermissions`	Modo para novos volumes. **Deixe em branco para volumes SMB.**	""
`securityStyle`	Estilo de segurança para novos volumes. Suporte a NFS `mixed` e `unix` estilos de segurança. Suporte para PMEs `mixed` e `ntfs` estilos de segurança.	O padrão do NFS é `unix` . O padrão SMB é `ntfs` .

Prepare-se para provisionar volumes SMB

Você pode provisionar volumes SMB usando o `ontap-nas` motorista. Antes de concluirIntegração de drivers ONTAP SAN e NAS Complete os seguintes passos.

Antes de começar

Antes de poder provisionar volumes SMB usando o `ontap-nas` Motorista, você deve ter o seguinte.

- Um cluster Kubernetes com um nó controlador Linux e pelo menos um nó de trabalho Windows executando o Windows Server 2019. O Trident suporta volumes SMB montados em pods executados apenas em nós Windows.
- Pelo menos um segredo Trident contendo suas credenciais do Active Directory. Para gerar segredos `smbcreds` :

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Um proxy CSI configurado como um serviço do Windows. Para configurar um `csi-proxy` , consulte"GitHub: Proxy CSI" ou"GitHub: CSI Proxy para Windows" para nós do Kubernetes executados no Windows.

Passos

1. Criar compartilhamentos SMB. Você pode criar os compartilhamentos administrativos SMB de duas maneiras: usando o"Console de gerenciamento da Microsoft" Acesse as Pastas Compartilhadas pelo snap-in ou usando a CLI do ONTAP . Para criar compartilhamentos SMB usando a CLI do ONTAP :
   a. Se necessário, crie a estrutura de diretórios para o compartilhamento.

      O `vserver cifs share create` O comando verifica o caminho especificado na opção -path durante a criação do compartilhamento. Se o caminho especificado não existir, o comando falhará.

   b. Crie um compartilhamento SMB associado à SVM especificada:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

c. Verifique se o compartilhamento foi criado:

```
vserver cifs share show -share-name share_name
```

Consulte"Criar um compartilhamento SMB" Para obter detalhes completos.

2. Ao criar o backend, você deve configurar o seguinte para especificar os volumes SMB. Para todas as opções de configuração do backend FSx para ONTAP , consulte"Opções e exemplos de configuração do FSx para ONTAP" .

Parâmetro	Descrição	Exemplo
`smbShare`	Você pode especificar uma das seguintes opções: o nome de um compartilhamento SMB criado usando o Console de Gerenciamento da Microsoft ou a CLI do ONTAP , ou um nome para permitir que o Trident crie o compartilhamento SMB. Este parâmetro é necessário para os backends do Amazon FSx para ONTAP .	`smb-share`
`nasType`	**Deve ser configurado para `smb` .** Se for nulo, o valor padrão é `nfs` .	`smb`
`securityStyle`	Estilo de segurança para novos volumes. **Deve ser configurado para `ntfs` ou `mixed` para volumes SMB.**	`ntfs`ou `mixed` para volumes SMB
`unixPermissions`	Modo para novos volumes. **Deve ficar vazio para volumes SMB.**	""

Configure uma classe de armazenamento e um PVC.

Configure um objeto StorageClass do Kubernetes e crie a classe de armazenamento para instruir o Trident sobre como provisionar volumes. Crie um PersistentVolumeClaim (PVC) que utilize a StorageClass do Kubernetes configurada para solicitar acesso ao PV. Em seguida, você pode montar o painel fotovoltaico em um suporte.

Criar uma classe de armazenamento

Configure um objeto StorageClass do Kubernetes.

O "Objeto StorageClass do Kubernetes" O objeto identifica o Trident como o provisionador usado para essa classe e instrui o Trident sobre como provisionar um volume. Use este exemplo para configurar o Storageclass para volumes usando NFS (consulte a seção Atributo Trident abaixo para obter a lista completa de atributos):

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

Use este exemplo para configurar o Storageclass para volumes usando iSCSI:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  provisioningType: "thin"
  snapshots: "true"
```

Para provisionar volumes NFSv3 no AWS Bottlerocket, adicione os seguintes itens necessários. `mountOptions` para a classe de armazenamento:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock
```

Consulte"Objetos Kubernetes e Trident" Para obter detalhes sobre como as classes de armazenamento interagem com o `PersistentVolumeClaim` e parâmetros para controlar como o Trident provisiona volumes.

Criar uma classe de armazenamento

Passos

1. Este é um objeto do Kubernetes, então use `kubectl` para criá-lo no Kubernetes.

```
kubectl create -f storage-class-ontapnas.yaml
```

2. Agora você deverá ver uma classe de armazenamento **basic-csi** tanto no Kubernetes quanto no Trident, e o Trident deverá ter descoberto os pools no backend.

```
kubectl get sc basic-csi
```

```
NAME         PROVISIONER             AGE
basic-csi    csi.trident.netapp.io   15h
```

Criar o PVC

UM "*PersistentVolumeClaim*" (PVC) é uma solicitação de acesso ao PersistentVolume no cluster.

O PVC pode ser configurado para solicitar armazenamento de um determinado tamanho ou modo de acesso. Utilizando a StorageClass associada, o administrador do cluster pode controlar mais do que apenas o tamanho e o modo de acesso do PersistentVolume, como o desempenho ou o nível de serviço.

Após criar o tubo de PVC, você pode montar o volume em um compartimento.

Exemplos de manifestos

Manifestações de exemplo de PersistentVolumeClaim

Estes exemplos mostram opções básicas de configuração de PVC.

PVC com acesso RWX

Este exemplo mostra um PVC básico com acesso RWX associado a uma StorageClass chamada `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

Exemplo de PVC usando iSCSI

Este exemplo mostra um PVC básico para iSCSI com acesso RWO associado a uma StorageClass chamada `protection-gold`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 1Gi
storageClassName: protection-gold
```

Criar PVC

Passos

1. Crie o PVC.

```
kubectl create -f pvc.yaml
```

2. Verifique o status do PVC.

```
kubectl get pvc
```

```
NAME          STATUS VOLUME     CAPACITY ACCESS MODES STORAGECLASS AGE
pvc-storage Bound  pv-name 2Gi      RWO                           5m
```

Consulte"Objetos Kubernetes e Trident" Para obter detalhes sobre como as classes de armazenamento interagem com o `PersistentVolumeClaim` e parâmetros para controlar como o Trident provisiona volumes.

Atributos do Trident

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

Implantar aplicação de exemplo

Após a criação da classe de armazenamento e do PVC, você pode montar o PV em um pod. Esta seção lista os comandos e configurações de exemplo para anexar o PV a um pod.

Passos

1. Instale o volume em um pod.

```
kubectl create -f pv-pod.yaml
```

Estes exemplos mostram configurações básicas para conectar o PVC a um pod: **Configuração básica**:

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

Você pode monitorar o progresso usando `kubectl get pod --watch`.

2. Verifique se o volume está montado em `/my/mount/path`.

```
kubectl exec -it pv-pod -- df -h /my/mount/path
```

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

Agora você pode excluir o Pod. O aplicativo Pod deixará de existir, mas o volume permanecerá.

```
kubectl delete pod pv-pod
```

Configure o complemento Trident EKS em um cluster EKS.

O NetApp Trident simplifica o gerenciamento de armazenamento do Amazon FSx for NetApp ONTAP no Kubernetes, permitindo que seus desenvolvedores e administradores se concentrem na implantação de aplicativos. O complemento NetApp Trident EKS inclui os patches de segurança e correções de bugs mais recentes, e é validado pela AWS para funcionar com o Amazon EKS. O complemento EKS permite garantir de forma consistente que seus clusters Amazon EKS estejam seguros e estáveis, além de reduzir o trabalho necessário para instalar, configurar e atualizar complementos.

Pré-requisitos

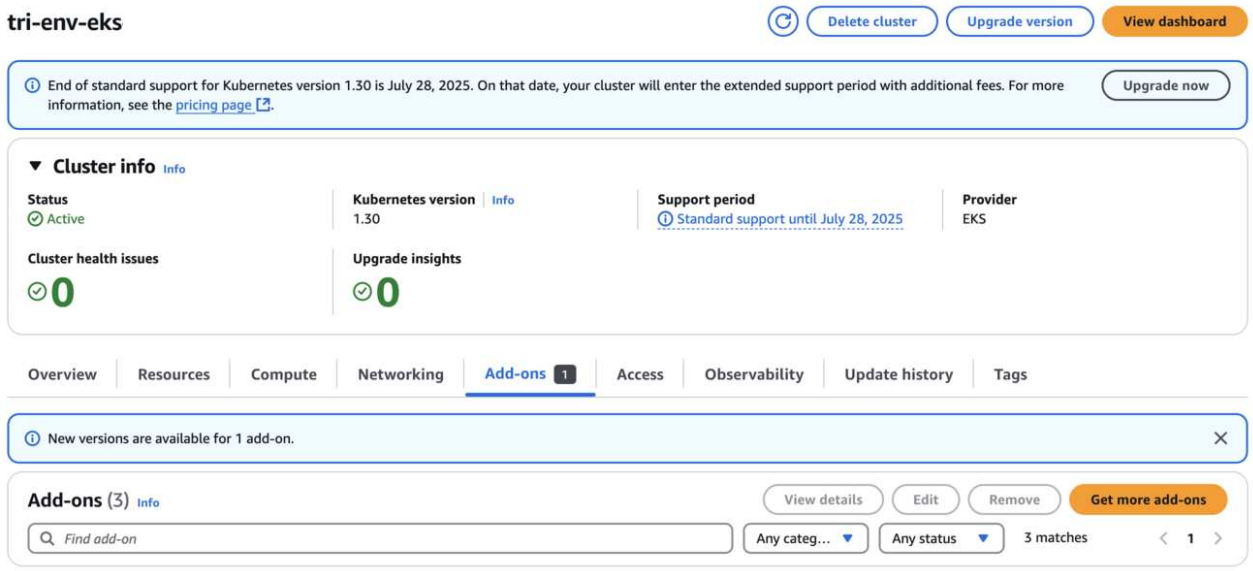
Certifique-se de ter o seguinte antes de configurar o complemento Trident para AWS EKS:

- Uma conta de cluster Amazon EKS com permissões para trabalhar com complementos. Consulte"Complementos do Amazon EKS" .
- Permissões da AWS para o marketplace da AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo de AMI: Amazon Linux 2 (AL2_x86_64) ou Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo de nó: AMD ou ARM
- Um sistema de arquivos Amazon FSx for NetApp ONTAP

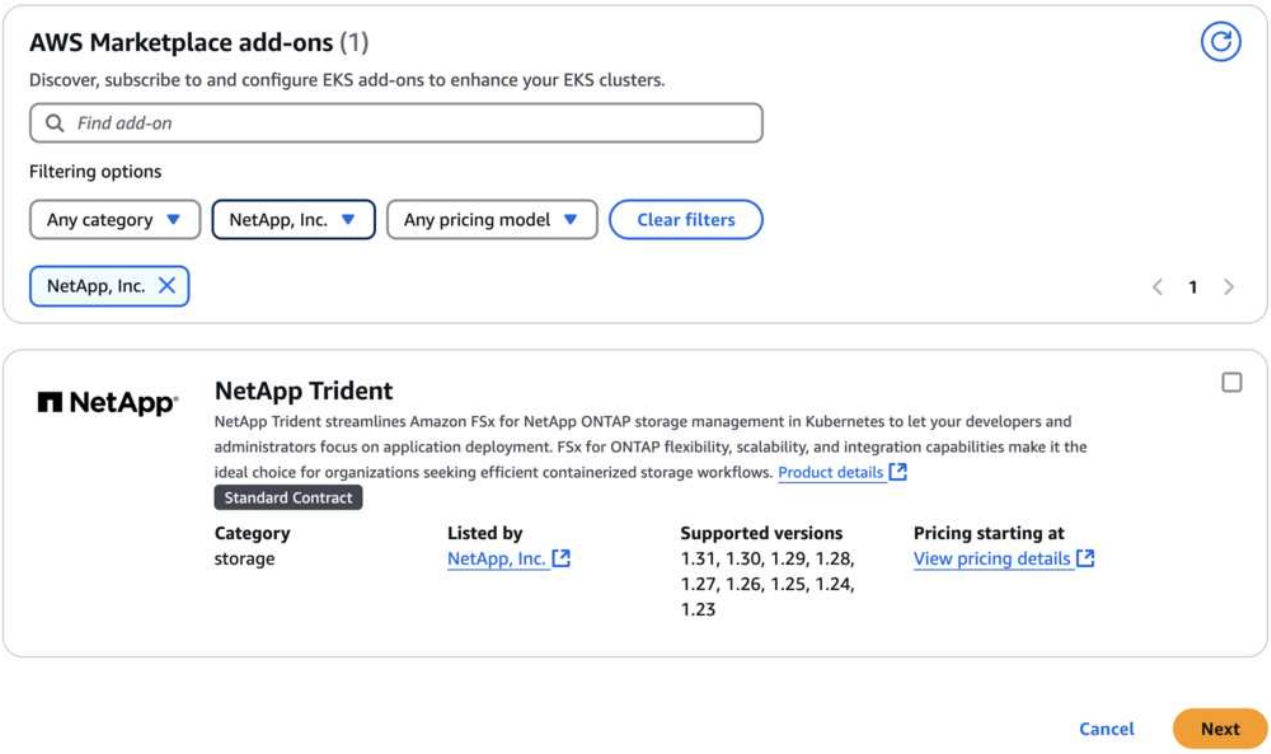
Passos

1. Certifique-se de criar uma função do IAM e um segredo da AWS para permitir que os pods do EKS acessem os recursos da AWS. Para obter instruções, consulte"Crie uma função do IAM e um segredo da AWS." .

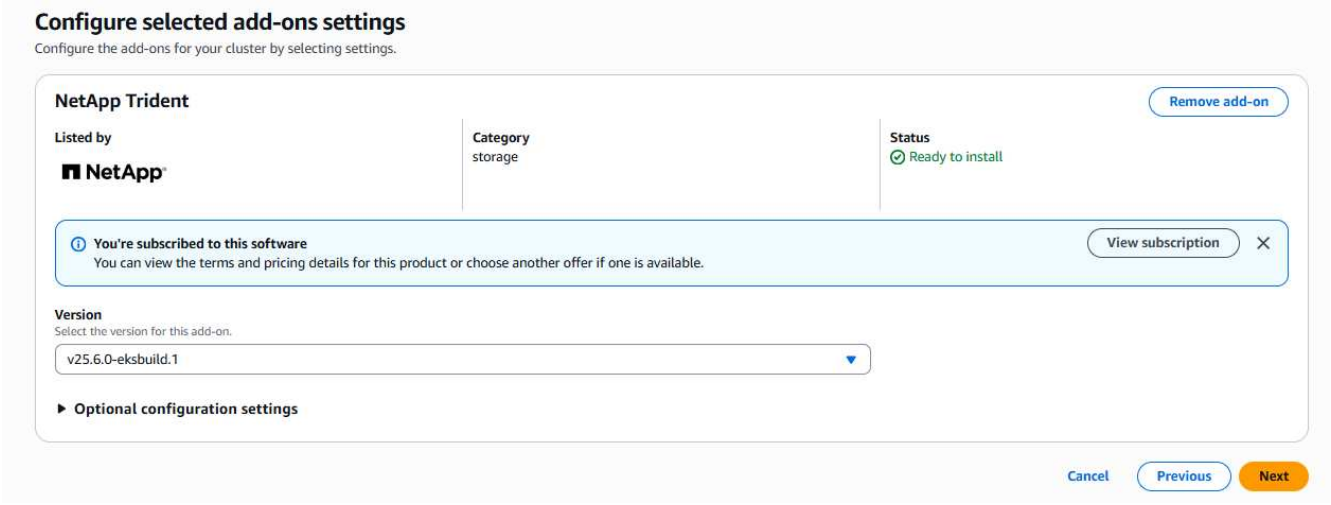
2. No seu cluster Kubernetes EKS, navegue até a guia **Complementos**.



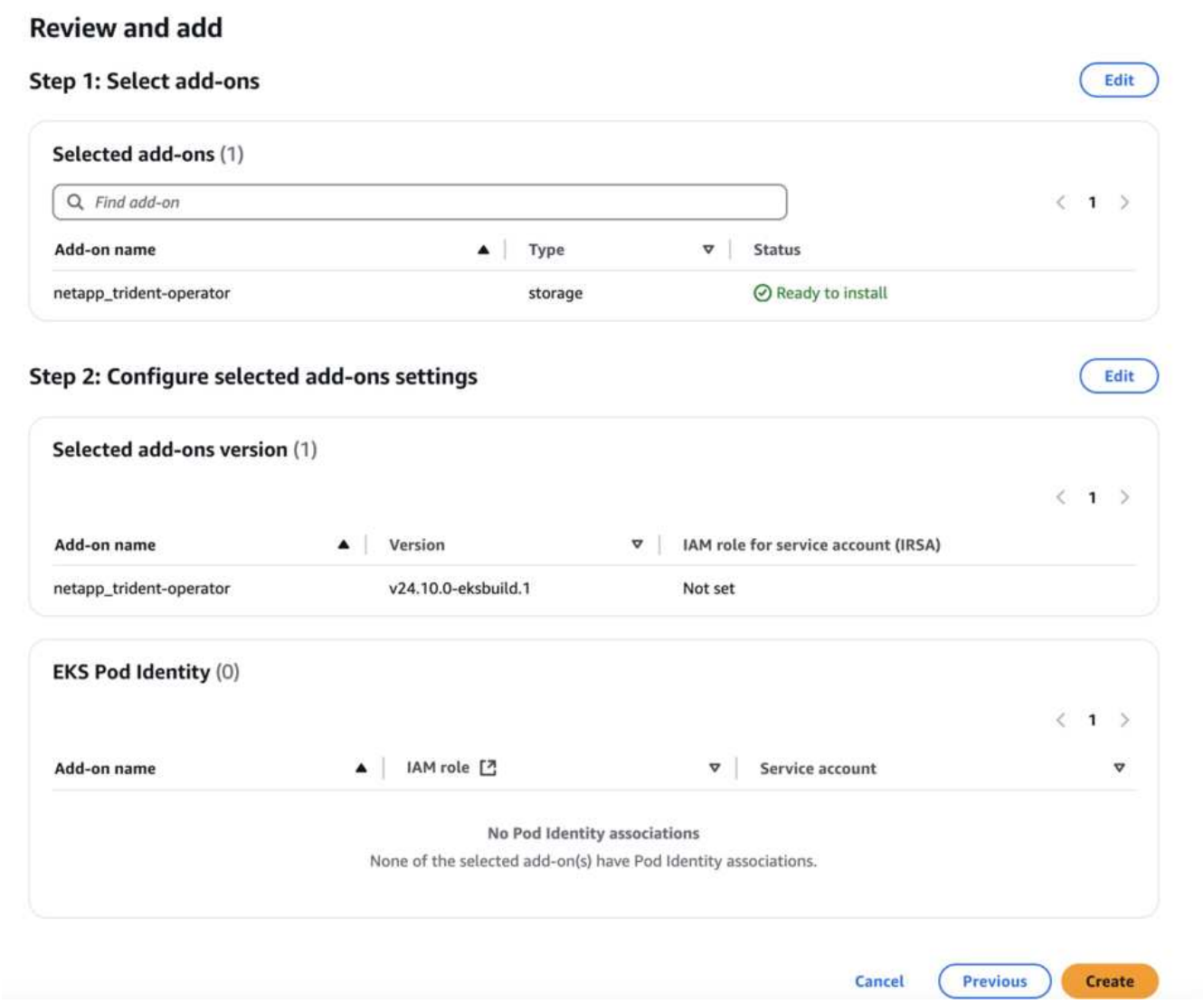
3. Acesse os complementos do **AWS Marketplace** e escolha a categoria *armazenamento*.



4. Localize * NetApp Trident*, selecione a caixa de seleção do complemento Trident e clique em **Avançar**.
5. Escolha a versão desejada do complemento.

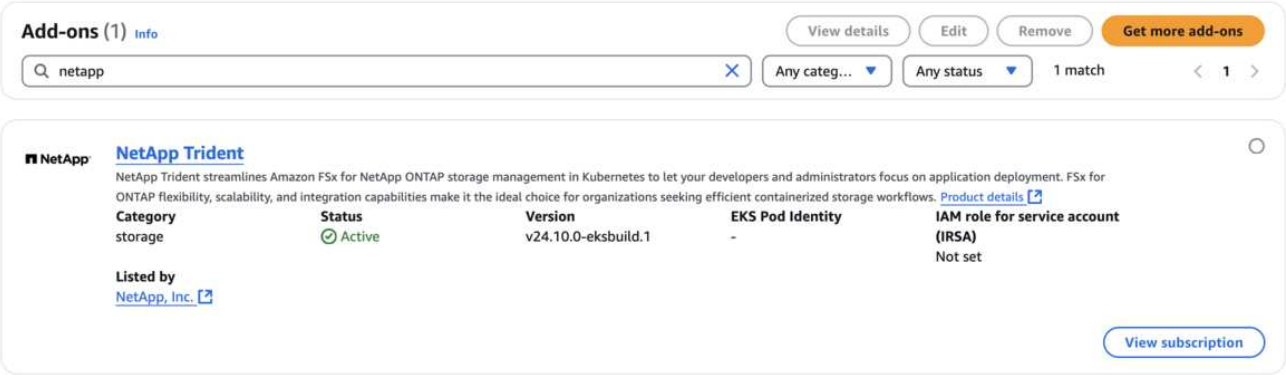


6. Configure as configurações necessárias do complemento.



7. Se você estiver usando o IRSA (funções do IAM para conta de serviço), consulte as etapas de configuração adicionais."aqui" .
8. Selecione **Criar**.

9. Verifique se o status do complemento é *Ativo*.



10. Execute o seguinte comando para verificar se o Trident está instalado corretamente no cluster:

```
kubectl get pods -n trident
```

11. Continue a configuração e configure o backend de armazenamento. Para obter informações, consulte"Configure o backend de armazenamento" .

Instale/desinstale o complemento Trident EKS usando a CLI.

Instale o complemento NetApp Trident EKS usando a CLI:

O comando de exemplo a seguir instala o complemento Trident EKS: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (com uma versão dedicada)

Desinstale o complemento NetApp Trident EKS usando a CLI:

O comando a seguir desinstala o complemento Trident EKS:

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Crie backends com kubectl

Um backend define a relação entre o Trident e um sistema de armazenamento. Ele informa ao Trident como se comunicar com esse sistema de armazenamento e como o Trident deve provisionar volumes a partir dele. Após a instalação do Trident , o próximo passo é criar um backend. O `TridentBackendConfig` A Definição de Recurso Personalizado (CRD) permite criar e gerenciar backends do Trident diretamente através da interface do Kubernetes. Você pode fazer isso usando `kubectl` ou a ferramenta de linha de comando equivalente para sua distribuição Kubernetes.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig` , `tbackendconfig`) é um CRD de front-end com namespace que permite gerenciar back-ends do Trident usando `kubectl` . Administradores de Kubernetes e de armazenamento agora podem criar e gerenciar backends diretamente por meio da CLI do Kubernetes, sem a

necessidade de um utilitário de linha de comando dedicado.(`tridentctl`).

Ao criar um `TridentBackendConfig` objeto, o seguinte acontece:

- Um ambiente de backend é criado automaticamente pelo Trident com base na configuração fornecida. Isso é representado internamente como um `TridentBackend` (`tbe` , `tridentbackend`) CR.
- O `TridentBackendConfig` está exclusivamente ligado a um `TridentBackend` que foi criado pela Trident.

Cada `TridentBackendConfig` mantém um mapeamento um-para-um com um `TridentBackend` A primeira é a interface fornecida ao usuário para projetar e configurar backends; a segunda é como o Trident representa o objeto backend propriamente dito.

> `TridentBackend`Os CRs são criados automaticamente pelo Trident. Você **não deve** modificá-los. Se você deseja fazer atualizações nos backends, faça isso modificando o `TridentBackendConfig` objeto.

Veja o exemplo a seguir para o formato do `TridentBackendConfig` CR:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Você também pode conferir os exemplos em "instalador de tridente" Diretório com exemplos de configurações para a plataforma/serviço de armazenamento desejado.

O `spec` Utiliza parâmetros de configuração específicos do backend. Neste exemplo, o backend usa o `ontap-san` O driver de armazenamento utiliza os parâmetros de configuração que estão tabelados aqui. Para obter a lista de opções de configuração para o driver de armazenamento desejado, consulte o"Informações de configuração do backend para o seu driver de armazenamento." .

O `spec` esta seção também inclui `credentials` e `deletionPolicy` campos, que são recentemente introduzidos no `TridentBackendConfig` CR:

- `credentials`Este parâmetro é um campo obrigatório e contém as credenciais usadas para autenticar com o sistema/serviço de armazenamento. Isso é definido como um segredo do Kubernetes criado pelo usuário. As credenciais não podem ser transmitidas em texto sem formatação e resultarão em um erro.
- `deletionPolicy`Este campo define o que deve acontecer quando o

`TridentBackendConfig foi apagado. Pode assumir um de dois valores possíveis:

- `delete`Isso resulta na exclusão de ambos. `TridentBackendConfig` CR e o backend associado. Este é o valor padrão.
- `retain`: Quando um `TridentBackendConfig` O CR é excluído, a definição do backend ainda estará presente e poderá ser gerenciada com `tridentctl` . Definir a política de exclusão para `retain` Permite que os usuários façam o downgrade para uma versão anterior (anterior à 21.04) e mantenham os backends criados. O valor deste campo pode ser atualizado após um `TridentBackendConfig` é criado.

O nome de um backend é definido usando `spec.backendName` . Se não for especificado, o nome do backend será definido como o nome do `TridentBackendConfig` objeto (nome.metadados). É recomendável definir explicitamente os nomes de backend usando `spec.backendName` .

Backends que foram criados com `tridentctl` não possuem um associado `TridentBackendConfig` objeto. Você pode optar por gerenciar esses back-ends com `kubectl` criando um `TridentBackendConfig` CR. É preciso ter cuidado para especificar parâmetros de configuração idênticos (como, por exemplo, `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , e assim por diante). O Trident irá vincular automaticamente o dispositivo recém-criado. `TridentBackendConfig` com o backend pré-existente.

Visão geral das etapas

Para criar um novo backend usando `kubectl` Você deve fazer o seguinte:

1. Criar um "Segredo do Kubernetes" O segredo contém as credenciais que o Trident precisa para se comunicar com o cluster/serviço de armazenamento.
2. Criar um `TridentBackendConfig` objeto. Esta seção contém detalhes específicos sobre o cluster/serviço de armazenamento e faz referência ao segredo criado na etapa anterior.

Após criar um backend, você pode observar seu status usando `kubectl get tbc <tbc-name> -n <trident-namespace>` e coletar detalhes adicionais.

Passo 1: Crie um segredo do Kubernetes

Crie um segredo que contenha as credenciais de acesso ao backend. Isso é específico para cada serviço/plataforma de armazenamento. Aqui está um exemplo:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

Esta tabela resume os campos que devem ser incluídos no Segredo para cada plataforma de armazenamento:

Descrição dos campos secretos da plataforma de armazenamento	Segredo	Descrição dos campos
Azure NetApp Files	ID do cliente	O ID do cliente obtido durante o cadastro no aplicativo.
Cloud Volumes Service para GCP	id_da_chave_privada	ID da chave privada. Parte da chave de API para conta de serviço do GCP com função de administrador do CVS
Cloud Volumes Service para GCP	chave_privada	Chave privada. Parte da chave de API para conta de serviço do GCP com função de administrador do CVS
Elemento (NetApp HCI/ SolidFire)	Ponto final	MVIP para o cluster SolidFire com credenciais de locatário
ONTAP	nome de usuário	Nome de usuário para conectar ao cluster/SVM. Utilizado para autenticação baseada em credenciais.
ONTAP	senha	Senha para conectar ao cluster/SVM. Utilizado para autenticação baseada em credenciais.
ONTAP	chavePrivadaDoCliente	Valor da chave privada do cliente codificado em Base64. Utilizado para autenticação baseada em certificado.

Descrição dos campos secretos da plataforma de armazenamento	Segredo	Descrição dos campos
ONTAP	chapUsername	Nome de usuário de entrada. Obrigatório se useCHAP=true. Para `ontap-san` e `ontap-san-economy`
ONTAP	chapIniciadorSecreto	Segredo do iniciador CHAP. Obrigatório se useCHAP=true. Para `ontap-san` e `ontap-san-economy`
ONTAP	chapTargetUsername	Nome de usuário alvo. Obrigatório se useCHAP=true. Para `ontap-san` e `ontap-san-economy`
ONTAP	chapTargetInitiatorSecret	Segredo iniciador do alvo CHAP. Obrigatório se useCHAP=true. Para `ontap-san` e `ontap-san-economy`

O segredo criado nesta etapa será referenciado em `spec.credentials` campo do `TridentBackendConfig` objeto que será criado na próxima etapa.

Passo 2: Crie o `TridentBackendConfig` CR

Agora você está pronto para criar o seu `TridentBackendConfig` CR. Neste exemplo, um backend que usa o `ontap-san` O driver é criado usando o `TridentBackendConfig` Objeto mostrado abaixo:

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Etapa 3: Verifique o status do `TridentBackendConfig` CR

Agora que você criou o `TridentBackendConfig` CR, você pode verificar o status. Veja o exemplo a seguir:

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                       BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

Um backend foi criado e vinculado com sucesso ao `TridentBackendConfig` CR.

A fase pode assumir um dos seguintes valores:

- `Bound`: O `TridentBackendConfig` O CR está associado a um backend, e esse backend contém `configRef` definido para o `TridentBackendConfig` UID do CR.
- `Unbound`Representado usando `""` . O `TridentBackendConfig` O objeto não está vinculado a um backend. Tudo recém-criado `TridentBackendConfig` Os CRs estão nessa fase por padrão. Após a mudança de fase, não é possível reverter para o estado Não Vinculado novamente.
- `Deleting`: O `TridentBackendConfig` CR's `deletionPolicy` estava configurado para excluir. Quando o `TridentBackendConfig` Quando o CR é excluído, ele entra no estado de exclusão.
  - Se não existirem reivindicações de volume persistentes (PVCs) no backend, exclua o `TridentBackendConfig` resultará na exclusão do backend e também do Trident . `TridentBackendConfig` CR.
  - Se um ou mais PVCs estiverem presentes no backend, ele entra em estado de exclusão. O `TridentBackendConfig` Posteriormente, o CR também entra na fase de deleção. O backend e `TridentBackendConfig` São apagados somente depois que todos os PVCs forem apagados.
- `Lost`O backend associado ao `TridentBackendConfig` CR foi apagado acidentalmente ou deliberadamente e o `TridentBackendConfig` O CR ainda possui uma referência ao backend excluído. O `TridentBackendConfig` O CR ainda pode ser excluído independentemente do `deletionPolicy` valor.
- `Unknown`O Trident não consegue determinar o estado ou a existência do backend associado ao `TridentBackendConfig` CR. Por exemplo, se o servidor da API não estiver respondendo ou se o `tridentbackends.trident.netapp.io` O CRD está faltando. Isso pode exigir intervenção.

Nesta etapa, o backend foi criado com sucesso! Existem diversas operações adicionais que podem ser realizadas, tais como:"atualizações e exclusões de backend" .

(Opcional) Passo 4: Obtenha mais detalhes

Você pode executar o seguinte comando para obter mais informações sobre seu backend:

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

```
NAME                      BACKEND NAME         BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success   ontap-san         delete
```

Além disso, você também pode obter um dump YAML/JSON de `TridentBackendConfig` .

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: 2021-04-21T20:45:11Z
  finalizers:
    - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo`contém o `backendName` e o `backendUUID` do backend que foi criado em resposta ao `TridentBackendConfig` CR. O `lastOperationStatus` O campo representa o status da última operação do `TridentBackendConfig` CR, que pode ser acionada pelo usuário (por exemplo, o usuário alterou algo em `spec`) ou acionado pelo Trident (por exemplo, durante reinicializações do Trident). Pode ser sucesso ou fracasso. `phase` representa o estado da relação entre o `TridentBackendConfig` CR e o backend. No

exemplo acima, `phase` tem o valor Bound, o que significa que o `TridentBackendConfig` CR está associado ao backend.

Você pode executar o `kubectl -n trident describe tbc <tbc-cr-name>` comando para obter detalhes dos registros de eventos.

> Não é possível atualizar ou excluir um backend que contenha um associado. `TridentBackendConfig` objeto usando `tridentctl` . Para entender os passos envolvidos na transição entre `tridentctl` e `TridentBackendConfig` ,"veja aqui" .

Gerenciar back-ends

Realize o gerenciamento de backend com kubectl

Aprenda como executar operações de gerenciamento de back-end usando `kubectl` .

Excluir um backend

Ao excluir um `TridentBackendConfig` , você instrui o Trident a excluir/reter backends (com base em `deletionPolicy`). Para excluir um backend, certifique-se de que `deletionPolicy` Está configurado para excluir. Para excluir apenas o `TridentBackendConfig` , assegure-se de que `deletionPolicy` está definido para manter. Isso garante que o backend ainda esteja presente e possa ser gerenciado usando `tridentctl` .

Execute o seguinte comando:

```
kubectl delete tbc <tbc-name> -n trident
```

O Trident não exclui os segredos do Kubernetes que estavam em uso pelo `TridentBackendConfig` . O usuário do Kubernetes é responsável por limpar os segredos. É preciso ter cuidado ao apagar segredos. Você deve excluir segredos somente se eles não estiverem sendo usados pelos servidores de backend.

Veja os backends existentes

Execute o seguinte comando:

```
kubectl get tbc -n trident
```

Você também pode executar `tridentctl get backend -n trident` ou `tridentctl get backend -o yaml -n trident` Para obter uma lista de todos os backends existentes. Esta lista também incluirá backends que foram criados com `tridentctl` .

Atualizar um backend

Existem diversos motivos para atualizar um backend:

- As credenciais de acesso ao sistema de armazenamento foram alteradas. Para atualizar as credenciais, o segredo do Kubernetes que é usado no `TridentBackendConfig` O objeto precisa ser atualizado. O Trident atualizará automaticamente o backend com as credenciais mais recentes fornecidas. Execute o

seguinte comando para atualizar o segredo do Kubernetes:

```
kubectl apply -f <updated-secret-file.yaml> -n trident
```

- É necessário atualizar os parâmetros (como o nome da SVM do ONTAP que está sendo usada).
  - Você pode atualizar `TridentBackendConfig` objetos diretamente através do Kubernetes usando o seguinte comando:

    ```
    kubectl apply -f <updated-backend-file.yaml>
    ```

  - Alternativamente, você pode fazer alterações no existente. `TridentBackendConfig` CR usando o seguinte comando:

    ```
    kubectl edit tbc <tbc-name> -n trident
    ```

> - Se uma atualização do backend falhar, o backend permanecerá em sua última configuração conhecida. Você pode visualizar os registros para determinar a causa executando o seguinte comando: `kubectl get tbc <tbc-name> -o yaml -n trident` ou `kubectl describe tbc <tbc-name> -n trident`.
> - Após identificar e corrigir o problema com o arquivo de configuração, você pode executar o comando de atualização novamente.

Realize a gestão de backend com o tridentctl

Aprenda como executar operações de gerenciamento de back-end usando `tridentctl`.

Crie um backend

Depois de criar um"arquivo de configuração do backend" Execute o seguinte comando:

```
tridentctl create backend -f <backend-file> -n trident
```

Se a criação do backend falhar, algo estava errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs -n trident
```

Após identificar e corrigir o problema com o arquivo de configuração, você pode simplesmente executar o `create` Comande novamente.

Excluir um backend

Para excluir um backend do Trident, faça o seguinte:

1. Recuperar o nome do backend:

```
tridentctl get backend -n trident
```

2. Exclua o backend:

```
tridentctl delete backend <backend-name> -n trident
```

> Se o Trident tiver provisionado volumes e snapshots desse backend que ainda existam, a exclusão do backend impede que novos volumes sejam provisionados por ele. O sistema de backend continuará existindo em um estado de "Exclusão".

Veja os backends existentes

Para visualizar os backends que o Trident conhece, faça o seguinte:

- Para obter um resumo, execute o seguinte comando:

```
tridentctl get backend -n trident
```

- Para obter todos os detalhes, execute o seguinte comando:

```
tridentctl get backend -o json -n trident
```

Atualizar um backend

Após criar um novo arquivo de configuração de backend, execute o seguinte comando:

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

Se a atualização do servidor falhar, algo estava errado com a configuração do servidor ou você tentou uma atualização inválida. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs -n trident
```

Após identificar e corrigir o problema com o arquivo de configuração, você pode simplesmente executar o `update` Comande novamente.

Identifique as classes de armazenamento que utilizam um backend.

Este é um exemplo do tipo de pergunta que você pode responder com o JSON que `tridentctl` Saídas para objetos de backend. Isso usa o `jq` utilitário, que você precisa instalar.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name,
storageClasses: [.storage[].storageClasses]|unique}]'
```

Isso também se aplica a backends que foram criados usando `TridentBackendConfig` .

Alternar entre opções de gerenciamento de back-end

Aprenda sobre as diferentes maneiras de gerenciar back-ends no Trident.

Opções para gerenciar back-ends

Com a introdução de `TridentBackendConfig` , os administradores agora têm duas maneiras exclusivas de gerenciar os sistemas de back-end. Isso levanta as seguintes questões:

- É possível criar back-ends usando `tridentctl` ser gerenciado com `TridentBackendConfig` ?
- É possível criar back-ends usando `TridentBackendConfig` ser gerenciado usando `tridentctl` ?

Gerenciar `tridentctl` backends usando `TridentBackendConfig`

Esta seção aborda os passos necessários para gerenciar backends criados usando `tridentctl` diretamente através da interface do Kubernetes, criando `TridentBackendConfig` objetos.

Isso se aplica aos seguintes cenários:

- Backends pré-existentes que não possuem um `TridentBackendConfig` porque foram criados com `tridentctl` .
- Novos backends que foram criados com `tridentctl` , enquanto outros `TridentBackendConfig` Os objetos existem.

Em ambos os cenários, os servidores de backend continuarão presentes, com o Trident agendando volumes e operando sobre eles. Os administradores têm duas opções aqui:

- Continuar usando `tridentctl` para gerenciar os backends que foram criados usando-o.
- Vincule os backends criados usando `tridentctl` para um novo `TridentBackendConfig` objeto. Fazer isso significaria que os back-ends seriam gerenciados usando `kubectl` e não `tridentctl` .

Para gerenciar um backend pré-existente usando `kubectl` , você precisará criar um `TridentBackendConfig` que se integra ao backend existente. Segue abaixo uma visão geral de como isso funciona:

1. Criar um segredo do Kubernetes. O segredo contém as credenciais que o Trident precisa para se comunicar com o cluster/serviço de armazenamento.
2. Criar um `TridentBackendConfig` objeto. Esta seção contém detalhes específicos sobre o cluster/serviço de armazenamento e faz referência ao segredo criado na etapa anterior. É preciso ter

cuidado para especificar parâmetros de configuração idênticos (como, por exemplo, `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , e assim por diante). `spec.backendName` Deve ser definido com o nome do backend existente.

Etapa 0: Identificar o backend

Para criar um `TridentBackendConfig` Para se conectar a um backend existente, você precisará obter a configuração do backend. Neste exemplo, vamos supor que um backend foi criado usando a seguinte definição JSON:

```
tridentctl get backend ontap-nas-backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

```
cat ontap-nas-backend.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",
  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels": {
    "store": "nas_store"
  },
  "region": "us_east_1",
  "storage": [
    {
      "labels": {
        "app": "msoffice",
        "cost": "100"
      },
      "zone": "us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
      "labels": {
        "app": "mysqldb",
        "cost": "25"
      },
      "zone": "us_east_1d",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "false",
        "unixPermissions": "0775"
      }
    }
  ]
}
```

Passo 1: Crie um segredo do Kubernetes

Crie um segredo que contenha as credenciais para o backend, conforme mostrado neste exemplo:

```
cat tbc-ontap-nas-backend-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password
```

```
kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

Passo 2: Crie um `TridentBackendConfig` CR

O próximo passo é criar um `TridentBackendConfig` CR que se vinculará automaticamente ao preexistente `ontap-nas-backend` (como neste exemplo). Assegure-se de que os seguintes requisitos sejam atendidos:

- O mesmo nome de backend está definido em `spec.backendName` .
- Os parâmetros de configuração são idênticos aos do backend original.
- Os pools virtuais (se presentes) devem manter a mesma ordem que no backend original.
- As credenciais são fornecidas por meio de um segredo do Kubernetes e não em texto simples.

Neste caso, o `TridentBackendConfig` Vai ficar assim:

```
cat backend-tbc-ontap-nas.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'
```

```
kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

Etapa 3: Verifique o status do `TridentBackendConfig` CR

Depois do `TridentBackendConfig` foi criada, sua fase deve ser `Bound` . Deve também refletir o mesmo nome de backend e UUID do backend existente.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

O backend agora será totalmente gerenciado usando o `tbc-ontap-nas-backend` `TridentBackendConfig` objeto.

Gerenciar `TridentBackendConfig` backends usando `tridentctl`

```
`tridentctl`pode ser usado para listar backends que foram criados usando
`TridentBackendConfig` .  Além disso, os administradores também podem
optar por gerenciar completamente esses back-ends por meio de `tridentctl`
ao excluir `TridentBackendConfig` e garantindo `spec.deletionPolicy` está
definido para `retain` .
```

Etapa 0: Identificar o backend

Por exemplo, vamos supor que o seguinte backend foi criado usando `TridentBackendConfig` :

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

A partir dos resultados, observa-se que `TridentBackendConfig` Foi criado com sucesso e está vinculado a um backend [observe o UUID do backend].

Passo 1: Confirmar `deletionPolicy` está definido para `retain`

Vamos analisar o valor de `deletionPolicy` . Isso precisa ser configurado para `retain` . Isso garante que quando um `TridentBackendConfig` O CR é excluído, mas a definição do backend ainda estará presente e poderá ser gerenciada com `tridentctl` .

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        retain
```

Não prossiga para a próxima etapa a menos que `deletionPolicy` está definido para `retain` .

Passo 2: Exclua o `TridentBackendConfig` CR

O passo final é excluir o `TridentBackendConfig` CR. Após confirmar o `deletionPolicy` está definido para `retain` Você pode prosseguir com a exclusão:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

tridentctl get backend ontap-san-backend -n trident
+------------------+---------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+------------------+---------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+------------------+---------------
+--------------------------------------+--------+---------+
```

Após a exclusão do `TridentBackendConfig` O objeto, o Trident simplesmente o remove sem realmente excluir o próprio backend.

Criar e gerenciar classes de armazenamento

Criar uma classe de armazenamento

Configure um objeto StorageClass do Kubernetes e crie a classe de armazenamento para instruir o Trident sobre como provisionar volumes.

Configure um objeto StorageClass do Kubernetes.

O "Objeto StorageClass do Kubernetes" Identifica o Trident como o provisionador usado para essa classe e instrui o Trident sobre como provisionar um volume. Por exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Consulte"Objetos Kubernetes e Trident" Para obter detalhes sobre como as classes de armazenamento interagem com o `PersistentVolumeClaim` e parâmetros para controlar como o Trident provisiona volumes.

Criar uma classe de armazenamento

Após criar o objeto StorageClass, você pode criar a classe de armazenamento. Exemplos de classe de armazenamento Fornece alguns exemplos básicos que você pode usar ou modificar.

Passos

1. Este é um objeto do Kubernetes, então use `kubectl` para criá-lo no Kubernetes.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. Agora você deverá ver uma classe de armazenamento **basic-csi** tanto no Kubernetes quanto no Trident, e o Trident deverá ter descoberto os pools no backend.

```
kubectl get sc basic-csi
```

```
NAME         PROVISIONER             AGE
basic-csi    csi.trident.netapp.io   15h
```

```
./tridentctl -n trident get storageclass basic-csi -o json
```

```
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}
```


Exemplos de classe de armazenamento

A Trident fornece "Definições simples de classes de armazenamento para backends específicos" .

Alternativamente, você pode editar `sample-input/storage-class-csi.yaml.templ` arquivo que acompanha o instalador e substituir *`BACKEND_TYPE`* com o nome do driver de armazenamento.

```
./tridentctl -n trident get backend
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

Gerenciar classes de armazenamento

Você pode visualizar as classes de armazenamento existentes, definir uma classe de armazenamento padrão, identificar o backend da classe de armazenamento e excluir classes de armazenamento.

Veja as classes de armazenamento existentes.

- Para visualizar as classes de armazenamento do Kubernetes existentes, execute o seguinte comando:

```
kubectl get storageclass
```

- Para visualizar os detalhes da classe de armazenamento do Kubernetes, execute o seguinte comando:

```
kubectl get storageclass <storage-class> -o json
```

- Para visualizar as classes de armazenamento sincronizadas do Trident, execute o seguinte comando:

```
tridentctl get storageclass
```

- Para visualizar os detalhes da classe de armazenamento sincronizado do Trident, execute o seguinte comando:

```
tridentctl get storageclass <storage-class> -o json
```

Defina uma classe de armazenamento padrão.

O Kubernetes 1.6 adicionou a capacidade de definir uma classe de armazenamento padrão. Esta é a classe de armazenamento que será usada para provisionar um Volume Persistente caso o usuário não especifique uma em uma Solicitação de Volume Persistente (PVC).

- Defina uma classe de armazenamento padrão configurando a anotação. `storageclass.kubernetes.io/is-default-class` para verdadeiro na definição da classe de armazenamento. De acordo com a especificação, qualquer outro valor ou a ausência da anotação é interpretada como falsa.
- Você pode configurar uma classe de armazenamento existente para ser a classe de armazenamento padrão usando o seguinte comando:

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- Da mesma forma, você pode remover a anotação da classe de armazenamento padrão usando o seguinte comando:

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Existem também exemplos no pacote de instalação do Trident que incluem essa anotação.

> Deve haver apenas uma classe de armazenamento padrão em seu cluster por vez. Tecnicamente, o Kubernetes não impede que você tenha mais de uma classe de armazenamento, mas ele se comportará como se não houvesse nenhuma classe de armazenamento padrão.

Identifique o backend para uma classe de armazenamento.

Este é um exemplo do tipo de pergunta que você pode responder com o JSON que `tridentctl` Saídas para objetos do backend Trident . Isso usa o `jq` utilitário, que você talvez precise instalar primeiro.

```
tridentctl get storageclass -o json | jq  '[.items[] | {storageClass:
.Config.name, backends: [.storage]|unique}]'
```

Excluir uma classe de armazenamento

Para excluir uma classe de armazenamento do Kubernetes, execute o seguinte comando:

```
kubectl delete storageclass <storage-class>
```

`<storage-class>`deve ser substituído pela sua classe de armazenamento.

Quaisquer volumes persistentes criados por meio dessa classe de armazenamento permanecerão intactos e o Trident continuará a gerenciá-los.

> Trident impõe um vazio `fsType` pelos volumes que gera. Para backends iSCSI, recomenda-se impor `parameters.fsType` na classe de armazenamento. Você deve excluir as StorageClasses existentes e recriá-las com `parameters.fsType` especificado.

Provisionar e gerenciar volumes

Forneça um volume

Crie um PersistentVolumeClaim (PVC) que utilize a StorageClass do Kubernetes configurada para solicitar acesso ao PV. Em seguida, você pode montar o painel fotovoltaico em um suporte.

Visão geral

UM "*PersistentVolumeClaim*" (PVC) é uma solicitação de acesso ao PersistentVolume no cluster.

O PVC pode ser configurado para solicitar armazenamento de um determinado tamanho ou modo de acesso. Utilizando a StorageClass associada, o administrador do cluster pode controlar mais do que apenas o tamanho e o modo de acesso do PersistentVolume, como o desempenho ou o nível de serviço.

Após criar o tubo de PVC, você pode montar o volume em um compartimento.

Criar o PVC

Passos

1. Crie o PVC.

```
kubectl create -f pvc.yaml
```

2. Verifique o status do PVC.

```
kubectl get pvc
```

```
NAME          STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS AGE
pvc-storage   Bound    pv-name    1Gi        RWO                    5m
```

1. Instale o volume em um pod.

```
kubectl create -f pv-pod.yaml
```

Você pode monitorar o progresso usando `kubectl get pod --watch`.

2. Verifique se o volume está montado em `/my/mount/path`.

```
kubectl exec -it task-pv-pod -- df -h /my/mount/path
```

3. Agora você pode excluir o Pod. O aplicativo Pod deixará de existir, mas o volume permanecerá.

```
kubectl delete pod pv-pod
```

Exemplos de manifestos

Manifestações de exemplo de PersistentVolumeClaim

Estes exemplos mostram opções básicas de configuração de PVC.

PVC com acesso RWO

Este exemplo mostra um PVC básico com acesso RWO associado a uma StorageClass chamada `basic-csi` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

PVC com NVMe/TCP

Este exemplo mostra um PVC básico para NVMe/TCP com acesso RWO associado a uma StorageClass chamada `protection-gold` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

Exemplos de manifesto de pod

Estes exemplos mostram configurações básicas para fixar o PVC a um pod.

Configuração básica

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

Configuração básica de NVMe/TCP

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

Consulte"Objetos Kubernetes e Trident" Para obter detalhes sobre como as classes de armazenamento interagem com o `PersistentVolumeClaim` e parâmetros para controlar como o Trident provisiona volumes.

Expandir volumes

O Trident oferece aos usuários do Kubernetes a capacidade de expandir seus volumes depois que eles são criados. Encontre informações sobre as configurações necessárias para expandir volumes iSCSI, NFS, SMB, NVMe/TCP e FC.

Expandir um volume iSCSI

Você pode expandir um Volume Persistente (PV) iSCSI usando o provisionador CSI.

> A expansão de volume iSCSI é suportada pelo `ontap-san` , `ontap-san-economy` , `solidfire-san` requer drivers e Kubernetes 1.16 ou posterior.

Etapa 1: Configure a classe de armazenamento para suportar a expansão de volume.

Edite a definição da classe de armazenamento para definir o `allowVolumeExpansion` campo para `true` .

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Para uma StorageClass já existente, edite-a para incluir o `allowVolumeExpansion` parâmetro.

Passo 2: Crie um PVC com a StorageClass que você criou.

Edite a definição de PVC e atualize o `spec.resources.requests.storage` para refletir o novo tamanho desejado, que deve ser maior que o tamanho original.

```
cat pvc-ontapsan.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

A Trident cria um Volume Persistente (PV) e o associa a esta Reivindicação de Volume Persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

Etapa 3: Defina um compartimento que conecte o PVC.

Conecte o PV a um pod para que ele possa ser redimensionado. Existem dois cenários ao redimensionar um PV iSCSI:

- Se o PV estiver conectado a um pod, o Trident expande o volume no backend de armazenamento, verifica novamente o dispositivo e redimensiona o sistema de arquivos.
- Ao tentar redimensionar um PV não anexado, o Trident expande o volume no servidor de armazenamento. Após o PVC ser vinculado a um pod, o Trident verifica novamente o dispositivo e redimensiona o sistema de arquivos. Em seguida, o Kubernetes atualiza o tamanho do PVC após a conclusão bem-sucedida da operação de expansão.

Neste exemplo, é criado um pod que utiliza o `san-pvc` .

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Etapa 4: Expandir o PV

Para redimensionar o PV criado de 1Gi para 2Gi, edite a definição do PVC e atualize o `spec.resources.requests.storage` para 2Gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Etapa 5: Valide a expansão

Você pode verificar se a expansão funcionou corretamente conferindo o tamanho do PVC, do PV e o volume do Trident :

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Expandir um volume FC

Você pode expandir um Volume Persistente (PV) FC usando o provisionador CSI.

> A expansão do volume FC é suportada pelo `ontap-san` O driver requer Kubernetes 1.16 ou posterior.

Etapa 1: Configure a classe de armazenamento para suportar a expansão de volume.

Edite a definição da classe de armazenamento para definir o `allowVolumeExpansion` campo para `true` .

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Para uma StorageClass já existente, edite-a para incluir o `allowVolumeExpansion` parâmetro.

Passo 2: Crie um PVC com a StorageClass que você criou.

Edite a definição de PVC e atualize o `spec.resources.requests.storage` para refletir o novo tamanho desejado, que deve ser maior que o tamanho original.

```
cat pvc-ontapsan.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

A Trident cria um Volume Persistente (PV) e o associa a esta Reivindicação de Volume Persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

Etapa 3: Defina um compartimento que conecte o PVC.

Conecte o PV a um pod para que ele possa ser redimensionado. Existem dois cenários ao redimensionar um painel fotovoltaico de célula de combustível:

- Se o PV estiver conectado a um pod, o Trident expande o volume no backend de armazenamento, verifica novamente o dispositivo e redimensiona o sistema de arquivos.
- Ao tentar redimensionar um PV não anexado, o Trident expande o volume no servidor de armazenamento. Após o PVC ser vinculado a um pod, o Trident verifica novamente o dispositivo e redimensiona o sistema de arquivos. Em seguida, o Kubernetes atualiza o tamanho do PVC após a conclusão bem-sucedida da

operação de expansão.

Neste exemplo, é criado um pod que utiliza o `san-pvc` .

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Etapa 4: Expandir o PV

Para redimensionar o PV criado de 1Gi para 2Gi, edite a definição do PVC e atualize o `spec.resources.requests.storage` para 2Gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Etapa 5: Valide a expansão

Você pode verificar se a expansão funcionou corretamente conferindo o tamanho do PVC, do PV e o volume do Trident :

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Expandir um volume NFS

O Trident suporta a expansão de volume para PVs NFS provisionados em `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `gcp-cvs` , e `azure-netapp-files` back-ends.

Etapa 1: Configure a classe de armazenamento para suportar a expansão de volume.

Para redimensionar um PV NFS, o administrador primeiro precisa configurar a classe de armazenamento para permitir a expansão do volume, definindo o seguinte: `allowVolumeExpansion` campo para `true` :

```
cat storageclass-ontapnas.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

Se você já criou uma classe de armazenamento sem essa opção, basta editar a classe de armazenamento

existente usando `kubectl edit storageclass` para permitir a expansão do volume.

Passo 2: Crie um PVC com a StorageClass que você criou.

```
cat pvc-ontapnas.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

A Trident deve criar um PV NFS de 20 MiB para este PVC:

```
kubectl get pvc
NAME           STATUS    VOLUME
CAPACITY     ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

Etapa 3: Expandir o PV

Para redimensionar o PV recém-criado de 20 MiB para 1 GiB, edite o PVC e defina `spec.resources.requests.storage` para 1 GiB:

```
kubectl edit pvc ontapnas20mb
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
# ...
```

Etapa 4: Valide a expansão

Você pode verificar se o redimensionamento funcionou corretamente conferindo o tamanho do PVC, do PV e do volume Trident :

```
kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Volumes de importação

Você pode importar volumes de armazenamento existentes como um PV do Kubernetes usando `tridentctl import`.

Visão geral e considerações

Você pode importar um volume para o Trident para:

- Containerize uma aplicação e reutilize seu conjunto de dados existente.
- Utilize um clone de um conjunto de dados para uma aplicação efêmera.
- Reconstruir um cluster Kubernetes com falha
- Migrar dados de aplicativos durante a recuperação de desastres

Considerações

Antes de importar um volume, revise as seguintes considerações.

- O Trident só pode importar volumes ONTAP do tipo RW (leitura e gravação). Os volumes do tipo DP (proteção de dados) são volumes de destino do SnapMirror . Você deve quebrar a relação de espelhamento antes de importar o volume para o Trident.

- Sugerimos importar volumes sem conexões ativas. Para importar um volume que esteja sendo usado ativamente, clone o volume e, em seguida, execute a importação.

Isso é especialmente importante para volumes de bloco, pois o Kubernetes não teria conhecimento da conexão anterior e poderia facilmente anexar um volume ativo a um pod. Isso pode resultar em corrupção de dados.

- No entanto `StorageClass` Deve ser especificado em um PVC; o Trident não usa esse parâmetro durante a importação. As classes de armazenamento são usadas durante a criação de volumes para selecionar entre os pools disponíveis com base nas características de armazenamento. Como o volume já existe, não é necessário selecionar um pool durante a importação. Portanto, a importação não falhará mesmo que o volume exista em um backend ou pool que não corresponda à classe de armazenamento especificada no PVC.
- O volume existente é determinado e definido no PVC. Após o volume ser importado pelo driver de armazenamento, o PV é criado com uma referência ClaimRef para o PVC.
  - A política de recuperação é inicialmente definida como `retain` no PV. Após o Kubernetes vincular com sucesso o PVC e o PV, a política de recuperação é atualizada para corresponder à política de recuperação da classe de armazenamento.
  - Se a política de recuperação da Classe de Armazenamento for `delete` O volume de armazenamento será excluído quando o PV for excluído.
- Por padrão, o Trident gerencia o PVC e renomeia o FlexVol volume e o LUN no backend. Você pode passar o `--no-manage` sinalizador para importar um volume não gerenciado. Se você usar `--no-manage` A Trident não realiza nenhuma operação adicional no PVC ou PV durante o ciclo de vida dos objetos. O volume de armazenamento não é excluído quando o PV é excluído, e outras operações como clonagem e redimensionamento de volume também são ignoradas.

Essa opção é útil se você deseja usar o Kubernetes para cargas de trabalho em contêineres, mas, fora isso, prefere gerenciar o ciclo de vida do volume de armazenamento fora do Kubernetes.

- Uma anotação é adicionada ao PVC e ao PV com a dupla função de indicar que o volume foi importado e se o PVC e o PV são gerenciados. Esta anotação não deve ser modificada nem removida.

Importar um volume

Você pode usar `tridentctl import` Importar um volume.

Passos

1. Crie o arquivo Persistent Volume Claim (PVC) (por exemplo, `pvc.yaml`) que será usado para criar o PVC. O arquivo PVC deve incluir `name` , `namespace` , `accessModes` , e `storageClassName` . Opcionalmente, você pode especificar `unixPermissions` na sua definição de PVC.

   Segue abaixo um exemplo de especificação mínima:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

Não inclua parâmetros adicionais como nome do PV ou tamanho do volume. Isso pode fazer com que o comando de importação falhe.

2. Use o `tridentctl import` Comando para especificar o nome do backend Trident que contém o volume e o nome que identifica exclusivamente o volume no armazenamento (por exemplo: ONTAP FlexVol, Element Volume, caminho do Cloud Volumes Service). O `-f` É necessário fornecer o argumento para especificar o caminho para o arquivo PVC.

```
tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-file>
```

Exemplos

Consulte os exemplos de importação de volume a seguir para obter informações sobre os drivers compatíveis.

ONTAP NAS e ONTAP NAS FlexGroup

O Trident suporta a importação de volumes usando o `ontap-nas` e `ontap-nas-flexgroup` motoristas.

- O Trident não suporta importação de volume usando o `ontap-nas-economy` motorista.
- O `ontap-nas` e `ontap-nas-flexgroup` Os drivers não permitem nomes de volume duplicados.

Cada volume criado com o `ontap-nas` O driver é um FlexVol volume no cluster ONTAP . Importando volumes FlexVol com o `ontap-nas` O driver funciona da mesma forma. Um volume FlexVol que já existe em um cluster ONTAP pode ser importado como um `ontap-nas` PVC. Da mesma forma, os volumes do FlexGroup podem ser importados como `ontap-nas-flexgroup` PVCs.

Exemplos de ONTAP NAS

A seguir, apresentamos um exemplo de importação de um volume gerenciado e de um volume não gerenciado.

Volume gerenciado

O exemplo a seguir importa um volume chamado `managed_volume` em um backend chamado `ontap_nas` :

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>

+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
| pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
```

Volume não gerenciado

Ao usar o `--no-manage` argumento: Trident não renomeia o volume.

O exemplo a seguir importa `unmanaged_volume` no `ontap_nas` backend:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
```

ONTAP SAN

O Trident suporta importação de volume usando o `ontap-san` (iSCSI, NVMe/TCP e FC) e `ontap-san-economy` motoristas.

O Trident pode importar volumes ONTAP SAN FlexVol que contêm um único LUN. Isto é consistente com o `ontap-san` driver, que cria um FlexVol volume para cada PVC e um LUN dentro do FlexVol volume. O Trident importa o FlexVol volume e o associa à definição de PVC. Trident pode importar `ontap-san-economy`

volumes que contêm vários LUNs.

Exemplos de SAN ONTAP

A seguir, apresentamos um exemplo de importação de um volume gerenciado e de um volume não gerenciado.

Volume gerenciado

Para volumes gerenciados, o Trident renomeia o FlexVol volume para `pvc-<uuid>` formato e o LUN dentro do FlexVol volume para `lun0` .

O exemplo a seguir importa o `ontap-san-managed` FlexVol volume que está presente no `ontap_san_default` backend:

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

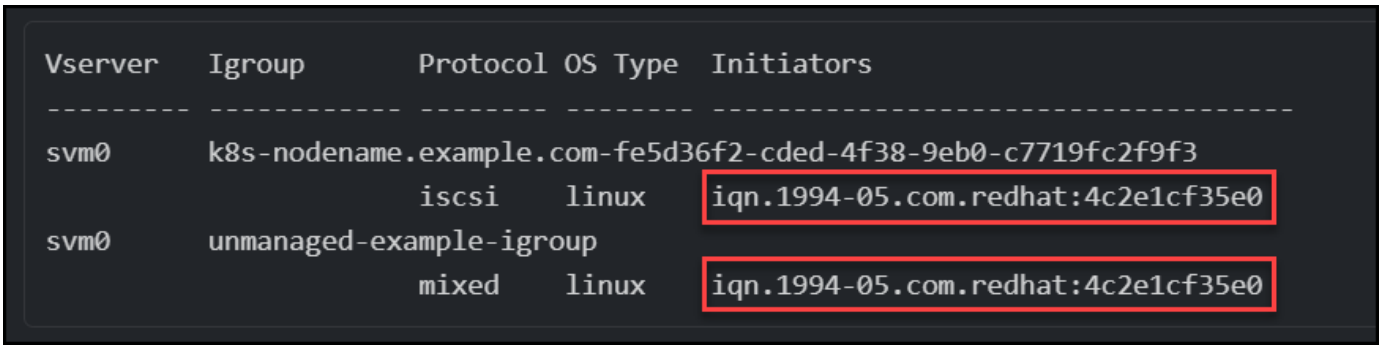
Volume não gerenciado

O exemplo a seguir importa `unmanaged_example_volume` no `ontap_san` backend:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

Se você tiver LUNs mapeadas para igroups que compartilham um IQN com o IQN de um nó do Kubernetes,

como mostrado no exemplo a seguir, você receberá o seguinte erro: `LUN already mapped to initiator(s) in this group`. Você precisará remover o iniciador ou desvincular o LUN para importar o volume.



Elemento

O Trident oferece suporte ao software NetApp Element e à importação de volumes NetApp HCI usando o `solidfire-san` motorista.

> O driver Element suporta nomes de volume duplicados. No entanto, o Trident retorna um erro se houver nomes de volume duplicados. Como solução alternativa, clone o volume, forneça um nome exclusivo para o volume e importe o volume clonado.

Exemplo de elemento

O exemplo a seguir importa um `element-managed` volume no backend `element_default`.

```
tridentctl import volume element_default element-managed -f pvc-basic-
import.yaml -n trident -d

+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
```

Plataforma Google Cloud

O Trident suporta a importação de volumes usando o `gcp-cvs` motorista.

Para importar um volume com suporte do NetApp Cloud Volumes Service no Google Cloud Platform, identifique o volume pelo seu caminho. O caminho do volume é a porção do caminho de exportação do volume após o `:/` . Por exemplo, se o caminho de exportação for `10.0.0.1:/adroit-jolly-swift` , o caminho do volume é `adroit-jolly-swift` .

Exemplo do Google Cloud Platform

O exemplo a seguir importa um `gcp-cvs` volume no backend `gcpcvs_YEppr` com o caminho de volume de `adroit-jolly-swift` .

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-
file> -n trident

+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | gcp-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true    |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
```

Azure NetApp Files

O Trident suporta a importação de volumes usando o `azure-netapp-files` motorista.

Para importar um volume do Azure NetApp Files , identifique o volume pelo seu caminho. O caminho do volume é a porção do caminho de exportação do volume após o `:/` . Por exemplo, se o caminho de montagem for `10.0.0.2:/importvol1` , o caminho do volume é `importvol1` .

Exemplo de Azure NetApp Files

O exemplo a seguir importa um `azure-netapp-files` volume no backend `azurenetappfiles_40517` com o caminho do volume `importvol1` .

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

Google Cloud NetApp Volumes

O Trident suporta a importação de volumes usando o `google-cloud-netapp-volumes` motorista.

Exemplo de Google Cloud NetApp Volumes

O exemplo a seguir importa um `google-cloud-netapp-volumes` volume no backend `backend-tbc-gcnv1` com o volume `testvoleasiaeast1` .

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

O exemplo a seguir importa um `google-cloud-netapp-volumes` Volume quando dois volumes estão presentes na mesma região:

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personalize os nomes e rótulos dos volumes.

Com o Trident, você pode atribuir nomes e rótulos significativos aos volumes que criar. Isso ajuda você a identificar e mapear facilmente os volumes aos seus respectivos recursos do Kubernetes (PVCs). Você também pode definir modelos no nível do backend para criar nomes de volume e rótulos personalizados; quaisquer volumes que você criar, importar ou clonar seguirão os modelos.

Antes de começar

Suporte para nomes e rótulos de volume personalizáveis:

1. Operações de criação, importação e clonagem de volumes.
2. No caso do driver ontap-nas-economy, apenas o nome do volume Qtree está em conformidade com o modelo de nome.
3. No caso do driver ontap-san-economy, apenas o nome LUN está em conformidade com o modelo de nome.

Limitações

1. Os nomes de volume personalizáveis são compatíveis apenas com drivers ONTAP locais.
2. Os nomes de volume personalizáveis não se aplicam a volumes existentes.

Principais comportamentos dos nomes de volume personalizáveis

1. Se ocorrer uma falha devido a uma sintaxe inválida em um modelo de nome, a criação do backend falhará. No entanto, se a aplicação do modelo falhar, o volume será nomeado de acordo com a convenção de nomenclatura existente.

2. O prefixo de armazenamento não se aplica quando um volume é nomeado usando um modelo de nome da configuração do backend. Qualquer valor de prefixo desejado pode ser adicionado diretamente ao modelo.

Exemplos de configuração de backend com modelo de nome e rótulos

É possível definir modelos de nomes personalizados no nível raiz e/ou no nível do pool.

Exemplo de nível raiz

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "defaults": {
    "nameTemplate":
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.volume.Requ
estName}}"
  },
  "labels": {
    "cluster": "ClusterA",
    "PVC": "{{.volume.Namespace}}_{{.volume.RequestName}}"
  }
}
```

Exemplo de nível de piscina

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels": {
        "labelname": "label1",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels": {
        "cluster": "label2",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

Exemplos de modelos de nome

Exemplo 1:

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Exemplo 2:

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Pontos a considerar

1. No caso de importações em volume, os rótulos são atualizados somente se o volume existente possuir rótulos em um formato específico. Por exemplo: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. No caso de importações de volumes gerenciados, o nome do volume segue o modelo de nome definido no nível raiz na definição do backend.
3. O Trident não suporta o uso de um operador de fatiamento com o prefixo de armazenamento.
4. Se os modelos não resultarem em nomes de volume exclusivos, o Trident acrescentará alguns caracteres aleatórios para criar nomes de volume exclusivos.
5. Se o nome personalizado para um volume econômico do NAS exceder 64 caracteres, o Trident nomeará os volumes de acordo com a convenção de nomenclatura existente. Para todos os outros drivers ONTAP , se o nome do volume exceder o limite de nomes, o processo de criação do volume falhará.

Compartilhar um volume NFS entre namespaces

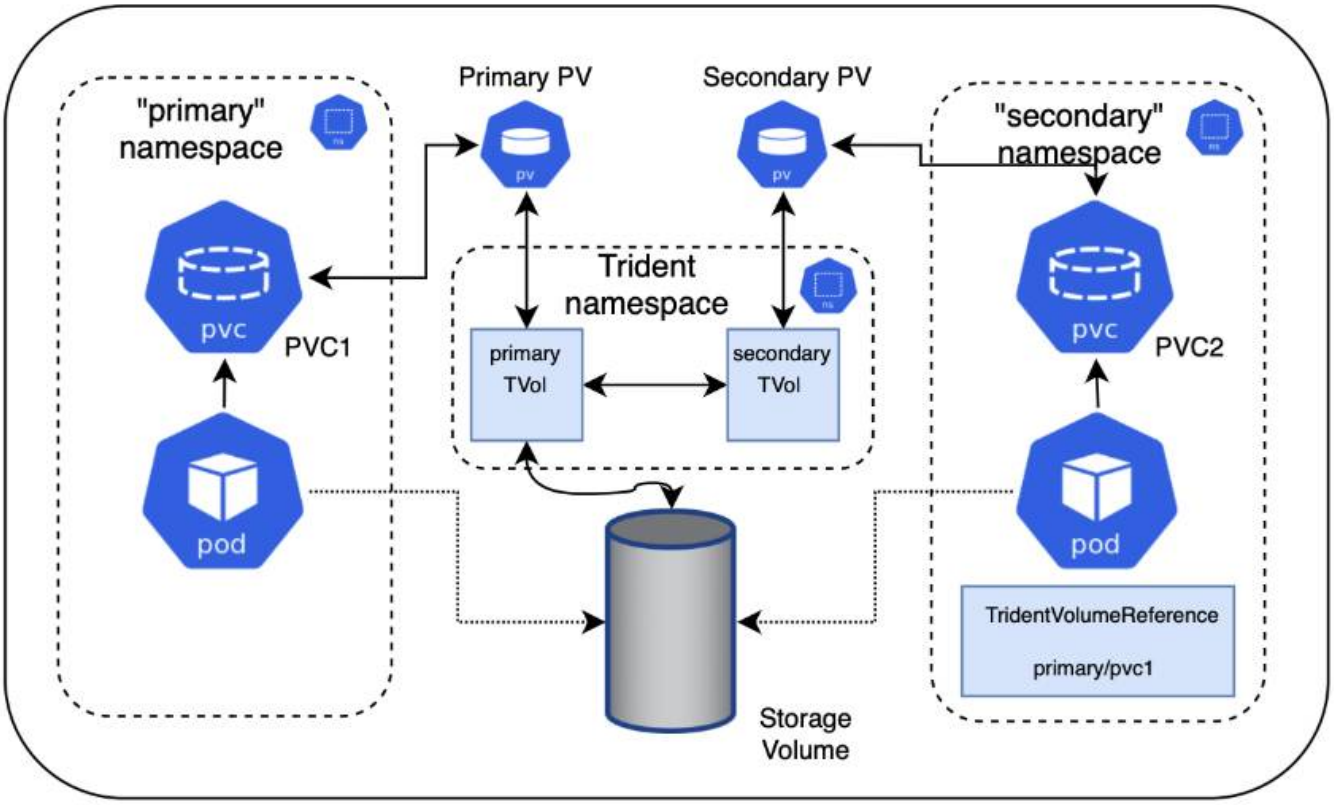
Usando o Trident, você pode criar um volume em um namespace primário e compartilhá-lo em um ou mais namespaces secundários.

Características

O CR TridentVolumeReference permite compartilhar com segurança volumes NFS ReadWriteMany (RWX) em um ou mais namespaces do Kubernetes. Essa solução nativa do Kubernetes oferece os seguintes benefícios:

- Múltiplos níveis de controle de acesso para garantir a segurança.
- Compatível com todos os drivers de volume Trident NFS.
- Sem dependência do tridentctl ou de qualquer outro recurso não nativo do Kubernetes.

Este diagrama ilustra o compartilhamento de volumes NFS entre dois namespaces do Kubernetes.



Início rápido

Você pode configurar o compartilhamento de volumes NFS em apenas alguns passos.

1 **Configure o PVC de origem para compartilhar o volume.**

O proprietário do namespace de origem concede permissão para acessar os dados no PVC de origem.

2 **Conceder permissão para criar uma solicitação de configuração (CR) no espaço de nomes de destino.**

O administrador do cluster concede permissão ao proprietário do namespace de destino para criar o CR TridentVolumeReference.

3 **Crie uma TridentVolumeReference no namespace de destino.**

O proprietário do namespace de destino cria o CR TridentVolumeReference para se referir ao PVC de origem.

4 **Crie o PVC subordinado no namespace de destino.**

O proprietário do namespace de destino cria o PVC subordinado para usar a fonte de dados do PVC de origem.

Configure os namespaces de origem e destino.

Para garantir a segurança, o compartilhamento entre namespaces exige colaboração e ação do proprietário do namespace de origem, do administrador do cluster e do proprietário do namespace de destino. A função do usuário é definida em cada etapa.

Passos

1. **Proprietário do namespace de origem:** Criar o PVC(`pvc1`) no namespace de origem que concede permissão para compartilhar com o namespace de destino(`namespace2`) usando o `shareToNamespace` anotação.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

   O Trident cria o PV e seu volume de armazenamento NFS de backend.

   - Você pode compartilhar o PVC com vários namespaces usando uma lista separada por vírgulas. Por exemplo, `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4` .
   - Você pode compartilhar com todos os namespaces usando `*` . Por exemplo, `trident.netapp.io/shareToNamespace: *`
   - Você pode atualizar o PVC para incluir o `shareToNamespace` Anotações a qualquer momento.

2. **Administrador do cluster:** Certifique-se de que o RBAC adequado esteja em vigor para conceder permissão ao proprietário do namespace de destino para criar o TridentVolumeReference CR no namespace de destino.
3. **Proprietário do namespace de destino:** Crie uma solicitação de configuração (CR) TridentVolumeReference no namespace de destino que faça referência ao namespace de origem. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Proprietário do namespace de destino:** Criar um PVC(`pvc2`) no espaço de nomes de destino(`namespace2`) usando o `shareFromPVC` Anotação para designar o PVC de origem.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

O diâmetro do tubo de PVC de destino deve ser menor ou igual ao do tubo de PVC de origem.

Resultados

Trident lê o `shareFromPVC` A anotação no PVC de destino cria o PV de destino como um volume subordinado sem recurso de armazenamento próprio, que aponta para o PV de origem e compartilha o recurso de armazenamento do PV de origem. O PVC e o PV de destino parecem estar conectados normalmente.

Excluir um volume compartilhado

Você pode excluir um volume que é compartilhado entre vários namespaces. O Trident removerá o acesso ao volume no namespace de origem e manterá o acesso para outros namespaces que compartilham o volume. Quando todos os namespaces que fazem referência ao volume são removidos, o Trident exclui o volume.

Usar `tridentctl get` para consultar volumes subordinados

Usando o[`tridentctl` utilitário, você pode executar o `get` comando para obter volumes subordinados. Para

obter mais informações, consulte o `tridentctl` comandos e opções.

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

Bandeiras:

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

Limitações

- O Trident não pode impedir que os namespaces de destino gravem no volume compartilhado. Você deve usar bloqueio de arquivos ou outros processos para evitar a sobrescrita de dados em volumes compartilhados.
- Não é possível revogar o acesso ao PVC de origem removendo o `shareToNamespace` ou `shareFromNamespace` anotações ou exclusão das `TridentVolumeReference` CR. Para revogar o acesso, você deve excluir o PVC subordinado.
- Não é possível criar snapshots, clones ou espelhar volumes subordinados.

Para maiores informações

Para saber mais sobre acesso a volumes entre namespaces:

- Visita"Compartilhamento de volumes entre namespaces: Dê as boas-vindas ao acesso a volumes entre namespaces." .
- Veja a demonstração em"NetAppTV" .

Clonar volumes entre namespaces

Usando o Trident, você pode criar novos volumes utilizando volumes existentes ou snapshots de volumes de um namespace diferente dentro do mesmo cluster Kubernetes.

Pré-requisitos

Antes de clonar volumes, certifique-se de que os backends de origem e destino sejam do mesmo tipo e tenham a mesma classe de armazenamento.

> A clonagem entre namespaces é suportada apenas para o `ontap-san` e `ontap-nas` drivers de armazenamento. Clones somente leitura não são suportados.

Início rápido

Você pode configurar a clonagem de volumes em apenas alguns passos.

1. **Configure o PVC de origem para clonar o volume.**

O proprietário do namespace de origem concede permissão para acessar os dados no PVC de origem.

2 **Conceder permissão para criar uma solicitação de configuração (CR) no espaço de nomes de destino.**

O administrador do cluster concede permissão ao proprietário do namespace de destino para criar o CR TridentVolumeReference.

3 **Crie uma TridentVolumeReference no namespace de destino.**

O proprietário do namespace de destino cria o CR TridentVolumeReference para se referir ao PVC de origem.

4 **Crie o PVC clonado no namespace de destino.**

O proprietário do namespace de destino cria um PVC para clonar o PVC do namespace de origem.

Configure os namespaces de origem e destino.

Para garantir a segurança, a clonagem de volumes entre namespaces requer a colaboração e a ação do proprietário do namespace de origem, do administrador do cluster e do proprietário do namespace de destino. A função do usuário é definida em cada etapa.

Passos

1. **Proprietário do namespace de origem:** Criar o PVC(`pvc1`) no espaço de nomes de origem(`namespace1`) que concede permissão para compartilhar com o espaço de nomes de destino(`namespace2`) usando o `cloneToNamespace` anotação.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/cloneToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

   A Trident cria o PV e seu volume de armazenamento de backend.

- Você pode compartilhar o PVC com vários namespaces usando uma lista separada por vírgulas. Por exemplo, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4` .
- Você pode compartilhar com todos os namespaces usando `*` . Por exemplo, `trident.netapp.io/cloneToNamespace: *`
- Você pode atualizar o PVC para incluir o `cloneToNamespace` Anotações a qualquer momento.

2. **Administrador do cluster:** Certifique-se de que o RBAC adequado esteja implementado para conceder permissão ao proprietário do namespace de destino para criar o CR TridentVolumeReference no namespace de destino.(`namespace2`).
3. **Proprietário do namespace de destino:** Crie uma solicitação de configuração (CR) TridentVolumeReference no namespace de destino que faça referência ao namespace de origem. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Proprietário do namespace de destino:** Criar um PVC(`pvc2`) no espaço de nomes de destino(`namespace2`) usando o `cloneFromPVC` ou `cloneFromSnapshot` , e `cloneFromNamespace` Anotações para designar o PVC de origem.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/cloneFromPVC: pvc1
    trident.netapp.io/cloneFromNamespace: namespace1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Limitações

- Para PVCs provisionadas usando drivers ontap-nas-economy, clones somente leitura não são suportados.

Replicar volumes usando o SnapMirror

O Trident suporta relações de espelhamento entre um volume de origem em um cluster e o volume de destino no cluster emparelhado para replicação de dados em casos de recuperação de desastres. Você pode usar uma Definição de Recurso Personalizado (CRD) com namespace, chamada Trident Mirror Relationship (TMR), para executar as seguintes operações:

- Criar relações de espelhamento entre volumes (PVCs)
- Remover relações de espelhamento entre volumes
- Quebre os relacionamentos espelhados
- Promover o volume secundário em situações de desastre (falhas).
- Realizar a transição sem perda de dados de aplicações de um cluster para outro (durante failovers ou migrações planejadas).

Pré-requisitos para replicação

Certifique-se de que os seguintes pré-requisitos sejam atendidos antes de começar:

Clusters ONTAP

- * Trident*: A versão 22.10 ou posterior do Trident deve estar presente nos clusters Kubernetes de origem e destino que utilizam o ONTAP como backend.
- **Licenças**: As licenças assíncronas do ONTAP SnapMirror que utilizam o pacote Data Protection devem estar habilitadas nos clusters ONTAP de origem e destino. Consulte "Visão geral do licenciamento do SnapMirror no ONTAP" para mais informações.

  A partir do ONTAP 9.10.1, todas as licenças são fornecidas como um arquivo de licença NetApp (NLF), que é um único arquivo que habilita vários recursos. Consulte"Licenças incluídas no ONTAP One" para mais informações.

  > Somente a proteção assíncrona SnapMirror é suportada.

Espionagem

- **Cluster e SVM**: Os backends de armazenamento ONTAP devem estar interligados. Consulte "Visão geral do peering de clusters e SVMs" para mais informações.

  > Certifique-se de que os nomes SVM usados na relação de replicação entre dois clusters ONTAP sejam únicos.

- * Trident e SVM*: Os SVMs remotos emparelhados devem estar disponíveis para o Trident no cluster de destino.

Motoristas com suporte

O NetApp Trident oferece suporte à replicação de volumes com a tecnologia NetApp SnapMirror, utilizando classes de armazenamento com suporte dos seguintes drivers: **`ontap-nas` : NFS** `ontap-san` : iSCSI

`ontap-san` : FC `ontap-san` : NVMe/TCP (requer versão mínima do ONTAP 9.15.1)

> A replicação de volumes usando o SnapMirror não é compatível com sistemas ASA r2. Para obter informações sobre sistemas ASA r2, consulte"Saiba mais sobre os sistemas de armazenamento ASA r2" .

Crie um PVC espelhado

Siga estes passos e utilize os exemplos de CRD para criar uma relação de espelhamento entre os volumes primário e secundário.

Passos

1. Execute os seguintes passos no cluster Kubernetes primário:
   a. Crie um objeto StorageClass com o `trident.netapp.io/replication: true` parâmetro.

      Exemplo

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        backendType: "ontap-nas"
        fsType: "nfs"
        trident.netapp.io/replication: "true"
      ```

   b. Crie um PVC com uma StorageClass criada anteriormente.

      Exemplo

      ```
      kind: PersistentVolumeClaim
      apiVersion: v1
      metadata:
        name: csi-nas
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        storageClassName: csi-nas
      ```

   c. Crie uma relação de espelhamento (MirrorRelationship) com informações locais.

Exemplo

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

O Trident obtém as informações internas do volume e o estado atual de proteção de dados (DP) do volume e, em seguida, preenche o campo de status do MirrorRelationship.

d. Obtenha o CR TridentMirrorRelationship para obter o nome interno e o SVM do PVC.

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. Execute os seguintes passos no cluster Kubernetes secundário:

   a. Crie uma StorageClass com o parâmetro trident.netapp.io/replication: true.

Exemplo

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

b. Crie um relacionamento de espelhamento (MirrorRelationship) com informações de destino e origem.

Exemplo

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
  - localPVCName: csi-nas
    remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

O Trident criará um relacionamento SnapMirror com o nome da política de relacionamento configurada (ou padrão para ONTAP) e o inicializará.

c. Crie um PVC com uma StorageClass previamente criada para atuar como destino secundário (SnapMirror).

Exemplo

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

O Trident verificará a existência do CRD TridentMirrorRelationship e não conseguirá criar o volume caso o relacionamento não exista. Caso exista a relação, o Trident garantirá que o novo FlexVol volume seja colocado em uma SVM que esteja emparelhada com a SVM remota definida no MirrorRelationship.

Estados de replicação de volume

Uma relação de espelhamento Trident (TMR, do inglês Trident Mirror Relationship) é um CRD (Componente de Referência Completa) que representa uma das extremidades de uma relação de replicação entre PVCs (Componentes Variáveis Parciais). O TMR de destino possui um estado, que informa ao Trident qual é o estado desejado. O TMR de destino possui os seguintes estados:

- **Estabelecido**: o PVC local é o volume de destino de uma relação de espelhamento, e esta é uma nova relação.
- **Em destaque**: o PVC local é ReadWrite e pode ser montado, sem nenhuma relação de espelhamento em vigor no momento.
- **Reestabelecido**: o PVC local é o volume de destino de uma relação de espelhamento e também estava anteriormente nessa relação de espelhamento.
  - O estado restabelecido deve ser usado se o volume de destino já teve alguma relação com o volume de origem, pois ele sobrescreve o conteúdo do volume de destino.
  - O estado restabelecido falhará se o volume não estiver previamente relacionado com a fonte.

Promover o PVC secundário durante uma falha não planejada.

Execute a seguinte etapa no cluster Kubernetes secundário:

- Atualize o campo *spec.state* de TridentMirrorRelationship para `promoted` .

Promover o PVC secundário durante uma falha planejada

Durante uma migração (failover) planejada, execute os seguintes passos para promover o PVC secundário:

Passos

1. No cluster Kubernetes primário, crie um snapshot do PVC e aguarde até que o snapshot seja criado.
2. No cluster Kubernetes primário, crie o CR SnapshotInfo para obter detalhes internos.

   Exemplo

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. No cluster Kubernetes secundário, atualize o campo *spec.state* do CR *TridentMirrorRelationship* para *promoted* e o campo *spec.promotedSnapshotHandle* para o nome interno do snapshot.
4. No cluster Kubernetes secundário, confirme o status (campo status.state) do TridentMirrorRelationship como promovido.

Restaurar uma relação de espelhamento após uma falha.

Antes de restaurar a relação de espelhamento, escolha o lado que deseja definir como o novo lado principal.

Passos

1. No cluster Kubernetes secundário, verifique se os valores do campo *spec.remoteVolumeHandle* no TridentMirrorRelationship estão atualizados.
2. No cluster Kubernetes secundário, atualize o campo *spec.mirror* do TridentMirrorRelationship para `reestablished`.

Operações adicionais

O Trident suporta as seguintes operações nos volumes primário e secundário:

Replicar o PVC primário em um novo PVC secundário.

Certifique-se de já possuir um tubo de PVC primário e um tubo de PVC secundário.

Passos

1. Exclua os CRDs PersistentVolumeClaim e TridentMirrorRelationship do cluster secundário (de destino) estabelecido.
2. Exclua o CRD TridentMirrorRelationship do cluster primário (de origem).
3. Crie um novo CRD TridentMirrorRelationship no cluster primário (de origem) para o novo PVC secundário (de destino) que você deseja estabelecer.

Redimensionar um PVC espelhado, primário ou secundário

O PVC pode ser redimensionado normalmente; o ONTAP expandirá automaticamente qualquer flexbox de destino se a quantidade de dados exceder o tamanho atual.

Remover a replicação de um PVC

Para remover a replicação, execute uma das seguintes operações no volume secundário atual:

- Exclua o MirrorRelationship no PVC secundário. Isso rompe a relação de replicação.
- Ou, atualize o campo spec.state para *promoted*.

Excluir um PVC (que foi previamente espelhado)

O Trident verifica se há PVCs replicados e libera a relação de replicação antes de tentar excluir o volume.

Excluir um TMR

A exclusão de um TMR em um dos lados de um relacionamento espelhado faz com que o TMR restante passe para o estado *promovido* antes que o Trident conclua a exclusão. Se o TMR selecionado para exclusão já estiver no estado *promovido*, não haverá relação de espelhamento existente e o TMR será removido, e o Trident promoverá o PVC local para *Leitura/Gravação*. Essa exclusão libera os metadados do SnapMirror para o volume local no ONTAP. Caso este volume seja utilizado em uma relação de espelhamento no futuro, será necessário usar um novo TMR com um estado de replicação de volume *estabelecido* ao criar a nova relação de espelhamento.

Atualize os relacionamentos de espelhamento quando o ONTAP estiver online.

As relações de espelhamento podem ser atualizadas a qualquer momento após serem estabelecidas. Você pode usar o `state: promoted` ou `state: reestablished` campos para atualizar os relacionamentos. Ao promover um volume de destino para um volume ReadWrite regular, você pode usar *promotedSnapshotHandle* para especificar um snapshot específico para restaurar o volume atual.

Atualizar relações de espelhamento quando o ONTAP estiver offline

Você pode usar um CRD para executar uma atualização do SnapMirror sem que o Trident tenha conectividade direta com o cluster ONTAP . Consulte o seguinte exemplo de formato do TridentActionMirrorUpdate:

Exemplo

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state`Reflete o estado do CRD TridentActionMirrorUpdate. Pode assumir um valor entre *Succeeded*, *In Progress* ou *Failed*.

Utilizar a topologia CSI

O Trident pode criar e anexar volumes seletivamente aos nós presentes em um cluster Kubernetes, utilizando o "Recurso de topologia CSI" .

Visão geral

Usando o recurso Topologia CSI, o acesso aos volumes pode ser limitado a um subconjunto de nós, com base em regiões e zonas de disponibilidade. Atualmente, os provedores de nuvem permitem que os administradores do Kubernetes criem nós baseados em zonas. Os nós podem estar localizados em diferentes zonas de disponibilidade dentro de uma região ou em várias regiões. Para facilitar o provisionamento de volumes para cargas de trabalho em uma arquitetura multizona, o Trident usa a topologia CSI.

Saiba mais sobre o recurso de Topologia CSI "aqui" .

O Kubernetes oferece dois modos exclusivos de vinculação de volumes:

- Com `VolumeBindingMode` definido para `Immediate` O Trident cria o volume sem qualquer conhecimento de topologia. A vinculação de volumes e o provisionamento dinâmico são tratados quando o PVC é criado. Este é o padrão `VolumeBindingMode` e é adequado para clusters que não impõem restrições de topologia. Volumes persistentes são criados sem qualquer dependência dos requisitos de agendamento do pod solicitante.
- Com `VolumeBindingMode` definido para `WaitForFirstConsumer` A criação e vinculação de um Volume Persistente para um PVC é adiada até que um pod que utilize o PVC seja agendado e criado. Dessa forma, os volumes são criados para atender às restrições de agendamento impostas pelos requisitos de topologia.

O `WaitForFirstConsumer` O modo de vinculação não requer rótulos de topologia. Isso pode ser usado independentemente do recurso de Topologia CSI.

O que você vai precisar

Para utilizar a topologia CSI, você precisa do seguinte:

- Um cluster Kubernetes executando um"versão do Kubernetes suportada"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Os nós do cluster devem ter rótulos que introduzam informações sobre a topologia.(`topology.kubernetes.io/region` e `topology.kubernetes.io/zone`). Essas etiquetas **devem estar presentes nos nós do cluster** antes da instalação do Trident para que ele reconheça a topologia.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Passo 1: Crie um backend com reconhecimento de topologia

Os sistemas de armazenamento Trident podem ser configurados para provisionar volumes seletivamente com base em zonas de disponibilidade. Cada backend pode conter um opcional `supportedTopologies` Bloco que representa uma lista de zonas e regiões suportadas. Para StorageClasses que fazem uso desse backend, um volume só seria criado se solicitado por um aplicativo agendado em uma região/zona suportada.

Aqui está um exemplo de definição de backend:

YAML

```yaml
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-a
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-b
```

JSON

```json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-a"
    },
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-b"
    }
  ]
}
```

`supportedTopologies`É utilizado para fornecer uma lista de regiões e zonas por backend. Essas regiões e zonas representam a lista de valores permitidos que podem ser fornecidos em uma StorageClass. Para StorageClasses que contêm um subconjunto das regiões e zonas fornecidas em um backend, o Trident cria um volume no backend.

Você pode definir `supportedTopologies` por pool de armazenamento também. Veja o exemplo a seguir:

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-b
```

Neste exemplo, o `region` e `zone` As etiquetas indicam a localização da piscina de armazenamento. `topology.kubernetes.io/region` e `topology.kubernetes.io/zone` ditam de onde os pools de armazenamento podem ser consumidos.

Etapa 2: Defina StorageClasses que levem em consideração a topologia.

Com base nos rótulos de topologia fornecidos aos nós do cluster, as StorageClasses podem ser definidas para conter informações de topologia. Isso determinará os pools de armazenamento que servem como candidatos para solicitações de PVC feitas e o subconjunto de nós que podem usar os volumes provisionados pelo Trident.

Veja o exemplo a seguir:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata: null
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions: null
  - key: topology.kubernetes.io/zone
    values:
      - us-east1-a
      - us-east1-b
  - key: topology.kubernetes.io/region
    values:
      - us-east1
parameters:
  fsType: ext4
```

Na definição de StorageClass fornecida acima, `volumeBindingMode` está definido para `WaitForFirstConsumer` . Os PVCs solicitados com esta StorageClass não serão processados até que sejam referenciados em um pod. E, `allowedTopologies` Fornece as zonas e a região a serem utilizadas. O `netapp-san-us-east1` O StorageClass cria PVCs no `san-backend-us-east1` backend definido acima.

Etapa 3: Criar e usar um PVC

Com a StorageClass criada e mapeada para um backend, agora você pode criar PVCs.

Veja o exemplo `spec` abaixo:

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

A criação de um PVC usando este manifesto resultaria no seguinte:

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type    Reason                Age   From                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

Para que o Trident crie um volume e o fixe ao PVC, utilize o PVC em um invólucro. Veja o exemplo a seguir:

```
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Este podSpec instrui o Kubernetes a agendar o pod em nós que estejam presentes no `us-east1` região, e escolha qualquer nó que esteja presente na `us-east1-a` ou `us-east1-b` zonas.

Veja a seguinte saída:

```
kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

Atualizar os backends para incluir `supportedTopologies`

Os backends preexistentes podem ser atualizados para incluir uma lista de `supportedTopologies` usando `tridentctl backend update` . Isso não afetará os volumes que já foram provisionados e será usado apenas para PVCs subsequentes.

Encontre mais informações

- "Gerenciar recursos para contêineres"
- "seletor de nó"
- "Afinidade e antiafinidade"
- "Manchas e Tolerâncias"

Trabalhar com instantâneos

Os snapshots de volumes persistentes (PVs) do Kubernetes permitem a criação de cópias pontuais dos volumes. Você pode criar um snapshot de um volume criado usando o Trident, importar um snapshot criado fora do Trident, criar um novo volume a partir de um snapshot existente e recuperar dados de volumes a partir de snapshots.

Visão geral

O snapshot de volume é suportado por `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , e `google-cloud-netapp-volumes` motoristas.

Antes de começar

Para trabalhar com snapshots, você precisa de um controlador de snapshots externo e definições de recursos personalizados (CRDs). Essa é a responsabilidade do orquestrador do Kubernetes (por exemplo: Kubeadm, GKE, OpenShift).

Se a sua distribuição Kubernetes não incluir o controlador de snapshots e os CRDs, consulteImplante um controlador de instantâneo de volume .

> Não crie um controlador de snapshots se estiver criando snapshots de volume sob demanda em um ambiente GKE. O GKE utiliza um controlador de snapshots integrado e oculto.

Criar um instantâneo de volume

Passos

1. Criar um `VolumeSnapshotClass` Para obter mais informações, consulte"VolumeSnapshotClass" .
   - O `driver` Aponta para o motorista do Trident CSI.
   - `deletionPolicy`pode ser `Delete` ou `Retain` . Quando definido para `Retain` , o snapshot físico subjacente no cluster de armazenamento é mantido mesmo quando o `VolumeSnapshot` O objeto foi excluído.

     Exemplo

     ```
     cat snap-sc.yaml
     ```

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

2. Crie uma imagem instantânea de um tubo de PVC existente.

   Exemplos

   - Este exemplo cria um instantâneo de um PVC existente.

     ```
     cat snap.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Este exemplo cria um objeto de instantâneo de volume para um PVC chamado `pvc1` e o nome do instantâneo está definido como `pvc1-snap` . Um VolumeSnapshot é análogo a um PVC e está associado a um `VolumeSnapshotContent` objeto que representa a captura instantânea real.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Você pode identificar o `VolumeSnapshotContent` objeto para o `pvc1-snap` O VolumeSnapshot é descrito dessa forma. O `Snapshot Content Name` Identifica o objeto VolumeSnapshotContent que fornece este snapshot. O `Ready To Use` O parâmetro indica que o instantâneo pode ser usado para criar um novo PVC.

```
kubectl describe volumesnapshots pvc1-snap
Name:         pvc1-snap
Namespace:    default
...
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:       PersistentVolumeClaim
    Name:       pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
...
```

Criar um PVC a partir de um instantâneo de volume

Você pode usar `dataSource` para criar um PVC usando um VolumeSnapshot chamado `<pvc-name>` como fonte dos dados. Após a fabricação do PVC, ele pode ser fixado a um invólucro e utilizado como qualquer outro PVC.

> O PVC será criado no mesmo backend que o volume de origem. Consulte"KB: A criação de um PVC a partir de um Snapshot de PVC do Trident não pode ser feita em um backend alternativo." .

O exemplo a seguir cria o PVC usando `pvc1-snap` como fonte de dados.

```
cat pvc-from-snap.yaml
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importar um instantâneo de volume

A Trident apoia o"Processo de snapshot pré-provisionado do Kubernetes" para permitir que o administrador do cluster crie um `VolumeSnapshotContent` snapshots de objetos e importações criados fora do Trident.

Antes de começar

O Trident deve ter criado ou importado o volume pai do snapshot.

Passos

1. **Administrador do cluster:** Criar um `VolumeSnapshotContent` objeto que faz referência ao snapshot do backend. Isso inicia o fluxo de trabalho de captura de instantâneos no Trident.
   - Especifique o nome do snapshot do backend em `annotations` como `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">` .
   - Especificar `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` em `snapshotHandle` Esta é a única informação fornecida ao Trident pelo snapshotter externo. `ListSnapshots` chamar.

     > O `<volumeSnapshotContentName>` Nem sempre é possível que o nome do snapshot do backend corresponda ao nome original devido a restrições de nomenclatura do CR.

     Exemplo

     O exemplo a seguir cria um `VolumeSnapshotContent` objeto que faz referência a um snapshot do backend `snap-01` .

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01"  # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Administrador do cluster:** Crie o `VolumeSnapshot` CR que faz referência ao `VolumeSnapshotContent` objeto. Este pedido permite o acesso para usar o `VolumeSnapshot` em um determinado espaço de nomes.

   Exemplo

   O exemplo a seguir cria um `VolumeSnapshot` CR nomeado `import-snap` que faz referência ao `VolumeSnapshotContent` nomeado `import-snap-content` .

   ```
   apiVersion: snapshot.storage.k8s.io/v1
   kind: VolumeSnapshot
   metadata:
     name: import-snap
   spec:
     # volumeSnapshotClassName: csi-snapclass (not required for pre-
   provisioned or imported snapshots)
     source:
       volumeSnapshotContentName: import-snap-content
   ```

3. **Processamento interno (nenhuma ação necessária):** O snapshotter externo reconhece o recém-criado `VolumeSnapshotContent` e executa o `ListSnapshots` chamar. Trident cria o `TridentSnapshot` .
   - O capturador de instantâneos externo define o `VolumeSnapshotContent` para `readyToUse` e o `VolumeSnapshot` para `true` .
   - Trident retorna `readyToUse=true` .
4. **Qualquer usuário:** Criar um `PersistentVolumeClaim` para fazer referência ao novo `VolumeSnapshot` , onde o `spec.dataSource` (ou `spec.dataSourceRef`) o nome é o `VolumeSnapshot` nome.

Exemplo

O exemplo a seguir cria um PVC que faz referência a `VolumeSnapshot` nomeado `import-snap` .

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Recupere dados de volume usando snapshots

O diretório de snapshots fica oculto por padrão para facilitar a máxima compatibilidade dos volumes provisionados usando o `ontap-nas` e `ontap-nas-economy` motoristas. Ative o `.snapshot` diretório para recuperar dados diretamente de snapshots.

Utilize o comando `volume snapshot restore` da CLI do ONTAP para restaurar um volume a um estado registrado em um snapshot anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Ao restaurar uma cópia de snapshot, a configuração de volume existente é sobrescrita. As alterações feitas nos dados do volume após a criação da cópia instantânea serão perdidas.

Restauração de volume in-place a partir de um snapshot

O Trident proporciona restauração de volume rápida e in situ a partir de uma imagem instantânea usando o `TridentActionSnapshotRestore` (TASR) CR. Essa solicitação de configuração (CR) funciona como uma ação imperativa do Kubernetes e não persiste após a conclusão da operação.

O Trident suporta a restauração de snapshots no `ontap-san` , `ontap-san-economy` , `ontap-nas` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` , `google-cloud-netapp-volumes` , e `solidfire-san` motoristas.

Antes de começar

Você precisa ter um PVC encadernado e um snapshot de volume disponível.

- Verifique se o status do PVC está vinculado.

```
kubectl get pvc
```

- Verifique se o snapshot do volume está pronto para uso.

```
kubectl get vs
```

Passos

1. Crie o TASR CR. Este exemplo cria um CR para PVC `pvc1` e instantâneo de volume `pvc1-snapshot` .

   O TASR CR deve estar em um namespace onde o PVC e o VS existam.

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Aplique a solicitação de restauração (CR) para restaurar a partir do snapshot. Este exemplo restaura a partir de um snapshot. `pvc1` .

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Resultados

O Trident restaura os dados a partir do snapshot. Você pode verificar o status da restauração do snapshot:

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- Na maioria dos casos, o Trident não tentará novamente a operação automaticamente em caso de falha. Você precisará repetir a operação.
- Usuários do Kubernetes sem acesso de administrador podem precisar de permissão do administrador para criar um CR do TASR em seu namespace de aplicação.

Excluir um PV com snapshots associados

Ao excluir um Volume Persistente com snapshots associados, o volume Trident correspondente é atualizado para o estado "Excluindo". Remova os snapshots de volume para excluir o volume Trident .

Implante um controlador de instantâneo de volume

Se a sua distribuição Kubernetes não incluir o controlador de snapshots e os CRDs, você pode implantá-los da seguinte forma.

Passos

1. Criar CRDs de snapshot de volume.

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yam
l
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Crie o controlador de instantâneo.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

Se necessário, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` e atualizar `namespace` para o seu espaço de nomes.

Links relacionados

- "Instantâneos de volume"
- "VolumeSnapshotClass"

Trabalhar com snapshots de grupos de volumes

O NetApp Trident permite criar snapshots de vários volumes (um grupo de snapshots de volumes) em grupos de volumes do Kubernetes. Este instantâneo do grupo de volumes representa cópias de vários volumes feitas no mesmo momento.

VolumeGroupSnapshot é um recurso beta do Kubernetes com APIs beta. O Kubernetes 1.32 é a versão mínima necessária para o VolumeGroupSnapshot.

Criar snapshots de grupos de volumes

O snapshot do grupo de volumes é compatível com o `ontap-san` Driver compatível apenas com o protocolo

iSCSI, ainda não suportado com Fibre Channel (FCP) nem NVMe/TCP. Antes de começar

- Certifique-se de que sua versão do Kubernetes seja K8s 1.32 ou superior.
- Para trabalhar com snapshots, você precisa de um controlador de snapshots externo e definições de recursos personalizados (CRDs). Essa é a responsabilidade do orquestrador do Kubernetes (por exemplo: Kubeadm, GKE, OpenShift).

  Se a sua distribuição Kubernetes não incluir o controlador de snapshots externo e os CRDs, consulteImplante um controlador de instantâneo de volume .

  > Não crie um controlador de snapshots se estiver criando snapshots de grupos de volumes sob demanda em um ambiente GKE. O GKE utiliza um controlador de snapshots integrado e oculto.

- No arquivo YAML do controlador de snapshots, defina o `CSIVolumeGroupSnapshot` Defina o recurso como 'true' para garantir que o snapshot do grupo de volumes esteja habilitado.
- Crie as classes de snapshot de grupo de volumes necessárias antes de criar um snapshot de grupo de volumes.
- Certifique-se de que todos os PVCs/volumes estejam no mesmo SVM para poder criar o VolumeGroupSnapshot.

Passos

- Crie uma classe VolumeGroupSnapshotClass antes de criar um VolumeGroupSnapshot. Para obter mais informações, consulte"VolumeGroupSnapshotClass" .

```
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
```

- Crie PVCs com os rótulos necessários usando as classes de armazenamento existentes ou adicione esses rótulos aos PVCs existentes.

  O exemplo a seguir cria o PVC usando `pvc1-group-snap` como fonte de dados e rótulo `consistentGroupSnapshot: groupA` . Defina a chave e o valor do rótulo com base em suas necessidades.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- Crie um VolumeGroupSnapshot com o mesmo rótulo.(`consistentGroupSnapshot: groupA`) especificado no PVC.

  Este exemplo cria um instantâneo de um grupo de volumes:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Recupere dados de volume usando um instantâneo de grupo.

Você pode restaurar Volumes Persistentes individuais usando os snapshots individuais que foram criados como parte do Snapshot do Grupo de Volumes. Não é possível recuperar o Snapshot do Grupo de Volumes como uma unidade.

Utilize o comando `volume snapshot restore` da CLI do ONTAP para restaurar um volume a um estado registrado em um snapshot anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Ao restaurar uma cópia de snapshot, a configuração de volume existente é sobrescrita. As alterações feitas nos dados do volume após a criação da cópia instantânea serão perdidas.

Restauração de volume in-place a partir de um snapshot

O Trident proporciona restauração de volume rápida e in situ a partir de uma imagem instantânea usando o `TridentActionSnapshotRestore` (TASR) CR. Essa solicitação de configuração (CR) funciona como uma ação imperativa do Kubernetes e não persiste após a conclusão da operação.

Para mais informações, consulte "Restauração de volume in-place a partir de um snapshot".

Excluir um PV com snapshots de grupo associados

Ao excluir um instantâneo de volume de grupo:

- Você pode excluir o grupo de snapshots VolumeGroupSnapshots como um todo, e não snapshots individuais dentro do grupo.
- Se os PersistentVolumes forem excluídos enquanto existir um snapshot para esse PersistentVolume, o Trident moverá esse volume para um estado de "exclusão", pois o snapshot deve ser removido antes que o volume possa ser removido com segurança.
- Se um clone tiver sido criado usando um snapshot agrupado e, em seguida, o grupo for excluído, uma operação de divisão no clone será iniciada e o grupo não poderá ser excluído até que a divisão seja concluída.

Implante um controlador de instantâneo de volume

Se a sua distribuição Kubernetes não incluir o controlador de snapshots e os CRDs, você pode implantá-los da seguinte forma.

Passos

1. Criar CRDs de snapshot de volume.

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.
yaml
```

2. Crie o controlador de instantâneo.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

Se necessário, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` e atualizar `namespace` para o seu espaço de nomes.

Links relacionados

- "VolumeGroupSnapshotClass"
- "Instantâneos de volume"

Gerenciar e monitorar o Trident

Trident aprimorado

Trident aprimorado

A partir da versão 24.02, o Trident segue um ciclo de lançamentos de quatro meses, disponibilizando três versões principais a cada ano civil. Cada nova versão se baseia nas versões anteriores e oferece novos recursos, melhorias de desempenho, correções de bugs e aprimoramentos. Recomendamos que você atualize pelo menos uma vez por ano para aproveitar os novos recursos do Trident.

Considerações antes de atualizar

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

- Deve haver apenas uma instância do Trident instalada em todos os namespaces de um determinado cluster Kubernetes.
- O Trident 23.07 e versões posteriores exigem snapshots de volume v1 e não são mais compatíveis com snapshots alfa ou beta.
- Se você criou o Cloud Volumes Service para o Google Cloud no"Tipo de serviço CVS" Você precisa atualizar a configuração do backend para usar o `standardsw` ou `zoneredundantstandardsw` Nível de serviço ao atualizar do Trident 23.01. A falha na atualização do `serviceLevel` Problemas no backend podem causar falhas nos volumes. Consulte "Exemplos de tipos de serviço da CVS" para mais detalhes.
- Ao fazer o upgrade, é importante que você forneça `parameter.fsType` em `StorageClasses` Usado pela Trident. Você pode excluir e recriar `StorageClasses` sem interromper os volumes preexistentes.
  - Este é um **requisito** para a aplicação da lei. "contextos de segurança" para volumes SAN.
  - O diretório https://github.com/NetApp/trident/tree/master/trident-installer/sample-input [sample input^] contém exemplos, como `storage-class-basic.yaml.templ` e xref:./trident-managing-k8s/`storage-class-bronze-default.yaml` .
  - Para obter mais informações, consulte"Problemas conhecidos" .

Passo 1: Selecione uma versão

As versões do Trident seguem um padrão baseado em datas. `YY.MM` convenção de nomenclatura, onde "AA" são os dois últimos dígitos do ano e "MM" é o mês. Os lançamentos do Dot seguem um `YY.MM.X` convenção, onde "X" é o nível de patch. Você selecionará a versão para a qual deseja atualizar com base na versão atual.

- Você pode realizar uma atualização direta para qualquer versão de destino que esteja dentro de um intervalo de quatro versões da sua versão instalada. Por exemplo, você pode atualizar diretamente da versão 24.06 (ou qualquer versão 24.06 com ponto) para a versão 25.06.
- Se você estiver atualizando de uma versão fora do período de quatro versões, execute uma atualização em várias etapas. Use as instruções de atualização para o"versão anterior" Você está atualizando para a versão mais recente que se encaixa no período de quatro versões. Por exemplo, se você estiver executando a versão 23.07 e quiser atualizar para a versão 25.06:
  a. Primeira atualização da versão 23.07 para a 24.06.
  b. Em seguida, atualize da versão 24.06 para a 25.06.

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

Etapa 2: Determine o método de instalação original.

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

1. Usar `kubectl get pods -n trident` para examinar as cápsulas.
   - Se não houver um módulo de operador, o Trident foi instalado usando `tridentctl` .
   - Caso exista um pod de operador, o Trident foi instalado usando o operador Trident, seja manualmente ou através do Helm.
2. Se houver um módulo de operador, use-o. `kubectl describe torc` Para determinar se o Trident foi instalado usando o Helm.
   - Se houver uma etiqueta "Helm", o Trident foi instalado usando o Helm.
   - Se não houver nenhuma etiqueta "Helm", o Trident foi instalado manualmente usando o operador Trident .

Etapa 3: Selecione um método de atualização

Geralmente, você deve atualizar usando o mesmo método que usou para a instalação inicial; no entanto, você pode"alternar entre métodos de instalação" . Existem duas opções para aprimorar o Trident.

- "Faça o upgrade usando o operador Trident ."

Sugerimos que você revise"Entenda o fluxo de trabalho de atualização do operador." antes de atualizar com a operadora.

*

Atualize com a operadora

Entenda 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 do controlador e nos Pods dos nós, bem como no DaemonSet dos nós, que permitem atualizações contínuas.

Gestão de atualização do operador Trident

Um dos muitos"Benefícios de usar o operador Trident" A instalação e atualização do Trident consiste no gerenciamento automático de objetos Trident e Kubernetes sem interromper os volumes montados existentes. Dessa forma, a Trident pode suportar atualizações sem tempo de inatividade, ou"*atualizações contínuas*" . Em particular, o operador Trident comunica-se com o cluster Kubernetes para:

- Exclua e recrie a implantação do Trident Controller e o DaemonSet do nó.

- Substitua os pods do controlador Trident e do nó Trident pelas novas versões.
  - Se um nó não for atualizado, isso não impede que os nós restantes sejam atualizados.
  - Somente nós com um Trident Node Pod em execução podem montar volumes.

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

Fluxo de trabalho de atualização do operador

Ao iniciar 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 do Kubernetes, incluindo CRDs, RBAC e Trident SVC.
   c. Exclui a implantação do Trident Controller para a versão *n*.
   d. Cria a implantação do Trident Controller para a versão *n+1*.
2. **Kubernetes** cria um Pod do Controlador Trident para *n+1*.
3. O *operador Trident *:
   a. Exclui o DaemonSet do nó Trident para *n*. O operador não aguarda o término do Node Pod.
   b. Cria o Daemonset Trident Node para *n+1*.
4. O **Kubernetes** cria Pods de nós Trident em nós que não executam o Pod de nó Trident *n*. Isso garante que nunca haja mais de um Trident Node Pod, de qualquer versão, em um nó.

Atualize uma instalação do Trident usando o operador Trident ou o Helm.

Você pode atualizar o Trident usando o operador Trident, seja manualmente ou usando o Helm. Você pode atualizar de uma instalação do operador Trident para outra instalação do operador Trident ou atualizar de uma para outra. `tridentctl` Instalação em uma versão de operador Trident . Análise"Selecione um método de atualização" antes de atualizar a instalação do operador Trident .

Atualize uma instalação manual.

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

Para atualizar a partir do Trident instalado usando o operador com escopo de namespace (versões 20.07 a 20.10), utilize as instruções de atualização para"sua versão instalada" do 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 executando Kubernetes 1.24, use"bundle_pre_1_25.yaml" .

- Para clusters executando Kubernetes 1.25 ou posterior, use"bundle_post_1_25.yaml" .

Antes de começar

Certifique-se de estar usando um cluster Kubernetes em execução."uma versão do Kubernetes compatível" .

Passos

1. Verifique sua versão do Trident :

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

2. Atualizar o `operator.yaml` , `tridentorchestrator_cr.yaml` , e `post_1_25_bundle.yaml` com o registro e os caminhos das imagens da versão para a qual você está atualizando (ex: 25.06) e a chave secreta correta.
3. Exclua o operador Trident que foi usado para instalar a instância atual do Trident . Por exemplo, se você estiver atualizando da versão 25.02, execute o seguinte comando:

```
kubectl delete -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

4. 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 espelhados do Trident e do CSI para o modo offline, ativar logs de depuração ou especificar segredos de extração de imagens.
5. Instale o Trident usando o arquivo YAML correto para o seu ambiente, onde *<bundle.yaml>* é `bundle_pre_1_25.yaml` ou `bundle_post_1_25.yaml` com base na sua versão do Kubernetes. Por exemplo, se você estiver instalando o Trident 25.06.0, execute o seguinte comando:

```
kubectl create -f 25.06.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

6. Edite o torque do tridente para incluir a imagem 25.06.0.

Atualize uma instalação do Helm

Você pode atualizar uma instalação do Trident Helm.

> Ao atualizar um cluster Kubernetes da versão 1.24 para a 1.25 ou posterior que tenha o Trident instalado, você deve atualizar o arquivo values.yaml para configurar `excludePodSecurityPolicy` para `true` ou adicione `--set excludePodSecurityPolicy=true` para o `helm upgrade` Execute o comando antes de atualizar o cluster.

Se você já atualizou seu cluster Kubernetes da versão 1.24 para a 1.25 sem atualizar o Trident Helm, a atualização do Helm falhará. Para que a atualização do leme seja concluída, execute estas etapas como pré-requisitos:

1. Instale o plugin helm-mapkubeapis a partir de https://github.com/helm/helm-mapkubeapis .
2. Execute um teste de lançamento do Trident no namespace onde o Trident está instalado. Esta lista contém 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 você"Instalei o Trident usando o Helm." , você pode usar `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` Para atualizar em uma única etapa. Se você não adicionou o repositório Helm ou não consegue usá-lo para atualizar:
   a. Baixe a versão mais recente do Trident em"a seção *Assets* no GitHub" .
   b. Use o `helm upgrade` comando onde `trident-operator-25.06.0.tgz` Reflete a versão para a qual você deseja atualizar.

```
helm upgrade <name> trident-operator-25.06.0.tgz
```

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

2. Correr `helm list` Para verificar se o gráfico e a versão do aplicativo foram atualizados. Correr `tridentctl logs` Para analisar quaisquer mensagens de depuração.

Atualize de um `tridentctl` instalação para o operador Trident

Você pode atualizar para a versão mais recente do operador Trident a partir de um `tridentctl` instalação. Os sistemas de backend e PVCs existentes estarão automaticamente disponíveis.

> Antes de alternar entre os métodos de instalação, revise"Alternando entre métodos de instalação" .

Passos

1. Baixe a versão mais recente do Trident .

```
# Download the release required [25.06.0]
mkdir 25.06.0
cd 25.06.0
wget
https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
installer-25.06.0.tar.gz
tar -xf trident-installer-25.06.0.tar.gz
cd trident-installer
```

2. Crie o `tridentorchestrator` CRD 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. Criar um `TridentOrchestrator` CR para instalação do 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:                v25.06.0
```

Atualize com tridentctl

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

Sobre esta tarefa

Desinstalar e reinstalar o Trident funciona como uma atualização. Ao desinstalar o Trident, o Persistent Volume Claim (PVC) e o Persistent Volume (PV) usados pela implementação do Trident não são excluídos. Os PVs que já foram provisionados permanecerão disponíveis enquanto o Trident estiver offline, e o Trident provisionará volumes para quaisquer PVCs que forem criados nesse período, após o seu retorno à internet.

Antes de começar

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

Passos

1. Execute o comando de desinstalação em `tridentctl` Remover todos os recursos associados ao Trident, exceto os CRDs e objetos relacionados.

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

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

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

Gerencie o Trident usando o tridentctl

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

Comandos e bandeiras globais

Você pode correr `tridentctl help` para obter uma lista dos comandos disponíveis para `tridentctl` ou acrescente o `--help` Adicione a flag a qualquer comando para obter uma lista de opções e parâmetros para esse comando específico.

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

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

Comandos

`create`
Adicionar um recurso ao Trident.

`delete`
Remova um ou mais recursos do Trident.

`get`
Obtenha um ou mais recursos da Trident.

`help`
Ajuda com qualquer comando.

`images`
Imprima uma tabela com as imagens dos contêineres que o Trident precisa.

`import`
Importe um recurso existente para o Trident.

`install`
Instale o Trident.

`logs`
Imprima os registros 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 backend.

`upgrade`
Aprimore 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 o `KUBECONFIG` Caminho para executar comandos localmente ou de um cluster Kubernetes para outro.

Alternativamente, você pode exportar o `KUBECONFIG` variável para apontar para um cluster Kubernetes específico e problema `tridentctl` comandos para esse cluster.

-n, --namespace string
Espaço de nomes da implantação do Trident .

-o, --output string
Formato de saída. Uma das opções: 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 escutar e servir apenas em 127.0.0.1 (para IPv4) ou [::1] (para IPv6).

Opções e parâmetros de comando

criar

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

```
tridentctl create [option]
```

Opções
`backend`Adicionar um backend ao Trident.

excluir

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

```
tridentctl delete [option]
```

Opções
`backend`Excluir um ou mais servidores 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.

pegar

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

```
tridentctl get [option]
```

Opções

`backend`Obtenha um ou mais backends de armazenamento da Trident.
`snapshot` Obtenha uma ou mais fotos instantâneas do Trident.
`storageclass` Obtenha uma ou mais classes de armazenamento da Trident.
`volume` Adquira um ou mais volumes da Trident.

Bandeiras

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

imagens

Usar `images` Bandeiras para imprimir uma tabela com as imagens dos contêineres que o Trident precisa.

```
tridentctl images [flags]
```

Bandeiras

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

volume de importação

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

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

Pseudônimos

`volume`, `v`

Bandeiras

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

instalar

Use o `install` sinalizadores para instalar o Trident.

```
tridentctl install [flags]
```

Bandeiras

`--autosupport-image string`: A imagem do contêiner para Telemetria de Suporte Automático (padrão "netapp/trident autosupport:<versão-atual>").
`--autosupport-proxy string` Endereço/porta de um proxy para envio de telemetria do Autosupport.
`--enable-node-prep` Tentativa de instalação dos pacotes necessários nos nós.
`--generate-custom-yaml` Gere arquivos YAML sem instalar nada.
`-h` , `--help` Ajuda para instalação.
`--http-request-timeout` : Substitui o tempo limite de requisição HTTP para a API REST do controlador Trident (padrão de 1m30s).
`--image-registry string` : O endereço/porta de um registro de imagens interno.
`--k8s-timeout duration` : O tempo limite para todas as operações do Kubernetes (padrão: 3m0s).
`--kubelet-dir string` : O local do host onde o estado interno do kubelet está armazenado (o padrão é "/var/lib/kubelet").
`--log-format string` : O formato de registro do Trident (texto, json) (padrão "texto").
`--node-prep` Permite que o Trident prepare os nós do cluster Kubernetes para gerenciar volumes usando o protocolo de armazenamento de dados especificado. **Atualmente, `iscsi` é o único valor suportado. A partir do OpenShift 4.19, a versão mínima do Trident compatível com este recurso é a 25.06.1.**
`--pv string` : O nome do PV legado usado pelo Trident; certifique-se de que ele não exista (padrão "trident").
`--pvc string` : O nome do PVC legado usado pelo Trident; certifique-se de que ele não exista (padrão "trident").
`--silence-autosupport` : Não enviar pacotes de suporte automático para a NetApp automaticamente (padrão: verdadeiro).
`--silent` Desative a maior parte das saídas durante a instalação.
`--trident-image string` : A imagem do Trident para instalar.
`--k8s-api-qps` : O limite de consultas por segundo (QPS) para solicitações da API do Kubernetes (padrão 100; opcional).
`--use-custom-yaml` Utilize quaisquer arquivos YAML existentes no diretório de configuração.
`--use-ipv6` Utilize IPv6 para a comunicação do Trident.

toras

Usar `logs` sinalizadores para imprimir os registros do Trident.

```
tridentctl logs [flags]
```

Bandeiras

`-a`, `--archive` Criar um arquivo de suporte com todos os registros, a menos que especificado de outra forma.
`-h` , `--help` Ajuda para registros.
`-l` , `--log string` : Registro do Trident para exibir. Uma das opções: trident|auto|trident-operator|all (padrão "auto").
`--node string` : O nome do nó Kubernetes do qual coletar os logs dos pods.
`-p` , `--previous` Obtenha os registros da instância de contêiner anterior, se existirem.
`--sidecars` Obtenha os registros dos contêineres auxiliares.

enviar

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

```
tridentctl send [option]
```

Opções

`autosupport`Envie um arquivo de suporte automático para a NetApp.

desinstalar

Usar `uninstall` sinalizadores para desinstalar o Trident.

```
tridentctl uninstall [flags]
```

Bandeiras

`-h, --help`Ajuda para desinstalar.
`--silent` Desative a maior parte das saídas durante a desinstalação.

atualizar

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

```
tridentctl update [option]
```

Opções

`backend`Atualizar um backend no Trident.

atualizar estado do backend

Use o `update backend state` Comando para suspender ou retomar operações de backend.

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

Pontos a considerar

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

  Bandeiras

  `-h`, `--help` Ajuda para o estado do backend.
  `--user-state` : Definir para `suspended` Para pausar as operações de backend. Definir para `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 do backend usando `userState` campo no arquivo de configuração do backend `TridentBackendConfig` ou `backend.json` . Para obter mais informações, consulte"Opções para gerenciar back-ends" e"Realize o gerenciamento de backend com kubectl" .

Exemplo:

JSON

Siga estes passos para atualizar o `userState` usando o `backend.json` arquivo:

1. Editar o `backend.json` arquivo para incluir o `userState` campo com o valor definido como 'suspenso'.
2. Atualize o backend usando o `tridentctl update backend` comando e o caminho para a versão atualizada `backend.json` arquivo.

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

```
{
  "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 for aplicado usando o `kubectl edit <tbc-name> -n <namespace>` comando. O exemplo a seguir atualiza o estado do backend para suspender usando o `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

Usar `version` bandeiras para imprimir a versão de `tridentctl` e o serviço Trident em funcionamento.

```
tridentctl version [flags]
```

Bandeiras

`--client`: Somente versão cliente (não requer servidor).
`-h, --help` Ajuda para a versão.

Suporte a plugins

O Tridentctl suporta plugins semelhantes ao kubectl. O Tridentctl detecta um plugin se o nome do arquivo binário do plugin seguir o esquema "tridentctl-<plugin>" e o binário estiver localizado em uma pasta listada na variável de ambiente PATH. Todos os plugins detectados estão listados na seção de plugins da ajuda do tridentctl. Opcionalmente, você também pode limitar a pesquisa especificando uma pasta de plugins na variável de ambiente TRIDENTCTL_PLUGIN_PATH (Exemplo: `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`). Se a variável for usada, o tridenctl pesquisará apenas na pasta especificada.

Monitor Trident

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

Visão geral

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

- Monitore o estado de saúde e a configuração do Trident. Você pode analisar o sucesso das operações e verificar se a comunicação com os sistemas de backend ocorre conforme o esperado.
- Analise as informações de utilização do backend e compreenda quantos volumes estão provisionados em um backend, a quantidade de espaço consumido e assim por diante.
- Mantenha um mapeamento da quantidade de volumes provisionados nos backends disponíveis.
- Acompanhe o desempenho. Você pode verificar quanto tempo o Trident leva para se comunicar com os servidores de backend e executar operações.

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

O que você vai precisar

- Um cluster Kubernetes com o Trident instalado.
- Uma instância do Prometheus. Isso pode ser um "implantação do Prometheus em contêineres" ou você pode optar por executar o Prometheus como um "aplicativo nativo" .

Etapa 1: Defina um alvo do Prometheus

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

Prometheus como um operador em seu cluster Kubernetes e a criação de um ServiceMonitor para obter métricas do Trident .

Etapa 2: Criar um ServiceMonitor do Prometheus

Para consumir as métricas do Trident , você deve criar um ServiceMonitor do Prometheus que monitore o `trident-csi` serviço e escuta no `metrics` porta. Um exemplo de ServiceMonitor tem a seguinte aparência:

```
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
```

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

Além das métricas disponíveis diretamente no Trident, o kubelet expõe muitas outras. `kubelet_volume_*` métricas através de seu próprio endpoint de métricas. O Kubelet pode fornecer informações sobre os volumes que estão anexados, bem como sobre os pods e outras operações internas que ele gerencia. Consulte "aqui" .

Etapa 3: Consultar métricas do Trident com PromQL

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

Aqui estão algumas consultas PromQL que você pode usar:

Obtenha informações sobre saúde da Trident

- **Percentagem 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
```

- **Percentagem 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 pelo Trident**

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

Obtenha informações sobre o uso do Trident

- **Tamanho médio do volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Espaço total provisionado por cada backend**

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

Obtenha o consumo de volume individual.

Isso só é habilitado se as métricas do kubelet também forem coletadas.

- **Percentagem de espaço utilizado para cada volume**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Saiba mais sobre a telemetria do Trident AutoSupport.

Por padrão, o Trident envia métricas do Prometheus e informações básicas de backend para a NetApp diariamente.

- Para impedir que o Trident envie métricas do Prometheus e informações básicas de backend para a NetApp, passe o seguinte: `--silence-autosupport` bandeira durante a instalação do Trident .
- O Trident também pode enviar logs de contêineres para o Suporte da NetApp sob demanda via `tridentctl send autosupport` . Você precisará acionar o Trident para que ele envie seus registros. Antes de enviar os registros, você deve aceitar os termos da NetApp.https://www.netapp.com/company/legal/privacy-policy/["política de Privacidade"^] .

- A menos que especificado de outra forma, o Trident busca os registros das últimas 24 horas.
- Você pode especificar o período de retenção de logs com o `--since` bandeira. Por exemplo: `tridentctl send autosupport --since=1h` . Essas informações são coletadas e enviadas por meio de um `trident-autosupport` contêiner que é instalado ao lado do Trident. Você pode obter a imagem do contêiner em "AutoSupport Trident" .
- A Trident AutoSupport não coleta nem transmite informações de identificação pessoal (PII) ou informações pessoais. Ele vem com um "Contrato de licença de usuário final (EULA)" Isso não se aplica à própria imagem do contêiner Trident . Você pode saber mais sobre o compromisso da NetApp com a segurança e a confiabilidade dos dados. "aqui" .

Um exemplo de carga útil enviada pelo Trident é semelhante a esta:

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

- As mensagens do AutoSupport são enviadas para o endpoint do AutoSupport da NetApp. Se você estiver usando um registro privado para armazenar imagens de contêiner, poderá usar o `--image-registry` bandeira.
- 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 a favor do `trident-autosupport` contêiner em `trident-deployment.yaml` .

Desativar métricas do Trident

Para **desativar** a exibição de métricas, você deve gerar arquivos YAML personalizados (usando o `--generate-custom-yaml` flag) e edite-os para remover o `--metrics` impedir que a flag seja invocada para o `trident-main` recipiente.

Desinstale o Trident

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

Sobre esta tarefa

- Se precisar corrigir erros observados após uma atualização, problemas de dependência ou uma atualização incompleta ou malsucedida, desinstale o Trident e reinstale a versão anterior seguindo as instruções específicas para essa versão."versão" . Esta é a única maneira recomendada de *fazer o 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 precisar remover completamente o Trident e todos os seus dados, consulte"Remova completamente o Trident e os CRDs." .

Antes de começar

Se você estiver desativando clusters Kubernetes, deverá excluir todos os aplicativos que usam volumes criados pelo Trident antes de desinstalá-lo. Isso garante que os PVCs sejam despublicados nos nós do Kubernetes antes de serem excluídos.

Determine o método de instalação original.

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

1. Usar `kubectl get pods -n trident` para examinar as cápsulas.
   - Se não houver um módulo de operador, o Trident foi instalado usando `tridentctl` .
   - Caso exista um pod de operador, o Trident foi instalado usando o operador Trident, seja manualmente ou através do Helm.
2. Se houver um módulo de operador, use-o. `kubectl describe tproc trident` Para determinar se o Trident foi instalado usando o Helm.
   - Se houver uma etiqueta "Helm", o Trident foi instalado usando o Helm.
   - Se não houver nenhuma etiqueta "Helm", o Trident foi instalado manualmente usando o operador Trident .

Desinstalar uma instalação do operador Trident

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

Desinstalar instalação manual

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

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

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

   Quando o `uninstall` A flag está definida como `true` O operador Trident desinstala o Trident, mas não remove o próprio TridentOrchestrator. Você deve limpar a pasta TridentOrchestrator e criar uma nova se quiser instalar o Trident novamente.

2. **Excluir `TridentOrchestrator`** : Removendo o `TridentOrchestrator` CR que foi usado para implantar o Trident, você instrui o operador a desinstalar o Trident. O operador processa a remoção de

`TridentOrchestrator` e procede à remoção da implantação e do daemonset do Trident , excluindo os pods do Trident que havia criado como parte da instalação.

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

Desinstalar a instalação do Helm

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

```
#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 um `tridentctl` instalação

Use o `uninstall` comando em `tridentctl` Remover todos os recursos associados ao Trident, exceto os CRDs e objetos relacionados:

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

Trident para Docker

Pré-requisitos para implantação

Você precisa instalar e configurar os pré-requisitos de protocolo necessários em seu host antes de poder implantar o Trident.

Verifique os requisitos

- Verifique se sua implantação atende a todos os requisitos."requisitos" .
- Verifique se você tem uma versão compatível do Docker instalada. Se a sua versão do Docker estiver desatualizada, "instalar ou atualizar" .

```
docker --version
```

- Verifique se os pré-requisitos do protocolo estão instalados e configurados no seu host.

Ferramentas NFS

Instale as ferramentas NFS usando os comandos para o seu sistema operacional.

RHEL 8+

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```

Reinicie os nós de trabalho após instalar as ferramentas NFS para evitar falhas ao conectar volumes aos contêineres.

Ferramentas iSCSI

Instale as ferramentas iSCSI usando os comandos para o seu sistema operacional.

RHEL 8+

1. Instale os seguintes pacotes do sistema:

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_utils device-
mapper-multipath
```

2. Verifique se a versão do iscsi-initiator-utils é 6.2.0.874-2.el7 ou posterior:

```
rpm -q iscsi-initiator-utils
```

3. Configure a digitalização para manual:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Ativar multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

Garantir `etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

5. Certifique-se de que `iscsid` e `multipathd` estão em execução:

```
sudo systemctl enable --now iscsid multipathd
```

6. Ative e inicie `iscsi` :

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. Instale os seguintes pacotes do sistema:

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools
scsitools
```

2. Verifique se a versão do open-iscsi é 2.0.874-5ubuntu2.10 ou posterior (para bionic) ou 2.0.874-7.1ubuntu6.1 ou posterior (para focal):

```
dpkg -l open-iscsi
```

3. Configure a digitalização para manual:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Ativar multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

Garantir `etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

5. Certifique-se de que `open-iscsi` e `multipath-tools` estão habilitados e em execução:

```
sudo systemctl status multipath-tools
sudo systemctl enable --now open-iscsi.service
sudo systemctl status open-iscsi
```

Ferramentas NVMe

Instale as ferramentas NVMe usando os comandos específicos do seu sistema operacional.

- O NVMe requer RHEL 9 ou posterior.
- Se a versão do kernel do seu nó Kubernetes for muito antiga ou se o pacote NVMe não estiver disponível para a sua versão do kernel, talvez seja necessário atualizar a versão do kernel do seu nó para uma que inclua o pacote NVMe.

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ferramentas FC

Instale as ferramentas FC usando os comandos para o seu sistema operacional.

- Ao usar nós de trabalho que executam RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) com PVs FC, especifique o `discard` A opção mountOption na StorageClass permite realizar a recuperação de espaço em linha. Consulte "Documentação da Red Hat" .

RHEL 8+

1. Instale os seguintes pacotes do sistema:

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. Ativar multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

> Garantir `etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

3. Certifique-se de que `multipathd` está em execução:

```
sudo systemctl enable --now multipathd
```

Ubuntu

1. Instale os seguintes pacotes do sistema:

```
sudo apt-get install -y lsscsi sg3-utils multipath-tools scsitools
```

2. Ativar multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

> Garantir `etc/multipath.conf` contém `find_multipaths no` sob `defaults` .

3. Certifique-se de que `multipath-tools` Está habilitado e em execução:

```
sudo systemctl status multipath-tools
```

Implantar Trident

O Trident for Docker oferece integração direta com o ecossistema Docker para plataformas de armazenamento NetApp . Ele oferece suporte ao provisionamento e gerenciamento de recursos de armazenamento da plataforma de armazenamento para hosts Docker, com uma estrutura para adicionar plataformas adicionais no futuro.

Várias instâncias do Trident podem ser executadas simultaneamente no mesmo host. Isso permite conexões simultâneas a múltiplos sistemas e tipos de armazenamento, com a possibilidade de personalizar o armazenamento usado para os volumes do Docker.

O que você vai precisar

Veja o"Pré-requisitos para implantação" . Após garantir que os pré-requisitos sejam atendidos, você estará pronto para implantar o Trident.

Método de plugin gerenciado pelo Docker (versão 1.13/17.03 e posterior)

Antes de começar

Se você utilizou o Trident em versões anteriores ao Docker 1.13/17.03 com o método tradicional de daemon, certifique-se de interromper o processo do Trident e reiniciar o daemon do Docker antes de usar o método de plugin gerenciado.

1. Interrompa todas as instâncias em execução:

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Reinicie o Docker.

```
systemctl restart docker
```

3. Certifique-se de ter o Docker Engine 17.03 (novo 1.13) ou posterior instalado.

```
docker --version
```

Se a sua versão estiver desatualizada, "Instale ou atualize sua instalação." .

Passos

1. Crie um arquivo de configuração e especifique as opções da seguinte forma:
   - `config`O nome de arquivo padrão é `config.json` No entanto, você pode usar qualquer nome que escolher, especificando o `config` opção com o nome do arquivo. O arquivo de configuração deve estar localizado em `/etc/netappdvp` diretório no sistema host.
   - `log-level`Especifique o nível de registro.(`debug`, `info`, `warn`, `error`, `fatal`). O padrão é `info` .

- `debug`: Especifique se o registro de depuração está ativado. O valor padrão é falso. Substitui o nível de registro se verdadeiro.

  i. Crie um local para o arquivo de configuração:

  ```
  sudo mkdir -p /etc/netappdvp
  ```

  ii. Crie o arquivo de configuração:

  ```
  cat << EOF > /etc/netappdvp/config.json
  ```

  ```
  {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
  }
  EOF
  ```

2. Inicie o Trident usando o sistema de plugins gerenciados. Substituir `<version>` com a versão do plugin (xxx.xx.x) que você está usando.

   ```
   docker plugin install --grant-all-permissions --alias netapp
   netapp/trident-plugin:<version> config=myConfigFile.json
   ```

3. Comece a usar o Trident para consumir armazenamento do sistema configurado.

   a. Crie um volume chamado "firstVolume":

   ```
   docker volume create -d netapp --name firstVolume
   ```

   b. Criar um volume padrão quando o contêiner for iniciado:

   ```
   docker run --rm -it --volume-driver netapp --volume
   secondVolume:/my_vol alpine ash
   ```

   c. Remova o volume "firstVolume":

```
docker volume rm firstVolume
```

Método tradicional (versão 1.12 ou anterior)

Antes de começar

1. Certifique-se de ter o Docker versão 1.10 ou posterior.

   ```
   docker --version
   ```

   Se a sua versão estiver desatualizada, atualize a sua instalação.

   ```
   curl -fsSL https://get.docker.com/ | sh
   ```

   Ou, "Siga as instruções para a sua distribuição." .

2. Certifique-se de que o NFS e/ou iSCSI estejam configurados em seu sistema.

Passos

1. Instale e configure o plug-in de volume Docker da NetApp :
   a. Baixe e descompacte o aplicativo:

      ```
      wget
      https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
      installer-25.06.0.tar.gz
      tar zxf trident-installer-25.06.0.tar.gz
      ```

   b. Mover para um local no caminho da lixeira:

      ```
      sudo mv trident-installer/extras/bin/trident /usr/local/bin/
      sudo chown root:root /usr/local/bin/trident
      sudo chmod 755 /usr/local/bin/trident
      ```

   c. Crie um local para o arquivo de configuração:

      ```
      sudo mkdir -p /etc/netappdvp
      ```

   d. Crie o arquivo de configuração:

      ```
      cat << EOF > /etc/netappdvp/ontap-nas.json
      ```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
EOF
```

2. Após inserir o arquivo binário e criar o arquivo de configuração, inicie o daemon Trident usando o arquivo de configuração desejado.

```
sudo trident --config=/etc/netappdvp/ontap-nas.json
```

> A menos que seja especificado, o nome padrão para o driver de volume é "netapp".

Após o daemon ser iniciado, você pode criar e gerenciar volumes usando a interface de linha de comando do Docker.

3. Criar um volume:

```
docker volume create -d netapp --name trident_1
```

4. Provisione um volume Docker ao iniciar um contêiner:

```
docker run --rm -it --volume-driver netapp --volume trident_2:/my_vol
alpine ash
```

5. Remover um volume do Docker:

```
docker volume rm trident_1
```

```
docker volume rm trident_2
```

Inicie o Trident na inicialização do sistema.

Um arquivo de unidade de exemplo para sistemas baseados em systemd pode ser encontrado em

`contrib/trident.service.example` no repositório Git. Para usar o arquivo com o RHEL, faça o seguinte:

1. Copie o arquivo para o local correto.

   Você deve usar nomes exclusivos para os arquivos de unidade se tiver mais de uma instância em execução.

   ```
   cp contrib/trident.service.example
   /usr/lib/systemd/system/trident.service
   ```

2. Edite o arquivo, altere a descrição (linha 2) para corresponder ao nome do driver e o caminho do arquivo de configuração (linha 9) para refletir seu ambiente.
3. Recarregue o systemd para que ele incorpore as alterações:

   ```
   systemctl daemon-reload
   ```

4. Ative o serviço.

   Esse nome varia dependendo do nome que você deu ao arquivo. `/usr/lib/systemd/system` diretório.

   ```
   systemctl enable trident
   ```

5. Inicie o serviço.

   ```
   systemctl start trident
   ```

6. Veja o status.

   ```
   systemctl status trident
   ```

Sempre que você modificar o arquivo de unidade, execute o `systemctl daemon-reload` comando para que ele esteja ciente das mudanças.

Atualize ou desinstale o Trident

Você pode atualizar o Trident para Docker com segurança, sem afetar os volumes que estão em uso. Durante o processo de atualização, haverá um breve período em que `docker volume` Os comandos direcionados ao plugin não serão executados com sucesso e os aplicativos não conseguirão montar volumes até que o plugin esteja em execução novamente. Na maioria das circunstâncias, isso é uma questão de segundos.

Atualizar

Siga os passos abaixo para atualizar o Trident para Docker.

Passos

1. Liste os volumes existentes:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

2. Desative o plugin:

```
docker plugin disable -f netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Atualize o plugin:

```
docker plugin upgrade --skip-remote-check --grant-all-permissions
netapp:latest netapp/trident-plugin:21.07
```

> A versão 18.01 do Trident substitui o nDVP. Você deve atualizar diretamente do `netapp/ndvp-plugin` imagem para o `netapp/trident-plugin` imagem.

4. Ative o plugin:

```
docker plugin enable netapp:latest
```

5. Verifique se o plugin está ativado:

```
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       Trident - NetApp Docker Volume
Plugin   true
```

6. Verifique se os volumes estão visíveis:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

Se você estiver atualizando de uma versão antiga do Trident (anterior à 20.10) para o Trident 20.10 ou posterior, poderá encontrar um erro. Para obter mais informações, consulte"Problemas conhecidos" . Caso se depare com o erro, primeiro desative o plugin, depois remova-o e, em seguida, instale a versão necessária do Trident , passando um parâmetro de configuração adicional: `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all-permissions config=config.json`

Desinstalar

Siga os passos abaixo para desinstalar o Trident para Docker.

Passos

1. Remova todos os volumes criados pelo plugin.
2. Desative o plugin:

```
docker plugin disable netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Remova o plugin:

```
docker plugin rm netapp:latest
```

Trabalhar com volumes

Você pode facilmente criar, clonar e remover volumes usando o padrão. `docker volume` comandos com o nome do driver Trident especificado quando necessário.

Criar um volume

- Crie um volume com um driver usando o nome padrão:

```
docker volume create -d netapp --name firstVolume
```

- Criar um volume com uma instância específica do Trident :

```
docker volume create -d ntap_bronze --name bronzeVolume
```

Se você não especificar nenhum"opções" , os valores padrão do driver são utilizados.

- Substituir o tamanho padrão do volume. Veja o exemplo a seguir para criar um volume de 20 GiB com um driver:

```
docker volume create -d netapp --name my_vol --opt size=20G
```

Os tamanhos dos volumes são expressos como strings contendo um valor inteiro com unidades opcionais (exemplo: 10G, 20GB, 3TiB). Se nenhuma unidade for especificada, o padrão é G. As unidades de tamanho podem ser expressas como potências de 2 (B, KiB, MiB, GiB, TiB) ou potências de 10 (B, KB, MB, GB, TB). As unidades abreviadas usam potências de 2 (G = GiB, T = TiB, …).

Remover um volume

- Remova o volume da mesma forma que qualquer outro volume do Docker:

```
docker volume rm firstVolume
```

Ao usar o `solidfire-san` O driver, no exemplo acima, exclui e limpa o volume.

Siga os passos abaixo para atualizar o Trident para Docker.

Clonar um volume

Ao usar o `ontap-nas` , `ontap-san` , `solidfire-san` , e `gcp-cvs storage drivers` O Trident pode clonar volumes. Ao usar o `ontap-nas-flexgroup` ou `ontap-nas-economy` Drivers, clonagem não é suportada. Criar um novo volume a partir de um volume existente resultará na criação de um novo snapshot.

- Inspecione o volume para enumerar os snapshots:

```
docker volume inspect <volume_name>
```

- Criar um novo volume a partir de um volume existente. Isso resultará na criação de um novo instantâneo:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume>
```

- Criar um novo volume a partir de um snapshot existente em um volume. Isso não criará um novo instantâneo:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

Exemplo

```
docker volume inspect firstVolume

[
  {
    "Driver": "ontap-nas",
    "Labels": null,
    "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
    "Name": "firstVolume",
    "Options": {},
    "Scope": "global",
    "Status": {
      "Snapshots": [
        {
          "Created": "2017-02-10T19:05:00Z",
          "Name": "hourly.2017-02-10_1505"
        }
      ]
    }
  }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap
```

Acessar volumes criados externamente

Você pode acessar dispositivos de bloco criados externamente (ou seus clones) por contêineres usando o Trident **somente** se eles não tiverem partições e se o sistema de arquivos for compatível com o Trident (por exemplo: um `ext4` -formatado `/dev/sdc1` Não estará acessível via Trident).

Opções de volume específicas do motorista

Cada driver de armazenamento possui um conjunto diferente de opções, que você pode especificar no momento da criação do volume para personalizar o resultado. Veja abaixo as opções que se aplicam ao seu sistema de armazenamento configurado.

Utilizar essas opções durante a operação de criação de volume é simples. Forneça a opção e o valor usando o `-o` operador durante a operação da CLI. Essas configurações substituem quaisquer valores equivalentes do arquivo de configuração JSON.

opções de volume ONTAP

As opções de criação de volumes para NFS, iSCSI e FC incluem o seguinte:

Opção	Descrição
`size`	O tamanho do volume, por padrão, é de 1 GiB.
`spaceReserve`	O parâmetro "thin" ou "thick" define o volume; o padrão é "thin". Os valores válidos são `none` (provisionamento dinâmico) e `volume` (provisionado em espessura).
`snapshotPolicy`	Isso definirá a política de instantâneo para o valor desejado. O padrão é `none` Isso significa que nenhum snapshot será criado automaticamente para o volume. A menos que seja modificada pelo administrador de armazenamento, existe em todos os sistemas ONTAP uma política chamada "default" que cria e retém seis snapshots por hora, dois por dia e dois por semana. Os dados preservados em um instantâneo podem ser recuperados navegando até o `.snapshot` diretório em qualquer diretório do volume.
`snapshotReserve`	Isso definirá a reserva de instantâneos para a porcentagem desejada. O valor padrão é nenhum, o que significa que o ONTAP selecionará o snapshotReserve (geralmente 5%) se você tiver selecionado uma snapshotPolicy, ou 0% se a snapshotPolicy for nenhuma. Você pode definir o valor padrão snapshotReserve no arquivo de configuração para todos os backends do ONTAP e usá-lo como uma opção de criação de volume para todos os backends do ONTAP, exceto o ontap-nas-economy.

Opção	Descrição
`splitOnClone`	Ao clonar um volume, isso fará com que o ONTAP separe imediatamente o clone de seu volume pai. O padrão é `false` . Algumas situações de uso para clonagem de volumes são melhor atendidas separando o clone de seu volume original imediatamente após a criação, pois é improvável que haja qualquer oportunidade para otimizar o armazenamento. Por exemplo, clonar um banco de dados vazio pode oferecer grande economia de tempo, mas pouca economia de armazenamento; portanto, é melhor dividir o clone imediatamente.
`encryption`	Ative a Criptografia de Volume NetApp (NVE) no novo volume; o padrão é `false` . Para usar esta opção, o NVE precisa estar licenciado e habilitado no cluster. Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE" .
`tieringPolicy`	Define a política de hierarquização a ser usada para o volume. Isso decide se os dados serão movidos para a camada de nuvem quando se tornarem inativos (frios).

As seguintes opções adicionais são **exclusivas** para NFS:

Opção	Descrição
`unixPermissions`	Isso controla o conjunto de permissões para o próprio volume. Por padrão, as permissões serão definidas como `---rwxr-xr-x , ou em notação numérica 0755, e `root` será o proprietário. Tanto o formato de texto quanto o numérico funcionarão.
`snapshotDir`	Definir isso para `true` fará o `.snapshot` Diretório visível para clientes que acessam o volume. O valor padrão é `false` , o que significa que a visibilidade do `.snapshot` O diretório está desativado por padrão. Algumas imagens, como por exemplo a imagem oficial do MySQL, não funcionam como esperado quando… `.snapshot` O diretório está visível.
`exportPolicy`	Define a política de exportação a ser usada para o volume. O padrão é `default` .

Opção	Descrição
`securityStyle`	Define o estilo de segurança a ser usado para acesso ao volume. O padrão é `unix` . Os valores válidos são `unix` e `mixed` .

As seguintes opções adicionais são **exclusivas** para iSCSI:

Opção	Descrição
`fileSystemType`	Define o sistema de arquivos usado para formatar volumes iSCSI. O padrão é `ext4` . Os valores válidos são `ext3` , `ext4` , e `xfs` .
`spaceAllocation`	Definir isso para `false` irá desativar o recurso de alocação de espaço do LUN. O valor padrão é `true` Isso significa que o ONTAP notifica o host quando o volume fica sem espaço e o LUN no volume não pode aceitar gravações. Essa opção também permite que o ONTAP recupere espaço automaticamente quando o host excluir dados.

Exemplos

Veja os exemplos abaixo:

- Crie um volume de 10 GiB:

```
docker volume create -d netapp --name demo -o size=10G -o
encryption=true
```

- Crie um volume de 100 GiB com snapshots:

```
docker volume create -d netapp --name demo -o size=100G -o
snapshotPolicy=default -o snapshotReserve=10
```

- Crie um volume com o bit setUID ativado:

```
docker volume create -d netapp --name demo -o unixPermissions=4755
```

O tamanho mínimo do volume é 20 MiB.

Se a reserva de snapshots não for especificada e a política de snapshots for `none` A Trident utiliza uma reserva instantânea de 0%.

- Crie um volume sem política de snapshots e sem reserva de snapshots:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
```

- Crie um volume sem política de snapshots e com uma reserva de snapshots personalizada de 10%:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
--opt snapshotReserve=10
```

- Crie um volume com uma política de snapshot e uma reserva de snapshot personalizada de 10%:

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy --opt snapshotReserve=10
```

- Crie um volume com uma política de snapshot e aceite a reserva de snapshot padrão do ONTAP (normalmente 5%):

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

Opções de volume do software Element

As opções do software Element expõem o tamanho e as políticas de qualidade de serviço (QoS) associadas ao volume. Quando o volume é criado, a política de QoS associada a ele é especificada usando o `-o type=service_level` nomenclatura.

O primeiro passo para definir um nível de serviço QoS com o driver Element é criar pelo menos um tipo e especificar os valores mínimo, máximo e de burst de IOPS associados a um nome no arquivo de configuração.

Outras opções de criação de volume do software Element incluem o seguinte:

Opção	Descrição
`size`	O tamanho do volume, por padrão, é 1 GiB ou entrada de configuração … "defaults": {"size": "5G"}.
`blocksize`	Use 512 ou 4096, o padrão é 512 ou a entrada de configuração DefaultBlockSize.

Exemplo

Veja o seguinte arquivo de configuração de exemplo com definições de QoS:

```
{
  "Types": [
    {
      "Type": "Bronze",
      "Qos": {
        "minIOPS": 1000,
        "maxIOPS": 2000,
        "burstIOPS": 4000
      }
    },
    {
      "Type": "Silver",
      "Qos": {
        "minIOPS": 4000,
        "maxIOPS": 6000,
        "burstIOPS": 8000
      }
    },
    {
      "Type": "Gold",
      "Qos": {
        "minIOPS": 6000,
        "maxIOPS": 8000,
        "burstIOPS": 10000
      }
    }
  ]
}
```

Na configuração acima, temos três definições de política: Bronze, Prata e Ouro. Esses nomes são arbitrários.

- Crie um volume Gold de 10 GiB:

```
docker volume create -d solidfire --name sfGold -o type=Gold -o size=10G
```

- Crie um volume Bronze de 100 GiB:

```
docker volume create -d solidfire --name sfBronze -o type=Bronze -o
size=100G
```

Coletar registros

Você pode coletar registros para auxiliar na resolução de problemas. O método que você

usa para coletar os logs varia dependendo de como você está executando o plugin do Docker.

Colete registros para solucionar problemas.

Passos

1. Se você estiver executando o Trident usando o método de plugin gerenciado recomendado (ou seja, usando `docker plugin` comandos), veja-os da seguinte forma:

```
docker plugin ls
```

```
ID                  NAME                DESCRIPTION
ENABLED
4fb97d2b956b        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
journalctl -u docker | grep 4fb97d2b956b
```

   O nível de registro padrão deve permitir que você diagnostique a maioria dos problemas. Se achar que isso não é suficiente, você pode ativar o registro de depuração.

2. Para ativar o registro de depuração, instale o plugin com o registro de depuração ativado:

```
docker plugin install netapp/trident-plugin:<version> --alias <alias>
debug=true
```

   Ou, ative o registro de depuração quando o plugin já estiver instalado:

```
docker plugin disable <plugin>
```

```
docker plugin set <plugin> debug=true
```

```
docker plugin enable <plugin>
```

3. Se você estiver executando o próprio binário no host, os logs estarão disponíveis no diretório do host. `/var/log/netappdvp` diretório. Para ativar o registro de depuração, especifique `-debug` quando você executar o plugin.

Dicas gerais de resolução de problemas

- O problema mais comum que os novos usuários enfrentam é uma configuração incorreta que impede a inicialização do plugin. Quando isso acontecer, você provavelmente verá uma mensagem como esta ao tentar instalar ou ativar o plugin:

`Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock: connect: no such file or directory`

Isso significa que o plugin não pôde ser iniciado. Felizmente, o plugin foi desenvolvido com um recurso abrangente de registro de logs que deve ajudá-lo a diagnosticar a maioria dos problemas que você provavelmente encontrará.

- Se houver problemas com a montagem de um painel fotovoltaico em um contêiner, certifique-se de que `rpcbind` Está instalado e em funcionamento. Utilize o gerenciador de pacotes necessário para o sistema operacional do host e verifique se `rpcbind` Está em execução. Você pode verificar o status do serviço rpcbind executando um `systemctl status rpcbind` ou seu equivalente.

Gerenciar várias instâncias do Trident

São necessárias várias instâncias do Trident quando se deseja ter várias configurações de armazenamento disponíveis simultaneamente. A chave para múltiplas instâncias é dar-lhes nomes diferentes usando o `--alias` opção com o plugin conteinerizado, ou `--volume-driver` opção ao instanciar o Trident no host.

Etapas para o plugin gerenciado do Docker (versão 1.13/17.03 ou posterior)

1. Inicie a primeira instância especificando um alias e um arquivo de configuração.

```
docker plugin install --grant-all-permissions --alias silver
netapp/trident-plugin:21.07 config=silver.json
```

2. Inicie a segunda instância, especificando um alias e um arquivo de configuração diferentes.

```
docker plugin install --grant-all-permissions --alias gold
netapp/trident-plugin:21.07 config=gold.json
```

3. Crie volumes especificando o alias como o nome do driver.

   Por exemplo, para o volume de ouro:

```
docker volume create -d gold --name ntapGold
```

   Por exemplo, para o volume de prata:

```
docker volume create -d silver --name ntapSilver
```

Passos para a versão tradicional (1.12 ou anterior)

1. Inicie o plugin com uma configuração NFS usando um ID de driver personalizado:

```
sudo trident --volume-driver=netapp-nas --config=/path/to/config
-nfs.json
```

2. Inicie o plugin com uma configuração iSCSI usando um ID de driver personalizado:

```
sudo trident --volume-driver=netapp-san --config=/path/to/config
-iscsi.json
```

3. Provisione volumes Docker para cada instância de driver:

   Por exemplo, para NFS:

```
docker volume create -d netapp-nas --name my_nfs_vol
```

   Por exemplo, para iSCSI:

```
docker volume create -d netapp-san --name my_iscsi_vol
```

Opções de configuração de armazenamento

Veja as opções de configuração disponíveis para suas configurações do Trident .

Opções de configuração global

Essas opções de configuração se aplicam a todas as configurações do Trident , independentemente da plataforma de armazenamento utilizada.

Opção	Descrição	Exemplo
`version`	Número da versão do arquivo de configuração	`1`
`storageDriverName`	Nome do driver de armazenamento	`ontap-nas`, `ontap-san` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `solidfire-san`
`storagePrefix`	Prefixo opcional para nomes de volumes. Padrão: `netappdvp_` .	`staging_`
`limitVolumeSize`	Restrição opcional nos tamanhos dos volumes. Padrão: "" (não aplicado)	`10g`

Não use `storagePrefix` (incluindo o padrão) para back-ends do Element. Por padrão, o `solidfire-san` O driver irá ignorar essa configuração e não usará um prefixo. A NetApp recomenda o uso de um tenantID específico para o mapeamento de volumes do Docker ou o uso dos dados de atributo preenchidos com a versão do Docker, informações do driver e nome bruto do Docker, nos casos em que possa ter ocorrido alguma alteração de nome.

As opções padrão estão disponíveis para evitar que você precise especificá-las em cada volume que criar. O `size` Essa opção está disponível para todos os tipos de controladores. Consulte a seção de configuração do ONTAP para obter um exemplo de como definir o tamanho padrão do volume.

Opção	Descrição	Exemplo
`size`	Tamanho padrão opcional para novos volumes. Padrão: `1G`	`10G`

Configuração ONTAP

Além dos valores de configuração global acima, ao usar o ONTAP, as seguintes opções de nível superior estão disponíveis.

Opção	Descrição	Exemplo
`managementLIF`	Endereço IP da LIF de gerenciamento do ONTAP . Você pode especificar um nome de domínio totalmente qualificado (FQDN).	`10.0.0.1`
`dataLIF`	Endereço IP do protocolo LIF. • Drivers ONTAP NAS *: A NetApp recomenda especificar `dataLIF` . Caso não sejam fornecidos, o Trident obtém os dataLIFs da SVM. Você pode especificar um nome de domínio totalmente qualificado (FQDN) para ser usado nas operações de montagem NFS, permitindo criar um DNS round-robin para balancear a carga entre várias dataLIFs. • Drivers ONTAP SAN*: Não especifique para iSCSI ou FC. Trident usa"Mapa LUN Seletivo ONTAP" para descobrir as LIFs iSCSI ou FC necessárias para estabelecer uma sessão de múltiplos caminhos. Um aviso é gerado se `dataLIF` está explicitamente definido.	`10.0.0.2`

Opção	Descrição	Exemplo
`svm`	Máquina virtual de armazenamento a ser usada (obrigatória se a LIF de gerenciamento for uma LIF de cluster)	`svm_nfs`
`username`	Nome de usuário para conectar-se ao dispositivo de armazenamento	`vsadmin`
`password`	Senha para conectar ao dispositivo de armazenamento	`secret`
`aggregate`	Agregado para provisionamento (opcional; se definido, deve ser atribuído à SVM). Para o `ontap-nas-flexgroup` motorista, esta opção é ignorada. Todos os agregados atribuídos ao SVM são usados para provisionar um volume FlexGroup .	`aggr1`
`limitAggregateUsage`	Opcional: o provisionamento pode falhar se a utilização for superior a esta percentagem.	`75%`
`nfsMountOptions`	Controle preciso das opções de montagem NFS; o padrão é "-o nfsvers=3". **Disponível apenas para o `ontap-nas` e `ontap-nas-economy` motoristas**. "Consulte as informações de configuração do host NFS aqui." .	`-o nfsvers=4`
`igroupName`	O Trident cria e gerencia por nó. `igroups` como `netappdvp` . Este valor não pode ser alterado nem omitido. **Disponível apenas para o `ontap-san` motorista**.	`netappdvp`
`limitVolumeSize`	Tamanho máximo do volume solicitado.	`300g`

Opção	Descrição	Exemplo
`qtreesPerFlexvol`	Número máximo de qtrees por FlexVol, deve estar no intervalo [50, 300], o padrão é 200. **Para o `ontap-nas-economy` driver, esta opção permite personalizar o número máximo de qtrees por FlexVol**.	`300`
`sanType`	**Compatível com `ontap-san` Somente o motorista.** Use para selecionar `iscsi` para iSCSI, `nvme` para NVMe/TCP ou `fcp` para SCSI sobre Fibre Channel (FC).	`iscsi`se estiver em branco
`limitVolumePoolSize`	**Compatível com `ontap-san-economy` e `ontap-san-economy` Somente motoristas.** Limita os tamanhos do FlexVol nos drivers ONTAP ontap-nas-economy e ontap-SAN-economy.	`300g`

As opções padrão estão disponíveis para evitar que você precise especificá-las em cada volume que criar:

Opção	Descrição	Exemplo
`spaceReserve`	Modo de reserva de espaço; `none` (provisionamento dinâmico) ou `volume` (espesso)	`none`
`snapshotPolicy`	A política de instantâneo a ser usada, padrão é `none`	`none`
`snapshotReserve`	Percentual de reserva de snapshots; o padrão é "" para aceitar o padrão do ONTAP.	`10`
`splitOnClone`	Separar um clone de seu pai no momento da criação; o padrão é `false`	`false`
`encryption`	Habilita a Criptografia de Volume NetApp (NVE) no novo volume; o padrão é `false` . Para usar esta opção, o NVE precisa estar licenciado e habilitado no cluster. Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE" .	verdadeiro

Opção	Descrição	Exemplo
`unixPermissions`	Opção NAS para volumes NFS provisionados, o padrão é `777`	`777`
`snapshotDir`	Opção NAS para acesso ao `.snapshot` diretório.	"verdadeiro" para NFSv4 "falso" para NFSv3
`exportPolicy`	Opção NAS para a política de exportação NFS a ser usada, o padrão é `default`	`default`
`securityStyle`	Opção NAS para acesso ao volume NFS provisionado. Suporte a NFS `mixed` e `unix` estilos de segurança. O padrão é `unix` .	`unix`
`fileSystemType`	Opção SAN para selecionar o tipo de sistema de arquivos; o padrão é `ext4`	`xfs`
`tieringPolicy`	A política de hierarquização a ser usada, padrão é `none` .	`none`

Opções de escala

O `ontap-nas` e `ontap-san` Os drivers criam um ONTAP FlexVol para cada volume Docker. O ONTAP suporta até 1000 FlexVols por nó de cluster, com um máximo de 12.000 volumes FlexVol por cluster. Se os seus requisitos de volume do Docker estiverem dentro dessa limitação, o `ontap-nas` O driver é a solução NAS preferida devido aos recursos adicionais oferecidos pelo FlexVols, como snapshots granulares de volume Docker e clonagem.

Se você precisar de mais volumes Docker do que os limites do FlexVol permitem, escolha a opção `ontap-nas-economy` ou o `ontap-san-economy` motorista.

O `ontap-nas-economy` O driver cria volumes Docker como Qtrees ONTAP dentro de um conjunto de volumes FlexVol gerenciados automaticamente. As Qtrees oferecem uma escalabilidade muito maior, até 100.000 por nó de cluster e 2.400.000 por cluster, à custa de algumas funcionalidades. O `ontap-nas-economy` O driver não suporta snapshots granulares de volume do Docker nem clonagem.

> O `ontap-nas-economy` O driver não é suportado atualmente no Docker Swarm, pois o Docker Swarm não orquestra a criação de volumes em vários nós.

O `ontap-san-economy` O driver cria volumes Docker como LUNs ONTAP dentro de um pool compartilhado de volumes FlexVol gerenciados automaticamente. Dessa forma, cada FlexVol não fica restrito a apenas um LUN, oferecendo melhor escalabilidade para cargas de trabalho SAN. Dependendo do array de armazenamento, o ONTAP suporta até 16384 LUNs por cluster. Como os volumes são LUNs subjacentes, este driver suporta snapshots e clonagem granulares de volume do Docker.

Escolha o `ontap-nas-flexgroup` Driver para aumentar o paralelismo em um único volume que pode crescer até a faixa de petabytes com bilhões de arquivos. Alguns casos de uso ideais para FlexGroups

incluem IA/ML/DL, big data e análise de dados, compilação de software, streaming, repositórios de arquivos e assim por diante. O Trident utiliza todos os agregados atribuídos a uma SVM ao provisionar um volume FlexGroup . O suporte do FlexGroup no Trident também leva em consideração os seguintes aspectos:

- Requer ONTAP versão 9.2 ou superior.
- Até o momento da redação deste texto, o FlexGroups suporta apenas NFS v3.
- Recomenda-se habilitar os identificadores NFSv3 de 64 bits para a SVM.
- O tamanho mínimo recomendado do membro/volume do FlexGroup é 100 GiB.
- A clonagem não é suportada para volumes FlexGroup .

Para obter informações sobre FlexGroups e cargas de trabalho adequadas para FlexGroups, consulte o seguinte: "Guia de Melhores Práticas e Implementação do NetApp FlexGroup para volumes" .

Para obter recursos avançados e grande escalabilidade no mesmo ambiente, você pode executar várias instâncias do Docker Volume Plugin, com uma delas usando `ontap-nas` e outro usando `ontap-nas-economy` .

Função ONTAP personalizada para Trident

Você pode criar uma função de cluster ONTAP com privilégios mínimos para que não precise usar a função de administrador do ONTAP para executar operações no Trident. Ao incluir o nome de usuário em uma configuração de backend do Trident , o Trident usa a função de cluster ONTAP que você criou para executar as operações.

Consulte"Gerador de funções personalizadas Trident" Para obter mais informações sobre como criar funções personalizadas do Trident .

Utilizando a CLI do ONTAP

1. Crie uma nova função usando o seguinte comando:

```
security login role create <role_name\> -cmddirname "command" -access all
-vserver <svm_name\>
```

2. Crie um nome de usuário para o usuário do Trident :

```
security login create -username <user_name\> -application ontapi
-authmethod password -role <name_of_role_in_step_1\> -vserver <svm_name\>
-comment "user_description"
security login create -username <user_name\> -application http -authmethod
password -role <name_of_role_in_step_1\> -vserver <svm_name\> -comment
"user_description"
```

3. Atribua a função ao usuário:

```
security login modify username <user_name\> -vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilizando o Gerenciador de Sistemas

Execute as seguintes etapas no ONTAP System Manager:

1. **Criar uma função personalizada**:
   a. Para criar uma função personalizada no nível do cluster, selecione **Cluster > Configurações**.

      (Ou) Para criar uma função personalizada no nível da SVM, selecione **Armazenamento > VMs de armazenamento > `required SVM` > Configurações > Usuários e funções**.

   b. Selecione o ícone de seta (→) ao lado de **Usuários e Funções**.
   c. Selecione **+Adicionar** em **Funções**.
   d. Defina as regras para a função e clique em **Salvar**.
2. **Atribua a função ao usuário do Trident *: + Execute as seguintes etapas na página *Usuários e Funções**:
   a. Selecione o ícone Adicionar * em **Usuários**.
   b. Selecione o nome de usuário desejado e, em seguida, selecione uma função no menu suspenso **Função**.
   c. Clique em **Salvar**.

Consulte as páginas seguintes para obter mais informações:

- "Funções personalizadas para administração do ONTAP"ou"Defina funções personalizadas"
- "Trabalhar com funções e usuários"

Arquivos de configuração de exemplo do ONTAP

Exemplo de NFS para o driver <code>ontap-nas</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "defaults": {
    "size": "10G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Exemplo de NFS para o driver <code>ontap-nas-flexgroup</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-flexgroup",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "defaults": {
    "size": "100G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Exemplo de NFS para o driver <code>ontap-nas-economy</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
```

Exemplo de iSCSI para o driver ontap-san.

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Exemplo de NFS para o driver <code>ontap-san-economy</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-san-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi_eco",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Exemplo NVMe/TCP para o driver ontap-san.

```
{
  "version": 1,
  "backendName": "NVMeBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nvme",
  "username": "vsadmin",
  "password": "password",
  "sanType": "nvme",
  "useREST": true
}
```

Exemplo de SCSI sobre FC para o driver <code>ontap-san</code>

```
{
  "version": 1,
  "backendName": "ontap-san-backend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "sanType": "fcp",
  "svm": "trident_svm",
  "username": "vsadmin",
  "password": "password",
  "useREST": true
}
```

Configuração do software Element

Além dos valores de configuração global, ao usar o software Element (NetApp HCI/ SolidFire), estas opções estão disponíveis.

Opção	Descrição	Exemplo
`Endpoint`	<a href="https://<login>:<senha>@<mvip>/json-rpc/<element-version&gt" class="bare">https://<login>:<senha>@<mvip>/json-rpc/<element-version&gt;	https://admin:admin@192.168.160.3/json-rpc/8.0

Opção	Descrição	Exemplo
`SVIP`	Endereço IP e porta iSCSI	10.0.0.7:3260
`TenantName`	Tenant do SolidFireF a ser usado (criado se não for encontrado)	`docker`
`InitiatorIFace`	Especifique a interface ao restringir o tráfego iSCSI a uma interface não padrão.	`default`
`Types`	Especificações de QoS	Veja o exemplo abaixo.
`LegacyNamePrefix`	Prefixo para instalações Trident atualizadas. Se você utilizou uma versão do Trident anterior à 1.3.2 e realizar uma atualização com volumes existentes, será necessário definir esse valor para acessar seus volumes antigos que foram mapeados pelo método de nome do volume.	`netappdvp-`

O `solidfire-san` O driver não é compatível com o Docker Swarm.

Exemplo de arquivo de configuração do software Element

```
{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
  "SVIP": "10.0.0.7:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "Types": [
    {
      "Type": "Bronze",
      "Qos": {
        "minIOPS": 1000,
        "maxIOPS": 2000,
        "burstIOPS": 4000
      }
    },
    {
      "Type": "Silver",
      "Qos": {
        "minIOPS": 4000,
        "maxIOPS": 6000,
        "burstIOPS": 8000
      }
    },
    {
      "Type": "Gold",
      "Qos": {
        "minIOPS": 6000,
        "maxIOPS": 8000,
        "burstIOPS": 10000
      }
    }
  ]
}
```

Problemas e limitações conhecidos

Encontre informações sobre problemas e limitações conhecidos ao usar o Trident com o Docker.

A atualização do plugin Trident Docker Volume para a versão 20.10 ou posterior a partir de versões mais antigas resulta em falha na atualização com o erro "arquivo ou diretório inexistente".

Solução alternativa

1. Desative o plugin.

```
docker plugin disable -f netapp:latest
```

2. Remova o plugin.

```
docker plugin rm -f netapp:latest
```

3. Reinstale o plugin fornecendo o arquivo extra. `config` parâmetro.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant
-all-permissions config=config.json
```

Os nomes dos volumes devem ter no mínimo 2 caracteres.

Essa é uma limitação do cliente Docker. O cliente interpretará um nome de caractere único como sendo um caminho do Windows. "Veja o bug 25773" .

O Docker Swarm possui certos comportamentos que impedem o Trident de oferecer suporte a todas as combinações de armazenamento e driver.

- Atualmente, o Docker Swarm utiliza o nome do volume em vez do ID do volume como seu identificador único de volume.
- As solicitações de volume são enviadas simultaneamente para cada nó em um cluster Swarm.
- Os plugins de volume (incluindo o Trident) devem ser executados independentemente em cada nó de um cluster Swarm. Devido ao modo como o ONTAP funciona e como o `ontap-nas` e `ontap-san` Os drivers são a única função que consegue operar dentro dessas limitações.

Os demais drivers estão sujeitos a problemas como condições de corrida, que podem resultar na criação de um grande número de volumes para uma única solicitação sem um "vencedor" claro; por exemplo, o Element possui um recurso que permite que os volumes tenham o mesmo nome, mas IDs diferentes.

A NetApp forneceu feedback à equipe do Docker, mas não há indicação de que medidas serão tomadas no futuro.

Se um FlexGroup estiver sendo provisionado, o ONTAP não provisionará um segundo FlexGroup se este segundo tiver um ou mais FlexGroup em comum com o FlexGroup que está sendo provisionado.

Melhores práticas e recomendações

Implantação

Utilize as recomendações listadas aqui ao implantar o Trident.

Implantar em um namespace dedicado

"Espaços de nomes"Proporcionam separação administrativa entre diferentes aplicações e constituem uma barreira ao compartilhamento de recursos. Por exemplo, um PVC de um namespace não pode ser consumido por outro. O Trident fornece recursos PV para todos os namespaces no cluster Kubernetes e, consequentemente, utiliza uma conta de serviço com privilégios elevados.

Além disso, o acesso ao módulo Trident pode permitir que um usuário acesse credenciais do sistema de armazenamento e outras informações confidenciais. É importante garantir que os usuários do aplicativo e os aplicativos de gerenciamento não tenham a capacidade de acessar as definições de objeto do Trident ou os próprios pods.

Utilize quotas e limites de intervalo para controlar o consumo de armazenamento.

O Kubernetes possui duas funcionalidades que, quando combinadas, fornecem um mecanismo poderoso para limitar o consumo de recursos pelas aplicações. O "mecanismo de cota de armazenamento" Permite ao administrador implementar limites globais e específicos da classe de armazenamento para o consumo de capacidade e contagem de objetos por namespace. Além disso, usando um "limite de alcance" Garante que as solicitações de PVC estejam dentro de um valor mínimo e máximo antes de serem encaminhadas ao provisionador.

Esses valores são definidos para cada namespace individualmente, o que significa que cada namespace deve ter valores definidos que estejam de acordo com seus requisitos de recursos. Veja aqui para obter informações sobre "como aproveitar as quotas" .

Configuração de armazenamento

Cada plataforma de armazenamento no portfólio da NetApp possui recursos exclusivos que beneficiam os aplicativos, sejam eles conteinerizados ou não.

Visão geral da plataforma

O Trident funciona com ONTAP e Element. Não existe uma plataforma que seja mais adequada para todas as aplicações e cenários do que outra; no entanto, as necessidades da aplicação e da equipe que administra o dispositivo devem ser levadas em consideração ao escolher uma plataforma.

Você deve seguir as melhores práticas básicas para o sistema operacional do host com o protocolo que você está utilizando. Opcionalmente, você pode considerar incorporar as melhores práticas de aplicação, quando disponíveis, nas configurações de backend, classe de armazenamento e PVC para otimizar o armazenamento para aplicações específicas.

ONTAP e Cloud Volumes ONTAP: Melhores práticas

Aprenda as melhores práticas para configurar o ONTAP e o Cloud Volumes ONTAP para Trident.

As recomendações a seguir são diretrizes para configurar o ONTAP para cargas de trabalho conteinerizadas, que consomem volumes provisionados dinamicamente pelo Trident. Cada um deve ser considerado e avaliado quanto à sua adequação ao seu ambiente.

Use SVM(s) dedicada(s) ao Trident.

Máquinas virtuais de armazenamento (SVMs) fornecem isolamento e separação administrativa entre locatários em um sistema ONTAP . Dedicar uma SVM a aplicações permite a delegação de privilégios e a aplicação das melhores práticas para limitar o consumo de recursos.

Existem diversas opções disponíveis para o gerenciamento da SVM:

- Forneça a interface de gerenciamento do cluster na configuração do backend, juntamente com as credenciais apropriadas, e especifique o nome da SVM.
- Crie uma interface de gerenciamento dedicada para a SVM usando o ONTAP System Manager ou a CLI.
- Compartilhe a função de gerenciamento com uma interface de dados NFS.

Em todos os casos, a interface deve estar no DNS e o nome DNS deve ser usado ao configurar o Trident. Isso ajuda a facilitar alguns cenários de recuperação de desastres (DR), por exemplo, SVM-DR sem a necessidade de retenção de identidade de rede.

Não há preferência entre ter uma LIF de gerenciamento dedicada ou compartilhada para a SVM; no entanto, você deve garantir que suas políticas de segurança de rede estejam alinhadas com a abordagem escolhida. Independentemente disso, o LIF de gerenciamento deve ser acessível via DNS para facilitar a máxima flexibilidade. "SVM-DR" Pode ser usado em conjunto com o Trident.

Limitar a contagem máxima de volume

Os sistemas de armazenamento ONTAP possuem um número máximo de volumes, que varia de acordo com a versão do software e a plataforma de hardware. Consulte "Hardware Universe da NetApp" Para determinar os limites exatos, consulte a plataforma específica e a versão do ONTAP . Quando o número de volumes disponíveis se esgota, as operações de provisionamento falham não apenas para o Trident, mas para todas as solicitações de armazenamento.

Tridente `ontap-nas` e `ontap-san` Os drivers provisionam um FlexVolume para cada Volume Persistente (PV) do Kubernetes que é criado. O `ontap-nas-economy` O driver cria aproximadamente um FlexVolume para cada 200 PVs (configurável entre 50 e 300). O `ontap-san-economy` O driver cria aproximadamente um FlexVolume para cada 100 PVs (configurável entre 50 e 200). Para evitar que o Trident consuma todos os volumes disponíveis no sistema de armazenamento, você deve definir um limite no SVM. Você pode fazer isso na linha de comando:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

O valor para `max-volumes` varia de acordo com diversos critérios específicos do seu ambiente:

- O número de volumes existentes no cluster ONTAP
- O número de volumes que você espera provisionar fora do Trident para outros aplicativos.
- O número de volumes persistentes que se espera que sejam consumidos por aplicações Kubernetes.

O `max-volumes` O valor representa o total de volumes provisionados em todos os nós do cluster ONTAP , e não em um nó ONTAP individual. Como resultado, você pode encontrar algumas condições em que um nó de

cluster ONTAP pode ter muito mais ou menos volumes provisionados Trident do que outro nó.

Por exemplo, um cluster ONTAP de dois nós tem a capacidade de hospedar um máximo de 2000 volumes FlexVol . Definir o número máximo de volumes em 1250 parece bastante razoável. No entanto, se apenas "agregados" Se os volumes atribuídos a partir de um nó forem alocados ao SVM, ou se os agregados atribuídos a partir de um nó não puderem ser provisionados (por exemplo, devido à capacidade), o outro nó se torna o destino de todos os volumes provisionados Trident . Isso significa que o limite de volume pode ser atingido para esse nó antes do `max-volumes` O valor é atingido, impactando tanto o Trident quanto outras operações de volume que utilizam esse nó. **Você pode evitar essa situação garantindo que os agregados de cada nó do cluster sejam atribuídos ao SVM usado pelo Trident em números iguais.**

Clonar um volume

O NetApp Trident suporta a clonagem de volumes ao usar o `ontap-nas` , `ontap-san` , `solidfire-san` , e `gcp-cvs` drivers de armazenamento. Ao usar o `ontap-nas-flexgroup` ou `ontap-nas-economy` Drivers, clonagem não é suportada. Criar um novo volume a partir de um volume existente resultará na criação de um novo snapshot.

Evite clonar um PVC associado a uma StorageClass diferente. Execute operações de clonagem dentro da mesma StorageClass para garantir a compatibilidade e evitar comportamentos inesperados.

Limitar o tamanho máximo dos volumes criados pelo Trident

Para configurar o tamanho máximo dos volumes que podem ser criados pelo Trident, use o `limitVolumeSize` parâmetro em seu `backend.json` definição.

Além de controlar o tamanho do volume no array de armazenamento, você também deve aproveitar os recursos do Kubernetes.

Limitar o tamanho máximo dos FlexVols criados pelo Trident.

Para configurar o tamanho máximo dos FlexVols usados como pools para os drivers ontap-san-economy e ontap-nas-economy, utilize o `limitVolumePoolSize` parâmetro em seu `backend.json` definição.

Configure o Trident para usar CHAP bidirecional.

Você pode especificar os nomes de usuário e senhas do iniciador e do destino CHAP na definição do seu backend e configurar o Trident para habilitar o CHAP na SVM. Usando o `useCHAP` Ao configurar o parâmetro na sua configuração de backend, o Trident autentica as conexões iSCSI para backends ONTAP com CHAP.

Criar e usar uma política de QoS para SVM

Utilizando uma política de QoS do ONTAP , aplicada ao SVM, limita-se o número de IOPS que podem ser consumidos pelos volumes provisionados Trident . Isso ajuda a "prevenir um valentão" ou um contêiner fora de controle que afete cargas de trabalho fora do Trident SVM.

Você pode criar uma política de QoS para a SVM em poucos passos. Consulte a documentação da sua versão do ONTAP para obter as informações mais precisas. O exemplo abaixo cria uma política de QoS que limita o total de IOPS disponíveis para a SVM a 5000.

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

Além disso, se a sua versão do ONTAP suportar, você pode considerar usar um mínimo de QoS para garantir uma quantidade de throughput para cargas de trabalho em contêineres. A QoS adaptativa não é compatível com uma política de nível SVM.

O número de IOPS dedicados às cargas de trabalho em contêineres depende de muitos fatores. Entre outras coisas, isso inclui:

- Outras cargas de trabalho que utilizam o array de armazenamento. Caso existam outras cargas de trabalho, não relacionadas à implementação do Kubernetes, utilizando os recursos de armazenamento, deve-se tomar cuidado para garantir que essas cargas de trabalho não sejam afetadas negativamente por acidente.
- Cargas de trabalho esperadas em execução em contêineres. Se cargas de trabalho com altos requisitos de IOPS forem executadas em contêineres, uma política de QoS baixa resultará em uma experiência ruim.

É importante lembrar que uma política de QoS atribuída no nível da SVM resulta em todos os volumes provisionados para a SVM compartilhando o mesmo pool de IOPS. Se uma, ou um pequeno número, das aplicações conteinerizadas tiverem uma alta exigência de IOPS, isso poderá sobrecarregar as outras cargas de trabalho conteinerizadas. Nesse caso, você pode considerar usar automação externa para atribuir políticas de QoS por volume.

> Você deve atribuir o grupo de políticas de QoS à SVM **somente** se a sua versão do ONTAP for anterior à 9.8.

Crie grupos de políticas de QoS para o Trident.

A Qualidade de Serviço (QoS) garante que o desempenho de cargas de trabalho críticas não seja prejudicado por cargas de trabalho concorrentes. Os grupos de políticas de QoS do ONTAP fornecem opções de QoS para volumes e permitem que os usuários definam o limite de taxa de transferência para uma ou mais cargas de trabalho. Para obter mais informações sobre QoS, consulte: "Garantindo a taxa de transferência com QoS" . Você pode especificar grupos de políticas de QoS no backend ou em um pool de armazenamento, e eles serão aplicados a cada volume criado nesse pool ou backend.

O ONTAP possui dois tipos de grupos de políticas de QoS: tradicionais e adaptativos. Os grupos de políticas tradicionais fornecem uma taxa de transferência máxima (ou mínima, em versões posteriores) fixa em IOPS. O QoS adaptativo dimensiona automaticamente a taxa de transferência de acordo com o tamanho da carga de trabalho, mantendo a proporção de IOPS para TBs|GBs à medida que o tamanho da carga de trabalho muda. Isso proporciona uma vantagem significativa ao gerenciar centenas ou milhares de cargas de trabalho em uma implantação de grande porte.

Ao criar grupos de políticas de QoS, considere o seguinte:

- Você deve definir o `qosPolicy` chave no `defaults` bloco da configuração do backend. Veja o exemplo de configuração de backend a seguir:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
  - labels:
      performance: extreme
    defaults:
      adaptiveQosPolicy: extremely-adaptive-pg
  - labels:
      performance: premium
    defaults:
      qosPolicy: premium-pg
```

- Você deve aplicar os grupos de políticas por volume, para que cada volume receba toda a taxa de transferência especificada pelo grupo de políticas. Grupos de políticas compartilhadas não são suportados.

Para obter mais informações sobre grupos de políticas de QoS, consulte: "Referência de comandos ONTAP" .

Limitar o acesso a recursos de armazenamento aos membros do cluster Kubernetes

Limitar o acesso aos volumes NFS, LUNs iSCSI e LUNs FC criados pelo Trident é um componente crítico da postura de segurança da sua implementação do Kubernetes. Isso evita que hosts que não fazem parte do cluster do Kubernetes acessem os volumes e potencialmente modifiquem dados inesperadamente.

É importante entender que os namespaces são o limite lógico para recursos no Kubernetes. A premissa é que os recursos no mesmo espaço de nomes podem ser compartilhados; no entanto, é importante ressaltar que não há capacidade de compartilhamento entre espaços de nomes diferentes. Isso significa que, embora os PVs sejam objetos globais, quando vinculados a um PVC, eles só são acessíveis por pods que estejam no mesmo namespace. **É fundamental garantir que os namespaces sejam usados para fornecer separação quando apropriado.**

A principal preocupação da maioria das organizações em relação à segurança de dados em um contexto Kubernetes é que um processo em um contêiner possa acessar o armazenamento montado no host, mas que não se destina ao contêiner. "Espaços de nomes" são projetadas para evitar esse tipo de comprometimento. No entanto, existe uma exceção: os contêineres privilegiados.

Um contêiner privilegiado é aquele que é executado com permissões de nível de host substancialmente maiores do que o normal. Essas opções não são negadas por padrão, portanto, certifique-se de desativar a funcionalidade usando "políticas de segurança de pod" .

Para volumes onde o acesso é desejado tanto do Kubernetes quanto de hosts externos, o armazenamento deve ser gerenciado de maneira tradicional, com o PV (Volume de Processo) introduzido pelo administrador e

não gerenciado pelo Trident. Isso garante que o volume de armazenamento seja destruído somente quando o Kubernetes e os hosts externos estiverem desconectados e não estiverem mais usando o volume. Além disso, é possível aplicar uma política de exportação personalizada, que permite o acesso a partir dos nós do cluster Kubernetes e de servidores de destino fora do cluster Kubernetes.

Para implantações que possuem nós de infraestrutura dedicados (por exemplo, OpenShift) ou outros nós que não conseguem agendar aplicativos de usuário, políticas de exportação separadas devem ser usadas para limitar ainda mais o acesso aos recursos de armazenamento. Isso inclui a criação de uma política de exportação para serviços que são implantados nesses nós de infraestrutura (por exemplo, os serviços OpenShift Metrics e Logging) e aplicativos padrão que são implantados em nós que não são de infraestrutura.

Utilize uma política de exportação específica.

Você deve garantir que exista uma política de exportação para cada backend que permita acesso apenas aos nós presentes no cluster Kubernetes. O Trident pode criar e gerenciar automaticamente políticas de exportação. Dessa forma, o Trident limita o acesso aos volumes que provisiona aos nós no cluster Kubernetes e simplifica a adição/exclusão de nós.

Alternativamente, você também pode criar uma política de exportação manualmente e preenchê-la com uma ou mais regras de exportação que processem cada solicitação de acesso ao nó:

- Use o `vserver export-policy create` Comando da CLI do ONTAP para criar a política de exportação.
- Adicione regras à política de exportação usando o `vserver export-policy rule create` Comando ONTAP CLI.

Executar esses comandos permite restringir quais nós do Kubernetes têm acesso aos dados.

Desativar `showmount` para a aplicação SVM

O `showmount` Esse recurso permite que um cliente NFS consulte a SVM para obter uma lista de exportações NFS disponíveis. Um pod implantado no cluster Kubernetes pode emitir o `showmount -e` Comande o inimigo e receba uma lista de montarias disponíveis, incluindo aquelas às quais ele não tem acesso. Embora isso, por si só, não represente uma falha de segurança, fornece informações desnecessárias que podem auxiliar um usuário não autorizado a se conectar a uma exportação NFS.

Você deve desativar `showmount` utilizando o comando ONTAP CLI em nível de SVM:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

Melhores práticas do SolidFire

Aprenda as melhores práticas para configurar o armazenamento SolidFire para Trident.

Criar conta Solidfire

Cada conta SolidFire representa um proprietário de volume único e recebe seu próprio conjunto de credenciais do Protocolo de Autenticação por Desafio e Aperto de Mão (CHAP). Você pode acessar os volumes atribuídos a uma conta usando o nome da conta e as credenciais CHAP correspondentes ou por meio de um grupo de acesso a volumes. Uma conta pode ter até dois mil volumes atribuídos a ela, mas um volume só pode pertencer a uma conta.

Criar uma política de QoS

Utilize as políticas de Qualidade de Serviço (QoS) do SolidFire se desejar criar e salvar uma configuração padronizada de qualidade de serviço que possa ser aplicada a vários volumes.

Você pode definir parâmetros de QoS para cada volume individualmente. O desempenho de cada volume pode ser garantido definindo três parâmetros configuráveis que definem a QoS: IOPS mínimo, IOPS máximo e IOPS de rajada.

Aqui estão os possíveis valores mínimos, máximos e de burst de IOPS para um tamanho de bloco de 4Kb.

Parâmetro IOPS	Definição	Valor mínimo	Valor padrão	Valor máximo (4 KB)
IOPS mínimo	Nível de desempenho garantido para um determinado volume.	50	50	15000
IOPS máximo	O desempenho não ultrapassará esse limite.	50	15000	200.000
IOPS de rajada	Máximo de IOPS permitido em um cenário de pico curto.	50	15000	200.000

Embora os valores de Max IOPS e Burst IOPS possam ser configurados para até 200.000, o desempenho máximo real de um volume é limitado pelo uso do cluster e pelo desempenho de cada nó.

O tamanho do bloco e a largura de banda influenciam diretamente o número de IOPS. À medida que o tamanho dos blocos aumenta, o sistema aumenta a largura de banda para um nível necessário para processar os blocos maiores. À medida que a largura de banda aumenta, o número de IOPS que o sistema consegue atingir diminui. Consulte "Qualidade de serviço SolidFire" Para obter mais informações sobre QoS e desempenho.

Autenticação SolidFire

O Element suporta dois métodos de autenticação: CHAP e Grupos de Acesso por Volume (VAG). O CHAP utiliza o protocolo CHAP para autenticar o host no servidor de backend. Os Grupos de Acesso a Volumes controlam o acesso aos volumes que provisionam. A NetApp recomenda o uso do CHAP para autenticação, pois é mais simples e não possui limites de escalabilidade.

O Trident com o provisionador CSI aprimorado suporta o uso da autenticação CHAP. Os VAGs devem ser usados apenas no modo de operação tradicional, não CSI.

A autenticação CHAP (verificação de que o iniciador é o usuário do volume pretendido) é suportada apenas com controle de acesso baseado em conta. Se você estiver usando CHAP para autenticação, duas opções estão disponíveis: CHAP unidirecional e CHAP bidirecional. O CHAP unidirecional autentica o acesso ao

volume usando o nome da conta SolidFire e o segredo do iniciador. A opção CHAP bidirecional oferece a maneira mais segura de autenticar o volume, pois o volume autentica o host por meio do nome da conta e do segredo do iniciador, e então o host autentica o volume por meio do nome da conta e do segredo do destino.

No entanto, se o CHAP não puder ser ativado e os VAGs forem necessários, crie o grupo de acesso e adicione os iniciadores de host e os volumes ao grupo de acesso. Cada IQN adicionado a um grupo de acesso pode acessar todos os volumes do grupo, com ou sem autenticação CHAP. Se o iniciador iSCSI estiver configurado para usar autenticação CHAP, o controle de acesso baseado em conta será utilizado. Se o iniciador iSCSI não estiver configurado para usar autenticação CHAP, o controle de acesso do Grupo de Acesso ao Volume será utilizado.

Onde posso encontrar mais informações?

Abaixo, encontra-se listada parte da documentação sobre as melhores práticas. Pesquise o "Biblioteca NetApp" para as versões mais recentes.

- ONTAP*
- "Guia de Melhores Práticas e Implementação do NFS"
- "Administração SAN"(para iSCSI)
- "Configuração iSCSI Express para RHEL"

Software Elemento

- "Configurando o SolidFire para Linux"
- NetApp HCI*
- "Pré-requisitos para implantação do NetApp HCI"
- "Acesse o mecanismo de implantação da NetApp"

Informações sobre as melhores práticas de aplicação

- "Melhores práticas para MySQL no ONTAP"
- "Melhores práticas para MySQL no SolidFire"
- "NetApp SolidFire e Cassandra"
- "Melhores práticas da Oracle no SolidFire"
- "Melhores práticas do PostgreSQL no SolidFire"

Nem todas as aplicações possuem diretrizes específicas; é importante trabalhar com sua equipe da NetApp e utilizar o… "Biblioteca NetApp" Para encontrar a documentação mais atualizada.

Integrar Trident

Para integrar o Trident, os seguintes elementos de design e arquitetura precisam ser integrados: seleção e implantação de drivers, design de classes de armazenamento, design de pools virtuais, impactos do Persistent Volume Claim (PVC) no provisionamento de armazenamento, operações de volume e implantação de serviços OpenShift usando o Trident.

Seleção e alocação de motoristas

Selecione e implemente um driver de backend para o seu sistema de armazenamento.

Drivers de backend ONTAP

Os drivers de backend do ONTAP são diferenciados pelo protocolo utilizado e pela forma como os volumes são provisionados no sistema de armazenamento. Portanto, considere cuidadosamente ao decidir qual driver implantar.

Em um nível mais elevado, se sua aplicação possui componentes que necessitam de armazenamento compartilhado (vários pods acessando o mesmo PVC), os drivers baseados em NAS seriam a escolha padrão, enquanto os drivers iSCSI baseados em bloco atendem às necessidades de armazenamento não compartilhado. Escolha o protocolo com base nos requisitos da aplicação e no nível de conforto das equipes de armazenamento e infraestrutura. De um modo geral, existe pouca diferença entre eles para a maioria das aplicações, pelo que a decisão costuma basear-se na necessidade ou não de armazenamento partilhado (em que mais do que um pod necessitará de acesso simultâneo).

Os drivers de backend ONTAP disponíveis são:

- `ontap-nas`Cada PV provisionado é um FlexVolume ONTAP completo.
- `ontap-nas-economy`Cada PV provisionado é uma qtree, com um número configurável de qtrees por FlexVolume (o padrão é 200).
- `ontap-nas-flexgroup`Cada PV provisionado como um ONTAP FlexGroup completo e todos os agregados atribuídos a um SVM são utilizados.
- `ontap-san`Cada PV provisionado é um LUN dentro de seu próprio FlexVolume.
- `ontap-san-economy`Cada PV provisionado é um LUN, com um número configurável de LUNs por FlexVolume (o padrão é 100).

A escolha entre os três drivers NAS tem algumas implicações nos recursos disponibilizados para o aplicativo.

Observe que, nas tabelas abaixo, nem todas as funcionalidades estão expostas através do Trident. Algumas configurações devem ser aplicadas pelo administrador de armazenamento após o provisionamento, caso essa funcionalidade seja desejada. As notas de rodapé em sobrescrito distinguem a funcionalidade de cada recurso e driver.

Drivers ONTAP NAS	Instantâneos	Clones	Políticas de exportação dinâmicas	Conexão múltipla	Qualidade de Serviço	Redimensionar	Replicação
`ontap-nas`	Sim	Sim	Sim, nota de rodapé:5[]	Sim	Sim, nota de rodapé:1[]	Sim	Sim, nota de rodapé:1[]
`ontap-nas-economy`	NO [3]	NO [3]	Sim, nota de rodapé:5[]	Sim	NO [3]	Sim	NO [3]
`ontap-nas-flexgroup`	Sim, nota de rodapé:1[]	NÃO	Sim, nota de rodapé:5[]	Sim	Sim, nota de rodapé:1[]	Sim	Sim, nota de rodapé:1[]

A Trident oferece 2 drivers SAN para ONTAP, cujas funcionalidades são apresentadas abaixo.

Motoristas ONTAP SAN	Instantâneos	Clones	Conexão múltipla	CHAP bidirecional	Qualidade de Serviço	Redimensionar	Replicação
`ontap-san`	Sim	Sim	Sim, nota de rodapé:4[]	Sim	Sim, nota de rodapé:1[]	Sim	Sim, nota de rodapé:1[]
`ontap-san-economy`	Sim	Sim	Sim, nota de rodapé:4[]	Sim	NO [3]	Sim	NO [3]

> Nota de rodapé para as tabelas acima: SimNota1[]: Não gerenciado pelo Trident SimNota2[]: Gerenciado pelo Trident, mas não granularmente em PV NãoNota3[]: Não gerenciado pelo Trident e não granularmente em PV SimNota4[]: Suportado para volumes de bloco bruto SimNota5[]: Suportado pelo Trident

Os recursos que não são granulares em relação ao PV são aplicados a todo o FlexVolume e todos os PVs (ou seja, qtrees ou LUNs em FlexVols compartilhados) compartilharão um agendamento comum.

Como podemos ver nas tabelas acima, grande parte da funcionalidade entre os `ontap-nas` e `ontap-nas-economy` é o mesmo. No entanto, porque o `ontap-nas-economy` O driver limita a capacidade de controlar o agendamento em nível de granularidade por PV, o que pode afetar, em particular, o planejamento de recuperação de desastres e backup. Para equipes de desenvolvimento que desejam aproveitar a funcionalidade de clonagem de PVC no armazenamento ONTAP , isso só é possível ao usar o `ontap-nas` , `ontap-san` ou `ontap-san-economy` motoristas.

ⓘ O `solidfire-san` O driver também é capaz de clonar PVCs.

Drivers de backend do Cloud Volumes ONTAP

O Cloud Volumes ONTAP oferece controle de dados juntamente com recursos de armazenamento de nível empresarial para diversos casos de uso, incluindo compartilhamento de arquivos e armazenamento em nível de bloco, atendendo a protocolos NAS e SAN (NFS, SMB/CIFS e iSCSI). Os drivers compatíveis com o Cloud Volume ONTAP são: `ontap-nas` , `ontap-nas-economy` , `ontap-san` e `ontap-san-economy` . Estas opções são aplicáveis ao Cloud Volume ONTAP para Azure e ao Cloud Volume ONTAP para GCP.

Drivers de backend do Amazon FSx para ONTAP

O Amazon FSx for NetApp ONTAP permite que você aproveite os recursos, o desempenho e as capacidades administrativas da NetApp com os quais você já está familiarizado, ao mesmo tempo que desfruta da simplicidade, agilidade, segurança e escalabilidade do armazenamento de dados na AWS. O FSx para ONTAP oferece suporte a muitos recursos do sistema de arquivos ONTAP e APIs de administração. Os drivers compatíveis com o Cloud Volume ONTAP são: `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` e `ontap-san-economy` .

Drivers de backend NetApp HCI/ SolidFire

O `solidfire-san` O driver usado com as plataformas NetApp HCI/ SolidFire ajuda o administrador a configurar um backend Element para o Trident com base nos limites de QoS. Se você deseja projetar seu

backend para definir limites de QoS específicos nos volumes provisionados pelo Trident, use o `type` parâmetro no arquivo de backend. O administrador também pode restringir o tamanho do volume que pode ser criado no armazenamento usando o `limitVolumeSize` parâmetro. Atualmente, recursos de armazenamento do Element, como redimensionamento e replicação de volumes, não são suportados pelo sistema. `solidfire-san` motorista. Essas operações devem ser realizadas manualmente através da interface web do Element Software.

Driver SolidFire	Instantâneos	Clones	Conexão múltipla	RACHAR	Qualidade de Serviço	Redimensionar	Replicação
`solidfire-san`	Sim	Sim	Sim, nota de rodapé:2[]	Sim	Sim	Sim	Sim, nota de rodapé:1[]

> Nota de rodapé: SimNota de rodapé:1[]: Não gerenciado pelo Trident SimNota de rodapé:2[]: Compatível com volumes de bloco bruto

Drivers de back-end do Azure NetApp Files

Trident usa o `azure-netapp-files` motorista para gerenciar o"Azure NetApp Files" serviço.

Mais informações sobre este driver e como configurá-lo podem ser encontradas em"Configuração do backend Trident para Azure NetApp Files" .

Driver de Azure NetApp Files	Instantâneos	Clones	Conexão múltipla	Qualidade de Serviço	Expandir	Replicação
`azure-netapp-files`	Sim	Sim	Sim	Sim	Sim	Sim, nota de rodapé:1[]

> Nota de rodapé: Sim [1]: Não gerenciado pela Trident

Cloud Volumes Service no driver de backend do Google Cloud

Trident usa o `gcp-cvs` Driver para conectar com o Cloud Volumes Service no Google Cloud.

O `gcp-cvs` O driver utiliza pools virtuais para abstrair o backend e permitir que o Trident determine o posicionamento dos volumes. O administrador define os pools virtuais no `backend.json` arquivos. As classes de armazenamento usam seletores para identificar pools virtuais por rótulo.

- Se os pools virtuais forem definidos no backend, o Trident tentará criar um volume nos pools de armazenamento do Google Cloud aos quais esses pools virtuais estão limitados.
- Caso os pools virtuais não estejam definidos no backend, o Trident selecionará um pool de armazenamento do Google Cloud dentre os pools de armazenamento disponíveis na região.

Para configurar o backend do Google Cloud no Trident, você deve especificar `projectNumber` , `apiRegion` , e `apiKey` no arquivo de backend. Você pode encontrar o número do projeto no console do Google Cloud. A chave da API é obtida do arquivo de chave privada da conta de serviço que você criou ao configurar o acesso à API do Cloud Volumes Service no Google Cloud.

Para obter detalhes sobre os tipos e níveis de serviço do Cloud Volumes Service no Google Cloud,

consulte:"Saiba mais sobre o suporte do Trident para CVS em GCP" .

Driver do Cloud Volumes Service para Google Cloud	Instantâneos	Clones	Conexão múltipla	Qualidade de Serviço	Expandir	Replicação
`gcp-cvs`	Sim	Sim	Sim	Sim	Sim	Disponível apenas no tipo de serviço CVS-Performance.

Notas de replicação

- A replicação não é gerenciada pelo Trident.
- O clone será criado no mesmo pool de armazenamento que o volume de origem.

projeto de classe de armazenamento

É necessário configurar e aplicar classes de armazenamento individuais para criar um objeto de classe de armazenamento do Kubernetes. Esta seção discute como projetar uma classe de armazenamento para sua aplicação.

Utilização específica do backend

A filtragem pode ser usada dentro de um objeto de classe de armazenamento específico para determinar qual pool de armazenamento ou conjunto de pools deve ser usado com essa classe de armazenamento específica. É possível definir três conjuntos de filtros na classe de armazenamento: `storagePools` , `additionalStoragePools` , e/ou `excludeStoragePools` .

O `storagePools` O parâmetro ajuda a restringir o armazenamento ao conjunto de pools que correspondem a quaisquer atributos especificados. O `additionalStoragePools` O parâmetro é usado para estender o conjunto de pools que o Trident usa para provisionamento, juntamente com o conjunto de pools selecionados pelos atributos e `storagePools` parâmetros. Você pode usar qualquer um dos parâmetros individualmente ou ambos em conjunto para garantir que o conjunto apropriado de pools de armazenamento seja selecionado.

O `excludeStoragePools` O parâmetro é usado para excluir especificamente o conjunto listado de pools que correspondem aos atributos.

Emular políticas de QoS

Se você deseja projetar classes de armazenamento para emular políticas de Qualidade de Serviço, crie uma classe de armazenamento com o `media` atributo como `hdd` ou `ssd` . Com base em `media` Com base no atributo mencionado na classe de armazenamento, o Trident selecionará o backend apropriado que atende ao serviço. `hdd` ou `ssd` os agregados correspondem ao atributo de mídia e, em seguida, direcionam o provisionamento dos volumes para o agregado específico. Portanto, podemos criar uma classe de armazenamento PREMIUM que teria `media` atributo definido como `ssd` que poderia ser classificada como a política de QoS PREMIUM. Podemos criar outra classe de armazenamento STANDARD que teria o atributo de mídia definido como `hdd`, o que poderia ser classificado como a política de QoS STANDARD. Também podemos usar o atributo "IOPS" na classe de armazenamento para redirecionar o provisionamento para um dispositivo Element que pode ser definido como uma Política de QoS.

Utilize o backend com base em funcionalidades específicas.

As classes de armazenamento podem ser projetadas para direcionar o provisionamento de volumes em um backend específico, onde recursos como provisionamento fino e espesso, snapshots, clones e criptografia estão habilitados. Para especificar qual armazenamento usar, crie classes de armazenamento que especifiquem o backend apropriado com o recurso necessário habilitado.

Piscinas virtuais

Pools virtuais estão disponíveis para todos os backends do Trident . Você pode definir pools virtuais para qualquer backend, usando qualquer driver fornecido Trident .

Os pools virtuais permitem que um administrador crie um nível de abstração sobre os backends, que podem ser referenciados por meio de classes de armazenamento, para maior flexibilidade e alocação eficiente de volumes nos backends. Diferentes backends podem ser definidos com a mesma classe de serviço. Além disso, é possível criar vários pools de armazenamento no mesmo backend, mas com características diferentes. Quando uma classe de armazenamento é configurada com um seletor contendo rótulos específicos, o Trident escolhe um backend que corresponda a todos os rótulos do seletor para alocar o volume. Se os rótulos do seletor de Classe de Armazenamento corresponderem a vários pools de armazenamento, o Trident escolherá um deles para provisionar o volume.

Projeto de piscina virtual

Ao criar um backend, geralmente é possível especificar um conjunto de parâmetros. Era impossível para o administrador criar outro backend com as mesmas credenciais de armazenamento e um conjunto diferente de parâmetros. Com a introdução de pools virtuais, esse problema foi amenizado. Um pool virtual é uma abstração de nível introduzida entre o backend e a Classe de Armazenamento do Kubernetes para que o administrador possa definir parâmetros junto com rótulos que podem ser referenciados por meio das Classes de Armazenamento do Kubernetes como um seletor, de forma independente do backend. Pools virtuais podem ser definidos para todos os backends NetApp suportados com o Trident. Essa lista inclui SolidFire/ NetApp HCI, ONTAP, Cloud Volumes Service no GCP, bem como Azure NetApp Files.

Ao definir pools virtuais, recomenda-se não tentar reorganizar a ordem dos pools virtuais existentes em uma definição de backend. Também é recomendável não editar/modificar os atributos de um pool virtual existente e, em vez disso, definir um novo pool virtual.

Emulação de diferentes níveis de serviço/QoS

É possível projetar pools virtuais para emular classes de serviço. Usando a implementação de pool virtual para o Cloud Volume Service para Azure NetApp Files, vamos examinar como podemos configurar diferentes classes de serviço. Configure o backend do Azure NetApp Files com vários rótulos, representando diferentes níveis de desempenho. Definir `servicelevel` Atribua o aspecto ao nível de desempenho apropriado e adicione outros aspectos necessários a cada rótulo. Agora, crie diferentes classes de armazenamento do Kubernetes que serão mapeadas para diferentes pools virtuais. Usando o `parameters.selector` No campo `StorageClass`, cada `StorageClass` especifica quais pools virtuais podem ser usados para hospedar um volume.

Atribuir um conjunto específico de aspectos

É possível projetar vários pools virtuais com um conjunto específico de características a partir de um único backend de armazenamento. Para isso, configure o backend com vários rótulos e defina os aspectos necessários em cada rótulo. Agora, crie diferentes classes de armazenamento do Kubernetes usando o `parameters.selector` campo que mapearia diferentes pools virtuais. Os volumes provisionados no backend terão os aspectos definidos no pool virtual escolhido.

Características do PVC que afetam o fornecimento de armazenamento

Alguns parâmetros além da classe de armazenamento solicitada podem afetar o processo de decisão de provisionamento do Trident ao criar um PVC.

Modo de acesso

Ao solicitar armazenamento por meio de um PVC, um dos campos obrigatórios é o modo de acesso. O modo desejado pode afetar o servidor selecionado para hospedar a solicitação de armazenamento.

O Trident tentará encontrar uma correspondência entre o protocolo de armazenamento utilizado e o método de acesso especificado, de acordo com a seguinte matriz. Isso é independente da plataforma de armazenamento subjacente.

	LerEscreverUmaVez	Somente leitura	LerEscreverMuitos
iSCSI	Sim	Sim	Sim (bloco bruto)
NFS	Sim	Sim	Sim

Uma solicitação de um PVC ReadWriteMany enviada a uma implementação do Trident sem um backend NFS configurado resultará na ausência de provisionamento de volume. Por essa razão, o solicitante deve usar o modo de acesso apropriado para sua aplicação.

Operações de volume

Modificar volumes persistentes

Os volumes persistentes são, com duas exceções, objetos imutáveis no Kubernetes. Uma vez criada, a política de recuperação e o tamanho podem ser modificados. No entanto, isso não impede que alguns aspectos do volume sejam modificados fora do Kubernetes. Isso pode ser desejável para personalizar o volume para aplicações específicas, garantir que a capacidade não seja consumida acidentalmente ou simplesmente mover o volume para um controlador de armazenamento diferente por qualquer motivo.

> Os provisionadores integrados do Kubernetes não suportam operações de redimensionamento de volumes para PVs NFS, iSCSI ou FC neste momento. O Trident suporta a expansão de volumes NFS, iSCSI e FC.

Os detalhes de conexão do sistema fotovoltaico não podem ser modificados após a sua criação.

Criar snapshots de volume sob demanda

O Trident suporta a criação de snapshots de volume sob demanda e a criação de PVCs a partir de snapshots usando a estrutura CSI. Os snapshots fornecem um método conveniente para manter cópias pontuais dos dados e têm um ciclo de vida independente do PV de origem no Kubernetes. Essas imagens podem ser usadas para clonar PVCs.

Criar volumes a partir de snapshots

O Trident também suporta a criação de PersistentVolumes a partir de snapshots de volume. Para fazer isso, basta criar um PersistentVolumeClaim e mencionar o `datasource` como o instantâneo necessário a partir do qual o volume precisa ser criado. O Trident lidará com esse PVC criando um volume com os dados presentes no snapshot. Com esse recurso, é possível duplicar dados entre regiões, criar ambientes de teste, substituir integralmente um volume de produção danificado ou corrompido, ou recuperar arquivos e diretórios

específicos e transferi-los para outro volume conectado.

Mover volumes no cluster

Os administradores de armazenamento têm a capacidade de mover volumes entre agregados e controladores no cluster ONTAP sem interromper o funcionamento do consumidor de armazenamento. Essa operação não afeta o Trident nem o cluster Kubernetes, desde que o agregado de destino seja um ao qual a SVM que o Trident está usando tenha acesso. É importante ressaltar que, se o agregado foi adicionado recentemente ao SVM, o backend precisará ser atualizado, adicionando-o novamente ao Trident. Isso fará com que o Trident reinvente o SVM para que o novo agregado seja reconhecido.

No entanto, a transferência de volumes entre sistemas de backend não é suportada automaticamente pelo Trident. Isso inclui transferências entre SVMs no mesmo cluster, entre clusters diferentes ou para uma plataforma de armazenamento diferente (mesmo que esse sistema de armazenamento esteja conectado ao Trident).

Se um volume for copiado para outro local, o recurso de importação de volume poderá ser usado para importar os volumes atuais para o Trident.

Expandir volumes

O Trident suporta o redimensionamento de PVs NFS, iSCSI e FC. Isso permite que os usuários redimensionem seus volumes diretamente através da camada Kubernetes. A expansão de volume é possível para todas as principais plataformas de armazenamento da NetApp , incluindo ONTAP, SolidFire/ NetApp HCI e backends do Cloud Volumes Service . Para permitir uma possível expansão posterior, defina `allowVolumeExpansion` para `true` na sua StorageClass associada ao volume. Sempre que o Volume Persistente precisar ser redimensionado, edite o `spec.resources.requests.storage` anotação na Solicitação de Volume Persistente referente ao tamanho de volume necessário. O Trident se encarregará automaticamente de redimensionar o volume no cluster de armazenamento.

Importar um volume existente para o Kubernetes

A importação de volumes permite importar um volume de armazenamento existente para um ambiente Kubernetes. Isso é atualmente suportado pelo `ontap-nas` , `ontap-nas-flexgroup` , `solidfire-san` , `azure-netapp-files` , e `gcp-cvs` motoristas. Essa funcionalidade é útil ao migrar um aplicativo existente para o Kubernetes ou em cenários de recuperação de desastres.

Ao usar o ONTAP e `solidfire-san` motoristas, usem o comando `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` Importar um volume existente para o Kubernetes para ser gerenciado pelo Trident. O arquivo PVC YAML ou JSON usado no comando de importação de volume aponta para uma classe de armazenamento que identifica o Trident como o provisionador. Ao usar um backend NetApp HCI/ SolidFire , certifique-se de que os nomes dos volumes sejam exclusivos. Se os nomes dos volumes estiverem duplicados, clone o volume para um nome exclusivo para que o recurso de importação de volumes possa distingui-los.

Se o `azure-netapp-files` ou `gcp-cvs` O driver está sendo usado, use o comando `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` Importar o volume para o Kubernetes para ser gerenciado pelo Trident. Isso garante uma referência de volume única.

Ao executar o comando acima, o Trident localizará o volume no servidor e lerá seu tamanho. Isso adicionará automaticamente (e sobrescreverá, se necessário) o tamanho do volume do PVC configurado. Em seguida, o Trident cria o novo PV e o Kubernetes vincula o PVC ao PV.

Caso um contêiner tenha sido implantado de forma a exigir o PVC importado específico, ele permanecerá em estado pendente até que o par PVC/PV seja vinculado por meio do processo de importação em volume. Após

a união do par PVC/PV, o contêiner deverá subir, desde que não haja outros problemas.

Serviço de registro

A implementação e o gerenciamento do armazenamento para o registro foram documentados em"netapp.io" no"blog" .

Serviço de registro

Assim como outros serviços do OpenShift, o serviço de registro de logs é implantado usando o Ansible com parâmetros de configuração fornecidos pelo arquivo de inventário, também conhecido como hosts, fornecido ao playbook. Serão abordados dois métodos de instalação: a implementação do registro de logs durante a instalação inicial do OpenShift e a implementação do registro de logs após a instalação do OpenShift.

> A partir da versão 3.9 do Red Hat OpenShift, a documentação oficial recomenda não utilizar NFS para o serviço de registro de logs devido a preocupações com a corrupção de dados. Isso se baseia nos testes que a Red Hat fez com seus produtos. O servidor ONTAP NFS não apresenta esses problemas e pode facilmente suportar uma implementação de registro de logs. Em última análise, a escolha do protocolo para o serviço de registro de logs é sua, mas saiba que ambos funcionarão perfeitamente com as plataformas NetApp e não há motivo para evitar o NFS se essa for sua preferência.

Se você optar por usar NFS com o serviço de registro de logs, precisará configurar a variável Ansible. `openshift_enable_unsupported_configurations` para `true` para evitar que o instalador falhe.

Começar

O serviço de registro de logs pode, opcionalmente, ser implementado tanto para aplicativos quanto para as operações principais do próprio cluster OpenShift. Se optar por implementar o registro de operações, especifique a variável. `openshift_logging_use_ops` como `true` Serão criadas duas instâncias do serviço. As variáveis que controlam a instância de registro de operações contêm "ops", enquanto a instância para aplicativos não contém.

Configurar as variáveis do Ansible de acordo com o método de implantação é importante para garantir que o armazenamento correto seja utilizado pelos serviços subjacentes. Vamos analisar as opções para cada um dos métodos de implantação.

> As tabelas abaixo contêm apenas as variáveis relevantes para a configuração de armazenamento relacionada ao serviço de registro de logs. Você pode encontrar outras opções em"Documentação de registro do Red Hat OpenShift" que devem ser revisados, configurados e utilizados de acordo com a sua implementação.

As variáveis na tabela abaixo farão com que o playbook do Ansible crie um PV (Perfil de Variável) e um PVC (Perfil de Variável Contínua) para o serviço de registro de logs, utilizando os detalhes fornecidos. Este método é significativamente menos flexível do que usar o playbook de instalação de componentes após a instalação do OpenShift; no entanto, se você já tiver volumes disponíveis, é uma opção viável.

Variável	Detalhes
`openshift_logging_storage_kind`	Definir para `nfs` Para que o instalador crie um PV NFS para o serviço de registro de logs.

Variável	Detalhes
`openshift_logging_storage_host`	O nome do host ou o endereço IP do host NFS. Este valor deve ser definido como o dataLIF da sua máquina virtual.
`openshift_logging_storage_nfs_directory`	O caminho de montagem para a exportação NFS. Por exemplo, se o volume for dividido como `/openshift_logging` , você usaria esse caminho para essa variável.
`openshift_logging_storage_volume_name`	O nome, por exemplo `pv_ose_logs` , do PV para criar.
`openshift_logging_storage_volume_size`	O tamanho da exportação NFS, por exemplo `100Gi` .

Se o seu cluster OpenShift já estiver em execução e, portanto, o Trident já tiver sido implantado e configurado, o instalador poderá usar o provisionamento dinâmico para criar os volumes. As seguintes variáveis precisarão ser configuradas.

Variável	Detalhes
`openshift_logging_es_pvc_dynamic`	Defina como verdadeiro para usar volumes provisionados dinamicamente.
`openshift_logging_es_pvc_storage_class_name`	O nome da classe de armazenamento que será utilizada no PVC.
`openshift_logging_es_pvc_size`	O tamanho do volume solicitado no PVC.
`openshift_logging_es_pvc_prefix`	Um prefixo para os PVCs usados pelo serviço de registro de dados.
`openshift_logging_es_ops_pvc_dynamic`	Definir para `true` Utilizar volumes provisionados dinamicamente para a instância de registro de operações.
`openshift_logging_es_ops_pvc_storage_class_name`	O nome da classe de armazenamento para a instância de registro de operações.
`openshift_logging_es_ops_pvc_size`	O tamanho da solicitação de volume para a instância de operações.
`openshift_logging_es_ops_pvc_prefix`	Um prefixo para os PVCs da instância de operações.

Implante a pilha de registro de logs.

Se você estiver implementando o registro de logs como parte do processo inicial de instalação do OpenShift, basta seguir o processo de implementação padrão. O Ansible irá configurar e implantar os serviços e objetos do OpenShift necessários para que o serviço esteja disponível assim que o Ansible for concluído.

No entanto, se você estiver realizando a implantação após a instalação inicial, o playbook do componente precisará ser usado pelo Ansible. Este processo pode variar ligeiramente dependendo da versão do OpenShift, portanto, certifique-se de ler e seguir as instruções."Documentação do Red Hat OpenShift Container Platform 3.11" para a sua versão.

Serviço de métricas

O serviço de métricas fornece informações valiosas ao administrador sobre o status, a utilização de recursos e a disponibilidade do cluster OpenShift. Também é necessário para a funcionalidade de dimensionamento automático de pods e muitas organizações usam dados do serviço de métricas para seus aplicativos de cobrança e/ou demonstração de resultados.

Assim como no serviço de registro de logs e no OpenShift como um todo, o Ansible é usado para implantar o serviço de métricas. Assim como o serviço de registro de logs, o serviço de métricas pode ser implantado durante a configuração inicial do cluster ou após sua entrada em operação, utilizando o método de instalação de componentes. As tabelas a seguir contêm as variáveis importantes na configuração do armazenamento persistente para o serviço de métricas.

As tabelas abaixo contêm apenas as variáveis relevantes para a configuração de armazenamento relacionada ao serviço de métricas. Existem muitas outras opções na documentação que devem ser analisadas, configuradas e utilizadas de acordo com a sua implementação.

Variável	Detalhes
`openshift_metrics_storage_kind`	Definir para `nfs` Para que o instalador crie um PV NFS para o serviço de registro de logs.
`openshift_metrics_storage_host`	O nome do host ou o endereço IP do host NFS. Este valor deve ser definido como o dataLIF da sua SVM.
`openshift_metrics_storage_nfs_directory`	O caminho de montagem para a exportação NFS. Por exemplo, se o volume for dividido como `/openshift_metrics` , você usaria esse caminho para essa variável.
`openshift_metrics_storage_volume_name`	O nome, por exemplo `pv_ose_metrics` , do PV para criar.
`openshift_metrics_storage_volume_size`	O tamanho da exportação NFS, por exemplo `100Gi` .

Se o seu cluster OpenShift já estiver em execução e, portanto, o Trident já tiver sido implantado e configurado, o instalador poderá usar o provisionamento dinâmico para criar os volumes. As seguintes variáveis precisarão ser configuradas.

Variável	Detalhes
`openshift_metrics_cassandra_pvc_prefix`	Um prefixo a ser usado para as métricas PVCs.
`openshift_metrics_cassandra_pvc_size`	O tamanho dos volumes a serem solicitados.
`openshift_metrics_cassandra_storage_type`	O tipo de armazenamento a ser usado para métricas; isso deve ser definido como dinâmico para que o Ansible crie PVCs com a classe de armazenamento apropriada.
`openshift_metrics_cassanda_pvc_storage_class_name`	O nome da classe de armazenamento a ser usada.

Implante o serviço de métricas

Com as variáveis Ansible apropriadas definidas no seu arquivo hosts/inventory, implante o serviço usando o Ansible. Se você estiver realizando a implantação durante a instalação do OpenShift, o PV será criado e utilizado automaticamente. Se você estiver realizando a implantação usando os playbooks de componentes, após a instalação do OpenShift, o Ansible cria todos os PVCs necessários e, depois que o Trident provisiona o armazenamento para eles, implanta o serviço.

As variáveis acima, bem como o processo de implantação, podem mudar a cada versão do OpenShift. Certifique-se de revisar e seguir as instruções."Guia de implantação do OpenShift da Red Hat" para a sua versão, de forma que esteja configurada para o seu ambiente.

Proteção de dados e recuperação de desastres

Saiba mais sobre as opções de proteção e recuperação para o Trident e para os volumes criados com o Trident. Você deve ter uma estratégia de proteção e recuperação de dados para cada aplicação que exija persistência.

Replicação e recuperação do Trident

Você pode criar um backup para restaurar o Trident em caso de desastre.

Replicação do Trident

O Trident usa CRDs do Kubernetes para armazenar e gerenciar seu próprio estado e o cluster Kubernetes etcd para armazenar seus metadados.

Passos

1. Faça backup do cluster Kubernetes etcd usando"Kubernetes: Fazendo backup de um cluster etcd" .
2. Coloque os artefatos de backup em um FlexVol volume.

> ⓘ A NetApp recomenda que você proteja a SVM onde o FlexVol reside com uma relação SnapMirror com outra SVM.

Recuperação Trident

Utilizando CRDs do Kubernetes e o snapshot do etcd do cluster Kubernetes, você pode recuperar o Trident.

Passos

1. A partir da SVM de destino, monte o volume que contém os arquivos de dados e certificados do Kubernetes etcd no host que será configurado como nó mestre.
2. Copie todos os certificados necessários referentes ao cluster Kubernetes em `/etc/kubernetes/pki` e os arquivos de membro do etcd em `/var/lib/etcd` .
3. Restaure o cluster Kubernetes a partir do backup do etcd usando"Kubernetes: Restaurando um cluster etcd" .
4. Correr `kubectl get crd` Para verificar se todos os recursos personalizados do Trident foram ativados e recuperar os objetos do Trident para confirmar se todos os dados estão disponíveis.

Replicação e recuperação de SVM

O Trident não consegue configurar relações de replicação; no entanto, o administrador de armazenamento pode usar "ONTAP SnapMirror" para replicar uma SVM.

Em caso de desastre, você pode ativar o SVM de destino do SnapMirror para começar a fornecer dados. Você poderá retornar ao servidor primário quando os sistemas forem restaurados.

Sobre esta tarefa

Ao usar o recurso de replicação SVM do SnapMirror, considere o seguinte:

- Você deve criar um backend distinto para cada SVM com SVM-DR habilitado.
- Configure as classes de armazenamento para selecionar os backends replicados somente quando necessário, evitando assim o provisionamento de volumes que não precisam de replicação nos backends que suportam SVM-DR.
- Os administradores de aplicativos devem compreender o custo e a complexidade adicionais associados à replicação e considerar cuidadosamente seu plano de recuperação antes de iniciar esse processo.

Replicação SVM

Você pode usar"ONTAP: Replicação SnapMirror SVM" para criar a relação de replicação SVM.

O SnapMirror permite que você defina opções para controlar o que será replicado. Você precisará saber quais opções selecionou ao realizar a operação.Recuperação de SVM usando Trident .

- "-preservar identidade verdadeiro"replica toda a configuração da SVM.
- "-descartar-configurações rede"Exclui LIFs e configurações de rede relacionadas.
- "-preservar identidade falso"replica apenas os volumes e a configuração de segurança.

Recuperação de SVM usando Trident

O Trident não detecta automaticamente falhas no SVM. Em caso de desastre, o administrador pode iniciar manualmente o failover do Trident para a nova SVM.

Passos

1. Cancele as transferências SnapMirror agendadas e em andamento, interrompa a relação de replicação, pare o SVM de origem e, em seguida, ative o SVM de destino do SnapMirror .
2. Se você especificou `-identity-preserve false` ou `-discard-config network` Ao configurar a replicação do seu SVM, atualize o `managementLIF` e `dataLIF` no arquivo de definição do backend Trident .
3. Confirmar `storagePrefix` está presente no arquivo de definição do backend Trident . Este parâmetro não pode ser alterado. omitindo `storagePrefix` fará com que a atualização do backend falhe.
4. Atualize todos os backends necessários para refletir o novo nome da SVM de destino usando:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Se você especificou `-identity-preserve false` ou `discard-config network` Você deve reiniciar todos os pods da aplicação.

Se você especificou `-identity-preserve true` Todos os volumes provisionados pelo Trident começam a fornecer dados quando o SVM de destino é ativado.

Replicação e recuperação de volume

O Trident não consegue configurar relações de replicação SnapMirror ; no entanto, o administrador de armazenamento pode usar"Replicação e recuperação do SnapMirror do ONTAP" para replicar volumes criados pelo Trident.

Em seguida, você pode importar os volumes recuperados para o Trident usando"importação de volume tridentctl" .

A importação não é suportada em `ontap-nas-economy` , `ontap-san-economy` , ou `ontap-flexgroup-economy` motoristas.

Proteção de dados instantâneos

Você pode proteger e restaurar dados usando:

- Um controlador de snapshots externo e CRDs para criar snapshots de volumes persistentes (PVs) do Kubernetes.

  "Instantâneos de volume"

- Os snapshots do ONTAP permitem restaurar todo o conteúdo de um volume ou recuperar arquivos ou LUNs individuais.

  "Capturas instantâneas do ONTAP"

Segurança

Segurança

Utilize as recomendações listadas aqui para garantir a segurança da sua instalação do Trident .

Execute o Trident em seu próprio espaço de nomes.

É importante impedir que aplicativos, administradores de aplicativos, usuários e aplicativos de gerenciamento acessem as definições de objetos do Trident ou os pods para garantir um armazenamento confiável e bloquear possíveis atividades maliciosas.

Para separar os outros aplicativos e usuários do Trident, sempre instale o Trident em seu próprio namespace do Kubernetes.(`trident`). Ao colocar o Trident em seu próprio namespace, garante-se que apenas a equipe administrativa do Kubernetes tenha acesso ao pod do Trident e aos artefatos (como segredos de backend e CHAP, se aplicável) armazenados nos objetos CRD do namespace. Você deve garantir que apenas administradores tenham acesso ao namespace Trident e, portanto, acesso ao… `tridentctl` aplicativo.

Utilize a autenticação CHAP com backends SAN do ONTAP.

O Trident oferece suporte à autenticação baseada em CHAP para cargas de trabalho ONTAP SAN (usando o `ontap-san` e `ontap-san-economy` motoristas). A NetApp recomenda o uso de CHAP bidirecional com Trident para autenticação entre um host e o backend de armazenamento.

Para backends ONTAP que utilizam drivers de armazenamento SAN, o Trident pode configurar CHAP bidirecional e gerenciar nomes de usuário e segredos CHAP através de `tridentctl` . Consulte"Prepare-se para configurar o backend com os drivers ONTAP SAN." Para entender como o Trident configura o CHAP em backends ONTAP .

Utilize a autenticação CHAP com os backends NetApp HCI e SolidFire.

A NetApp recomenda a implementação do CHAP bidirecional para garantir a autenticação entre um host e os backends do NetApp HCI e SolidFire . O Trident utiliza um objeto secreto que inclui duas senhas CHAP por locatário. Quando o Trident é instalado, ele gerencia os segredos CHAP e os armazena em um `tridentvolume` Objeto CR para o respectivo PV. Ao criar um PV, o Trident usa os segredos CHAP para iniciar uma sessão iSCSI e se comunicar com o sistema NetApp HCI e SolidFire via CHAP.

Os volumes criados pelo Trident não estão associados a nenhum Grupo de Acesso a Volumes.

Use Trident com NVE e NAE

O NetApp ONTAP oferece criptografia de dados em repouso para proteger dados confidenciais caso um disco seja roubado, devolvido ou reutilizado. Para mais detalhes, consulte"Visão geral da configuração da criptografia de volume do NetApp" .

- Se o NAE estiver habilitado no backend, qualquer volume provisionado no Trident terá o NAE habilitado.
  - Você pode definir o sinalizador de criptografia NVE para `""` Para criar volumes habilitados para NAE.
- Se o NAE não estiver habilitado no backend, qualquer volume provisionado no Trident terá o NVE habilitado, a menos que o sinalizador de criptografia NVE esteja definido como `false` (o valor padrão) na configuração do backend.

Os volumes criados no Trident em um backend habilitado para NAE devem ser criptografados com NVE ou NAE.

- Você pode definir o sinalizador de criptografia NVE para `true` Na configuração do backend do Trident , é possível substituir a criptografia NAE e usar uma chave de criptografia específica para cada volume.
- Definir o sinalizador de criptografia NVE para `false` Em um backend habilitado para NAE, cria-se um volume habilitado para NAE. Não é possível desativar a criptografia NAE definindo o sinalizador de criptografia NVE como `false` .

- Você pode criar manualmente um volume NVE no Trident definindo explicitamente o sinalizador de criptografia NVE para `true` .

Para obter mais informações sobre as opções de configuração do backend, consulte:

- "Opções de configuração do ONTAP SAN"
- "Opções de configuração do ONTAP NAS"

Configuração Unificada de Chaves do Linux (LUKS)

Você pode habilitar o Linux Unified Key Setup (LUKS) para criptografar volumes ONTAP SAN e ONTAP SAN ECONOMY no Trident. O Trident suporta rotação de senha e expansão de volume para volumes criptografados com LUKS.

No Trident, os volumes criptografados com LUKS usam a cifra e o modo aes-xts-plain64, conforme recomendado por"NIST" .

> A criptografia LUKS não é compatível com sistemas ASA r2. Para obter informações sobre sistemas ASA r2, consulte"Saiba mais sobre os sistemas de armazenamento ASA r2" .

Antes de começar

- Os nós de trabalho devem ter o cryptsetup 2.1 ou superior (mas inferior a 3.0) instalado. Para mais informações, visite"Gitlab: cryptsetup" .
- Por motivos de desempenho, a NetApp recomenda que os nós de trabalho suportem o padrão de criptografia avançada New Instructions (AES-NI). Para verificar a compatibilidade com AES-NI, execute o seguinte comando:

  ```
  grep "aes" /proc/cpuinfo
  ```

  Se nada for retornado, seu processador não suporta AES-NI. Para obter mais informações sobre AES-NI, visite:"Intel: Instruções do Padrão de Criptografia Avançada (AES-NI)" .

Ativar criptografia LUKS

Você pode habilitar a criptografia por volume no lado do host usando o Linux Unified Key Setup (LUKS) para volumes ONTAP SAN e ONTAP SAN ECONOMY.

Passos

1. Defina os atributos de criptografia LUKS na configuração do backend. Para obter mais informações sobre as opções de configuração de backend para ONTAP SAN, consulte"Opções de configuração do ONTAP SAN" .

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. Usar `parameters.selector` Para definir os pools de armazenamento usando criptografia LUKS. Por exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. Crie um segredo que contenha a senha LUKS. Por exemplo:

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

Limitações

Volumes criptografados com LUKS não podem aproveitar a deduplicação e a compressão do ONTAP .

Configuração de backend para importação de volumes LUKS

Para importar um volume LUKS, você deve configurar `luksEncryption` para(`true` nos bastidores. O `luksEncryption` Essa opção informa ao Trident se o volume é compatível com LUKS.(`true`) ou não compatível com LUKS(`false`) conforme mostrado no exemplo a seguir.

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

Configuração de PVC para importação de volumes LUKS

Para importar volumes LUKS dinamicamente, defina a anotação. `trident.netapp.io/luksEncryption` para `true` e inclua uma classe de armazenamento habilitada para LUKS no PVC, conforme mostrado neste exemplo.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

Rotacionar uma senha LUKS

Você pode alternar a senha LUKS e confirmar a alternância.

> Não se esqueça de uma senha até verificar se ela não está mais sendo referenciada por nenhum volume, snapshot ou segredo. Caso a senha referenciada seja perdida, você poderá não conseguir montar o volume e os dados permanecerão criptografados e inacessíveis.

Sobre esta tarefa

A rotação da senha LUKS ocorre quando um pod que monta o volume é criado após a especificação de uma nova senha LUKS. Quando um novo pod é criado, o Trident compara a senha LUKS no volume com a senha ativa no segredo.

- Se a senha no volume não corresponder à senha ativa no segredo, ocorrerá a rotação.
- Se a senha no volume corresponder à senha ativa no segredo, o `previous-luks-passphrase` O parâmetro é ignorado.

Passos

1. Adicione o `node-publish-secret-name` e `node-publish-secret-namespace` Parâmetros da classe de armazenamento. Por exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. Identifique as senhas existentes no volume ou no snapshot.

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Instantâneo

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["A"]
```

3. Atualize o segredo LUKS do volume para especificar as senhas novas e anteriores. Garantir `previous-luke-passphrase-name` e `previous-luks-passphrase` A senha corresponde à senha anterior.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. Crie um novo pod para montar o volume. Isso é necessário para iniciar a rotação.
5. Verifique se a senha foi rotacionada.

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Instantâneo

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

Resultados

A senha foi rotacionada quando apenas a nova senha foi retornada no volume e no snapshot.

> Se duas senhas forem retornadas, por exemplo `luksPassphraseNames: ["B", "A"]`, a rotação está incompleta. Você pode acionar um novo módulo para tentar completar a rotação.

Ativar expansão de volume

Você pode habilitar a expansão de volume em um volume criptografado com LUKS.

Passos

1. Ative o `CSINodeExpandSecret` recurso gate (beta 1.25+). Consulte "Kubernetes 1.25: Use segredos para expansão de volumes CSI orientada a nós" para mais detalhes.
2. Adicione o `node-expand-secret-name` e `node-expand-secret-namespace` Parâmetros da classe de armazenamento. Por exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

Resultados

Ao iniciar a expansão do armazenamento online, o kubelet passa as credenciais apropriadas para o driver.

Criptografia Kerberos em voo

Ao usar a criptografia Kerberos em trânsito, você pode melhorar a segurança do acesso aos dados, habilitando a criptografia para o tráfego entre seu cluster gerenciado e o backend de armazenamento.

O Trident oferece suporte à criptografia Kerberos para ONTAP como backend de armazenamento:

- * ONTAP local* - O Trident oferece suporte à criptografia Kerberos em conexões NFSv3 e NFSv4 de clusters Red Hat OpenShift e Kubernetes upstream para volumes ONTAP locais.

Você pode criar, excluir, redimensionar, criar snapshots, clonar, clonar em modo somente leitura e importar volumes que utilizam criptografia NFS.

Configure a criptografia Kerberos em trânsito com volumes ONTAP locais.

Você pode habilitar a criptografia Kerberos no tráfego de armazenamento entre seu cluster gerenciado e um backend de armazenamento ONTAP local.

> A criptografia Kerberos para tráfego NFS com backends de armazenamento ONTAP locais só é compatível com o uso do `ontap-nas` driver de armazenamento.

Antes de começar

- Certifique-se de ter acesso ao `tridentctl` utilidade.
- Certifique-se de ter acesso de administrador ao backend de armazenamento do ONTAP .
- Certifique-se de saber o nome do(s) volume(s) que você compartilhará do backend de armazenamento ONTAP .
- Certifique-se de ter preparado a máquina virtual de armazenamento ONTAP para suportar a criptografia Kerberos para volumes NFS. Consulte "Habilitar Kerberos em um dataLIF" para obter instruções.
- Certifique-se de que todos os volumes NFSv4 que você usa com criptografia Kerberos estejam configurados corretamente. Consulte a seção Configuração de Domínio NFSv4 da NetApp (página 13) do manual. "Guia de Aprimoramentos e Melhores Práticas do NetApp NFSv4" .

Adicionar ou modificar políticas de exportação do ONTAP

Você precisa adicionar regras às políticas de exportação ONTAP existentes ou criar novas políticas de exportação que suportem criptografia Kerberos para o volume raiz da VM de armazenamento ONTAP , bem como para quaisquer volumes ONTAP compartilhados com o cluster Kubernetes upstream. As regras de política de exportação que você adicionar, ou as novas políticas de exportação que você criar, precisam ser compatíveis com os seguintes protocolos de acesso e permissões de acesso:

Protocolos de acesso

Configure a política de exportação com os protocolos de acesso NFS, NFSv3 e NFSv4.

Detalhes de acesso

Você pode configurar uma das três versões diferentes de criptografia Kerberos, dependendo das suas necessidades para o volume:

- **Kerberos 5** - (autenticação e criptografia)
- **Kerberos 5i** - (autenticação e criptografia com proteção de identidade)

- **Kerberos 5p** - (autenticação e criptografia com proteção de identidade e privacidade)

Configure a regra de política de exportação do ONTAP com as permissões de acesso apropriadas. Por exemplo, se os clusters forem montar os volumes NFS com uma combinação de criptografia Kerberos 5i e Kerberos 5p, use as seguintes configurações de acesso:

Tipo	Acesso somente leitura	Acesso de leitura/gravação	Acesso de superusuário
UNIX	Habilitado	Habilitado	Habilitado
Kerberos 5i	Habilitado	Habilitado	Habilitado
Kerberos 5p	Habilitado	Habilitado	Habilitado

Consulte a seguinte documentação para obter informações sobre como criar políticas de exportação e regras de política de exportação do ONTAP :

- "Crie uma política de exportação"
- "Adicionar uma regra a uma política de exportação"

Crie um backend de armazenamento

Você pode criar uma configuração de backend de armazenamento Trident que inclua a capacidade de criptografia Kerberos.

Sobre esta tarefa

Ao criar um arquivo de configuração de backend de armazenamento que configura a criptografia Kerberos, você pode especificar uma das três versões diferentes de criptografia Kerberos usando o `spec.nfsMountOptions` parâmetro:

- `spec.nfsMountOptions: sec=krb5`(autenticação e criptografia)
- `spec.nfsMountOptions: sec=krb5i`(autenticação e criptografia com proteção de identidade)
- `spec.nfsMountOptions: sec=krb5p`(autenticação e criptografia com proteção de identidade e privacidade)

Especifique apenas um nível Kerberos. Se você especificar mais de um nível de criptografia Kerberos na lista de parâmetros, somente a primeira opção será utilizada.

Passos

1. No cluster gerenciado, crie um arquivo de configuração de backend de armazenamento usando o seguinte exemplo. Substitua os valores entre colchetes <> por informações do seu ambiente:

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. Utilize o arquivo de configuração que você criou na etapa anterior para criar o backend:

```
tridentctl create backend -f <backend-configuration-file>
```

Se a criação do backend falhar, há algo errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs
```

Após identificar e corrigir o problema com o arquivo de configuração, você poderá executar o comando de criação novamente.

Criar uma classe de armazenamento

Você pode criar uma classe de armazenamento para provisionar volumes com criptografia Kerberos.

Sobre esta tarefa

Ao criar um objeto de classe de armazenamento, você pode especificar uma das três versões diferentes de criptografia Kerberos usando o `mountOptions` parâmetro:

- `mountOptions: sec=krb5`(autenticação e criptografia)
- `mountOptions: sec=krb5i`(autenticação e criptografia com proteção de identidade)
- `mountOptions: sec=krb5p`(autenticação e criptografia com proteção de identidade e privacidade)

Especifique apenas um nível Kerberos. Se você especificar mais de um nível de criptografia Kerberos na lista de parâmetros, somente a primeira opção será utilizada. Se o nível de criptografia especificado na configuração do backend de armazenamento for diferente do nível especificado no objeto da classe de armazenamento, o objeto da classe de armazenamento terá precedência.

Passos

1. Crie um objeto Kubernetes do tipo StorageClass, utilizando o seguinte exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions:
  - sec=krb5i #can be krb5, krb5i, or krb5p
parameters:
  backendType: ontap-nas
  storagePools: ontapnas_pool
  trident.netapp.io/nasType: nfs
allowVolumeExpansion: true
```

2. Crie a classe de armazenamento:

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. Certifique-se de que a classe de armazenamento foi criada:

```
kubectl get sc ontap-nas-sc
```

   Você deverá ver uma saída semelhante à seguinte:

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

volumes de provisão

Após criar um backend de armazenamento e uma classe de armazenamento, você poderá provisionar um volume. Para obter instruções, consulte "Forneça um volume" .

Configure a criptografia Kerberos em trânsito com volumes do Azure NetApp Files.

Você pode habilitar a criptografia Kerberos no tráfego de armazenamento entre seu cluster gerenciado e um único backend de armazenamento do Azure NetApp Files ou um pool virtual de backends de armazenamento do Azure NetApp Files .

Antes de começar

- Certifique-se de ter habilitado o Trident no cluster Red Hat OpenShift gerenciado.
- Certifique-se de ter acesso ao `tridentctl` utilidade.
- Certifique-se de ter preparado o backend de armazenamento do Azure NetApp Files para criptografia Kerberos, observando os requisitos e seguindo as instruções em "Documentação do Azure NetApp Files" .
- Certifique-se de que todos os volumes NFSv4 que você usa com criptografia Kerberos estejam configurados corretamente. Consulte a seção Configuração de Domínio NFSv4 da NetApp (página 13) do manual. "Guia de Aprimoramentos e Melhores Práticas do NetApp NFSv4" .

Crie um backend de armazenamento

Você pode criar uma configuração de back-end de armazenamento do Azure NetApp Files que inclua a capacidade de criptografia Kerberos.

Sobre esta tarefa

Ao criar um arquivo de configuração de backend de armazenamento que configura a criptografia Kerberos, você pode defini-lo para ser aplicado em um dos dois níveis possíveis:

- O **nível de backend de armazenamento** usando o `spec.kerberos` campo
- O **nível da piscina virtual** usando o `spec.storage.kerberos` campo

Ao definir a configuração no nível do pool virtual, o pool é selecionado usando o rótulo na classe de armazenamento.

Em qualquer um dos níveis, você pode especificar uma das três versões diferentes de criptografia Kerberos:

- `kerberos: sec=krb5`(autenticação e criptografia)
- `kerberos: sec=krb5i`(autenticação e criptografia com proteção de identidade)
- `kerberos: sec=krb5p`(autenticação e criptografia com proteção de identidade e privacidade)

Passos

1. No cluster gerenciado, crie um arquivo de configuração de backend de armazenamento usando um dos exemplos a seguir, dependendo de onde você precisa definir o backend de armazenamento (nível de backend de armazenamento ou nível de pool virtual). Substitua os valores entre colchetes <> por informações do seu ambiente:

Exemplo de nível de backend de armazenamento

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

Exemplo de nível de piscina virtual

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

2. Utilize o arquivo de configuração que você criou na etapa anterior para criar o backend:

```
tridentctl create backend -f <backend-configuration-file>
```

Se a criação do backend falhar, há algo errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs
```

Após identificar e corrigir o problema com o arquivo de configuração, você poderá executar o comando de criação novamente.

Criar uma classe de armazenamento

Você pode criar uma classe de armazenamento para provisionar volumes com criptografia Kerberos.

Passos

1. Crie um objeto Kubernetes do tipo StorageClass, utilizando o seguinte exemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: azure-netapp-files
  trident.netapp.io/nasType: nfs
  selector: type=encryption
```

2. Crie a classe de armazenamento:

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. Certifique-se de que a classe de armazenamento foi criada:

```
kubectl get sc -sc-nfs
```

Você deverá ver uma saída semelhante à seguinte:

```
NAME          PROVISIONER             AGE
sc-nfs        csi.trident.netapp.io   15h
```

volumes de provisão

Após criar um backend de armazenamento e uma classe de armazenamento, você poderá provisionar um volume. Para obter instruções, consulte "Forneça um volume" .

Proteja seus aplicativos com o Trident Protect.

Saiba mais sobre o Trident Protect.

O NetApp Trident Protect oferece recursos avançados de gerenciamento de dados de aplicativos que aprimoram a funcionalidade e a disponibilidade de aplicativos Kubernetes com estado, com suporte dos sistemas de armazenamento NetApp ONTAP e do provisionador de armazenamento NetApp Trident CSI. O Trident Protect simplifica o gerenciamento, a proteção e a movimentação de cargas de trabalho conteinerizadas em nuvens públicas e ambientes locais. Ele também oferece recursos de automação por meio de sua API e CLI.

Você pode proteger aplicativos com o Trident Protect criando recursos personalizados (CRs) ou usando a CLI do Trident Protect.

O que vem a seguir?

Você pode consultar os requisitos do Trident Protect antes de instalá-lo:

- "Requisitos do Trident Protect"

Instale o Trident Protect

Requisitos do Trident Protect

Comece verificando se seu ambiente operacional, clusters de aplicativos, aplicativos e licenças estão prontos. Certifique-se de que seu ambiente atenda a esses requisitos para implantar e operar o Trident Protect.

Compatibilidade do cluster Kubernetes com o Trident Protect

O Trident Protect é compatível com uma ampla gama de ofertas de Kubernetes totalmente gerenciadas e autogerenciadas, incluindo:

- Serviço Amazon Elastic Kubernetes (EKS)
- Google Kubernetes Engine (GKE)
- Serviço Kubernetes do Microsoft Azure (AKS)
- Red Hat OpenShift
- Fazendeiro SUSE
- Portfólio VMware Tanzu
- Kubernetes upstream

- Os backups do Trident Protect são suportados apenas em nós de computação Linux. Os nós de computação do Windows não são suportados para operações de backup.
- Certifique-se de que o cluster no qual você instala o Trident Protect esteja configurado com um controlador de snapshots em execução e os CRDs relacionados. Para instalar um controlador de snapshots, consulte "estas instruções" .

Compatibilidade com o backend de armazenamento Trident Protect

O Trident Protect é compatível com os seguintes sistemas de armazenamento:

- Amazon FSx for NetApp ONTAP
- Cloud Volumes ONTAP
- matrizes de armazenamento ONTAP
- Google Cloud NetApp Volumes
- Azure NetApp Files

Certifique-se de que seu sistema de armazenamento atenda aos seguintes requisitos:

- Certifique-se de que o armazenamento NetApp conectado ao cluster esteja usando o Trident 24.02 ou mais recente (o Trident 24.10 é recomendado).
- Certifique-se de ter um backend de armazenamento NetApp ONTAP .
- Certifique-se de ter configurado um bucket de armazenamento de objetos para armazenar backups.
- Crie todos os namespaces de aplicativos que você planeja usar para aplicativos ou operações de gerenciamento de dados de aplicativos. O Trident Protect não cria esses namespaces para você; se você especificar um namespace inexistente em um recurso personalizado, a operação falhará.

Requisitos para volumes de economia nas

O Trident Protect oferece suporte a operações de backup e restauração para volumes NAS Economy. Atualmente, não há suporte para snapshots, clones e replicação do SnapMirror para volumes nas-economy. Você precisa habilitar um diretório de snapshots para cada volume nas-economy que planeja usar com o Trident Protect.

Algumas aplicações não são compatíveis com volumes que utilizam um diretório de instantâneos. Para essas aplicações, você precisa ocultar o diretório de snapshots executando o seguinte comando no sistema de armazenamento ONTAP :

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

Você pode habilitar o diretório de snapshots executando o seguinte comando para cada volume nas-economy, substituindo `<volume-UUID>` com o UUID do volume que você deseja alterar:

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level
=true -n trident
```

Você pode habilitar diretórios de snapshots por padrão para novos volumes definindo a opção de configuração do backend Trident . `snapshotDir` para `true` . Os volumes existentes não serão afetados.

Protegendo dados com VMs KubeVirt

O Trident Protect 24.10 e versões posteriores, incluindo a 24.10.1, apresentam comportamentos diferentes ao proteger aplicativos executados em VMs do KubeVirt. Em ambas as versões, você pode ativar ou desativar o congelamento e descongelamento do sistema de arquivos durante as operações de proteção de dados.

Durante as operações de restauração, qualquer `VirtualMachineSnapshots` Os dados criados para uma máquina virtual (VM) não são restaurados.

Trident Protect 24.10

O Trident Protect 24.10 não garante automaticamente um estado consistente para os sistemas de arquivos das VMs do KubeVirt durante as operações de proteção de dados. Se você deseja proteger os dados da sua máquina virtual KubeVirt usando o Trident Protect 24.10, precisa habilitar manualmente a funcionalidade de congelamento/descongelamento dos sistemas de arquivos antes da operação de proteção de dados. Isso garante que os sistemas de arquivos estejam em um estado consistente.

Você pode configurar o Trident Protect 24.10 para gerenciar o congelamento e o descongelamento do sistema de arquivos da VM durante as operações de proteção de dados."configurando a virtualização" e então usando o seguinte comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=true -n trident-protect
```

Trident Protect 24.10.1 e versões mais recentes

A partir da versão 24.10.1 do Trident Protect, o sistema congela e descongela automaticamente os sistemas de arquivos do KubeVirt durante as operações de proteção de dados. Opcionalmente, você pode desativar esse comportamento automático usando o seguinte comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

Requisitos para replicação do SnapMirror

A replicação NetApp SnapMirror está disponível para uso com o Trident Protect para as seguintes soluções ONTAP :

- Clusters NetApp FAS, AFF e ASA locais
- NetApp ONTAP Select
- NetApp Cloud Volumes ONTAP
- Amazon FSx for NetApp ONTAP

Requisitos do cluster ONTAP para replicação do SnapMirror

Certifique-se de que seu cluster ONTAP atenda aos seguintes requisitos caso planeje usar a replicação SnapMirror :

- * NetApp Trident*: O NetApp Trident deve estar presente nos clusters Kubernetes de origem e destino que utilizam o ONTAP como backend. O Trident Protect oferece suporte à replicação com a tecnologia NetApp SnapMirror usando classes de armazenamento com suporte dos seguintes drivers:
  - `ontap-nas`: NFS
  - `ontap-san`: iSCSI
  - `ontap-san`: FC
  - `ontap-san`: NVMe/TCP (requer versão mínima do ONTAP 9.15.1)
- **Licenças**: As licenças assíncronas do ONTAP SnapMirror que utilizam o pacote Data Protection devem estar habilitadas nos clusters ONTAP de origem e destino. Consulte "Visão geral do licenciamento do SnapMirror no ONTAP" para mais informações.

  A partir do ONTAP 9.10.1, todas as licenças são fornecidas como um arquivo de licença NetApp (NLF), que é um único arquivo que habilita vários recursos. Consulte"Licenças incluídas no ONTAP One" para mais informações.

  > Somente a proteção assíncrona SnapMirror é suportada.

Considerações sobre peering para replicação do SnapMirror

Certifique-se de que seu ambiente atenda aos seguintes requisitos caso planeje usar o peering de backend de armazenamento:

- **Cluster e SVM**: Os backends de armazenamento ONTAP devem estar interligados. Consulte "Visão geral do peering de clusters e SVMs" para mais informações.

  > Certifique-se de que os nomes SVM usados na relação de replicação entre dois clusters ONTAP sejam únicos.

- * NetApp Trident e SVM*: Os SVMs remotos emparelhados devem estar disponíveis para o NetApp Trident no cluster de destino.
- **Backends gerenciados**: Você precisa adicionar e gerenciar backends de armazenamento ONTAP no Trident Protect para criar uma relação de replicação.

Configuração do Trident / ONTAP para replicação do SnapMirror

O Trident Protect exige que você configure pelo menos um backend de armazenamento que suporte replicação para os clusters de origem e destino. Se os clusters de origem e destino forem os mesmos, o aplicativo de destino deverá usar um backend de armazenamento diferente do aplicativo de origem para obter a melhor resiliência.

Requisitos do cluster Kubernetes para replicação do SnapMirror

Certifique-se de que seus clusters Kubernetes atendam aos seguintes requisitos:

- **Acessibilidade do AppVault**: Tanto o cluster de origem quanto o de destino devem ter acesso à rede

para ler e gravar no AppVault para replicação de objetos de aplicativos.

- **Conectividade de rede**: Configure regras de firewall, permissões de bucket e listas de permissão de IP para habilitar a comunicação entre os dois clusters e o AppVault através de WANs.

> Muitos ambientes empresariais implementam políticas de firewall rigorosas em conexões WAN. Verifique esses requisitos de rede com sua equipe de infraestrutura antes de configurar a replicação.

Instale e configure o Trident Protect.

Se o seu ambiente atender aos requisitos do Trident Protect, você pode seguir estas etapas para instalar o Trident Protect em seu cluster. Você pode obter o Trident Protect da NetApp ou instalá-lo a partir do seu próprio registro privado. A instalação a partir de um registro privado é útil caso seu cluster não tenha acesso à Internet.

Instale o Trident Protect

Instale o Trident Protect da NetApp.

Passos

1. Adicione o repositório Trident Helm:

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

2. Use o Helm para instalar o Trident Protect. Substituir `<name-of-cluster>` com um nome de cluster, que será atribuído ao cluster e usado para identificar os backups e snapshots do cluster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect
```

Instale o Trident Protect a partir de um registro privado.

Você pode instalar o Trident Protect a partir de um registro de imagens privado caso seu cluster Kubernetes não tenha acesso à Internet. Nestes exemplos, substitua os valores entre colchetes por informações do seu ambiente:

Passos

1. Baixe as seguintes imagens para sua máquina local, atualize as tags e, em seguida, envie-as para seu registro privado:

```
netapp/controller:25.06.0
netapp/restic:25.06.0
netapp/kopia:25.06.0
netapp/trident-autosupport:25.06.0
netapp/exechook:25.06.0
netapp/resourcebackup:25.06.0
netapp/resourcerestore:25.06.0
netapp/resourcedelete:25.06.0
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

   Por exemplo:

```
docker pull netapp/controller:25.06.0
```

```
docker tag netapp/controller:25.06.0 <private-registry-
url>/controller:25.06.0
```

```
docker push <private-registry-url>/controller:25.06.0
```

2. Crie o namespace do sistema Trident Protect:

```
kubectl create ns trident-protect
```

3. Faça login no registro:

```
helm registry login <private-registry-url> -u <account-id> -p <api-token>
```

4. Crie um segredo de pull para usar na autenticação de registro privado:

```
kubectl create secret docker-registry regcred --docker-username=<registry-username> --docker-password=<api-token> -n trident-protect --docker-server=<private-registry-url>
```

5. Adicione o repositório Trident Helm:

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

6. Crie um arquivo chamado `protectValues.yaml` . Certifique-se de que contenha as seguintes configurações do Trident Protect:

```
---
image:
  registry: <private-registry-url>
imagePullSecrets:
  - name: regcred
controller:
  image:
    registry: <private-registry-url>
rbacProxy:
  image:
    registry: <private-registry-url>
crCleanup:
  imagePullSecrets:
    - name: regcred
webhooksCleanup:
  imagePullSecrets:
    - name: regcred
```

7. Use o Helm para instalar o Trident Protect. Substituir `<name_of_cluster>` com um nome de cluster, que será atribuído ao cluster e usado para identificar os backups e snapshots do cluster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect -f protectValues.yaml
```

Instale o plugin Trident Protect CLI

Você pode usar o plugin de linha de comando Trident Protect, que é uma extensão do Trident. `tridentctl` Utilitário para criar e interagir com recursos personalizados (CRs) do Trident Protect.

Instale o plugin Trident Protect CLI

Antes de usar o utilitário de linha de comando, você precisa instalá-lo na máquina que utiliza para acessar o cluster. Siga estes passos, dependendo se a sua máquina utiliza uma CPU x64 ou ARM .

Baixe o plugin para CPUs Linux AMD64

Passos

1. Baixe o plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-amd64
```

Baixe o plugin para CPUs Linux ARM64

Passos

1. Baixe o plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-arm64
```

Baixe o plugin para CPUs Mac AMD64

Passos

1. Baixe o plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-amd64
```

Baixe o plugin para CPUs Mac ARM64

Passos

1. Baixe o plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-arm64
```

1. Habilite as permissões de execução para o binário do plugin:

```
chmod +x tridentctl-protect
```

2. Copie o arquivo binário do plugin para um local definido na sua variável PATH. Por exemplo, `/usr/bin` ou `/usr/local/bin` (Você pode precisar de privilégios elevados):

```
cp ./tridentctl-protect /usr/local/bin/
```

3. Opcionalmente, você pode copiar o arquivo binário do plugin para um local em seu diretório pessoal. Nesse caso, recomenda-se garantir que o local faça parte da sua variável PATH:

```
cp ./tridentctl-protect ~/bin/
```

Copiar o plugin para um local na sua variável PATH permite que você o utilize digitando o comando. `tridentctl-protect` ou `tridentctl protect` de qualquer lugar.

Veja a ajuda do plugin Trident CLI

Você pode usar os recursos de ajuda integrados do plugin para obter ajuda detalhada sobre as funcionalidades do plugin:

Passos

1. Utilize a função de ajuda para visualizar as instruções de utilização:

```
tridentctl-protect help
```

Ativar o preenchimento automático de comandos

Após instalar o plugin Trident Protect CLI, você pode ativar o recurso de autocompletar para determinados comandos.

Ative o recurso de autocompletar para o shell Bash.

Passos

1. Baixe o script de conclusão:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.bash
```

2. Crie um novo diretório em seu diretório pessoal para armazenar o script:

```
mkdir -p ~/.bash/completions
```

3. Mova o script baixado para o `~/.bash/completions` diretório:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Adicione a seguinte linha ao `~/.bashrc` Arquivo no seu diretório pessoal:

```
source ~/.bash/completions/tridentctl-completion.bash
```

Ative o recurso de autocompletar para o shell Z.

Passos

1. Baixe o script de conclusão:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.zsh
```

2. Crie um novo diretório em seu diretório pessoal para armazenar o script:

```
mkdir -p ~/.zsh/completions
```

3. Mova o script baixado para o `~/.zsh/completions` diretório:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Adicione a seguinte linha ao `~/.zprofile` Arquivo no seu diretório pessoal:

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

Resultado

Na sua próxima sessão de login no shell, você poderá usar o recurso de autocompletar comandos com o plugin tridentctl-protect.

Personalize a instalação do Trident Protect

Você pode personalizar a configuração padrão do Trident Protect para atender aos requisitos específicos do seu ambiente.

Especifique os limites de recursos do contêiner Trident Protect.

Você pode usar um arquivo de configuração para especificar limites de recursos para os contêineres do Trident Protect após a instalação do Trident Protect. A definição de limites de recursos permite controlar a quantidade de recursos do cluster que são consumidos pelas operações do Trident Protect.

Passos

1. Crie um arquivo chamado `resourceLimits.yaml` .
2. Preencha o arquivo com as opções de limite de recursos para os contêineres do Trident Protect de acordo com as necessidades do seu ambiente.

   O seguinte arquivo de configuração de exemplo mostra as configurações disponíveis e contém os valores padrão para cada limite de recurso:

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
      memory: ""
```

```
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

3. Aplique os valores de `resourceLimits.yaml` arquivo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f resourceLimits.yaml --reuse-values
```

Personalizar restrições de contexto de segurança

Você pode usar um arquivo de configuração para modificar as restrições de contexto de segurança (SCCs) do OpenShift para contêineres Trident Protect após instalar o Trident Protect. Essas restrições definem as limitações de segurança para os pods em um cluster Red Hat OpenShift.

Passos

1. Crie um arquivo chamado `sccconfig.yaml` .
2. Adicione a opção SCC ao arquivo e modifique os parâmetros de acordo com as necessidades do seu ambiente.

   O exemplo a seguir mostra os valores padrão dos parâmetros para a opção SCC:

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

Esta tabela descreve os parâmetros para a opção SCC:

Parâmetro	Descrição	Padrão
criar	Determina se um recurso SCC pode ser criado. Um recurso SCC será criado somente se `scc.create` está definido para `true` e o processo de instalação do Helm identifica um ambiente OpenShift. Se não estiver operando no OpenShift, ou se `scc.create` está definido para `false` Nenhum recurso SCC será criado.	verdadeiro
nome	Especifica o nome do SCC.	tridente-proteger-trabalho
prioridade	Define a prioridade do SCC. Os carcinomas de células escamosas (CCEs) com valores de prioridade mais altos são avaliados antes daqueles com valores mais baixos.	1

3. Aplique os valores de `sccconfig.yaml` arquivo:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect -f
sccconfig.yaml --reuse-values
```

Isso substituirá os valores padrão pelos valores especificados em `sccconfig.yaml` arquivo.

Configure as definições adicionais do gráfico de navegação do Trident Protect.

Você pode personalizar as configurações do AutoSupport e a filtragem de namespace para atender aos seus requisitos específicos. A tabela a seguir descreve os parâmetros de configuração disponíveis:

Parâmetro	Tipo	Descrição
autoSupport.proxy	corda	Configura um URL de proxy para conexões do NetApp AutoSupport . Use isso para rotear uploads de pacotes de suporte por meio de um servidor proxy. Exemplo: `http://my.proxy.url` .

Parâmetro	Tipo	Descrição
autoSupport.inseguro	booleano	Ignora a verificação TLS para conexões de proxy do AutoSupport quando configurado para `true` . Use somente para conexões proxy inseguras. (padrão: `false`)
autoSupport.habilitado	booleano	Ativa ou desativa os uploads diários do pacote Trident Protect AutoSupport . Quando definido para `false` Os uploads diários agendados estão desativados, mas você ainda pode gerar pacotes de suporte manualmente. (padrão: `true`)
restaurarSkipNamespaceAnnotations	corda	Lista separada por vírgulas de anotações de namespace a serem excluídas das operações de backup e restauração. Permite filtrar namespaces com base em anotações.
restaurarIgnorarEtiquetasDeEspaçoDeNomes	corda	Lista separada por vírgulas de rótulos de namespace a serem excluídos das operações de backup e restauração. Permite filtrar namespaces com base em rótulos.

Você pode configurar essas opções usando um arquivo de configuração YAML ou sinalizadores de linha de comando:

Usar arquivo YAML

Passos

1. Crie um arquivo de configuração e dê um nome a ele. `values.yaml` .
2. No arquivo que você criou, adicione as opções de configuração que deseja personalizar.

```
autoSupport:
  enabled: false
  proxy: http://my.proxy.url
  insecure: true
restoreSkipNamespaceAnnotations: "annotation1,annotation2"
restoreSkipNamespaceLabels: "label1,label2"
```

3. Depois de preencher o `values.yaml` Arquivo com os valores corretos, aplique o arquivo de configuração:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f values.yaml --reuse-values
```

Usar sinalizador CLI

Passos

1. Use o seguinte comando com o `--set` Sinalizador para especificar parâmetros individuais:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set autoSupport.enabled=false \
  --set autoSupport.proxy=http://my.proxy.url \
  --set restoreSkipNamespaceAnnotations="annotation1,annotation2" \
  --set restoreSkipNamespaceLabels="label1,label2" \
  --reuse-values
```

Restrinja os pods do Trident Protect a nós específicos.

Você pode usar a restrição de seleção de nós `nodeSelector` do Kubernetes para controlar quais de seus nós são elegíveis para executar pods do Trident Protect, com base nos rótulos dos nós. Por padrão, o Trident Protect é restrito a nós que executam Linux. Você pode personalizar ainda mais essas restrições de acordo com suas necessidades.

Passos

1. Crie um arquivo chamado `nodeSelectorConfig.yaml` .
2. Adicione a opção nodeSelector ao arquivo e modifique-o para adicionar ou alterar rótulos de nós, restringindo a seleção de acordo com as necessidades do seu ambiente. Por exemplo, o arquivo a seguir

contém a restrição padrão do sistema operacional, mas também tem como alvo uma região e um nome de aplicativo específicos:

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. Aplique os valores de `nodeSelectorConfig.yaml` arquivo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

Isso substitui as restrições padrão pelas que você especificou em `nodeSelectorConfig.yaml` arquivo.

Gerenciar Trident Protect

Gerencie a autorização e o controle de acesso do Trident Protect.

O Trident Protect utiliza o modelo Kubernetes de controle de acesso baseado em funções (RBAC). Por padrão, o Trident Protect fornece um único namespace de sistema e sua respectiva conta de serviço padrão. Se a sua organização possui muitos usuários ou necessidades específicas de segurança, você pode usar os recursos RBAC do Trident Protect para obter um controle mais granular sobre o acesso a recursos e namespaces.

O administrador do cluster sempre tem acesso aos recursos no ambiente padrão. `trident-protect` namespace, e também pode acessar recursos em todos os outros namespaces. Para controlar o acesso a recursos e aplicativos, você precisa criar namespaces adicionais e adicionar recursos e aplicativos a esses namespaces.

Observe que nenhum usuário pode criar solicitações de configuração (CRs) de gerenciamento de dados de aplicativos no ambiente padrão. `trident-protect` espaço de nomes. Você precisa criar solicitações de configuração (CRs) de gerenciamento de dados de aplicativos em um namespace de aplicativo (como prática recomendada, crie as CRs de gerenciamento de dados de aplicativos no mesmo namespace do aplicativo associado).

Somente os administradores devem ter acesso aos objetos de recursos personalizados privilegiados do Trident Protect, que incluem:

- **AppVault**: Requer dados de credenciais do bucket
- **Pacote de Suporte Automático**: Coleta métricas, registros e outros dados confidenciais do Trident Protect.
- **AutoSupportBundleSchedule**: Gerencia os agendamentos de coleta de logs

Como prática recomendada, utilize o RBAC para restringir o acesso a objetos privilegiados apenas a administradores.

Para obter mais informações sobre como o RBAC regula o acesso a recursos e namespaces, consulte o "Documentação do RBAC do Kubernetes" .

Para obter informações sobre contas de serviço, consulte o "Documentação da conta de serviço do Kubernetes" .

Exemplo: Gerenciar o acesso para dois grupos de usuários

Por exemplo, uma organização possui um administrador de cluster, um grupo de usuários de engenharia e um grupo de usuários de marketing. O administrador do cluster deverá executar as seguintes tarefas para criar um ambiente onde o grupo de engenharia e o grupo de marketing tenham acesso apenas aos recursos atribuídos aos seus respectivos namespaces.

Etapa 1: Crie um espaço de nomes para conter os recursos de cada grupo.

Criar um espaço de nomes permite separar recursos logicamente e controlar melhor quem tem acesso a esses recursos.

Passos

1. Crie um espaço de nomes para o grupo de engenharia:

```
kubectl create ns engineering-ns
```

2. Crie um espaço de nomes para o grupo de marketing:

```
kubectl create ns marketing-ns
```

Etapa 2: Crie novas contas de serviço para interagir com os recursos em cada namespace.

Cada novo namespace que você cria vem com uma conta de serviço padrão, mas você deve criar uma conta de serviço para cada grupo de usuários para que possa dividir ainda mais os privilégios entre os grupos no futuro, se necessário.

Passos

1. Crie uma conta de serviço para o grupo de engenharia:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Criar uma conta de serviço para o grupo de marketing:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Etapa 3: Crie um segredo para cada nova conta de serviço.

Uma chave secreta da conta de serviço é usada para autenticar com a conta de serviço e pode ser facilmente excluída e recriada caso seja comprometida.

Passos

1. Crie um segredo para a conta do serviço de engenharia:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
type: kubernetes.io/service-account-token
```

2. Crie um segredo para a conta do serviço de marketing:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
type: kubernetes.io/service-account-token
```

Etapa 4: Crie um objeto RoleBinding para vincular o objeto ClusterRole a cada nova conta de serviço.

Um objeto ClusterRole padrão é criado quando você instala o Trident Protect. Você pode vincular essa ClusterRole à conta de serviço criando e aplicando um objeto RoleBinding.

Passos

1. Vincule a função ClusterRole à conta de serviço de engenharia:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

2. Vincule a função ClusterRole à conta do serviço de marketing:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

Etapa 5: Testar permissões

Verifique se as permissões estão corretas.

Passos

1. Confirme se os usuários de engenharia podem acessar os recursos de engenharia:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Confirme que os usuários de engenharia não têm acesso aos recursos de marketing:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n marketing-ns
```

Etapa 6: Conceda acesso aos objetos do AppVault

Para executar tarefas de gerenciamento de dados, como backups e snapshots, o administrador do cluster precisa conceder acesso aos objetos do AppVault a usuários individuais.

Passos

1. Crie e aplique um arquivo YAML de combinação de AppVault e segredo que conceda ao usuário acesso a um AppVault. Por exemplo, a seguinte solicitação de configuração (CR) concede acesso a um AppVault ao usuário. `eng-user` :

```
apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident Protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3
```

2. Crie e aplique uma solicitação de configuração (CR) de função para permitir que os administradores do cluster concedam acesso a recursos específicos em um namespace. Por exemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Crie e aplique uma solicitação de configuração (CR) de vinculação de função (RoleBinding) para associar as permissões ao usuário eng-user. Por exemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Verifique se as permissões estão corretas.

   a. Tentativa de recuperar informações de objetos do AppVault para todos os namespaces:

   ```
   kubectl get appvaults -n trident-protect
   --as=system:serviceaccount:engineering-ns:eng-user
   ```

   Você deverá ver uma saída semelhante à seguinte:

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is
forbidden: User "system:serviceaccount:engineering-ns:eng-user"
cannot list resource "appvaults" in API group
"protect.trident.netapp.io" in the namespace "trident-protect"
```

b. Teste para verificar se o usuário consegue obter as informações do AppVault às quais agora possui permissão de acesso:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n
trident-protect
```

Você deverá ver uma saída semelhante à seguinte:

```
yes
```

Resultado

Os usuários aos quais você concedeu permissões do AppVault devem poder usar objetos autorizados do AppVault para operações de gerenciamento de dados do aplicativo e não devem poder acessar recursos fora dos namespaces atribuídos nem criar novos recursos aos quais não tenham acesso.

Monitorar recursos do Trident Protect

Você pode usar as ferramentas de código aberto kube-state-metrics, Prometheus e Alertmanager para monitorar a integridade dos recursos protegidos pelo Trident Protect.

O serviço kube-state-metrics gera métricas a partir da comunicação da API do Kubernetes. Ao utilizá-lo com o Trident Protect, você obtém informações úteis sobre o estado dos recursos em seu ambiente.

O Prometheus é um conjunto de ferramentas que pode ingerir os dados gerados pelo kube-state-metrics e apresentá-los como informações facilmente legíveis sobre esses objetos. Em conjunto, o kube-state-metrics e o Prometheus oferecem uma maneira de monitorar a integridade e o status dos recursos que você está gerenciando com o Trident Protect.

O Alertmanager é um serviço que recebe os alertas enviados por ferramentas como o Prometheus e os encaminha para destinos que você configura.

> As configurações e orientações incluídas nestas etapas são apenas exemplos; você precisa personalizá-las para que correspondam ao seu ambiente. Consulte a seguinte documentação oficial para obter instruções específicas e suporte:
>
> - "Documentação do kube-state-metrics"
> - "Documentação do Prometheus"
> - "Documentação do Alertmanager"

Passo 1: Instale as ferramentas de monitoramento

Para habilitar o monitoramento de recursos no Trident Protect, você precisa instalar e configurar o kube-state-metrics, o Prometheus e o Alertmanager.

Instale o kube-state-metrics

Você pode instalar o kube-state-metrics usando o Helm.

Passos

1. Adicione o gráfico Helm kube-state-metrics. Por exemplo:

```
helm repo add prometheus-community https://prometheus-
community.github.io/helm-charts
helm repo update
```

2. Aplique o Prometheus ServiceMonitor CRD ao cluster:

```
kubectl apply -f https://raw.githubusercontent.com/prometheus-
operator/prometheus-operator/main/example/prometheus-operator-
crd/monitoring.coreos.com_servicemonitors.yaml
```

3. Crie um arquivo de configuração para o gráfico Helm (por exemplo, `metrics-config.yaml`). Você pode personalizar a seguinte configuração de exemplo para corresponder ao seu ambiente:

metrics-config.yaml: configuração do gráfico Helm kube-state-metrics

```
---
extraArgs:
  # Collect only custom metrics
  - --custom-resource-state-only=true

customResourceState:
  enabled: true
  config:
    kind: CustomResourceStateMetrics
    spec:
      resources:
      - groupVersionKind:
          group: protect.trident.netapp.io
          kind: "Backup"
          version: "v1"
        labelsFromPath:
          backup_uid: [metadata, uid]
          backup_name: [metadata, name]
          creation_time: [metadata, creationTimestamp]
        metrics:
        - name: backup_info
          help: "Exposes details about the Backup state"
          each:
            type: Info
            info:
              labelsFromPath:
                appVaultReference: ["spec", "appVaultRef"]
                appReference: ["spec", "applicationRef"]
rbac:
  extraRules:
  - apiGroups: ["protect.trident.netapp.io"]
    resources: ["backups"]
    verbs: ["list", "watch"]

# Collect metrics from all namespaces
namespaces: ""

# Ensure that the metrics are collected by Prometheus
prometheus:
  monitor:
    enabled: true
```

4. Instale o kube-state-metrics implantando o gráfico Helm. Por exemplo:

```
helm install custom-resource -f metrics-config.yaml prometheus-
community/kube-state-metrics --version 5.21.0
```

5. Configure o kube-state-metrics para gerar métricas para os recursos personalizados usados pelo Trident Protect, seguindo as instruções em "Documentação do recurso personalizado kube-state-metrics" .

Instalar o Prometheus

Você pode instalar o Prometheus seguindo as instruções em "Documentação do Prometheus" .

Instalar o Alertmanager

Você pode instalar o Alertmanager seguindo as instruções em "Documentação do Alertmanager" .

Etapa 2: Configure as ferramentas de monitoramento para funcionarem em conjunto.

Após instalar as ferramentas de monitoramento, você precisa configurá-las para que funcionem em conjunto.

Passos

1. Integre o kube-state-metrics com o Prometheus. Edite o arquivo de configuração do Prometheus(`prometheus.yaml`) e adicione as informações do serviço kube-state-metrics. Por exemplo:

   prometheus.yaml: integração do serviço kube-state-metrics com o Prometheus

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
  namespace: trident-protect
data:
  prometheus.yaml: |
    global:
      scrape_interval: 15s
    scrape_configs:
      - job_name: 'kube-state-metrics'
        static_configs:
          - targets: ['kube-state-metrics.trident-protect.svc:8080']
```

2. Configure o Prometheus para encaminhar alertas para o Alertmanager. Edite o arquivo de configuração do Prometheus(`prometheus.yaml`) e adicione a seguinte seção:

prometheus.yaml: Enviar alertas para o Alertmanager

```
alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - alertmanager.trident-protect.svc:9093
```

Resultado

O Prometheus agora pode coletar métricas do kube-state-metrics e enviar alertas para o Alertmanager. Agora você está pronto para configurar quais condições acionam um alerta e para onde os alertas devem ser enviados.

Etapa 3: Configurar alertas e destinos de alerta

Depois de configurar as ferramentas para funcionarem em conjunto, você precisa configurar que tipo de informação aciona os alertas e para onde os alertas devem ser enviados.

Exemplo de alerta: falha no backup

O exemplo a seguir define um alerta crítico que é acionado quando o status do recurso personalizado de backup é definido como `Error` por 5 segundos ou mais. Você pode personalizar este exemplo para corresponder ao seu ambiente e incluir este trecho de YAML no seu arquivo de configuração. `prometheus.yaml` arquivo de configuração:

rules.yaml: Defina um alerta do Prometheus para backups com falha.

```
rules.yaml: |
  groups:
    - name: fail-backup
        rules:
          - alert: BackupFailed
            expr: kube_customresource_backup_info{status="Error"}
            for: 5s
            labels:
              severity: critical
            annotations:
              summary: "Backup failed"
              description: "A backup has failed."
```

Configure o Alertmanager para enviar alertas para outros canais.

Você pode configurar o Alertmanager para enviar notificações para outros canais, como e-mail, PagerDuty, Microsoft Teams ou outros serviços de notificação, especificando a respectiva configuração no arquivo de configuração. `alertmanager.yaml` arquivo.

O exemplo a seguir configura o Alertmanager para enviar notificações para um canal do Slack. Para adaptar este exemplo ao seu ambiente, substitua o valor de `api_url` Chave com a URL do webhook do Slack usada em seu ambiente:

alertmanager.yaml: Enviar alertas para um canal do Slack

```
data:
  alertmanager.yaml: |
    global:
      resolve_timeout: 5m
    route:
      receiver: 'slack-notifications'
    receivers:
      - name: 'slack-notifications'
        slack_configs:
          - api_url: '<your-slack-webhook-url>'
            channel: '#failed-backups-channel'
            send_resolved: false
```

Gere um pacote de suporte Trident Protect

O Trident Protect permite que os administradores gerem pacotes que incluem informações úteis para o Suporte da NetApp , como logs, métricas e informações de topologia sobre os clusters e aplicativos gerenciados. Se você estiver conectado à Internet, poderá carregar pacotes de suporte no NetApp Support Site (NSS) usando um arquivo de recurso personalizado (CR).

Crie um pacote de suporte usando uma solicitação de crédito (CR).

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-support-bundle.yaml`).
2. Configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.triggerType**: (*Obrigatório*) Determina se o pacote de suporte é gerado imediatamente ou agendado. A geração programada de pacotes ocorre às 00h UTC. Valores possíveis:
     - Agendado
     - Manual
   - **spec.uploadEnabled**: (Opcional) Controla se o pacote de suporte deve ser carregado no site de suporte da NetApp após ser gerado. Se não for especificado, o valor padrão é `false` . Valores possíveis:
     - verdadeiro
     - falso (padrão)
   - **spec.dataWindowStart**: (Opcional) Uma string de data no formato RFC 3339 que especifica a data e a hora em que a janela de dados incluída no pacote de suporte deve começar. Caso não seja especificado, o padrão é de 24 horas atrás. A data mais antiga que você pode especificar é de 7 dias atrás.

     Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Depois de preencher o `trident-protect-support-bundle.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-support-bundle.yaml -n trident-protect
```

Crie um pacote de suporte usando a CLI

Passos

1. Crie o pacote de suporte, substituindo os valores entre colchetes pelas informações do seu ambiente.

O `trigger-type` Determina se o pacote é criado imediatamente ou se o tempo de criação é ditado pelo agendamento, e pode ser `Manual` ou `Scheduled` . A configuração padrão é `Manual` .

Por exemplo:

```
tridentctl-protect create autosupportbundle <my-bundle-name>
--trigger-type <trigger-type> -n trident-protect
```

Monitore e recupere o pacote de suporte

Depois de criar um pacote de suporte usando qualquer um dos métodos, você pode monitorar seu progresso de geração e recuperá-lo para seu sistema local.

Passos

1. Espere pelo `status.generationState` para alcançar `Completed` estado. Você pode monitorar o progresso da geração com o seguinte comando:

   ```
   kubectl get autosupportbundle trident-protect-support-bundle -n trident-protect
   ```

2. Recupere o pacote de suporte para seu sistema local. Obtenha o comando de cópia do pacote AutoSupport concluído:

   ```
   kubectl describe autosupportbundle trident-protect-support-bundle -n trident-protect
   ```

   Encontre o `kubectl cp` Extraia o comando da saída e execute-o, substituindo o argumento de destino pelo diretório local de sua preferência.

Aprimore o Trident Protect

Você pode atualizar o Trident Protect para a versão mais recente para aproveitar os novos recursos ou correções de bugs.

> Ao atualizar da versão 24.10, os snapshots executados durante a atualização podem falhar. Essa falha não impede a criação de snapshots futuros, sejam eles manuais ou agendados. Se um snapshot falhar durante a atualização, você pode criar manualmente um novo snapshot para garantir a proteção do seu aplicativo.
>
> Para evitar possíveis falhas, você pode desativar todos os agendamentos de snapshots antes da atualização e reativá-los posteriormente. No entanto, isso resulta na perda de quaisquer snapshots agendados durante o período de atualização.

Para atualizar o Trident Protect, siga os passos abaixo.

Passos

1. Atualize o repositório Trident Helm:

```
helm repo update
```

2. Atualize os CRDs do Trident Protect:

> Esta etapa é necessária se você estiver atualizando de uma versão anterior à 25.06, pois os CRDs agora estão incluídos no gráfico Helm do Trident Protect.

a. Execute este comando para transferir o gerenciamento de CRDs de `trident-protect-crds` para `trident-protect` :

```
kubectl get crd | grep protect.trident.netapp.io | awk '{print $1}' |
xargs -I {} kubectl patch crd {} --type merge -p '{"metadata":
{"annotations":{"meta.helm.sh/release-name": "trident-protect"}}}'
```

b. Execute este comando para excluir o segredo do Helm para o `trident-protect-crds` gráfico:

> Não desinstale o `trident-protect-crds` O uso do Helm para gerar o gráfico pode remover seus CRDs e quaisquer dados relacionados.

```
kubectl delete secret -n trident-protect -l name=trident-protect-
crds,owner=helm
```

3. Aprimore o Trident Protect:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect
--version 100.2506.0 --namespace trident-protect
```

Gerenciar e proteger aplicativos

Use objetos do Trident Protect AppVault para gerenciar buckets.

O recurso personalizado (CR) do bucket para o Trident Protect é conhecido como AppVault. Os objetos do AppVault são a representação declarativa do fluxo de trabalho do Kubernetes de um bucket de armazenamento. Um AppVault CR contém as configurações necessárias para que um bucket seja usado em operações de proteção, como backups, snapshots, operações de restauração e replicação do SnapMirror . Somente administradores podem criar AppVaults.

Você precisa criar uma CR do AppVault manualmente ou pela linha de comando ao executar operações de

proteção de dados em um aplicativo. A CR do AppVault é específica para o seu ambiente, e você pode usar os exemplos nesta página como guia para criar CRs do AppVault.

> Certifique-se de que o AppVault CR esteja no cluster onde o Trident Protect está instalado. Se o AppVault CR não existir ou você não conseguir acessá-lo, a linha de comando exibirá um erro.

Configure a autenticação e as senhas do AppVault.

Antes de criar um AppVault CR, certifique-se de que o AppVault e o movimentador de dados escolhido possam ser autenticados com o provedor e quaisquer recursos relacionados.

Senhas do repositório do Data Mover

Ao criar objetos AppVault usando CRs ou o plugin Trident Protect CLI, você pode especificar um segredo do Kubernetes com senhas personalizadas para criptografia Restic e Kopia. Se você não especificar uma senha secreta, o Trident Protect usará uma senha padrão.

- Ao criar manualmente solicitações de configuração (CRs) do AppVault, use o campo **spec.dataMoverPasswordSecretRef** para especificar o segredo.
- Ao criar objetos do AppVault usando a CLI do Trident Protect, utilize o `--data-mover-password-secret-ref` argumento para especificar o segredo.

Crie uma senha secreta para o repositório do Data Mover.

Utilize os exemplos a seguir para criar a senha secreta. Ao criar objetos do AppVault, você pode instruir o Trident Protect a usar esse segredo para autenticar com o repositório de movimentação de dados.

> - Dependendo do programa de transferência de dados que você estiver usando, basta incluir a senha correspondente a esse programa. Por exemplo, se você estiver usando o Restic e não planeja usar o Kopia no futuro, poderá incluir apenas a senha do Restic ao criar o segredo.
> - Guarde a senha em um local seguro. Você precisará dela para restaurar dados no mesmo cluster ou em um diferente. Se o cluster ou o `trident-protect` O namespace foi excluído e você não poderá restaurar seus backups ou snapshots sem a senha.

Use um CR

```
---
apiVersion: v1
data:
  KOPIA_PASSWORD: <base64-encoded-password>
  RESTIC_PASSWORD: <base64-encoded-password>
kind: Secret
metadata:
  name: my-optional-data-mover-secret
  namespace: trident-protect
type: Opaque
```

Use a CLI

```
kubectl create secret generic my-optional-data-mover-secret \
--from-literal=KOPIA_PASSWORD=<plain-text-password> \
--from-literal=RESTIC_PASSWORD=<plain-text-password> \
-n trident-protect
```

Permissões IAM de armazenamento compatíveis com S3

Ao acessar armazenamento compatível com S3, como Amazon S3, S3 genérico, "StorageGrid S3" , ou "ONTAP S3" Ao usar o Trident Protect, você precisa garantir que as credenciais de usuário fornecidas tenham as permissões necessárias para acessar o bucket. Segue abaixo um exemplo de uma política que concede as permissões mínimas necessárias para acesso ao Trident Protect. Você pode aplicar essa política ao usuário que gerencia as políticas de bucket compatíveis com o S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

Para obter mais informações sobre as políticas do Amazon S3, consulte os exemplos em "Documentação do Amazon S3" .

Identidade do Pod EKS para autenticação do Amazon S3 (AWS)

O Trident Protect oferece suporte à identidade do EKS Pod para operações de movimentação de dados do Kopia. Este recurso permite acesso seguro aos buckets do S3 sem armazenar credenciais da AWS em segredos do Kubernetes.

Requisitos para a identidade do EKS Pod com o Trident Protect

Antes de usar o EKS Pod Identity com o Trident Protect, certifique-se do seguinte:

- Seu cluster EKS tem a Identidade do Pod habilitada.
- Você criou uma função do IAM com as permissões de bucket do S3 necessárias. Para saber mais, consulte "Permissões IAM de armazenamento compatíveis com S3".
- A função IAM está associada às seguintes contas de serviço do Trident Protect:
  - `<trident-protect>-controller-manager`
  - `<trident-protect>-resource-backup`
  - `<trident-protect>-resource-restore`
  - `<trident-protect>-resource-delete`

Para obter instruções detalhadas sobre como habilitar a identidade do Pod e associar funções do IAM a contas de serviço, consulte o "Documentação de identidade do pod AWS EKS" .

Configuração do AppVault Ao usar a identidade do Pod do EKS, configure seu CR do AppVault com o `useIAM: true` sinalizador em vez de credenciais explícitas:

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: eks-protect-vault
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-aws
      endpoint: s3.example.com
      useIAM: true
```

Exemplos de geração de chaves AppVault para provedores de nuvem

Ao definir um AppVault CR, você precisa incluir credenciais para acessar os recursos hospedados pelo provedor, a menos que esteja usando a autenticação do IAM. A maneira como você gera as chaves para as credenciais varia de acordo com o provedor. A seguir estão exemplos de geração de chaves de linha de comando para vários provedores. Você pode usar os exemplos a seguir para criar chaves para as credenciais de cada provedor de nuvem.

Google Cloud

```
kubectl create secret generic <secret-name> \
--from-file=credentials=<mycreds-file.json> \
-n trident-protect
```

Amazon S3 (AWS)

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<amazon-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

Microsoft Azure

```
kubectl create secret generic <secret-name> \
--from-literal=accountKey=<secret-name> \
-n trident-protect
```

S3 genérico

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<generic-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

ONTAP S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<ontap-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

StorageGrid S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<storagegrid-s3-trident-protect-src
-bucket-secret> \
-n trident-protect
```

Exemplos de criação do AppVault

A seguir, apresentamos exemplos de definições do AppVault para cada provedor.

Exemplos de AppVault CR

Você pode usar os seguintes exemplos de CR para criar objetos do AppVault para cada provedor de nuvem.

- Opcionalmente, você pode especificar um segredo do Kubernetes que contenha senhas personalizadas para a criptografia dos repositórios Restic e Kopia. ConsulteSenhas do repositório do Data Mover para mais informações.
- Para objetos do Amazon S3 (AWS) AppVault, você pode opcionalmente especificar um sessionToken, o que é útil se você estiver usando o logon único (SSO) para autenticação. Este token é criado quando você gera chaves para o provedor emExemplos de geração de chaves AppVault para provedores de nuvem .
- Para objetos do AppVault no S3, você pode opcionalmente especificar um URL de proxy de saída para o tráfego de saída do S3 usando o `spec.providerConfig.S3.proxyURL` chave.

Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GCP
  providerConfig:
    gcp:
      bucketName: trident-protect-src-bucket
      projectID: project-id
  providerCredentials:
    credentials:
      valueFromSecret:
        key: credentials
        name: gcp-trident-protect-src-bucket-secret
```

Amazon S3 (AWS)

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
    sessionToken:
      valueFromSecret:
        key: sessionToken
        name: s3-secret
```

Para ambientes EKS que utilizam o Pod Identity com o movimentador de dados Kopia, você pode remover o `providerCredentials` seção e adicione `useIAM: true` sob o `s3` configuração em vez disso.

Microsoft Azure

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret
```

S3 genérico

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

StorageGrid S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

Exemplos de criação do AppVault usando a CLI do Trident Protect

Você pode usar os seguintes exemplos de comandos da CLI para criar solicitações de configuração (CRs) do AppVault para cada provedor.

- Opcionalmente, você pode especificar um segredo do Kubernetes que contenha senhas personalizadas para a criptografia dos repositórios Restic e Kopia. ConsulteSenhas do repositório do Data Mover para mais informações.
- Para objetos do AppVault no S3, você pode opcionalmente especificar um URL de proxy de saída para o tráfego de saída do S3 usando o `--proxy-url <ip_address:port>` argumento.

Google Cloud

```
tridentctl-protect create vault GCP <vault-name> \
--bucket <mybucket> \
--project <my-gcp-project> \
--secret <secret-name>/credentials \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> \
--account <account-name> \
--bucket <bucket-name> \
--secret <secret-name> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

S3 genérico

```
tridentctl-protect create vault GenericS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

StorageGrid S3

```
tridentctl-protect create vault StorageGridS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Veja as informações do AppVault

Você pode usar o plugin Trident Protect CLI para visualizar informações sobre os objetos do AppVault que você criou no cluster.

Passos

1. Visualizar o conteúdo de um objeto do AppVault:

```
tridentctl-protect get appvaultcontent gcp-vault \
--show-resources all \
-n trident-protect
```

Exemplo de saída:

```
+-------------+-------+----------+-----------------------------
+---------------------------+
|   CLUSTER   |  APP  |   TYPE   |            NAME             |
TIMESTAMP         |
+-------------+-------+----------+-----------------------------
+---------------------------+
|             | mysql | snapshot | mysnap                      | 2024-
08-09 21:02:11 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815180300 | 2024-
08-15 18:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815200300 | 2024-
08-15 20:03:06 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815180300 | 2024-
08-15 18:04:25 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:30 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815200300 | 2024-
08-15 20:04:21 (UTC) |
| production1 | mysql | backup   | mybackup5                   | 2024-
08-09 22:25:13 (UTC) |
|             | mysql | backup   | mybackup                    | 2024-
08-09 21:02:52 (UTC) |
+-------------+-------+----------+-----------------------------
+---------------------------+
```

2. Opcionalmente, para visualizar o AppVaultPath de cada recurso, use a flag `--show-paths` .

   O nome do cluster na primeira coluna da tabela só estará disponível se um nome de cluster tiver sido especificado na instalação do Trident Protect no Helm. Por exemplo: `--set clusterName=production1` .

Remover um AppVault

Você pode remover um objeto do AppVault a qualquer momento.

> Não remova o `finalizers` Digite a chave no CR do AppVault antes de excluir o objeto do AppVault. Caso isso aconteça, poderá resultar em dados residuais no bucket do AppVault e recursos órfãos no cluster.

Antes de começar

Certifique-se de ter excluído todos os CRs de snapshot e backup que estavam sendo usados pelo AppVault que você deseja excluir.

Remova um AppVault usando a CLI do Kubernetes

1. Remova o objeto AppVault e substitua-o. `appvault-name` com o nome do objeto AppVault a ser removido:

```
kubectl delete appvault <appvault-name> \
-n trident-protect
```

Remova um AppVault usando a CLI do Trident Protect.

1. Remova o objeto AppVault e substitua-o. `appvault-name` com o nome do objeto AppVault a ser removido:

```
tridentctl-protect delete appvault <appvault-name> \
-n trident-protect
```

Defina uma aplicação para gestão com o Trident Protect.

Você pode definir um aplicativo que deseja gerenciar com o Trident Protect criando um CR (Create Request) de aplicativo e um CR do AppVault associado.

Criar um AppVault CR

Você precisa criar um AppVault CR que será usado ao executar operações de proteção de dados no aplicativo, e o AppVault CR precisa residir no cluster onde o Trident Protect está instalado. O pedido de configuração (CR) do AppVault é específico para o seu ambiente; para exemplos de pedidos de configuração do AppVault, consulte [link para a documentação]."Recursos personalizados do AppVault."

Definir uma aplicação

Você precisa definir cada aplicativo que deseja gerenciar com o Trident Protect. Você pode definir um aplicativo para gerenciamento criando manualmente uma solicitação de configuração (CR) do aplicativo ou usando a CLI do Trident Protect.

Adicionar um aplicativo usando um CR

Passos

1. Crie o arquivo CR do aplicativo de destino:
   a. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `maria-app.yaml`).
   b. Configure os seguintes atributos:
      - **metadata.name**: (*Obrigatório*) O nome do recurso personalizado do aplicativo. Anote o nome escolhido, pois outros arquivos CR necessários para operações de proteção farão referência a esse valor.
      - **spec.includedNamespaces**: (*Obrigatório*) Use o seletor de namespace e rótulo para especificar os namespaces e recursos que o aplicativo utiliza. O namespace da aplicação deve fazer parte desta lista. O seletor de rótulos é opcional e pode ser usado para filtrar recursos dentro de cada namespace especificado.
      - **spec.includedClusterScopedResources**: (Opcional) Use este atributo para especificar recursos com escopo de cluster a serem incluídos na definição do aplicativo. Este atributo permite selecionar esses recursos com base em seu grupo, versão, tipo e rótulos.
        - **groupVersionKind**: (*Obrigatório*) Especifica o grupo, a versão e o tipo da API do recurso com escopo de cluster.
        - **labelSelector**: (Opcional) Filtra os recursos com escopo de cluster com base em seus rótulos.
      - **metadata.annotations.protect.trident.netapp.io/skip-vm-freeze**: (*Opcional*) Esta anotação só se aplica a aplicações definidas a partir de máquinas virtuais, como em ambientes KubeVirt, onde o congelamento do sistema de arquivos ocorre antes dos snapshots. Especifique se este aplicativo pode gravar no sistema de arquivos durante um snapshot. Se definido como verdadeiro, o aplicativo ignora a configuração global e pode gravar no sistema de arquivos durante um snapshot. Se definido como falso, o aplicativo ignora a configuração global e o sistema de arquivos é congelado durante a captura de um instantâneo. Se especificada, mas a aplicação não tiver máquinas virtuais na definição da aplicação, a anotação será ignorada. Caso não seja especificado, o aplicativo segue o"configuração de congelamento global do Trident Protect" .

Caso precise aplicar essa anotação após a criação de um aplicativo, você pode usar o seguinte comando:

```
kubectl annotate application -n <application CR namespace> <application CR name> protect.trident.netapp.io/skip-vm-freeze="true"
```

+
Exemplo de YAML:

+

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  annotations:
    protect.trident.netapp.io/skip-vm-freeze: "false"
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: namespace-1
      labelSelector:
        matchLabels:
          app: example-app
    - namespace: namespace-2
      labelSelector:
        matchLabels:
          app: another-example-app
  includedClusterScopedResources:
    - groupVersionKind:
        group: rbac.authorization.k8s.io
        kind: ClusterRole
        version: v1
      labelSelector:
        matchLabels:
          mylabel: test
```

1. (Opcional) Adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:

   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `Include` ou `Exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:

     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.
       - **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
       - **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.
       - **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.
       - **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.

- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

> Quando ambos `resourceFilter` e `labelSelector` são usados, `resourceFilter` primeiro corre e depois `labelSelector` é aplicado aos recursos resultantes.

Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

2. Após criar a solicitação de configuração (CR) do aplicativo que corresponda ao seu ambiente, aplique a CR. Por exemplo:

```
kubectl apply -f maria-app.yaml
```

Passos

1. Crie e aplique a definição do aplicativo usando um dos exemplos a seguir, substituindo os valores entre colchetes pelas informações do seu ambiente. Você pode incluir namespaces e recursos na definição do aplicativo usando listas separadas por vírgulas com os argumentos mostrados nos exemplos.

   Você pode, opcionalmente, usar uma anotação ao criar um aplicativo para especificar se o aplicativo pode gravar no sistema de arquivos durante um snapshot. Isso só se aplica a aplicativos definidos a partir de máquinas virtuais, como em ambientes KubeVirt, onde o congelamento do sistema de arquivos ocorre antes dos snapshots. Se você definir a anotação para `true` , o aplicativo ignora a configuração global e pode gravar no sistema de arquivos durante um snapshot. Se você definir para `false` , o aplicativo ignora a configuração global e o sistema de arquivos fica congelado durante a

captura de um snapshot. Se você usar a anotação, mas o aplicativo não tiver máquinas virtuais na definição do aplicativo, a anotação será ignorada. Se você não usar a anotação, o aplicativo seguirá o padrão."configuração de congelamento global do Trident Protect" .

Para especificar a anotação ao usar a CLI para criar um aplicativo, você pode usar o `--annotation` bandeira.

- Crie o aplicativo e utilize a configuração global para o comportamento de congelamento do sistema de arquivos:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace>
```

- Crie o aplicativo e configure as definições locais do aplicativo para o comportamento de congelamento do sistema de arquivos:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace> --annotation protect.trident.netapp.io/skip-vm-freeze
=<"true"|"false">
```

Você pode usar `--resource-filter-include` e `--resource-filter-exclude` sinalizadores para incluir ou excluir recursos com base em `resourceSelectionCriteria` tais como grupo, tipo, versão, rótulos, nomes e namespaces, conforme mostrado no exemplo a seguir:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
--resource-filter-include
'[{"Group":"apps","Kind":"Deployment","Version":"v1","Names":["my-
deployment"],"Namespaces":["my-
namespace"],"LabelSelectors":["app=my-app"]}]'
```

Proteja aplicativos usando o Trident Protect.

Você pode proteger todos os aplicativos gerenciados pelo Trident Protect criando snapshots e backups usando uma política de proteção automatizada ou de forma pontual.

Você pode configurar o Trident Protect para congelar e descongelar sistemas de arquivos durante operações de proteção de dados. "Saiba mais sobre como configurar o congelamento do sistema de arquivos com o Trident Protect.".

Criar um instantâneo sob demanda

Você pode criar um instantâneo sob demanda a qualquer momento.

Os recursos com escopo de cluster são incluídos em um backup, snapshot ou clone se forem explicitamente referenciados na definição do aplicativo ou se tiverem referências a qualquer um dos namespaces do aplicativo.

Criar um instantâneo usando um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-snapshot-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.applicationRef**: O nome do aplicativo no Kubernetes a ser criado a partir do snapshot.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do snapshot (metadados) deve ser armazenado.
   - **spec.reclaimPolicy**: (Opcional) Define o que acontece com o AppArchive de um snapshot quando o CR do snapshot é excluído. Isso significa que mesmo quando configurado para `Retain` , o instantâneo será excluído. Opções válidas:
     - `Retain`(padrão)
     - `Delete`

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: Snapshot
     metadata:
       namespace: my-app-namespace
       name: my-cr-name
     spec:
       applicationRef: my-application
       appVaultRef: appvault-name
       reclaimPolicy: Delete
     ```
3. Depois de preencher o `trident-protect-snapshot-cr.yaml` Arquivo com os valores corretos, aplique o CR:

   ```
   kubectl apply -f trident-protect-snapshot-cr.yaml
   ```

Crie um instantâneo usando a CLI

Passos

1. Crie o instantâneo, substituindo os valores entre colchetes pelas informações do seu ambiente. Por exemplo:

   ```
   tridentctl-protect create snapshot <my_snapshot_name> --appvault
   <my_appvault_name> --app <name_of_app_to_snapshot> -n
   <application_namespace>
   ```

Crie um backup sob demanda

Você pode fazer backup de um aplicativo a qualquer momento.

> Os recursos com escopo de cluster são incluídos em um backup, snapshot ou clone se forem explicitamente referenciados na definição do aplicativo ou se tiverem referências a qualquer um dos namespaces do aplicativo.

Antes de começar

Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de backup do S3 de longa duração. Se o token expirar durante a operação de backup, a operação poderá falhar.

- Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
- Consulte o "Documentação do AWS IAM" Para obter mais informações sobre credenciais com recursos da AWS.

Crie um backup usando um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-backup-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.applicationRef**: (*Obrigatório*) O nome do aplicativo no Kubernetes para o qual deseja fazer backup.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup deve ser armazenado.
   - **spec.dataMover**: (Opcional) Uma string indicando qual ferramenta de backup usar para a operação de backup. Valores possíveis (diferencia maiúsculas de minúsculas):
     - `Restic`
     - `Kopia`(padrão)
   - **spec.reclaimPolicy**: (Opcional) Define o que acontece com um backup quando ele é liberado de sua reivindicação. Valores possíveis:
     - `Delete`
     - `Retain`(padrão)
   - **spec.snapshotRef**: (*Opcional*): Nome do snapshot a ser usado como fonte do backup. Caso não seja fornecida, será criada e armazenada uma cópia de segurança temporária.
   - **metadata.annotations.protect.trident.netapp.io/full-backup** : (*Opcional*) Esta anotação é usada para especificar se um backup deve ser não incremental. Por padrão, todos os backups são incrementais. No entanto, se esta anotação estiver definida como `true` , o backup deixa de ser incremental. Caso não seja especificado, o backup seguirá a configuração padrão de backup incremental. A melhor prática é realizar um backup completo periodicamente e, em seguida, realizar backups incrementais entre os backups completos para minimizar o risco associado às restaurações.

     Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup: "true"
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Depois de preencher o `trident-protect-backup-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-backup-cr.yaml
```

Crie um backup usando a CLI

Passos

1. Crie o backup, substituindo os valores entre colchetes pelas informações do seu ambiente. Por exemplo:

```
tridentctl-protect create backup <my_backup_name> --appvault <my-
vault-name> --app <name_of_app_to_back_up> --data-mover
<Kopia_or_Restic> -n <application_namespace>
```

   Você pode usar opcionalmente o `--full-backup` Sinalizador para especificar se um backup deve ser não incremental. Por padrão, todos os backups são incrementais. Quando essa opção é usada, o backup deixa de ser incremental. A melhor prática é realizar um backup completo periodicamente e, em seguida, realizar backups incrementais entre os backups completos para minimizar o risco associado às restaurações.

Crie um cronograma de proteção de dados

Uma política de proteção protege um aplicativo criando instantâneos, backups ou ambos em um cronograma definido. Você pode optar por criar snapshots e backups por hora, dia, semana e mês, e pode especificar o número de cópias a serem mantidas. Você pode agendar um backup completo não incremental usando a anotação full-backup-rule. Por padrão, todos os backups são incrementais. Executar um backup completo periodicamente, juntamente com backups incrementais entre eles, ajuda a reduzir o risco associado às restaurações.

- Você pode criar agendamentos para snapshots somente configurando `backupRetention` para zero e `snapshotRetention` para um valor maior que zero. Contexto `snapshotRetention` Definir como zero significa que quaisquer backups agendados ainda criarão snapshots, mas estes serão temporários e excluídos imediatamente após a conclusão do backup.
- Os recursos com escopo de cluster são incluídos em um backup, snapshot ou clone se forem explicitamente referenciados na definição do aplicativo ou se tiverem referências a qualquer um dos namespaces do aplicativo.

Crie um cronograma usando uma CR (Central de Recursos).

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-schedule-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.dataMover**: (Opcional) Uma string indicando qual ferramenta de backup usar para a operação de backup. Valores possíveis (diferencia maiúsculas de minúsculas):
     - `Restic`
     - `Kopia`(padrão)
   - **spec.applicationRef**: O nome do aplicativo no Kubernetes para o qual será feito o backup.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup deve ser armazenado.
   - **spec.backupRetention**: O número de backups a serem retidos. Zero indica que nenhum backup deve ser criado (somente instantâneos).
   - **spec.snapshotRetention**: O número de snapshots a serem mantidos. Zero indica que nenhum instantâneo deve ser criado.
   - **spec.granularity**: A frequência com que a programação deve ser executada. Valores possíveis, juntamente com os campos associados obrigatórios:
     - `Hourly`(requer que você especifique) `spec.minute`)
     - `Daily`(requer que você especifique) `spec.minute` e `spec.hour`)
     - `Weekly`(requer que você especifique) `spec.minute, spec.hour` , e `spec.dayOfWeek`)
     - `Monthly`(requer que você especifique) `spec.minute, spec.hour` , e `spec.dayOfMonth`)
     - `Custom`
   - **spec.dayOfMonth**: (*Opcional*) O dia do mês (1 - 31) em que o agendamento deve ser executado. Este campo é obrigatório se a granularidade estiver definida como `Monthly` . O valor deve ser fornecido como uma string.
   - **spec.dayOfWeek**: (*Opcional*) O dia da semana (0 - 7) em que a programação deve ser executada. Valores de 0 ou 7 indicam domingo. Este campo é obrigatório se a granularidade estiver definida como `Weekly` . O valor deve ser fornecido como uma string.
   - **spec.hour**: (*Opcional*) A hora do dia (0 - 23) em que a programação deve ser executada. Este campo é obrigatório se a granularidade estiver definida como `Daily` , `Weekly` , ou `Monthly` . O valor deve ser fornecido como uma string.
   - **spec.minute**: (*Opcional*) O minuto da hora (0 - 59) em que a programação deve ser executada. Este campo é obrigatório se a granularidade estiver definida como `Hourly` , `Daily` , `Weekly` , ou `Monthly` . O valor deve ser fornecido como uma string.
   - **metadata.annotations.protect.trident.netapp.io/full-backup-rule**: (Opcional) Esta anotação é usada para especificar a regra para agendamento de backup completo. Você pode configurá-lo para `always` Para backup completo constante ou personalize de acordo com suas necessidades. Por exemplo, se você escolher a granularidade diária, poderá especificar os dias

da semana em que o backup completo deverá ocorrer.

Exemplo de YAML para agendamento de backup e snapshot:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup-rule: "Monday,Thursday"
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: Daily
  hour: "0"
  minute: "0"
```

Exemplo de YAML para programação somente de snapshot:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-snapshot-schedule
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "0"
  snapshotRetention: "15"
  granularity: Daily
  hour: "2"
  minute: "0"
```

3. Depois de preencher o `trident-protect-schedule-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

Crie um agendamento usando a CLI (linha de comando)

Passos

1. Crie o cronograma de proteção, substituindo os valores entre parênteses pelas informações do seu ambiente. Por exemplo:

   Você pode usar `tridentctl-protect create schedule --help` Para visualizar informações de ajuda detalhadas para este comando.

   ```
   tridentctl-protect create schedule <my_schedule_name> --appvault
   <my_appvault_name> --app <name_of_app_to_snapshot> --backup
   -retention <how_many_backups_to_retain> --data-mover
   <Kopia_or_Restic> --day-of-month <day_of_month_to_run_schedule>
   --day-of-week <day_of_month_to_run_schedule> --granularity
   <frequency_to_run> --hour <hour_of_day_to_run> --minute
   <minute_of_hour_to_run> --recurrence-rule <recurrence> --snapshot
   -retention <how_many_snapshots_to_retain> -n <application_namespace>
   --full-backup-rule <string>
   ```

   Você pode configurar o `--full-backup-rule` bandeira para `always` Para backup completo constante ou personalize de acordo com suas necessidades. Por exemplo, se você escolher a granularidade diária, poderá especificar os dias da semana em que o backup completo deverá ocorrer. Por exemplo, use `--full-backup-rule "Monday,Thursday"` Agendar backup completo às segundas e quintas-feiras.

   Para agendamentos somente de instantâneo, defina `--backup-retention 0` e especifique um valor maior que 0 para `--snapshot-retention` .

Excluir um instantâneo

Exclua os snapshots agendados ou sob demanda que você não precisa mais.

Passos

1. Remova o CR (Código de Referência) do instantâneo associado ao instantâneo:

   ```
   kubectl delete snapshot <snapshot_name> -n my-app-namespace
   ```

Excluir um backup

Exclua os backups agendados ou sob demanda que você não precisa mais.

Certifique-se de que a política de reembolso esteja definida como `Delete` Remover todos os dados de backup do armazenamento de objetos. A configuração padrão da política é `Retain` Para evitar a perda acidental de dados. Se a política não for alterada para `Delete` Os dados de backup permanecerão no armazenamento de objetos e precisarão ser excluídos manualmente.

Passos

1. Remova o CR de backup associado ao backup:

```
kubectl delete backup <backup_name> -n my-app-namespace
```

Verifique o status de uma operação de backup.

Você pode usar a linha de comando para verificar o status de uma operação de backup que está em andamento, foi concluída ou falhou.

Passos

1. Utilize o seguinte comando para recuperar o status da operação de backup, substituindo os valores entre colchetes pelas informações do seu ambiente:

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath
='{.status}'
```

Habilitar backup e restauração para operações do Azure NetApp Files (ANF)

Se você instalou o Trident Protect, pode habilitar a funcionalidade de backup e restauração com uso eficiente de espaço para back-ends de armazenamento que utilizam a classe de armazenamento azure-netapp-files e foram criados antes do Trident 24.06. Essa funcionalidade opera com volumes NFSv4 e não consome espaço adicional do pool de capacidade.

Antes de começar

Garanta o seguinte:

- Você instalou o Trident Protect.
- Você definiu um aplicativo no Trident Protect. Este aplicativo terá funcionalidade de proteção limitada até que você conclua este procedimento.
- Você tem `azure-netapp-files` Selecionada como a classe de armazenamento padrão para seu backend de armazenamento.

Expandir para ver as etapas de configuração

1. Se o volume ANF foi criado antes da atualização para o Trident 24.10, Trident os seguintes passos:

   a. Habilite o diretório de instantâneos para cada PV baseado em azure-netapp-files e associado ao aplicativo:

   ```
   tridentctl update volume <pv name> --snapshot-dir=true -n trident
   ```

   b. Confirme se o diretório de snapshots foi habilitado para cada PV associado:

   ```
   tridentctl get volume <pv name> -n trident -o yaml | grep
   snapshotDir
   ```

   Resposta:

   ```
   snapshotDirectory: "true"
   ```

   +
   Quando o diretório de instantâneos não está habilitado, o sistema escolhe a funcionalidade de backup regular, que consome temporariamente espaço no pool de capacidade durante o processo de backup. Nesse caso, certifique-se de que haja espaço suficiente disponível no pool de capacidade para criar um volume temporário do tamanho do volume que está sendo copiado.

Resultado

O aplicativo está pronto para backup e restauração usando o Trident Protect. Cada PVC também pode ser usado por outros aplicativos para backups e restaurações.

Restaurar aplicativos

Restaure aplicativos usando o Trident Protect.

Você pode usar o Trident Protect para restaurar seu aplicativo a partir de um snapshot ou backup. A restauração a partir de um snapshot existente será mais rápida ao restaurar o aplicativo no mesmo cluster.

- Ao restaurar um aplicativo, todos os ganchos de execução configurados para ele são restaurados juntamente com o aplicativo. Se houver um gancho de execução pós-restauração, ele será executado automaticamente como parte da operação de restauração.
- A restauração a partir de um backup para um namespace diferente ou para o namespace original é suportada para volumes qtree. No entanto, a restauração a partir de um snapshot para um namespace diferente ou para o namespace original não é suportada para volumes qtree.
- Você pode usar configurações avançadas para personalizar as operações de restauração. Para saber mais, consulte "Utilize as configurações avançadas de restauração do Trident Protect.".

Restaurar a partir de um backup para um namespace diferente.

Ao restaurar um backup para um namespace diferente usando um CR de BackupRestore, o Trident Protect restaura o aplicativo em um novo namespace e cria um CR de aplicativo para o aplicativo restaurado. Para proteger a aplicação restaurada, crie backups ou snapshots sob demanda, ou estabeleça um cronograma de proteção.

Restaurar um backup para um namespace diferente, mantendo os recursos existentes, não alterará nenhum recurso que compartilhe o mesmo nome que os recursos do backup. Para restaurar todos os recursos do backup, exclua e recrie o namespace de destino ou restaure o backup para um novo namespace.

Antes de começar

Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de restauração do S3 de longa duração. Se o token expirar durante a operação de restauração, a operação poderá falhar.

- Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
- Consulte o "Documentação do AWS IAM" Para obter mais informações sobre credenciais com recursos da AWS.

Ao restaurar backups usando o Kopia como o movimentador de dados, você pode, opcionalmente, especificar anotações no CR ou usar a CLI para controlar o comportamento do armazenamento temporário usado pelo Kopia. Consulte o "Documentação Kopia" Para obter mais informações sobre as opções que você pode configurar. Use o `tridentctl-protect create --help` Para obter mais informações sobre como especificar anotações com a CLI do Trident Protect, consulte o comando.

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-backup-restore-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do backup é armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup está armazenado.
   - **spec.namespaceMapping**: O mapeamento do namespace de origem da operação de restauração para o namespace de destino. Substituir `my-source-namespace` e `my-destination-namespace` com informações do seu ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: my-cr-name
       namespace: my-destination-namespace
     spec:
       appArchivePath: my-backup-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (Opcional) Se precisar selecionar apenas determinados recursos do aplicativo para restaurar, adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:

   > O Trident Protect seleciona alguns recursos automaticamente devido à sua relação com recursos que você seleciona. Por exemplo, se você selecionar um recurso de reivindicação de volume persistente e ele tiver um pod associado, o Trident Protect também restaurará o pod associado.

   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `Include` ou `Exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:
     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos

dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.

- **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
- **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.
- **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.
- **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.
- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

  Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Depois de preencher o `trident-protect-backup-restore-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Use a CLI

Passos

1. Restaure o backup para um namespace diferente, substituindo os valores entre colchetes pelas informações do seu ambiente. O `namespace-mapping` O argumento usa namespaces separados por dois pontos para mapear namespaces de origem para os namespaces de destino corretos no formato `source1:dest1,source2:dest2` . Por exemplo:

```
tridentctl-protect create backuprestore <my_restore_name> \
--backup <backup_namespace>/<backup_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

Restaurar a partir de um backup para o namespace original

Você pode restaurar um backup para o namespace original a qualquer momento.

Antes de começar

Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de restauração do S3 de longa duração. Se o token expirar durante a operação de restauração, a operação poderá falhar.

- Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
- Consulte o "Documentação do AWS IAM" Para obter mais informações sobre credenciais com recursos da AWS.

> ⓘ Ao restaurar backups usando o Kopia como o movimentador de dados, você pode, opcionalmente, especificar anotações no CR ou usar a CLI para controlar o comportamento do armazenamento temporário usado pelo Kopia. Consulte o "Documentação Kopia" Para obter mais informações sobre as opções que você pode configurar. Use o `tridentctl-protect create --help` Para obter mais informações sobre como especificar anotações com a CLI do Trident Protect, consulte o comando.

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-backup-ipr-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do backup é armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup está armazenado.

     Por exemplo:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appArchivePath: my-backup-path
  appVaultRef: appvault-name
```

3. (Opcional) Se precisar selecionar apenas determinados recursos do aplicativo para restaurar, adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:

   > O Trident Protect seleciona alguns recursos automaticamente devido à sua relação com recursos que você seleciona. Por exemplo, se você selecionar um recurso de reivindicação de volume persistente e ele tiver um pod associado, o Trident Protect também restaurará o pod associado.

   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `Include` ou `Exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:
     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.
       - **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
       - **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.

- **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.
- **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.
- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

  Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Depois de preencher o `trident-protect-backup-ipr-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

Use a CLI

Passos

1. Restaure o backup para o namespace original, substituindo os valores entre colchetes pelas informações do seu ambiente. O `backup` O argumento usa um namespace e um nome de backup no formato `<namespace>/<name>` . Por exemplo:

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

Restaurar a partir de um backup para um cluster diferente

Você pode restaurar um backup em um cluster diferente caso haja algum problema com o cluster original.

> Ao restaurar backups usando o Kopia como o movimentador de dados, você pode, opcionalmente, especificar anotações no CR ou usar a CLI para controlar o comportamento do armazenamento temporário usado pelo Kopia. Consulte o "Documentação Kopia" Para obter mais informações sobre as opções que você pode configurar. Use o `tridentctl-protect create --help` Para obter mais informações sobre como especificar anotações com a CLI do Trident Protect, consulte o comando.

Antes de começar

Certifique-se de que os seguintes pré-requisitos sejam atendidos:

- O cluster de destino tem o Trident Protect instalado.
- O cluster de destino tem acesso ao caminho do bucket do mesmo AppVault que o cluster de origem, onde o backup está armazenado.
- Ao executar o AppVault CR, certifique-se de que seu ambiente local possa se conectar ao bucket de armazenamento de objetos definido nele. `tridentctl-protect get appvaultcontent` comando. Caso as restrições de rede impeçam o acesso, execute a CLI do Trident Protect a partir de um pod no cluster de destino.
- Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de restauração de longa duração. Se o token expirar durante a operação de restauração, a operação poderá falhar.
  - Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
  - Consulte o "Documentação do AWS" Para obter mais informações sobre credenciais com recursos da AWS.

Passos

1. Verifique a disponibilidade do AppVault CR no cluster de destino usando o plugin Trident Protect CLI:

```
tridentctl-protect get appvault --context <destination_cluster_name>
```

> Certifique-se de que o namespace destinado à restauração do aplicativo exista no cluster de destino.

2. Visualize o conteúdo de backup do AppVault disponível no cluster de destino:

```
tridentctl-protect get appvaultcontent <appvault_name> \
--show-resources backup \
--show-paths \
--context <destination_cluster_name>
```

Executar este comando exibe os backups disponíveis no AppVault, incluindo seus clusters de origem, nomes de aplicativos correspondentes, registros de data e hora e caminhos de arquivamento.

Exemplo de saída:

```
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
|   CLUSTER   |    APP    |  TYPE  |      NAME       |        TIMESTAMP
|    PATH     |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
| production1 | wordpress | backup | wordpress-bkup-1| 2024-10-30
08:37:40 (UTC)| backuppath1 |
| production1 | wordpress | backup | wordpress-bkup-2| 2024-10-30
08:37:40 (UTC)| backuppath2 |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
```

3. Restaure o aplicativo no cluster de destino usando o nome do AppVault e o caminho do arquivo:

Use um CR

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-backup-restore-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup está armazenado.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do backup é armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

     Caso o BackupRestore CR não esteja disponível, você pode usar o comando mencionado na etapa 2 para visualizar o conteúdo do backup.

   - **spec.namespaceMapping**: O mapeamento do namespace de origem da operação de restauração para o namespace de destino. Substituir `my-source-namespace` e `my-destination-namespace` com informações do seu ambiente.

     Por exemplo:

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  namespaceMapping: [{"source": "my-source-namespace", "
destination": "my-destination-namespace"}]
```

3. Depois de preencher o `trident-protect-backup-restore-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Use a CLI

1. Use o seguinte comando para restaurar o aplicativo, substituindo os valores entre colchetes pelas informações do seu ambiente. O argumento namespace-mapping usa namespaces separados por

dois pontos para mapear namespaces de origem para os namespaces de destino corretos no formato source1:dest1,source2:dest2. Por exemplo:

```
tridentctl-protect create backuprestore <restore_name> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--appvault <appvault_name> \
--path <backup_path> \
--context <destination_cluster_name> \
-n <application_namespace>
```

Restaurar a partir de um snapshot para um namespace diferente.

Você pode restaurar dados de um snapshot usando um arquivo de recurso personalizado (CR) para um namespace diferente ou para o namespace de origem original. Ao restaurar um snapshot para um namespace diferente usando um CR de SnapshotRestore, o Trident Protect restaura o aplicativo em um novo namespace e cria um CR de aplicativo para o aplicativo restaurado. Para proteger a aplicação restaurada, crie backups ou snapshots sob demanda, ou estabeleça um cronograma de proteção.

> O SnapshotRestore é compatível com o `spec.storageClassMapping` atributo, mas somente quando as classes de armazenamento de origem e destino usam o mesmo backend de armazenamento. Se você tentar restaurar para um `StorageClass` Se usar um sistema de armazenamento diferente, a operação de restauração falhará.

Antes de começar

Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de restauração do S3 de longa duração. Se o token expirar durante a operação de restauração, a operação poderá falhar.

- Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
- Consulte o "Documentação do AWS IAM" Para obter mais informações sobre credenciais com recursos da AWS.

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-snapshot-restore-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do snapshot está armazenado.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do snapshot está armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get snapshots <SNAPHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.namespaceMapping**: O mapeamento do namespace de origem da operação de restauração para o namespace de destino. Substituir `my-source-namespace` e `my-destination-namespace` com informações do seu ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: my-app-namespace
     spec:
       appVaultRef: appvault-name
       appArchivePath: my-snapshot-path
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (Opcional) Se precisar selecionar apenas determinados recursos do aplicativo para restaurar, adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:

   > O Trident Protect seleciona alguns recursos automaticamente devido à sua relação com recursos que você seleciona. Por exemplo, se você selecionar um recurso de reivindicação de volume persistente e ele tiver um pod associado, o Trident Protect também restaurará o pod associado.

   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `Include` ou `Exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:
     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos

dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.

- **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
- **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.
- **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.
- **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.
- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

  Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Depois de preencher o `trident-protect-snapshot-restore-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Use a CLI

Passos

1. Restaure o snapshot para um namespace diferente, substituindo os valores entre colchetes pelas informações do seu ambiente.

   - O `snapshot` O argumento usa um namespace e um nome de snapshot no formato `<namespace>/<name>` .
   - O `namespace-mapping` O argumento usa namespaces separados por dois pontos para mapear

namespaces de origem para os namespaces de destino corretos no formato `source1:dest1,source2:dest2`.

Por exemplo:

```
tridentctl-protect create snapshotrestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

Restaurar de um snapshot para o namespace original

Você pode restaurar um snapshot para o namespace original a qualquer momento.

Antes de começar

Certifique-se de que o tempo de expiração do token de sessão da AWS seja suficiente para quaisquer operações de restauração do S3 de longa duração. Se o token expirar durante a operação de restauração, a operação poderá falhar.

- Consulte o "Documentação do AWS API" Para obter mais informações sobre como verificar a expiração do token de sessão atual.
- Consulte o "Documentação do AWS IAM" Para obter mais informações sobre credenciais com recursos da AWS.

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-snapshot-ipr-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do snapshot está armazenado.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do snapshot está armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotInplaceRestore
     metadata:
       name: my-cr-name
       namespace: my-app-namespace
     spec:
       appVaultRef: appvault-name
         appArchivePath: my-snapshot-path
     ```

3. (Opcional) Se precisar selecionar apenas determinados recursos do aplicativo para restaurar, adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:

   > O Trident Protect seleciona alguns recursos automaticamente devido à sua relação com recursos que você seleciona. Por exemplo, se você selecionar um recurso de reivindicação de volume persistente e ele tiver um pod associado, o Trident Protect também restaurará o pod associado.

   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `Include` ou `Exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:
     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.
       - **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
       - **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.
       - **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.

- **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.
- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

  Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Depois de preencher o `trident-protect-snapshot-ipr-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

Use a CLI

Passos

1. Restaure o snapshot para o namespace original, substituindo os valores entre colchetes pelas informações do seu ambiente. Por exemplo:

```
tridentctl-protect create snapshotinplacerestore <my_restore_name> \
--snapshot <snapshot_to_restore> \
-n <application_namespace>
```

Verifique o status de uma operação de restauração.

Você pode usar a linha de comando para verificar o status de uma operação de restauração que está em andamento, foi concluída ou falhou.

Passos

1. Utilize o seguinte comando para recuperar o status da operação de restauração, substituindo os valores entre colchetes pelas informações do seu ambiente:

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o
jsonpath='{.status}'
```

Utilize as configurações avançadas de restauração do Trident Protect.

Você pode personalizar operações de restauração usando configurações avançadas, como anotações, configurações de namespace e opções de armazenamento para atender aos seus requisitos específicos.

Anotações e rótulos de namespace durante operações de restauração e failover

Durante as operações de restauração e failover, os rótulos e anotações no namespace de destino são ajustados para corresponder aos rótulos e anotações no namespace de origem. Rótulos ou anotações do namespace de origem que não existem no namespace de destino são adicionados, e quaisquer rótulos ou anotações já existentes são sobrescritos para corresponder ao valor do namespace de origem. Rótulos ou anotações que existem apenas no espaço de nomes de destino permanecem inalterados.

> Se você usa o Red Hat OpenShift, é importante observar o papel crítico das anotações de namespace em ambientes OpenShift. As anotações de namespace garantem que os pods restaurados cumpram as permissões e configurações de segurança apropriadas definidas pelas restrições de contexto de segurança (SCCs) do OpenShift e possam acessar volumes sem problemas de permissão. Para mais informações, consulte o "Documentação sobre restrições de contexto de segurança do OpenShift" .

Você pode impedir que anotações específicas no namespace de destino sejam sobrescritas definindo a variável de ambiente do Kubernetes. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` antes de executar a operação de restauração ou failover. Por exemplo:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

> Ao executar uma operação de restauração ou failover, quaisquer anotações e rótulos de namespace especificados em `restoreSkipNamespaceAnnotations` e `restoreSkipNamespaceLabels` são excluídos da operação de restauração ou failover. Certifique-se de que essas configurações sejam definidas durante a instalação inicial do Helm. Para saber mais, consulte "Configurar opções de filtragem de namespace e AutoSupport".

Se você instalou o aplicativo de origem usando o Helm com o `--create-namespace` bandeira, tratamento

especial é dado ao `name` Legenda da etiqueta. Durante o processo de restauração ou failover, o Trident Protect copia esse rótulo para o namespace de destino, mas atualiza o valor para o valor do namespace de destino se o valor da origem corresponder ao namespace de origem. Se esse valor não corresponder ao namespace de origem, ele será copiado para o namespace de destino sem alterações.

Exemplo

O exemplo a seguir apresenta um namespace de origem e um de destino, cada um com anotações e rótulos diferentes. É possível visualizar o estado do namespace de destino antes e depois da operação, bem como a forma como as anotações e os rótulos são combinados ou sobrescritos no namespace de destino.

Antes da operação de restauração ou failover

A tabela a seguir ilustra o estado dos namespaces de origem e destino do exemplo antes da operação de restauração ou failover:

Espaço de nomes	Anotações	Etiquetas
Espaço de nomes ns-1 (fonte)	• anotação.um/chave: "valoratualizado" • anotação.dois/chave: "verdadeiro"	• ambiente=produção • conformidade=hipaa • nome=ns-1
Espaço de nomes ns-2 (destino)	• anotação.um/chave: "verdadeiro" • anotação.três/chave: "falso"	• função=banco de dados

Após a operação de restauração

A tabela a seguir ilustra o estado do namespace de destino de exemplo após a operação de restauração ou failover. Algumas chaves foram adicionadas, outras foram sobrescritas, e a `name` O rótulo foi atualizado para corresponder ao namespace de destino:

Espaço de nomes	Anotações	Etiquetas
Espaço de nomes ns-2 (destino)	• anotação.um/chave: "valoratualizado" • anotação.dois/chave: "verdadeiro" • anotação.três/chave: "falso"	• nome=ns-2 • conformidade=hipaa • ambiente=produção • função=banco de dados

Campos suportados

Esta seção descreve campos adicionais disponíveis para operações de restauração.

Mapeamento de classes de armazenamento

O `spec.storageClassMapping` O atributo define um mapeamento de uma classe de armazenamento presente no aplicativo de origem para uma nova classe de armazenamento no cluster de destino. Você pode usar isso ao migrar aplicativos entre clusters com diferentes classes de armazenamento ou ao alterar o backend de armazenamento para operações de BackupRestore.

Exemplo:

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

Anotações suportadas

Esta seção lista as anotações suportadas para configurar vários comportamentos no sistema. Se uma anotação não for definida explicitamente pelo usuário, o sistema usará o valor padrão.

Anotação	Tipo	Descrição	Valor padrão
protect.trident.netapp.io/data-mover-timeout-sec	corda	Tempo máximo (em segundos) permitido para a paralisação da operação de movimentação de dados.	"300"
protect.trident.netapp.io/kopia-content-cache-size-limit-mb	corda	O limite máximo de tamanho (em megabytes) para o cache de conteúdo do Kopia.	"1000"

Replique aplicações usando NetApp SnapMirror e Trident Protect.

Com o Trident Protect, você pode usar os recursos de replicação assíncrona da tecnologia NetApp SnapMirror para replicar dados e alterações de aplicativos de um backend de armazenamento para outro, no mesmo cluster ou entre clusters diferentes.

Anotações e rótulos de namespace durante operações de restauração e failover

Durante as operações de restauração e failover, os rótulos e anotações no namespace de destino são ajustados para corresponder aos rótulos e anotações no namespace de origem. Rótulos ou anotações do namespace de origem que não existem no namespace de destino são adicionados, e quaisquer rótulos ou anotações já existentes são sobrescritos para corresponder ao valor do namespace de origem. Rótulos ou anotações que existem apenas no espaço de nomes de destino permanecem inalterados.

> Se você usa o Red Hat OpenShift, é importante observar o papel crítico das anotações de namespace em ambientes OpenShift. As anotações de namespace garantem que os pods restaurados cumpram as permissões e configurações de segurança apropriadas definidas pelas restrições de contexto de segurança (SCCs) do OpenShift e possam acessar volumes sem problemas de permissão. Para mais informações, consulte o "Documentação sobre restrições de contexto de segurança do OpenShift" .

Você pode impedir que anotações específicas no namespace de destino sejam sobrescritas definindo a variável de ambiente do Kubernetes. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` antes de executar a operação de restauração ou failover. Por exemplo:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

Ao executar uma operação de restauração ou failover, quaisquer anotações e rótulos de namespace especificados em `restoreSkipNamespaceAnnotations` e `restoreSkipNamespaceLabels` são excluídos da operação de restauração ou failover. Certifique-se de que essas configurações sejam definidas durante a instalação inicial do Helm. Para saber mais, consulte "Configurar opções de filtragem de namespace e AutoSupport".

Se você instalou o aplicativo de origem usando o Helm com o `--create-namespace` bandeira, tratamento especial é dado ao `name` Legenda da etiqueta. Durante o processo de restauração ou failover, o Trident Protect copia esse rótulo para o namespace de destino, mas atualiza o valor para o valor do namespace de destino se o valor da origem corresponder ao namespace de origem. Se esse valor não corresponder ao namespace de origem, ele será copiado para o namespace de destino sem alterações.

Exemplo

O exemplo a seguir apresenta um namespace de origem e um de destino, cada um com anotações e rótulos diferentes. É possível visualizar o estado do namespace de destino antes e depois da operação, bem como a forma como as anotações e os rótulos são combinados ou sobrescritos no namespace de destino.

Antes da operação de restauração ou failover

A tabela a seguir ilustra o estado dos namespaces de origem e destino do exemplo antes da operação de restauração ou failover:

Espaço de nomes	Anotações	Etiquetas
Espaço de nomes ns-1 (fonte)	• anotação.um/chave: "valoratualizado" • anotação.dois/chave: "verdadeiro"	• ambiente=produção • conformidade=hipaa • nome=ns-1
Espaço de nomes ns-2 (destino)	• anotação.um/chave: "verdadeiro" • anotação.três/chave: "falso"	• função=banco de dados

Após a operação de restauração

A tabela a seguir ilustra o estado do namespace de destino de exemplo após a operação de restauração ou failover. Algumas chaves foram adicionadas, outras foram sobrescritas, e a `name` O rótulo foi atualizado para corresponder ao namespace de destino:

Espaço de nomes	Anotações	Etiquetas
Espaço de nomes ns-2 (destino)	• anotação.um/chave: "valoratualizado" • anotação.dois/chave: "verdadeiro" • anotação.três/chave: "falso"	• nome=ns-2 • conformidade=hipaa • ambiente=produção • função=banco de dados

Você pode configurar o Trident Protect para congelar e descongelar sistemas de arquivos durante operações de proteção de dados. "Saiba mais sobre como configurar o congelamento do sistema de arquivos com o Trident Protect.".

Ganchos de execução durante operações de failover e reversão

Ao usar o relacionamento AppMirror para proteger seu aplicativo, existem comportamentos específicos relacionados aos ganchos de execução que você deve conhecer durante operações de failover e reversão.

- Durante um failover, os hooks de execução são copiados automaticamente do cluster de origem para o cluster de destino. Você não precisa recriá-los manualmente. Após a falha, os ganchos de execução permanecem presentes na aplicação e serão executados durante quaisquer ações relevantes.
- Durante a reversão ou resincronização reversa, todos os ganchos de execução existentes no aplicativo são removidos. Quando a aplicação de origem se torna a aplicação de destino, esses ganchos de execução deixam de ser válidos e são excluídos para impedir sua execução.

Para saber mais sobre ganchos de execução, consulte"Gerenciar os ganchos de execução do Trident Protect" .

Estabeleça uma relação de replicação.

A configuração de uma relação de replicação envolve o seguinte:

- Escolher a frequência com que o Trident Protect deve tirar um snapshot do aplicativo (que inclui os recursos do Kubernetes do aplicativo, bem como os snapshots de volume para cada um dos volumes do aplicativo).
- Escolher o agendamento de replicação (inclui recursos do Kubernetes, bem como dados de volume persistentes)
- Definir a hora em que a foto será tirada.

Passos

1. No cluster de origem, crie um AppVault para o aplicativo de origem. Dependendo do seu provedor de armazenamento, modifique um exemplo em"Recursos personalizados do AppVault" para se adequar ao seu ambiente:

Crie um AppVault usando um CR

a. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-appvault-primary-source.yaml`).

b. Configure os seguintes atributos:

- **metadata.name**: (*Obrigatório*) O nome do recurso personalizado do AppVault. Anote o nome escolhido, pois outros arquivos CR necessários para uma relação de replicação farão referência a esse valor.
- **spec.providerConfig**: (*Obrigatório*) Armazena a configuração necessária para acessar o AppVault usando o provedor especificado. Escolha um nome para o bucket e quaisquer outros detalhes necessários para o seu provedor. Anote os valores que você escolher, pois outros arquivos CR necessários para uma relação de replicação fazem referência a esses valores. Consulte"Recursos personalizados do AppVault" Para exemplos de solicitações de configuração (CRs) do AppVault com outros provedores.
- **spec.providerCredentials**: (*Obrigatório*) Armazena referências a quaisquer credenciais necessárias para acessar o AppVault usando o provedor especificado.
  - **spec.providerCredentials.valueFromSecret**: (*Obrigatório*) Indica que o valor da credencial deve vir de um segredo.
    - **chave**: (*Obrigatório*) A chave válida do segredo a ser selecionado.
    - **nome**: (*Obrigatório*) Nome do segredo que contém o valor deste campo. Devem estar no mesmo espaço de nomes.
  - **spec.providerCredentials.secretAccessKey**: (*Obrigatório*) A chave de acesso usada para acessar o provedor. O **nome** deve corresponder a **spec.providerCredentials.valueFromSecret.name**.
- **spec.providerType**: (*Obrigatório*) Determina o que fornece o backup; por exemplo, NetApp ONTAP S3, S3 genérico, Google Cloud ou Microsoft Azure. Valores possíveis:
  - aws
  - azul
  - gcp
  - genérico-s3
  - ontap-s3
  - storagegrid-s3

c. Depois de preencher o `trident-protect-appvault-primary-source.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n
trident-protect
```

Crie um AppVault usando a CLI

a. Crie o AppVault, substituindo os valores entre colchetes pelas informações do seu ambiente:

```
tridentctl-protect create vault Azure <vault-name> --account
<account-name> --bucket <bucket-name> --secret <secret-name> -n
trident-protect
```

2. No cluster de origem, crie a solicitação de configuração (CR) do aplicativo de origem:

Crie o aplicativo de origem usando um CR (Código de Referência).

a. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-app-source.yaml`).

b. Configure os seguintes atributos:

- **metadata.name**: (*Obrigatório*) O nome do recurso personalizado do aplicativo. Anote o nome escolhido, pois outros arquivos CR necessários para uma relação de replicação farão referência a esse valor.
- **spec.includedNamespaces**: (*Obrigatório*) Uma matriz de namespaces e rótulos associados. Utilize nomes de namespaces e, opcionalmente, restrinja o escopo dos namespaces com rótulos para especificar recursos que existem nos namespaces listados aqui. O namespace da aplicação deve fazer parte deste array.

  Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
      labelSelector: {}
```

c. Depois de preencher o `trident-protect-app-source.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-
namespace
```

Crie o aplicativo de origem usando a CLI (linha de comando)

a. Crie o aplicativo de origem. Por exemplo:

```
tridentctl-protect create app <my-app-name> --namespaces
<namespaces-to-be-included> -n <my-app-namespace>
```

3. Opcionalmente, no cluster de origem, tire um snapshot do aplicativo de origem. Este instantâneo é usado como base para a aplicação no cluster de destino. Se você pular esta etapa, precisará aguardar a próxima captura de instantâneo agendada para obter um instantâneo recente. Para criar um instantâneo sob demanda, consulte "Criar um instantâneo sob demanda".

4. No cluster de origem, crie a solicitação de configuração (CR) para o agendamento de replicação:

Além da programação fornecida abaixo, recomenda-se criar uma programação de snapshots diários separada, com um período de retenção de 7 dias, para manter um snapshot comum entre os clusters ONTAP interligados. Isso garante que os instantâneos fiquem disponíveis por até 7 dias, mas o período de retenção pode ser personalizado de acordo com as necessidades do usuário.

Caso ocorra uma falha, o sistema pode usar esses snapshots por até 7 dias para operações de reversão. Essa abordagem torna o processo inverso mais rápido e eficiente, pois apenas as alterações feitas desde o último instantâneo serão transferidas, e não todos os dados.

Se um cronograma existente para o aplicativo já atender aos requisitos de retenção desejados, não serão necessários cronogramas adicionais.

Crie o cronograma de replicação usando uma CR (Código de Referência).

a. Crie um agendamento de replicação para o aplicativo de origem:

i. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-schedule.yaml`).

ii. Configure os seguintes atributos:

- **metadata.name**: (*Obrigatório*) O nome do recurso personalizado de agendamento.
- **spec.appVaultRef**: (*Obrigatório*) Este valor deve corresponder ao campo metadata.name do AppVault para o aplicativo de origem.
- **spec.applicationRef**: (*Obrigatório*) Este valor deve corresponder ao campo metadata.name do CR da aplicação de origem.
- **spec.backupRetention**: (*Obrigatório*) Este campo é obrigatório e o valor deve ser definido como 0.
- **spec.enabled**: Deve ser definido como verdadeiro.
- **spec.granularity**: Deve ser definido como `Custom` .
- **spec.recurrenceRule**: Define uma data de início em UTC e um intervalo de recorrência.
- **spec.snapshotRetention**: Deve ser definido como 2.

Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  name: appmirror-schedule
  namespace: my-app-namespace
spec:
  appVaultRef: my-appvault-name
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: Custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

i. Depois de preencher o `trident-protect-schedule.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-schedule.yaml -n my-app-namespace
```

Crie o agendamento de replicação usando a CLI

a. Crie o cronograma de replicação, substituindo os valores entre colchetes pelas informações do seu ambiente:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule <rule> --snapshot-retention
<snapshot_retention_count> -n <my_app_namespace>
```

Exemplo:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule  "DTSTART:20220101T000200Z
\nRRULE:FREQ=MINUTELY;INTERVAL=5" --snapshot-retention 2 -n
<my_app_namespace>
```

5. No cluster de destino, crie um AppVault CR de aplicativo de origem idêntico ao AppVault CR que você aplicou no cluster de origem e nomeie-o (por exemplo, `trident-protect-appvault-primary-destination.yaml`).
6. Aplicar o CR:

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n
trident-protect
```

7. Crie uma solicitação de configuração (CR) do AppVault de destino para o aplicativo de destino no cluster de destino. Dependendo do seu provedor de armazenamento, modifique um exemplo em"Recursos personalizados do AppVault" para se adequar ao seu ambiente:
   a. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-appvault-secondary-destination.yaml`).
   b. Configure os seguintes atributos:
      - **metadata.name**: (*Obrigatório*) O nome do recurso personalizado do AppVault. Anote o nome escolhido, pois outros arquivos CR necessários para uma relação de replicação farão referência a esse valor.
      - **spec.providerConfig**: (*Obrigatório*) Armazena a configuração necessária para acessar o AppVault usando o provedor especificado. Escolha um `bucketName` e quaisquer outros detalhes necessários para o seu fornecedor. Anote os valores que você escolher, pois outros arquivos CR necessários para uma relação de replicação fazem referência a esses valores. Consulte"Recursos personalizados do AppVault" Para exemplos de solicitações de configuração (CRs) do AppVault com outros provedores.
      - **spec.providerCredentials**: (*Obrigatório*) Armazena referências a quaisquer credenciais necessárias para acessar o AppVault usando o provedor especificado.

        - **spec.providerCredentials.valueFromSecret**: (*Obrigatório*) Indica que o valor da credencial deve vir de um segredo.
            - **chave**: (*Obrigatório*) A chave válida do segredo a ser selecionado.
            - **nome**: (*Obrigatório*) Nome do segredo que contém o valor deste campo. Devem estar no mesmo espaço de nomes.
        - **spec.providerCredentials.secretAccessKey**: (*Obrigatório*) A chave de acesso usada para acessar o provedor. O **nome** deve corresponder a **spec.providerCredentials.valueFromSecret.name**.
    - **spec.providerType**: (*Obrigatório*) Determina o que fornece o backup; por exemplo, NetApp ONTAP S3, S3 genérico, Google Cloud ou Microsoft Azure. Valores possíveis:
        - aws
        - azul
        - gcp
        - genérico-s3
        - ontap-s3
        - storagegrid-s3

c. Depois de preencher o `trident-protect-appvault-secondary-destination.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n trident-protect
```

8. No cluster de destino, crie um arquivo CR de relacionamento AppMirror:

Crie um relacionamento AppMirror usando um CR

a. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-relationship.yaml`).

b. Configure os seguintes atributos:

- **metadata.name:** (Obrigatório) O nome do recurso personalizado AppMirrorRelationship.
- **spec.destinationAppVaultRef**: (*Obrigatório*) Este valor deve corresponder ao nome do AppVault para o aplicativo de destino no cluster de destino.
- **spec.namespaceMapping**: (*Obrigatório*) Os namespaces de destino e de origem devem corresponder ao namespace do aplicativo definido no respectivo CR do aplicativo.
- **spec.sourceAppVaultRef**: (*Obrigatório*) Este valor deve corresponder ao nome do AppVault para o aplicativo de origem.
- **spec.sourceApplicationName**: (*Obrigatório*) Este valor deve corresponder ao nome do aplicativo de origem que você definiu no CR do aplicativo de origem.
- **spec.sourceApplicationUID**: (Obrigatório) Este valor deve corresponder ao UID do aplicativo de origem que você definiu no CR do aplicativo de origem.
- **spec.storageClassName**: (Opcional) Escolha o nome de uma classe de armazenamento válida no cluster. A classe de armazenamento deve ser vinculada a uma VM de armazenamento ONTAP que esteja emparelhada com o ambiente de origem. Caso a classe de armazenamento não seja especificada, a classe de armazenamento padrão do cluster será utilizada por padrão.
- **spec.recurrenceRule**: Define uma data de início em UTC e um intervalo de recorrência.

  Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppMirrorRelationship
metadata:
  name: amr-16061e80-1b05-4e80-9d26-d326dc1953d8
  namespace: my-app-namespace
spec:
  desiredState: Established
  destinationAppVaultRef: generic-s3-trident-protect-dst-bucket-
8fe0b902-f369-4317-93d1-ad7f2edc02b5
  namespaceMapping:
    - destination: my-app-namespace
      source: my-app-namespace
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  sourceAppVaultRef: generic-s3-trident-protect-src-bucket-
b643cc50-0429-4ad5-971f-ac4a83621922
  sourceApplicationName: my-app-name
  sourceApplicationUID: 7498d32c-328e-4ddd-9029-122540866aeb
  storageClassName: sc-vsim-2
```

c. Depois de preencher o `trident-protect-relationship.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-
namespace
```

Crie um relacionamento AppMirror usando a CLI

a. Crie e aplique o objeto AppMirrorRelationship, substituindo os valores entre colchetes pelas informações do seu ambiente:

```
tridentctl-protect create appmirrorrelationship
<name_of_appmirorrelationship> --destination-app-vault
<my_vault_name> --source-app-vault <my_vault_name> --recurrence
-rule <rule> --namespace-mapping <ns_mapping> --source-app-id
<source_app_UID> --source-app <my_source_app_name> --storage
-class <storage_class_name> -n <application_namespace>
```

Exemplo:

```
tridentctl-protect create appmirrorrelationship my-amr
--destination-app-vault appvault2 --source-app-vault appvault1
--recurrence-rule
"DTSTART:20220101T000200Z\nRRULE:FREQ=MINUTELY;INTERVAL=5"
--source-app my-app --namespace-mapping "my-source-ns1:my-dest-
ns1,my-source-ns2:my-dest-ns2" --source-app-id 373f24c1-5769-
404c-93c3-5538af6ccc36 --storage-class my-storage-class -n my-
dest-ns1
```

9. (Opcional) No cluster de destino, verifique o estado e o status da relação de replicação:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Failover para o cluster de destino

Com o Trident Protect, você pode realizar o failover de aplicativos replicados para um cluster de destino. Este procedimento interrompe a relação de replicação e coloca o aplicativo online no cluster de destino. O Trident Protect não interrompe o aplicativo no cluster de origem se ele estiver operacional.

Passos

1. No cluster de destino, edite o arquivo CR AppMirrorRelationship (por exemplo, `trident-protect-relationship.yaml`) e altere o valor de **spec.desiredState** para `Promoted` .
2. Salve o arquivo CR.
3. Aplicar o CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (Opcional) Crie quaisquer cronogramas de proteção que você precise no aplicativo em caso de falha.
5. (Opcional) Verifique o estado e o status da relação de replicação:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Ressincronizar uma relação de replicação que falhou

A operação de ressincronização restabelece a relação de replicação. Após a execução de uma operação de ressincronização, o aplicativo de origem original torna-se o aplicativo em execução e quaisquer alterações feitas no aplicativo em execução no cluster de destino são descartadas.

O processo interrompe o aplicativo no cluster de destino antes de restabelecer a replicação.

Quaisquer dados gravados no aplicativo de destino durante a transição de falha serão perdidos.

Passos

1. Opcional: No cluster de origem, crie um snapshot do aplicativo de origem. Isso garante que as alterações mais recentes do cluster de origem sejam capturadas.
2. No cluster de destino, edite o arquivo CR AppMirrorRelationship (por exemplo, `trident-protect-relationship.yaml`) e altere o valor de spec.desiredState para `Established` .
3. Salve o arquivo CR.
4. Aplicar o CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Se você criou algum agendamento de proteção no cluster de destino para proteger o aplicativo em failover, remova-o. Qualquer agendamento pendente causa falhas no snapshot de volume.

Reverter a resincronização de uma relação de replicação que falhou

Ao reverter a resincronização de uma relação de replicação que falhou, o aplicativo de destino torna-se o aplicativo de origem e a origem torna-se o destino. As alterações feitas no aplicativo de destino durante o failover são mantidas.

Passos

1. No cluster de destino original, exclua o CR AppMirrorRelationship. Isso faz com que o destino se torne a origem. Se ainda houver alguma programação de proteção no novo cluster de destino, remova-a.
2. Estabeleça uma relação de replicação aplicando os arquivos CR que você usou originalmente para configurar a relação aos clusters opostos.
3. Certifique-se de que o novo destino (cluster de origem original) esteja configurado com ambas as solicitações de configuração (CRs) do AppVault.
4. Configure uma relação de replicação no cluster oposto, definindo os valores para a direção inversa.

Direção de replicação inversa do aplicativo

Ao inverter a direção da replicação, o Trident Protect move a aplicação para o backend de armazenamento de destino, enquanto continua a replicar de volta para o backend de armazenamento de origem original. O Trident Protect interrompe o aplicativo de origem e replica os dados para o destino antes de realizar o failover para o aplicativo de destino.

Nessa situação, você está trocando a origem e o destino.

Passos

1. No cluster de origem, crie um snapshot de desligamento:

Crie um instantâneo de desligamento usando uma CR (Código de Requisição).

a. Desative os agendamentos da política de proteção para o aplicativo de origem.

b. Crie um arquivo CR de ShutdownSnapshot:

   i. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele (por exemplo, `trident-protect-shutdownsnapshot.yaml`).

   ii. Configure os seguintes atributos:

   - **metadata.name**: (*Obrigatório*) O nome do recurso personalizado.
   - **spec.AppVaultRef**: (*Obrigatório*) Este valor deve corresponder ao campo metadata.name do AppVault para o aplicativo de origem.
   - **spec.ApplicationRef**: (*Obrigatório*) Este valor deve corresponder ao campo metadata.name do arquivo CR do aplicativo de origem.

     Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ShutdownSnapshot
metadata:
  name: replication-shutdown-snapshot-afc4c564-e700-4b72-86c3-
c08a5dbe844e
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
```

c. Depois de preencher o `trident-protect-shutdownsnapshot.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-
namespace
```

Crie um instantâneo de desligamento usando a CLI

a. Crie o instantâneo de desligamento, substituindo os valores entre colchetes pelas informações do seu ambiente. Por exemplo:

```
tridentctl-protect create shutdownsnapshot <my_shutdown_snapshot>
--appvault <my_vault> --app <app_to_snapshot> -n
<application_namespace>
```

2. No cluster de origem, após a conclusão do snapshot de desligamento, obtenha o status do snapshot de desligamento:

```
kubectl get shutdownsnapshot -n my-app-namespace
<shutdown_snapshot_name> -o yaml
```

3. No cluster de origem, encontre o valor de **shutdownsnapshot.status.appArchivePath** usando o seguinte comando e registre a última parte do caminho do arquivo (também chamada de nome base; isso será tudo o que vem depois da última barra):

```
k get shutdownsnapshot -n my-app-namespace <shutdown_snapshot_name> -o
jsonpath='{.status.appArchivePath}'
```

4. Realize um failover do novo cluster de destino para o novo cluster de origem, com a seguinte alteração:

> Na etapa 2 do procedimento de failover, inclua o `spec.promotedSnapshot` No arquivo CR AppMirrorRelationship, defina o valor do campo para o nome base que você registrou na etapa 3 acima.

5. Execute as etapas de resincronização reversa emReverter a resincronização de uma relação de replicação que falhou .
6. Ative os agendamentos de proteção no novo cluster de origem.

Resultado

As seguintes ações ocorrem devido à replicação reversa:

- É feita uma captura instantânea dos recursos Kubernetes do aplicativo de origem original.
- Os pods do aplicativo de origem original são interrompidos corretamente excluindo os recursos do Kubernetes do aplicativo (mantendo os PVCs e PVs).
- Após o desligamento dos pods, são criados snapshots dos volumes do aplicativo e replicados.
- Os relacionamentos SnapMirror foram desfeitos, tornando os volumes de destino prontos para leitura/gravação.
- Os recursos do Kubernetes do aplicativo são restaurados a partir do snapshot anterior ao desligamento, usando os dados de volume replicados após o desligamento do aplicativo de origem original.
- A replicação é restabelecida na direção inversa.

Restaurar aplicativos para o cluster de origem original.

Utilizando o Trident Protect, você pode realizar o "failback" após uma operação de failover seguindo a seguinte sequência de operações. Nesse fluxo de trabalho para restaurar a direção de replicação original, o Trident Protect replica (ressincroniza) quaisquer alterações do aplicativo de volta para o aplicativo de origem original antes de inverter a direção da replicação.

Esse processo começa com um relacionamento que concluiu uma transição de failover para um destino e envolve as seguintes etapas:

- Comece com um estado de failover.
- Reverter a resincronização da relação de replicação.

> Não execute uma operação de ressincronização normal, pois isso descartará os dados gravados no cluster de destino durante o procedimento de failover.

- Inverta o sentido da replicação.

Passos

1. Realize oReverter a resincronização de uma relação de replicação que falhou passos.
2. Realize oDireção de replicação inversa do aplicativo passos.

Excluir uma relação de replicação

Você pode excluir uma relação de replicação a qualquer momento. Ao excluir a relação de replicação do aplicativo, o resultado são dois aplicativos separados, sem nenhuma relação entre eles.

Passos

1. No cluster de destino atual, exclua o CR AppMirrorRelationship:

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

Migre aplicativos usando o Trident Protect.

Você pode migrar seus aplicativos entre clusters ou para diferentes classes de armazenamento restaurando dados de backup.

> Ao migrar um aplicativo, todos os ganchos de execução configurados para ele são migrados juntamente com o aplicativo. Se houver um gancho de execução pós-restauração, ele será executado automaticamente como parte da operação de restauração.

Operações de backup e restauração

Para executar operações de backup e restauração nos seguintes cenários, você pode automatizar tarefas específicas de backup e restauração.

Clonar para o mesmo cluster

Para clonar um aplicativo para o mesmo cluster, crie um snapshot ou backup e restaure os dados para o mesmo cluster.

Passos

1. Faça um dos seguintes:
   a. "Criar um instantâneo".
   b. "Criar um backup".
2. No mesmo cluster, faça um dos seguintes procedimentos, dependendo se você criou um snapshot ou um backup:

a. "Restaure seus dados a partir do snapshot.".

b. "Restaure seus dados a partir do backup.".

Clonar para um cluster diferente

Para clonar um aplicativo para um cluster diferente (realizar uma clonagem entre clusters), crie um backup no cluster de origem e, em seguida, restaure o backup em um cluster diferente. Certifique-se de que o Trident Protect esteja instalado no cluster de destino.

Você pode replicar um aplicativo entre clusters diferentes usando"Replicação SnapMirror" .

Passos

1. "Criar um backup".
2. Certifique-se de que o AppVault CR para o bucket de armazenamento de objetos que contém o backup foi configurado no cluster de destino.
3. No cluster de destino,"Restaure seus dados a partir do backup." .

Migrar aplicações de uma classe de armazenamento para outra classe de armazenamento.

Você pode migrar aplicativos de uma classe de armazenamento para uma classe de armazenamento diferente restaurando um backup para a classe de armazenamento de destino.

Por exemplo (excluindo os segredos do CR de restauração):

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: "${snapshotRestoreCRName}"
spec:
  appArchivePath: "${snapshotArchivePath}"
  appVaultRef: "${appVaultCRName}"
  namespaceMapping:
    - destination: "${destinationNamespace}"
      source: "${sourceNamespace}"
  storageClassMapping:
    - destination: "${destinationStorageClass}"
      source: "${sourceStorageClass}"
  resourceFilter:
    resourceMatchers:
      kind: Secret
      version: v1
    resourceSelectionCriteria: exclude
```

Restaurar o snapshot usando um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-snapshot-restore-cr.yaml` .
2. No arquivo que você criou, configure os seguintes atributos:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do snapshot está armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do snapshot está armazenado.
   - **spec.namespaceMapping**: O mapeamento do namespace de origem da operação de restauração para o namespace de destino. Substituir `my-source-namespace` e `my-destination-namespace` com informações do seu ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: trident-protect
     spec:
       appArchivePath: my-snapshot-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. Opcionalmente, se precisar selecionar apenas determinados recursos do aplicativo para restaurar, adicione filtros que incluam ou excluam recursos marcados com rótulos específicos:
   - **resourceFilter.resourceSelectionCriteria**: (Obrigatório para filtragem) Use `include or exclude` Para incluir ou excluir um recurso definido em resourceMatchers. Adicione os seguintes parâmetros resourceMatchers para definir os recursos a serem incluídos ou excluídos:
     - **resourceFilter.resourceMatchers**: Um array de objetos resourceMatcher. Se você definir vários elementos nessa matriz, eles corresponderão como uma operação OR, e os campos dentro de cada elemento (grupo, tipo, versão) corresponderão como uma operação AND.
       - **resourceMatchers[].group**: (Opcional) Grupo do recurso a ser filtrado.
       - **resourceMatchers[].kind**: (*Opcional*) Tipo do recurso a ser filtrado.
       - **resourceMatchers[].version**: (Opcional) Versão do recurso a ser filtrado.

- **resourceMatchers[].names**: (Opcional) Nomes no campo metadata.name do Kubernetes do recurso a ser filtrado.
- **resourceMatchers[].namespaces**: (Opcional) Namespaces no campo metadata.name do recurso do Kubernetes a ser filtrado.
- **resourceMatchers[].labelSelectors**: (Opcional) String do seletor de rótulo no campo metadata.name do recurso do Kubernetes, conforme definido em "Documentação do Kubernetes" . Por exemplo: `"trident.netapp.io/os=linux"` .

  Por exemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Depois de preencher o `trident-protect-snapshot-restore-cr.yaml` Arquivo com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Restaure o snapshot usando a CLI

Passos

1. Restaure o snapshot para um namespace diferente, substituindo os valores entre colchetes pelas informações do seu ambiente.

   - O `snapshot` O argumento usa um namespace e um nome de snapshot no formato `<namespace>/<name>` .
   - O `namespace-mapping` O argumento usa namespaces separados por dois pontos para mapear namespaces de origem para os namespaces de destino corretos no formato `source1:dest1,source2:dest2` .

     Por exemplo:

```
tridentctl-protect create snapshotrestore <my_restore_name>
--snapshot <namespace/snapshot_to_restore> --namespace-mapping
<source_to_destination_namespace_mapping>
```

Gerenciar os ganchos de execução do Trident Protect

Um gancho de execução é uma ação personalizada que você pode configurar para ser executada em conjunto com uma operação de proteção de dados de um aplicativo gerenciado. Por exemplo, se você tiver um aplicativo de banco de dados, poderá usar um gancho de execução para pausar todas as transações do banco de dados antes de um instantâneo e retomar as transações após a conclusão do instantâneo. Isso garante instantâneos consistentes com o aplicativo.

Tipos de ganchos de execução

O Trident Protect suporta os seguintes tipos de ganchos de execução, com base em quando podem ser executados:

- Pré-instantâneo
- Pós-instantâneo
- Pré-backup
- Pós-backup
- Pós-restauração
- Pós-failover

Ordem de execução

Quando uma operação de proteção de dados é executada, os eventos de gancho de execução ocorrem na seguinte ordem:

1. Todos os ganchos de execução de pré-operação personalizados aplicáveis são executados nos contêineres apropriados. Você pode criar e executar quantos ganchos de pré-operação personalizados precisar, mas a ordem de execução desses ganchos antes da operação não é garantida nem configurável.
2. Congelamentos do sistema de arquivos ocorrem, se aplicável. "Saiba mais sobre como configurar o congelamento do sistema de arquivos com o Trident Protect.".
3. A operação de proteção de dados é realizada.
4. Sistemas de arquivos congelados são descongelados, se aplicável.
5. Todos os ganchos de execução pós-operação personalizados aplicáveis são executados nos contêineres apropriados. Você pode criar e executar quantos ganchos pós-operação personalizados precisar, mas a ordem de execução desses ganchos após a operação não é garantida nem configurável.

Se você criar vários ganchos de execução do mesmo tipo (por exemplo, pré-instantâneo), a ordem de execução desses ganchos não será garantida. Entretanto, a ordem de execução de ganchos de diferentes tipos é garantida. Por exemplo, a seguir está a ordem de execução de uma configuração que possui todos os diferentes tipos de ganchos:

1. Ganchos pré-instantâneos executados
2. Ganchos pós-instantâneos executados
3. Ganchos de pré-backup executados
4. Ganchos pós-backup executados

O exemplo de ordem anterior só se aplica quando você executa um backup que não utiliza um snapshot existente.

Você deve sempre testar seus scripts de gancho de execução antes de habilitá-los em um ambiente de produção. Você pode usar o comando 'kubectl exec' para testar os scripts convenientemente. Depois de habilitar os ganchos de execução em um ambiente de produção, teste os snapshots e backups resultantes para garantir que sejam consistentes. Você pode fazer isso clonando o aplicativo em um namespace temporário, restaurando o snapshot ou backup e, em seguida, testando o aplicativo.

Se um gancho de execução pré-snapshot adicionar, alterar ou remover recursos do Kubernetes, essas alterações serão incluídas no snapshot ou backup e em qualquer operação de restauração subsequente.

Notas importantes sobre ganchos de execução personalizados

Considere o seguinte ao planejar ganchos de execução para seus aplicativos.

- Um gancho de execução deve usar um script para executar ações. Muitos ganchos de execução podem referenciar o mesmo script.
- O Trident Protect exige que os scripts usados pelos ganchos de execução sejam escritos no formato de scripts shell executáveis.
- O tamanho do script é limitado a 96 KB.
- O Trident Protect utiliza as configurações de ganchos de execução e quaisquer critérios correspondentes para determinar quais ganchos são aplicáveis a uma operação de snapshot, backup ou restauração.

Como os ganchos de execução geralmente reduzem ou desabilitam completamente a funcionalidade do aplicativo em que estão sendo executados, você deve sempre tentar minimizar o tempo que seus ganchos de execução personalizados levam para serem executados. Se você iniciar uma operação de backup ou snapshot com ganchos de execução associados, mas depois cancelá-la, os ganchos ainda poderão ser executados se a operação de backup ou snapshot já tiver começado. Isso significa que a lógica usada em um gancho de execução pós-backup não pode assumir que o backup foi concluído.

Filtros de gancho de execução

Ao adicionar ou editar um gancho de execução para um aplicativo, você pode adicionar filtros ao gancho de execução para gerenciar quais contêineres o gancho corresponderá. Os filtros são úteis para aplicativos que usam a mesma imagem de contêiner em todos os contêineres, mas podem usar cada imagem para uma finalidade diferente (como o Elasticsearch). Os filtros permitem que você crie cenários em que os ganchos de execução são executados em alguns contêineres idênticos, mas não necessariamente em todos. Se você criar vários filtros para um único gancho de execução, eles serão combinados com um operador lógico AND. Você pode ter até 10 filtros ativos por gancho de execução.

Cada filtro que você adiciona a um gancho de execução usa uma expressão regular para corresponder aos contêineres no seu cluster. Quando um gancho corresponde a um contêiner, o gancho executará seu script associado naquele contêiner. Expressões regulares para filtros usam a sintaxe Regular Expression 2 (RE2), que não oferece suporte à criação de um filtro que exclua contêineres da lista de correspondências. Para obter informações sobre a sintaxe que o Trident Protect suporta para expressões regulares em filtros de gancho de execução, consulte "Suporte à sintaxe de Expressão Regular 2 (RE2)" .

> Se você adicionar um filtro de namespace a um gancho de execução executado após uma operação de restauração ou clonagem e a origem e o destino da restauração ou clonagem estiverem em namespaces diferentes, o filtro de namespace será aplicado somente ao namespace de destino.

Exemplos de ganchos de execução

Visite o "Projeto NetApp Verda GitHub" para baixar ganchos de execução reais para aplicativos populares, como Apache Cassandra e Elasticsearch. Você também pode ver exemplos e obter ideias para estruturar seus próprios ganchos de execução personalizados.

Crie um gancho de execução

Você pode criar um gancho de execução personalizado para um aplicativo usando . Você precisa ter permissões de Proprietário, Administrador ou Membro para criar ganchos de execução.

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-hook.yaml` .
2. Configure os seguintes atributos para corresponder ao seu ambiente Trident Protect e à configuração do cluster:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.applicationRef**: (*Obrigatório*) O nome do aplicativo Kubernetes para o qual o gancho de execução será executado.
   - **spec.stage**: (*Obrigatório*) Uma string que indica em qual etapa da ação o gancho de execução deve ser executado. Valores possíveis:
     - Pré
     - Publicar
   - **spec.action**: (*Obrigatório*) Uma string que indica qual ação o gancho de execução tomará, assumindo que todos os filtros de gancho de execução especificados sejam correspondidos. Valores possíveis:
     - Instantâneo
     - Backup
     - Restaurar
     - Failover
   - **spec.enabled**: (Opcional) Indica se este gancho de execução está habilitado ou desabilitado. Caso não seja especificado, o valor padrão é verdadeiro.
   - **spec.hookSource**: (*Obrigatório*) Uma string contendo o script de gancho codificado em base64.
   - **spec.timeout**: (Opcional) Um número que define por quanto tempo, em minutos, o gancho de execução pode ser executado. O valor mínimo é de 1 minuto, e o valor padrão é de 25 minutos caso não seja especificado.
   - **spec.arguments**: (Opcional) Uma lista YAML de argumentos que você pode especificar para o gancho de execução.
   - **spec.matchingCriteria**: (*Opcional*) Uma lista opcional de pares de chave-valor de critérios, cada par constituindo um filtro de gancho de execução. Você pode adicionar até 10 filtros por gancho de execução.
   - **spec.matchingCriteria.type**: (*Opcional*) Uma string que identifica o tipo de filtro do gancho de execução. Valores possíveis:
     - Imagem do contêiner
     - Nome do contêiner
     - Nome do Pod
     - PodLabel
     - Nome do Namespace
   - **spec.matchingCriteria.value**: (*Opcional*) Uma string ou expressão regular que identifica o valor do filtro do gancho de execução.

Exemplo de YAML:

```
apiVersion: protect.trident.netapp.io/v1
kind: ExecHook
metadata:
  name: example-hook-cr
  namespace: my-app-namespace
  annotations:
    astra.netapp.io/astra-control-hook-source-id:
/account/test/hookSource/id
spec:
  applicationRef: my-app-name
  stage: Pre
  action: Snapshot
  enabled: true
  hookSource: IyEvYmluL2Jhc2gKZWNobyAiZXhhbXBsZSBzY3JpcHQiCg==
  timeout: 10
  arguments:
    - FirstExampleArg
    - SecondExampleArg
  matchingCriteria:
    - type: containerName
      value: mysql
    - type: containerImage
      value: bitnami/mysql
    - type: podName
      value: mysql
    - type: namespaceName
      value: mysql-a
    - type: podLabel
      value: app.kubernetes.io/component=primary
    - type: podLabel
      value: helm.sh/chart=mysql-10.1.0
    - type: podLabel
      value: deployment-type=production
```

3. Após preencher o arquivo CR com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-hook.yaml
```

Use a CLI

Passos

1. Crie o gancho de execução, substituindo os valores entre colchetes por informações do seu ambiente. Por exemplo:

```
tridentctl-protect create exechook <my_exec_hook_name> --action
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>
--source-file <script-file> -n <application_namespace>
```

Executar manualmente um gancho de execução

Você pode executar manualmente um gancho de execução para fins de teste ou se precisar executá-lo novamente após uma falha. Você precisa ter permissões de Proprietário, Administrador ou Membro para executar ganchos de execução manualmente.

A execução manual de um gancho de execução consiste em duas etapas básicas:

1. Crie um backup de recursos, que coleta recursos e cria um backup deles, determinando onde o gancho será executado.
2. Execute o gancho de execução no backup.

Passo 1: Crie um backup de recursos

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-resource-backup.yaml` .
2. Configure os seguintes atributos para corresponder ao seu ambiente Trident Protect e à configuração do cluster:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.applicationRef**: (*Obrigatório*) O nome do aplicativo Kubernetes para o qual o backup do recurso será criado.
   - **spec.appVaultRef**: (*Obrigatório*) O nome do AppVault onde o conteúdo do backup está armazenado.
   - **spec.appArchivePath**: O caminho dentro do AppVault onde o conteúdo do backup é armazenado. Você pode usar o seguinte comando para encontrar esse caminho:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

     Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ResourceBackup
metadata:
  name: example-resource-backup
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
```

3. Após preencher o arquivo CR com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-resource-backup.yaml
```

Use a CLI

Passos

1. Crie o backup, substituindo os valores entre colchetes pelas informações do seu ambiente. Por exemplo:

```
tridentctl protect create resourcebackup <my_backup_name> --app
<my_app_name> --appvault <my_appvault_name> -n
<my_app_namespace> --app-archive-path <app_archive_path>
```

2. Veja o status do backup. Você pode usar este comando de exemplo repetidamente até que a operação seja concluída:

```
tridentctl protect get resourcebackup -n <my_app_namespace>
<my_backup_name>
```

3. Verifique se o backup foi bem-sucedido:

```
kubectl describe resourcebackup <my_backup_name>
```

Etapa 2: Execute o gancho de execução

Use um CR

Passos

1. Crie o arquivo de recurso personalizado (CR) e dê um nome a ele. `trident-protect-hook-run.yaml` .
2. Configure os seguintes atributos para corresponder ao seu ambiente Trident Protect e à configuração do cluster:
   - **metadata.name**: (*Obrigatório*) O nome deste recurso personalizado; escolha um nome único e adequado ao seu ambiente.
   - **spec.applicationRef**: (*Obrigatório*) Certifique-se de que este valor corresponda ao nome do aplicativo do ResourceBackup CR que você criou na etapa 1.
   - **spec.appVaultRef**: (*Obrigatório*) Certifique-se de que este valor corresponda ao appVaultRef do ResourceBackup CR que você criou na etapa 1.
   - **spec.appArchivePath**: Certifique-se de que esse valor corresponda ao appArchivePath do ResourceBackup CR que você criou na etapa 1.

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.action**: (*Obrigatório*) Uma string que indica qual ação o gancho de execução tomará, assumindo que todos os filtros de gancho de execução especificados sejam correspondidos. Valores possíveis:
     - Instantâneo
     - Backup
     - Restaurar
     - Failover
   - **spec.stage**: (*Obrigatório*) Uma string que indica em qual etapa da ação o gancho de execução deve ser executado. Esta sequência de ganchos não utilizará ganchos em nenhuma outra etapa. Valores possíveis:
     - Pré
     - Publicar

     Exemplo de YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ExecHooksRun
metadata:
  name: example-hook-run
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
  stage: Post
  action: Failover
```

3. Após preencher o arquivo CR com os valores corretos, aplique o CR:

```
kubectl apply -f trident-protect-hook-run.yaml
```

Use a CLI

Passos

1. Crie a solicitação de execução manual do gancho (hook):

```
tridentctl protect create exechooksrun <my_exec_hook_run_name>
-n <my_app_namespace> --action snapshot --stage <pre_or_post>
--app <my_app_name> --appvault <my_appvault_name> --path
<my_backup_name>
```

2. Verifique o status da execução do gancho (hook). Você pode executar este comando repetidamente até que a operação seja concluída:

```
tridentctl protect get exechooksrun -n <my_app_namespace>
<my_exec_hook_run_name>
```

3. Descreva o objeto exechooksrun para ver os detalhes finais e o status:

```
kubectl -n <my_app_namespace> describe exechooksrun
<my_exec_hook_run_name>
```

Desinstale o Trident Protect.

Você pode precisar remover componentes do Trident Protect se estiver atualizando de uma versão de avaliação para a versão completa do produto.

Para remover o Trident Protect, siga os passos abaixo.

Passos

1. Remova os arquivos CR do Trident Protect:

   Esta etapa não é necessária para a versão 25.06 e posteriores.

   ```
   helm uninstall -n trident-protect trident-protect-crds
   ```

2. Remover Trident Protect:

   ```
   helm uninstall -n trident-protect trident-protect
   ```

3. Remova o namespace Trident Protect:

   ```
   kubectl delete ns trident-protect
   ```

Blogs Trident e Trident Protect

Você pode encontrar ótimos blogs sobre NetApp Trident e Trident Protect aqui:

Blogs da Trident

- 09 de maio de 2025:"Configuração automática do backend Trident para FSx para ONTAP com o complemento Amazon EKS."
- 19 de agosto de 2025:"Aprimorando a consistência de dados: instantâneos de grupos de volume na virtualização OpenShift com Trident"
- 15 de abril de 2025:"NetApp Trident com Google Cloud NetApp Volumes para protocolo SMB"
- 14 de abril de 2025:"Aproveitando o Protocolo Fibre Channel com o Trident 25.02 para Armazenamento Persistente no Kubernetes"
- 14 de abril de 2025:"Desbloqueando o poder dos sistemas NetApp ASA r2 para armazenamento em blocos do Kubernetes"
- 31 de março de 2025:"Simplificando a instalação do Trident no Red Hat OpenShift com o novo operador certificado"
- 27 de março de 2025:"Provisionamento do Trident para PMEs com o Google Cloud NetApp Volumes"
- 05 de março de 2025:"Desbloqueie a integração perfeita de armazenamento iSCSI: um guia para FSxN em clusters ROSA para AWS."
- 27 de fevereiro de 2025:"Implantando identidade na nuvem com Trident, GKE e Google Cloud NetApp Volumes"
- 12 de dezembro de 2024:"Apresentando suporte a Fibre Channel no Trident"
- 26 de novembro de 2024:"Trident 25.01: Aprimorando a experiência de armazenamento do Kubernetes com novos recursos e melhorias"
- 11 de novembro de 2024:"NetApp Trident com Google Cloud NetApp Volumes"
- 29 de outubro de 2024:"Amazon FSx for NetApp ONTAP com Red Hat OpenShift Service na AWS (ROSA) usando Trident"
- 29 de outubro de 2024:"Migração ao vivo de VMs com OpenShift Virtualization no ROSA e Amazon FSx for NetApp ONTAP"
- 08 de julho de 2024:"Usando NVMe/TCP para consumir armazenamento ONTAP para seus aplicativos modernos em contêineres no Amazon EKS"
- 1º de julho de 2024:"Armazenamento Kubernetes integrado com Google Cloud NetApp Volumes Flex e Astra Trident"
- 11 de junho de 2024:"ONTAP como armazenamento de backend para o registro de imagem integrado no OpenShift"

Blogs da Trident Protect

- 16 de maio de 2025:"Automatizando o failover do registro para recuperação de desastres com os hooks pós-restauração do Trident Protect."
- 16 de maio de 2025:"Recuperação de desastres de virtualização OpenShift com NetApp Trident Protect"
- 13 de maio de 2025:"Migração de classe de armazenamento com backup e restauração do Trident Protect."

- 9 de maio de 2025:"Redimensione aplicações Kubernetes com os hooks de pós-restauração do Trident Protect."
- 03 de abril de 2025:"Trident Protect Power Up: Replicação do Kubernetes para Proteção e Recuperação de Desastres"
- 13 de março de 2025:"Operações de backup e restauração consistentes em caso de falha para VMs de virtualização do OpenShift"
- 11 de março de 2025:"Estendendo os padrões do GitOps à proteção de dados de aplicativos com o NetApp Trident."
- 03 de março de 2025:"Trident 25.02: Aprimorando a experiência do Red Hat OpenShift com novos e empolgantes recursos."
- 15 de janeiro de 2025:"Apresentamos o Trident Protect, um sistema de controle de acesso baseado em funções."
- 11 de novembro de 2024: "Apresentando o tridentctl protect: a poderosa CLI para Trident Protect."
- 11 de novembro de 2024:"Gerenciamento de dados orientado por Kubernetes: a nova era com o Trident Protect."

Conhecimento e suporte

Perguntas frequentes

Encontre respostas para as perguntas mais frequentes sobre instalação, configuração, atualização e resolução de problemas do Trident.

Perguntas gerais

Com que frequência o Trident é lançado?

A partir da versão 24.02, o Trident passa a ser lançado a cada quatro meses: fevereiro, junho e outubro.

O Trident é compatível com todos os recursos lançados em uma determinada versão do Kubernetes?

O Trident geralmente não oferece suporte a recursos alfa no Kubernetes. O Trident poderá oferecer suporte a recursos beta nas duas versões do Trident que seguem a versão beta do Kubernetes.

O Trident depende de outros produtos da NetApp para funcionar?

O Trident não possui dependências de outros produtos de software da NetApp e funciona como um aplicativo independente. No entanto, você deve ter um dispositivo de armazenamento de backend da NetApp .

Como posso obter detalhes completos da configuração do Trident ?

Use o `tridentctl get` comando para obter mais informações sobre a sua configuração do Trident .

Posso obter métricas sobre como o armazenamento é provisionado pelo Trident?

Sim. Os endpoints do Prometheus podem ser usados para coletar informações sobre a operação do Trident , como o número de backends gerenciados, o número de volumes provisionados, bytes consumidos e assim por diante. Você também pode usar"Cloud Insights" para monitoramento e análise.

A experiência do usuário muda ao usar o Trident como um provisionador CSI?

Não. Não haverá alterações em termos de experiência do usuário e funcionalidades. O nome do provisionador utilizado é `csi.trident.netapp.io` . Este método de instalação do Trident é recomendado se você deseja usar todos os novos recursos oferecidos pelas versões atuais e futuras.

Instale e utilize o Trident em um cluster Kubernetes.

O Trident suporta instalação offline a partir de um registro privado?

Sim, o Trident pode ser instalado offline. Consulte"Saiba mais sobre a instalação do Trident." .

Posso instalar o Trident remotamente?

Sim. O Trident 18.10 e versões posteriores suportam a capacidade de instalação remota a partir de qualquer máquina que possua `kubectl` Acesso ao cluster. Depois `kubectl` O acesso é verificado (por exemplo, iniciar um `kubectl get nodes` comando da máquina remota para verificar), siga as instruções de instalação.

Posso configurar alta disponibilidade com o Trident?

O Trident é instalado como um Deployment do Kubernetes (ReplicaSet) com uma única instância, portanto, possui alta disponibilidade integrada. Você não deve aumentar o número de réplicas no deployment. Se o nó onde o Trident está instalado for perdido ou o pod ficar inacessível por qualquer outro motivo, o Kubernetes reimplantará automaticamente o pod em um nó íntegro do cluster. O Trident é apenas um sistema de controle de voo, portanto, os pods atualmente instalados não serão afetados se o Trident for reimplantado.

O Trident precisa de acesso ao namespace kube-system?

O Trident lê dados do servidor da API do Kubernetes para determinar quando os aplicativos solicitam novos PVCs, portanto, ele precisa de acesso ao kube-system.

Quais são as funções e privilégios utilizados pela Trident?

O instalador do Trident cria um Kubernetes ClusterRole, que tem acesso específico aos recursos PersistentVolume, PersistentVolumeClaim, StorageClass e Secret do cluster Kubernetes. Consulte"Personalize a instalação do tridentctl" .

Posso gerar localmente os mesmos arquivos de manifesto que o Trident usa para a instalação?

Se necessário, você pode gerar e modificar localmente os arquivos de manifesto exatos que o Trident usa para a instalação. Consulte"Personalize a instalação do tridentctl" .

Posso compartilhar a mesma SVM de backend ONTAP para duas instâncias Trident separadas em dois clusters Kubernetes distintos?

Embora não seja recomendável, você pode usar o mesmo SVM de backend para duas instâncias do Trident . Especifique um nome de volume exclusivo para cada instância durante a instalação e/ou especifique um valor exclusivo. `StoragePrefix` parâmetro no `setup/backend.json` arquivo. Isso garante que o mesmo FlexVol volume não seja usado para ambas as instâncias.

É possível instalar o Trident no ContainerLinux (anteriormente CoreOS)?

O Trident é simplesmente um pod do Kubernetes e pode ser instalado em qualquer lugar onde o Kubernetes esteja em execução.

Posso usar o Trident com o NetApp Cloud Volumes ONTAP?

Sim, o Trident é compatível com AWS, Google Cloud e Azure.

O Trident funciona com o Cloud Volumes Services?

Sim, o Trident é compatível com o serviço Azure NetApp Files no Azure, bem como com o Cloud Volumes Service no GCP.

Solução de problemas e suporte

A NetApp oferece suporte ao Trident?

Embora o Trident seja de código aberto e fornecido gratuitamente, a NetApp oferece suporte completo, desde que seu sistema NetApp seja compatível.

Como faço para abrir um chamado de suporte?

Para abrir um chamado de suporte, faça um dos seguintes procedimentos:

1. Entre em contato com seu Gerente de Suporte e obtenha ajuda para abrir um chamado.
2. Abra um chamado de suporte entrando em contato com "Suporte NetApp" .

Como faço para gerar um pacote de logs de suporte?

Você pode criar um pacote de suporte executando o seguinte comando: `tridentctl logs -a` . Além dos logs capturados no pacote, capture também o log do kubelet para diagnosticar problemas de montagem no lado do Kubernetes. As instruções para obter o log do kubelet variam dependendo de como o Kubernetes foi instalado.

O que devo fazer se precisar solicitar uma nova funcionalidade?

Crie um problema em "Trident Github" e mencione **RFE** no assunto e na descrição do problema.

Onde posso reportar um defeito?

Crie um problema em "Trident Github" . Certifique-se de incluir todas as informações e registros necessários referentes ao problema.

O que acontece se eu tiver uma pergunta rápida sobre o Trident que precise de esclarecimento? Existe alguma comunidade ou fórum?

Se você tiver alguma dúvida, problema ou solicitação, entre em contato conosco através do nosso Trident."Canal do Discord" ou GitHub.

A senha do meu sistema de armazenamento foi alterada e o Trident não funciona mais. Como faço para recuperar o sistema?

Atualize a senha do backend com `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident` . Substituir `myBackend` no exemplo com o nome do seu backend, e `` `/path/to_new_backend.json `` com o caminho para o correto `backend.json` arquivo.

O Trident não consegue encontrar meu nó Kubernetes. Como posso resolver isso?

Existem dois cenários prováveis para o Trident não conseguir encontrar um nó do Kubernetes. Isso pode ser devido a um problema de rede no Kubernetes ou a um problema de DNS. O daemonset do nó Trident que é executado em cada nó do Kubernetes deve ser capaz de se comunicar com o controlador Trident para registrar o nó no Trident. Se ocorreram alterações na rede após a instalação do Trident , você encontrará esse problema apenas com novos nós do Kubernetes adicionados ao cluster.

Se a cápsula Trident for destruída, perderei os dados?

Os dados não serão perdidos se a cápsula Trident for destruída. Os metadados do Trident são armazenados em objetos CRD. Todos os sistemas fotovoltaicos (PVs) provisionados pela Trident funcionarão normalmente.

Trident aprimorado

Posso atualizar diretamente de uma versão antiga para uma versão mais recente (pulando algumas versões)?

A NetApp oferece suporte à atualização do Trident de uma versão principal para a próxima versão principal imediata. Você pode atualizar da versão 18.xx para a 19.xx, da 19.xx para a 20.xx e assim por diante. Você deve testar a atualização em um ambiente de laboratório antes de implementá-la em produção.

É possível fazer o downgrade do Trident para uma versão anterior?

Se você precisar de correções para erros observados após uma atualização, problemas de dependência ou uma atualização malsucedida ou incompleta, você deve"Desinstalar Trident" e reinstale a versão anterior seguindo as instruções específicas para essa versão. Esta é a única maneira recomendada de fazer o downgrade para uma versão anterior.

Gerenciar back-ends e volumes

Preciso definir tanto o Management quanto o DataLIF no mesmo arquivo de definição do backend ONTAP ?

A gestão do LIF é obrigatória. O DataLIF varia:

- ONTAP SAN: Não especifique para iSCSI. Trident usa"Mapa LUN Seletivo ONTAP" Descobrir os LIFs iSCI necessários para estabelecer uma sessão de múltiplos caminhos. Um aviso é gerado se `dataLIF` está explicitamente definido. Consulte "Opções e exemplos de configuração do ONTAP SAN" para mais detalhes.
- ONTAP NAS: A NetApp recomenda especificar `dataLIF` . Caso não sejam fornecidos, o Trident obtém os dataLIFs da SVM. Você pode especificar um nome de domínio totalmente qualificado (FQDN) para ser usado nas operações de montagem NFS, permitindo criar um DNS round-robin para balancear a carga entre várias dataLIFs. Consulte"Opções e exemplos de configuração do ONTAP NAS" para detalhes

O Trident consegue configurar CHAP para backends ONTAP ?

Sim. O Trident oferece suporte a CHAP bidirecional para backends ONTAP . Isso requer configuração. `useCHAP=true` na sua configuração de backend.

Como faço para gerenciar políticas de exportação com o Trident?

O Trident pode criar e gerenciar políticas de exportação dinamicamente a partir da versão 20.04. Isso permite que o administrador de armazenamento forneça um ou mais blocos CIDR em sua configuração de backend e que o Trident adicione os IPs dos nós que se enquadram nesses intervalos a uma política de exportação criada por ele. Dessa forma, o Trident gerencia automaticamente a adição e a exclusão de regras para nós com IPs dentro dos CIDRs especificados.

É possível usar endereços IPv6 para as LIFs de gerenciamento e de dados?

O Trident suporta a definição de endereços IPv6 para:

- `managementLIF`e `dataLIF` para backends ONTAP NAS.
- `managementLIF`para backends SAN ONTAP . Você não pode especificar `dataLIF` em um backend ONTAP SAN.

O Trident deve ser instalado usando a flag `--use-ipv6` (para `tridentctl` instalação), `IPv6` (para operador Trident), ou `tridentTPv6` (para instalação do Helm) para que funcione em IPv6.

É possível atualizar o LIF de gerenciamento no backend?

Sim, é possível atualizar o LIF de gerenciamento do backend usando o `tridentctl update backend` comando.

É possível atualizar o DataLIF no backend?

Você pode atualizar o DataLIF em `ontap-nas` e `ontap-nas-economy` apenas.

Posso criar vários backends no Trident para Kubernetes?

O Trident pode suportar vários backends simultaneamente, seja com o mesmo driver ou com drivers diferentes.

Como o Trident armazena as credenciais do backend?

O Trident armazena as credenciais do backend como segredos do Kubernetes.

Como o Trident seleciona um backend específico?

Se os atributos de backend não puderem ser usados para selecionar automaticamente os pools corretos para uma classe, o `storagePools` e `additionalStoragePools` Os parâmetros são usados para selecionar um conjunto específico de pools.

Como posso garantir que o Trident não provisione a partir de um backend específico?

O `excludeStoragePools` O parâmetro é usado para filtrar o conjunto de pools que o Trident usa para provisionamento e removerá quaisquer pools que correspondam.

Se existirem vários backends do mesmo tipo, como o Trident seleciona qual backend usar?

Se houver vários backends configurados do mesmo tipo, o Trident seleciona o backend apropriado com base nos parâmetros presentes em `StorageClass` e `PersistentVolumeClaim` . Por exemplo, se houver vários backends de driver ontap-nas, o Trident tenta corresponder aos parâmetros no `StorageClass` e `PersistentVolumeClaim` combinados e compatíveis com um backend que possa atender aos requisitos listados em `StorageClass` e `PersistentVolumeClaim` . Se houver vários servidores de backend que correspondam à solicitação, o Trident selecionará um deles aleatoriamente.

O Trident suporta CHAP bidirecional com Element/ SolidFire?

Sim.

Como o Trident implementa Qtrees em um volume ONTAP ? Quantas Qtrees podem ser implantadas em um único volume?

O `ontap-nas-economy` O driver cria até 200 Qtrees no mesmo FlexVol volume (configurável entre 50 e 300), 100.000 Qtrees por nó de cluster e 2,4 milhões por cluster. Ao inserir um novo `PersistentVolumeClaim` que é atendido pelo driver de economia, o driver verifica se já existe um FlexVol volume que possa atender à nova Qtree. Caso não exista um FlexVol volume capaz de atender à Qtree, um novo FlexVol volume será criado.

Como posso configurar as permissões Unix para volumes provisionados em um NAS ONTAP ?

Você pode definir permissões Unix no volume provisionado pelo Trident configurando um parâmetro no arquivo de definição do backend.

Como posso configurar um conjunto explícito de opções de montagem NFS do ONTAP durante o provisionamento de um volume?

Por padrão, o Trident não define opções de montagem com nenhum valor no Kubernetes. Para especificar as opções de montagem na classe de armazenamento do Kubernetes, siga o exemplo fornecido."aqui" .

Como faço para definir os volumes provisionados para uma política de exportação específica?

Para permitir que os hosts apropriados acessem um volume, use o `exportPolicy` parâmetro configurado no arquivo de definição do backend.

Como faço para configurar a criptografia de volume através do Trident com o ONTAP?

Você pode configurar a criptografia no volume provisionado pelo Trident usando o parâmetro de criptografia no arquivo de definição do backend. Para mais informações, consulte:"Como o Trident funciona com NVE e NAE"

Qual a melhor forma de implementar QoS para ONTAP através do Trident?

Usar `StorageClasses` Implementar QoS para ONTAP.

Como posso especificar provisionamento fino ou espesso através do Trident?

Os drivers ONTAP suportam provisionamento fino ou provisionamento espesso. Os drivers ONTAP utilizam, por padrão, o provisionamento dinâmico (thin provisioning). Se desejar provisionamento espesso, você deve configurar o arquivo de definição do backend ou o `StorageClass` . Se ambos estiverem configurados, `StorageClass` tem precedência. Configure o seguinte para o ONTAP:

1. Sobre `StorageClass` , defina o `provisioningType` atributo como espesso.
2. No arquivo de definição do backend, habilite volumes espessos definindo `backend spaceReserve parameter` como volume.

Como posso garantir que os volumes em uso não sejam excluídos mesmo se eu excluir acidentalmente o PVC?

A proteção contra PVCs é ativada automaticamente no Kubernetes a partir da versão 1.10.

Posso cultivar PVCs NFS criados pela Trident?

Sim. Você pode expandir um tubo de PVC que foi criado pela Trident. Note que o crescimento automático de volume é um recurso do ONTAP que não se aplica ao Trident.

Posso importar um volume enquanto ele estiver no modo de Proteção de Dados (DP) do SnapMirror ou no modo offline?

A importação do volume falha se o volume externo estiver no modo DP ou estiver offline. Você recebe a seguinte mensagem de erro:

```
Error: could not import volume: volume import failed to get size of
volume: volume <name> was not found (400 Bad Request) command terminated
with exit code 1.
Make sure to remove the DP mode or put the volume online before importing
the volume.
```

Como a quota de recursos é traduzida em um cluster NetApp ?

A cota de recursos de armazenamento do Kubernetes deve funcionar enquanto houver capacidade disponível no armazenamento da NetApp . Quando o armazenamento NetApp não consegue respeitar as configurações de cota do Kubernetes devido à falta de capacidade, o Trident tenta provisionar, mas ocorre um erro.

Posso criar snapshots de volume usando o Trident?

Sim. O Trident oferece suporte à criação de snapshots de volumes sob demanda e volumes persistentes a partir de snapshots. Para criar PVs a partir de snapshots, certifique-se de que o `VolumeSnapshotDataSource` O recurso de bloqueio foi ativado.

Quais são os drivers que suportam snapshots de volume do Trident ?

A partir de hoje, o suporte para snapshots sob demanda está disponível para nossos produtos. `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , e `azure-netapp-files` Drivers de backend.

Como faço para criar um backup instantâneo de um volume provisionado pelo Trident com o ONTAP?

Isso está disponível em `ontap-nas` , `ontap-san` , e `ontap-nas-flexgroup` motoristas. Você também pode especificar um `snapshotPolicy` para o `ontap-san-economy` driver no nível FlexVol .

Isso também está disponível em `ontap-nas-economy` drivers, mas na granularidade do nível de FlexVol volume e não na granularidade do nível de qtree. Para habilitar a capacidade de criar snapshots de volumes provisionados pelo Trident, defina a opção de parâmetro de backend. `snapshotPolicy` para a política de snapshots desejada, conforme definida no backend ONTAP . O Trident não tem conhecimento de quaisquer instantâneos capturados pelo controlador de armazenamento.

Posso definir uma porcentagem de reserva de snapshots para um volume provisionado através do Trident?

Sim, você pode reservar uma porcentagem específica de espaço em disco para armazenar as cópias de snapshots por meio do Trident , definindo a configuração. `snapshotReserve` atributo no arquivo de definição do backend. Se você configurou `snapshotPolicy` e `snapshotReserve` No arquivo de definição do backend, a porcentagem de reserva de snapshots é definida de acordo com o `snapshotReserve` porcentagem mencionada no arquivo de backend. Se o `snapshotReserve` A porcentagem não é mencionada; por padrão, o ONTAP considera 5% para a reserva de snapshots. Se o `snapshotPolicy` A opção está definida como nenhuma e a porcentagem de reserva de instantâneo está definida como 0.

Posso acessar diretamente o diretório de snapshots do volume e copiar arquivos?

Sim, você pode acessar o diretório de snapshots no volume provisionado pelo Trident definindo o `snapshotDir` parâmetro no arquivo de definição do backend.

Posso configurar o SnapMirror para volumes através do Trident?

Atualmente, o SnapMirror precisa ser configurado externamente usando a CLI do ONTAP ou o OnCommand System Manager.

Como faço para restaurar volumes persistentes para um snapshot específico do ONTAP ?

Para restaurar um volume para um snapshot do ONTAP , execute os seguintes passos:

1. Suspenda o pod da aplicação que está utilizando o volume persistente.
2. Retorne ao snapshot desejado através da CLI do ONTAP ou do OnCommand System Manager.
3. Reinicie o pod da aplicação.

O Trident consegue provisionar volumes em SVMs que tenham um espelhamento de compartilhamento de carga configurado?

É possível criar espelhos de balanceamento de carga para volumes raiz de SVMs que fornecem dados via NFS. O ONTAP atualiza automaticamente os espelhos de balanceamento de carga para volumes que foram criados pelo Trident. Isso pode resultar em atrasos na montagem dos volumes. Quando vários volumes são criados usando o Trident, o provisionamento de um volume depende da atualização do espelho de balanceamento de carga ONTAP .

Como posso separar o uso de classes de armazenamento para cada cliente/inquilino?

O Kubernetes não permite classes de armazenamento em namespaces. No entanto, você pode usar o Kubernetes para limitar o uso de uma classe de armazenamento específica por namespace usando Cotas de Recursos de Armazenamento, que são por namespace. Para negar o acesso de um namespace específico a um armazenamento específico, defina a cota de recursos como 0 para essa classe de armazenamento.

Solução de problemas

Utilize as dicas fornecidas aqui para solucionar problemas que você possa encontrar durante a instalação e o uso do Trident.

> Para obter ajuda com o Trident, crie um pacote de suporte usando `tridentctl logs -a -n trident` e envie para o suporte da NetApp .

Solução de problemas gerais

- Se a cápsula Trident não subir corretamente (por exemplo, quando a cápsula Trident fica presa no `ContainerCreating` fase com menos de dois contêineres prontos), em execução `kubectl -n trident describe deployment trident` e `kubectl -n trident describe pod trident-**` pode fornecer informações adicionais. Obtenção de logs do kubelet (por exemplo, via `journalctl -xeu kubelet`) também pode ser útil.
- Se não houver informações suficientes nos registros do Trident , você pode tentar ativar o modo de depuração do Trident passando o parâmetro `-d` Adicione um parâmetro à configuração de instalação com base na sua opção de instalação.

  Em seguida, confirme se o modo de depuração está ativado usando `./tridentctl logs -n trident` e procurando por `level=debug msg` no registro.

Instalado com operador

```
kubectl patch torc trident -n <namespace> --type=merge -p
'{"spec":{"debug":true}}'
```

Isso reiniciará todos os pods do Trident , o que pode levar vários segundos. Você pode verificar isso observando a coluna 'IDADE' na saída de `kubectl get pod -n trident` .

Para Trident 20.07 e 20.10 use `tprov` em vez de `torc` .

Instalado com Helm

```
helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
tridentDebug=true`
```

Instalado com tridentctl

```
./tridentctl uninstall -n trident
./tridentctl install -d -n trident
```

- Você também pode obter logs de depuração para cada backend incluindo `debugTraceFlags` na sua definição de backend. Por exemplo, inclua `debugTraceFlags: {"api":true, "method":true,}` Para obter chamadas de API e percursos de métodos nos logs do Trident . Os backends existentes podem ter `debugTraceFlags` configurado com um `tridentctl backend update` .
- Ao usar o Red Hat Enterprise Linux CoreOS (RHCOS), certifique-se de que `iscsid` Está habilitado nos nós de trabalho e é iniciado por padrão. Isso pode ser feito usando as configurações de máquina do OpenShift ou modificando os modelos de ignição.
- Um problema comum que você pode encontrar ao usar o Trident com "Azure NetApp Files" Isso ocorre quando os segredos do locatário e do cliente provêm de um registro de aplicativo com permissões insuficientes. Para obter uma lista completa dos requisitos do Trident , consulte:"Azure NetApp Files" configuração.
- Se houver problemas com a montagem de um painel fotovoltaico em um contêiner, certifique-se de que `rpcbind` Está instalado e em funcionamento. Utilize o gerenciador de pacotes necessário para o sistema operacional do host e verifique se `rpcbind` Está em execução. Você pode verificar o status do `rpcbind` serviço executando um `systemctl status rpcbind` ou seu equivalente.
- Se um backend Trident relatar que está no `failed` Embora o estado tenha funcionado anteriormente, é provável que seja causado pela alteração das credenciais SVM/admin associadas ao backend. Atualizando as informações do backend usando `tridentctl update backend` Ou então, sacudir o pod Trident resolverá esse problema.
- Se você encontrar problemas de permissão ao instalar o Trident com o Docker como ambiente de execução de contêineres, tente instalar o Trident com o... `--in cluster=false` bandeira. Isso não usará um pod de instalação e evitará problemas de permissão observados devido ao `trident-installer` usuário.
- Use o `uninstall parameter <Uninstalling Trident>` para limpar após uma execução malsucedida. Por padrão, o script não remove os CRDs criados pelo Trident, tornando seguro desinstalar e instalar novamente, mesmo em uma implantação em andamento.

- Se você deseja fazer o downgrade para uma versão anterior do Trident, primeiro execute o seguinte comando: `tridentctl uninstall` comando para remover Trident. Baixe o arquivo desejado "Versão Trident" e instale usando o `tridentctl install` comando.
- Após uma instalação bem-sucedida, se um tubo de PVC ficar preso no `Pending` fase, em execução `kubectl describe pvc` Pode fornecer informações adicionais sobre o motivo pelo qual a Trident não conseguiu provisionar um PV para este PVC.

Implantação malsucedida do Trident usando o operador

Se você estiver implantando o Trident usando o operador, o status de `TridentOrchestrator` mudanças de `Installing` para `Installed` . Se você observar o `Failed` Se o operador não conseguir se recuperar sozinho, verifique os registros do operador executando o seguinte comando:

```
tridentctl logs -l trident-operator
```

Analisar os registros do contêiner trident-operator pode indicar onde está o problema. Por exemplo, um desses problemas poderia ser a impossibilidade de obter as imagens de contêiner necessárias dos registros upstream em um ambiente isolado da internet (airgapped).

Para entender por que a instalação do Trident não teve sucesso, você deve dar uma olhada em… `TridentOrchestrator` status.

```
kubectl describe torc trident-2
Name:         trident-2
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Image Pull Secrets:         <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:                      Trident is bound to another CR 'trident'
  Namespace:                    trident-2
  Status:                       Error
  Version:
Events:
  Type     Reason  Age                From                         Message
  ----     ------  ----               ----                         -------
  Warning  Error   16s (x2 over 16s)  trident-operator.netapp.io  Trident
is bound to another CR 'trident'
```

Este erro indica que já existe um `TridentOrchestrator` que foi usado para instalar o Trident. Como cada cluster Kubernetes só pode ter uma instância do Trident, o operador garante que, a qualquer momento, exista apenas uma instância ativa. `TridentOrchestrator` que pode criar.

Além disso, observar o estado das cápsulas Trident pode muitas vezes indicar se algo está errado.

```
kubectl get pods -n trident

NAME                                READY   STATUS             RESTARTS
AGE
trident-csi-4p5kq                   1/2     ImagePullBackOff   0
5m18s
trident-csi-6f45bfd8b6-vfrkw        4/5     ImagePullBackOff   0
5m19s
trident-csi-9q5xc                   1/2     ImagePullBackOff   0
5m18s
trident-csi-9v95z                   1/2     ImagePullBackOff   0
5m18s
trident-operator-766f7b8658-ldzsv   1/1     Running            0
8m17s
```

É possível ver claramente que os pods não conseguem inicializar completamente porque uma ou mais imagens de contêiner não foram obtidas.

Para resolver o problema, você deve editar o `TridentOrchestrator` CR. Alternativamente, você pode excluir `TridentOrchestrator` e crie uma nova com a definição modificada e precisa.

Implantação malsucedida do Trident usando `tridentctl`

Para ajudar a descobrir o que deu errado, você pode executar o instalador novamente usando o`-d` argumento, que ativará o modo de depuração e ajudará você a entender qual é o problema:

```
./tridentctl install -n trident -d
```

Após resolver o problema, você pode limpar a instalação da seguinte forma e, em seguida, executar o programa. `tridentctl install` Comande novamente:

```
./tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Removed Trident user from security context constraint.
INFO Trident uninstallation succeeded.
```

Remova completamente o Trident e os CRDs.

Você pode remover completamente o Trident e todos os CRDs criados, bem como os recursos personalizados associados.

Isso não pode ser desfeito. Não faça isso a menos que queira uma instalação completamente nova do Trident. Para desinstalar o Trident sem remover os CRDs, consulte"Desinstale o Trident" .

Operador do Trident

Para desinstalar o Trident e remover completamente os CRDs usando o operador Trident :

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

Leme

Para desinstalar o Trident e remover completamente os CRDs usando o Helm:

```
kubectl patch torc trident --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

<code>tridentctl</code>

Para remover completamente os CRDs após desinstalar o Trident, utilize `tridentctl`

```
tridentctl obliviate crd
```

Falha na desempacotamento de nó NVMe com namespaces de bloco bruto RWX no Kubernetes 1.26

Se você estiver executando o Kubernetes 1.26, a desempacotamento de nós pode falhar ao usar NVMe/TCP com namespaces de bloco bruto RWX. Os cenários a seguir fornecem soluções alternativas para a falha. Alternativamente, você pode atualizar o Kubernetes para a versão 1.27.

O namespace e o pod foram excluídos.

Considere um cenário em que você tenha um namespace gerenciado Trident (volume persistente NVMe) anexado a um pod. Se você excluir o namespace diretamente do backend do ONTAP , o processo de descompactação ficará travado após a tentativa de exclusão do pod. Este cenário não afeta o cluster Kubernetes nem outras funcionalidades.

Solução alternativa

Desmonte o volume persistente (correspondente a esse namespace) do respectivo nó e exclua-o.

dados bloqueados LIFs

```
 If you block (or bring down) all the dataLIFs of the NVMe Trident
backend, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Solução alternativa
Reinicie o dataLIFS para restaurar a funcionalidade completa.
```

Mapeamento de namespace excluído

```
 If you remove the `hostNQN` of the worker node from the corresponding
subsystem, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Solução alternativa
Adicione o `hostNQN` Voltar ao subsistema.
```

Os clientes NFSv4.2 relatam "argumento inválido" após atualizar o ONTAP quando esperavam que "v4.2-xattrs" estivesse habilitado

Após a atualização do ONTAP, os clientes NFSv4.2 podem relatar erros de "argumento inválido" ao tentar montar exportações NFSv4.2. Este problema ocorre quando o `v4.2-xattrs` a opção não está habilitada no SVM. .Solução alternativa Habilite o `v4.2-xattrs` opção no SVM ou atualize para o ONTAP 9.12.1 ou posterior, onde esta opção é habilitada por padrão.

Apoiar

A NetApp oferece suporte ao Trident de diversas maneiras. Diversas opções gratuitas de autoatendimento estão disponíveis 24 horas por dia, 7 dias por semana, como artigos da base de conhecimento (KB) e um canal no Discord.

Suporte Trident

A Trident oferece três níveis de suporte com base na sua versão. Consulte"Suporte da versão de software NetApp para definições" .

Suporte total
A Trident oferece suporte completo por doze meses a partir da data de lançamento.

Suporte limitado
A Trident oferece suporte limitado durante os meses 13 a 24 a partir da data de lançamento.

Autossuficiência
A documentação do Trident estará disponível para os meses 25 a 36 a partir da data de lançamento.

Versão	Suporte total	Suporte limitado	Autossuficiência

"25,06"	Junho de 2026	Junho de 2027	Junho de 2028
"25,02"	Fevereiro de 2026	Fevereiro de 2027	Fevereiro de 2028
"24,10"	Outubro de 2025	Outubro de 2026	Outubro de 2027
"24,06"	Junho de 2025	Junho de 2026	Junho de 2027
"24,02"	Fevereiro de 2025	Fevereiro de 2026	Fevereiro de 2027
"23,10"	—	Outubro de 2025	Outubro de 2026
"23,07"	—	Julho de 2025	Julho de 2026
"23,04"	—	Abril de 2025	Abril de 2026
"23,01"	—	—	Janeiro de 2026
"22,10"	—	—	Outubro de 2025

Autossuficiência

Para obter uma lista completa de artigos de resolução de problemas, consulte "Base de conhecimento da NetApp (login necessário)" .

Apoio comunitário

Existe uma comunidade pública ativa de usuários de contêineres (incluindo desenvolvedores do Trident) em nosso site."Canal do Discord" . Este é um ótimo lugar para fazer perguntas gerais sobre o projeto e discutir tópicos relacionados com colegas que compartilham os mesmos interesses.

Suporte técnico da NetApp

Para obter ajuda com o Trident, crie um pacote de suporte usando `tridentctl logs -a -n trident` e envie para `NetApp Support <Getting Help>`.

Para maiores informações

- "Recursos Trident"
- "Hub do Kubernetes"

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`

Avisos legais

Avisos legais fornecem acesso a declarações de direitos autorais, marcas registradas, patentes e muito mais.

Direitos autorais

"https://www.netapp.com/company/legal/copyright/"

Marcas Registradas

NETAPP, o logotipo da NETAPP e as marcas listadas na página de Marcas Registradas da NetApp são marcas registradas da NetApp, Inc. Outros nomes de empresas e produtos podem ser marcas registradas de seus respectivos proprietários.

"https://www.netapp.com/company/legal/trademarks/"

Patentes

Uma lista atual de patentes de propriedade da NetApp pode ser encontrada em:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Política de Privacidade

"https://www.netapp.com/company/legal/privacy-policy/"

Código aberto

Você pode consultar os direitos autorais e as licenças de terceiros usados no software NetApp para Trident no arquivo de avisos de cada versão em https://github.com/NetApp/trident/ .

Informações sobre direitos autorais

Copyright © 2026 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.