使用Astra Control配置程序

Astra Control Center

NetApp
March 12, 2024

目录

使用Astra Control配置程序

配置存储后端加密

通过使用Astra Control配置程序、您可以对受管集群和存储后端之间的流量启用加密、从而提高数据访问安全性。

Astra Control配置程序支持对两种类型的存储后端进行Kerberos加密：

- 内部部署的**Kubernet**—Astra配置程序支持通过从ONTAP OpenShift和上游Kubernetes集群到内部ONTAP卷的NFS3和NFSv4连接进行Kerberos加密。
- **NFSv**-控件配置程序支持通过从上游Azure NetApp Files集群到Azure NetApp Files卷的NFSv4.1连接进行Kerberos加密。

您可以创建、删除、调整大小、创建快照、克隆、 只读克隆、并导入使用NFS加密的卷。

使用内部ONTAP卷配置传输中的Kerberos加密

您可以对受管集群与内部ONTAP存储后端之间的存储流量启用Kerberos加密。

> 仅支持使用对使用内部ONTAP存储后端的NFS流量进行Kerberos加密 `ontap-nas` 存储驱动程序。

开始之前

- 确保已在受管集群上启用Asta Control配置程序。请参见 "启用Asta Control配置程序" 有关说明，请参见。
- 确保您可以访问 `tridentctl` 实用程序。
- 确保您对ONTAP存储后端具有管理员访问权限。
- 确保您知道要从ONTAP存储后端共享的一个或多个卷的名称。
- 确保已准备好ONTAP Storage VM以支持NFS卷的Kerberos加密。请参见 "在数据 LIF 上启用 Kerberos" 有关说明，请参见。
- 确保已正确配置使用Kerberos加密的任何NFSv4卷。请参阅的NetApp NFSv4域配置一节(第13页) "《NetApp NFSv4增强功能和最佳实践指南》"。

添加或修改ONTAP导出策略

您需要向现有ONTAP导出策略添加规则、或者创建新的导出策略、以便对ONTAP Storage VM根卷以及与上游Kubornetes集群共享的任何ONTAP卷支持Kerberos加密。您添加的导出策略规则或创建的新导出策略需要支持以下访问协议和访问权限：

访问协议

使用NFS、NFSv3和NFSv4访问协议配置导出策略。

访问详细信息

您可以根据卷的需求配置以下三种不同版本的Kerberos加密之一：

- **Kerberos 5**-(身份验证和加密)
- **Kerberos 5i**-(身份验证和加密与身份保护)
- **Kerberos 5p**-(身份验证和加密、具有身份和隐私保护功能)

使用适当的访问权限配置ONTAP导出策略规则。例如、如果集群要挂载混合使用Kerberos 5i和Kerberos 5p加密的NFS卷、请使用以下访问设置：

Type	只读访问	读/写访问	超级用户访问
"unix"	enabled	enabled	enabled
Kerberos 5i	enabled	enabled	enabled
Kerberos 5p	enabled	enabled	enabled

有关如何创建ONTAP导出策略和导出策略规则、请参见以下文档：

- "创建导出策略"
- "向导出策略添加规则"

创建存储后端

您可以创建包含Kerberos加密功能的A作用 力控制配置程序存储后端配置。

关于此任务

在创建用于配置Kerberos加密的存储后端配置文件时、您可以使用指定三个不同版本的Kerberos加密之一 `spec.nfsMountOptions` 参数：

- `spec.nfsMountOptions: sec=krb5` (身份验证和加密)
- `spec.nfsMountOptions: sec=krb5i` (身份验证和加密以及身份保护)
- `spec.nfsMountOptions: sec=krb5p` (身份验证和加密以及身份和隐私保护)

请仅指定一个Kerberos级别。如果在参数列表中指定多个Kerberos加密级别、则仅会使用第一个选项。

步骤

1. 在受管集群上、使用以下示例创建存储后端配置文件。将括号<>中的值替换为您环境中的信息：

```
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. 使用您在上一步中创建的配置文件创建后端：

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

如果后端创建失败，则后端配置出现问题。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs
```

确定并更正配置文件中的问题后，您可以再次运行 create 命令。

创建存储类。

您可以创建存储类来配置采用Kerberos加密的卷。

关于此任务

创建存储类对象时、您可以使用指定三个不同版本的Kerberos加密之一 `mountOptions` 参数：

- `mountOptions: sec=krb5` (身份验证和加密)
- `mountOptions: sec=krb5i` (身份验证和加密以及身份保护)
- `mountOptions: sec=krb5p` (身份验证和加密以及身份和隐私保护)

请仅指定一个Kerberos级别。如果在参数列表中指定多个Kerberos加密级别、则仅会使用第一个选项。如果您在存储后端配置中指定的加密级别与您在存储类对象中指定的加密级别不同、则存储类对象优先。

步骤

1. 使用以下示例创建StorageClass Kubenetes对象：

```
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. 创建存储类：

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. 确保已创建存储类：

```
kubectl get sc ontap-nas-sc
```

您应看到类似于以下内容的输出：

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

配置卷

创建存储后端和存储类后、您现在可以配置卷。有关说明，请参见 "配置卷"。

为Azure NetApp Files卷配置传输中的Kerberos加密

您可以对受管集群与单个Azure NetApp Files存储后端或Azure NetApp Files存储后端虚拟池之间的存储流量启用Kerberos加密。

开始之前

- 确保已在受管Red Hat OpenShift集群上启用Asta Control配置程序。请参见 "启用Asta Control配置程序" 有关说明，请参见。
- 确保您可以访问 `tridentctl` 实用程序。
- 请注意中的要求并按照中的说明、确保您已为Kerberos加密准备好Azure NetApp Files存储后端 "Azure NetApp Files 文档"。
- 确保已正确配置使用Kerberos加密的任何NFSv4卷。请参阅的NetApp NFSv4域配置一节(第13页) "《NetApp NFSv4增强功能和最佳实践指南》"。

创建存储后端

您可以创建包含Kerberos加密功能的Azure NetApp Files存储后端配置。

关于此任务

在创建配置Kerberos加密的存储后端配置文件时、您可以对其进行定义、使其应用于以下两个可能的级别之一：

- 使用的*存储后端级别* `spec.kerberos` 字段
- 使用的*虚拟池级别* `spec.storage.kerberos` 字段

在虚拟池级别定义配置时、系统会使用存储类中的标签来选择该池。

在任一级别、您都可以指定以下三种不同版本的Kerberos加密之一：

- `kerberos: sec=krb5` (身份验证和加密)
- `kerberos: sec=krb5i` (身份验证和加密以及身份保护)
- `kerberos: sec=krb5p` (身份验证和加密以及身份和隐私保护)

步骤

1. 在受管集群上、根据需要定义存储后端的位置(存储后端级别或虚拟池级别)、使用以下示例之一创建存储后端配置文件。将括号<>中的值替换为您环境中的信息：

存储后端级别示例

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
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-anf-secret
```

虚拟池级别示例

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
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-anf-secret
```

2. 使用您在上一步中创建的配置文件创建后端：

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

如果后端创建失败，则后端配置出现问题。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs
```

确定并更正配置文件中的问题后，您可以再次运行 create 命令。

创建存储类。

您可以创建存储类来配置采用Kerberos加密的卷。

步骤

1. 使用以下示例创建StorageClass Kubenetes对象：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "nfs"
  selector: "type=encryption"
```

2. 创建存储类：

```
kubectl create -f sample-input/storage-class-anf-sc-nfs.yaml
```

3. 确保已创建存储类：

```
kubectl get sc anf-sc-nfs
```

您应看到类似于以下内容的输出：

```
NAME          PROVISIONER             AGE
anf-sc-nfs    csi.trident.netapp.io   15h
```

配置卷

创建存储后端和存储类后、您现在可以配置卷。有关说明，请参见 "配置卷"。

使用快照恢复卷数据

Asta Control配置程序可使用从快照快速原位还原卷 `TridentActionSnapshotRestore` (TSR) CR。此CR用作要务Kubbernetes操作、在操作完成后不会持久保留。

Asta Control配置程序支持在上执行快照还原 `ontap-san`， `ontap-san-economy`， `ontap-nas`， `ontap-`

nas-flexgroup， azure-netapp-files， gcp-cvs，和 solidfire-san 驱动程序。

开始之前

您必须具有绑定的PVC和可用的卷快照。

- 验证PVC状态是否已绑定。

```
kubectl get pvc
```

- 确认卷快照已准备就绪、可以使用。

```
kubectl get vs
```

步骤

1. 创建TSR CR。此示例将为PVC创建CR pvc1 和卷快照 pvc1-snapshot。

```
cat tasr-pvc1-snapshot.yaml

apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. 应用CR以从快照还原。此示例将从Snapshot还原 pvc1。

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

结果

Asta Control配置程序从快照还原数据。您可以验证快照还原状态。

```
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: this-doesnt-matter
    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: ""
```

- 在大多数情况下、如果发生故障、Asta Control配置程序不会自动重试此操作。您需要再次执行此操作。
- 没有管理员访问权限的Kubbernetes用户可能必须获得管理员授予的权限、才能在其应用程序命名空间中创建TSR CR。

使用SnapMirror复制卷

您可以使用Astra Control配置程序在一个集群上的源卷和对等集群上的目标卷之间创建镜像关系、以便为灾难恢复复制数据。您可以使用具有名称流的自定义资源定义(CRD)执行以下操作：

- 在卷之间创建镜像关系(PVC)
- 删除卷之间的镜像关系
- 中断镜像关系
- 在灾难情况下提升二级卷(故障转移)
- 在集群之间执行应用程序无中断过渡(在计划内故障转移或迁移期间)

复制前提条件

开始之前、请确保满足以下前提条件：

ONTAP 集群

- **Astra**控件配置程序：Astra控件配置程序版本22.10或更高版本必须同时位于使用ONTAP作为后端的源和目标Kubernetes集群上。
- 许可证：必须在源和目标ONTAP集群上启用使用数据保护包的ONTAP SnapMirror异步许可证。请参见 "ONTAP 中的SnapMirror许可概述" 有关详细信息 …

对等

- 集群和**SVM**：ONTAP存储后端必须建立对等状态。请参见 "集群和 SVM 对等概述" 有关详细信息 …

  确保两个ONTAP集群之间的复制关系中使用的SVM名称是唯一的。

- **Astra Control**置备程序和**SVM**：对等远程SVM必须可供目标集群上的Astra Control置备程序使用。

支持的驱动程序

- ONTAP -NAS和ONTAP SAN驱动程序支持卷复制。

创建镜像**PVC**

按照以下步骤并使用CRD示例在主卷和二级卷之间创建镜像关系。

步骤

1. 在主Kubbernetes集群上执行以下步骤：
   a. 使用创建StorageClass对象 `trident.netapp.io/replication: true` 参数。

      示例

      ```
      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. 使用先前创建的StorageClass创建PVC。

示例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

c. 使用本地信息创建镜像关系CR。

示例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Astra Control配置程序会提取卷的内部信息以及卷的当前数据保护(DP)状态、然后填充镜像关系的状态字段。

d. 获取TridentMirorRelationship CR以获取PVC的内部名称和SVM。

```
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. 在二级Kubbernetes集群上执行以下步骤：

   a. 使用trident.netapp.io/replication: true参数创建StorageClass。

      示例

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        trident.netapp.io/replication: true
      ```

   b. 使用目标和源信息创建镜像关系CR。

      示例

      ```
      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"
      ```

Asta控件配置程序将使用配置的关系策略名称(或ONTAP的默认策略名称)创建SnapMirror关系并对其进行初始化。

c. 使用先前创建的StorageClass创建一个PVC以用作二级(SnapMirror目标)。

示例

```
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
```

Astra Control配置程序将检查是否存在TridentMirorRelationship CRD、如果此关系不存在、则无法创建卷。如果存在此关系、Astra控件配置程序将确保将新FlexVol卷放置到与镜像关系中定义的远程SVM建立对等关系的SVM上。

卷复制状态

三级镜像关系(TCR)是一种CRD、表示PVC之间复制关系的一端。目标T关系 管理器具有一个状态、该状态会告诉Astra Control配置程序所需的状态是什么。目标T关系 管理器具有以下状态：

- 已建立：本地PVC是镜像关系的目标卷、这是一个新关系。
- 提升：本地PVC可读写并可挂载、当前未建立任何有效的镜像关系。
- 重新建立：本地PVC是镜像关系的目标卷、以前也位于该镜像关系中。
  - 如果目标卷曾经与源卷建立关系、因为它会覆盖目标卷的内容、则必须使用重新建立的状态。
  - 如果卷之前未与源建立关系、则重新建立的状态将失败。

在计划外故障转移期间提升辅助**PVC**

在二级Kubbernetes集群上执行以下步骤：

- 将TridentMirorRelationship的_spec.state_字 段更新为 `promoted`。

在计划内故障转移期间提升辅助**PVC**

在计划内故障转移(迁移)期间、执行以下步骤以提升二级PVC：

步骤

1. 在主Kubbernetes集群上、创建PVC的快照、并等待创建快照。
2. 在主Kubnetes集群上、创建SnapshotInfo CR以获取内部详细信息。

   示例

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. 在二级Kubernetes集群上、将_TridentMirorRelationship_ CR的_spec.state_字 段更新为_promoted_和_spec.promotedSnapshotHandle_、以成为快照的内部名称。
4. 在二级Kubernetes集群上、确认Trident镜像 关系的状态(stats.state字段)为已提升。

在故障转移后还原镜像关系

在还原镜像关系之前、请选择要用作新主卷的那一端。

步骤

1. 在二级Kubernetes集群上、确保已更新TundentMirorRelationship上的_spic.netVolumeHandle_字段的值。
2. 在二级Kubernetes集群上、将Trident镜像 关系的_spec.mirector_字段更新到 `reestablished`。

其他操作

Asta Control配置程序支持在主卷和二级卷上执行以下操作：

将主**PVC**复制到新的二级**PVC**

确保您已有一个主PVC和一个次要PVC。

步骤

1. 从已建立的二级(目标)集群中删除PerbestentVolumeClaim和TridentMirorRelationship CRD。
2. 从主(源)集群中删除TridentMirorRelationship CRD。
3. 在主(源)集群上为要建立的新二级(目标) PVC创建新的TridentMirorRelationship CRD。

调整镜像、主**PVC**或二级**PVC**的大小

可以正常调整PVC的大小、如果数据量超过当前大小、ONTAP将自动扩展任何目标flevxvol。

从**PVC**中删除复制

要删除复制、请对当前二级卷执行以下操作之一：

- 删除次要PVC上的镜像关系。此操作将中断复制关系。
- 或者、将spec.state字段更新为_promoted_。

删除**PVC (**之前已镜像**)**

ASRA Control配置程序会检查是否存在复制的PVC、并在尝试删除卷之前释放复制关系。

删除**TTr**

删除镜像关系一端的T磁 还原会导致剩余的T磁 还原在Astra Control配置程序完成删除之前过渡到_promoted状态。如果选定要删除的TMirror已处于_Promote 状态、则不存在现有镜像关系、此时*TMirror*将被删除、*Astra Control*配置程序会将本地*PVC*提升为*_ReadWrite*。此删除操作将释放ONTAP中本地卷的SnapMirror元数据。如果此卷将来要在镜像关系中使用、则在创建新镜像关系时、它必须使用具有_re设立_卷复制状态的新TMirror。

在**ONTAP**联机时更新镜像关系

建立镜像关系后、可以随时更新这些关系。您可以使用 `state: promoted` 或 `state: reestablished` 用于更新关系的字段。
将目标卷提升为常规ReadWrite卷时、可以使用_promotedSnapshotHandle_指定要将当前卷还原到的特定快照。

在**ONTAP**脱机时更新镜像关系

您可以使用CRD执行SnapMirror更新、而Astra Control不直接连接到ONTAP集群。请参阅以下TridentAction镜像 更新的示例格式：

示例

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` 反映TridentAction镜像 更新CRD的状态。它可以从_suced_、_in Progress _或_failed中获取值。