安全性
Trident

NetApp
July 01, 2026

目录

安全性

安全性

请使用此处列出的建议，以确保 Trident 的安装安全。

在自己的命名空间中运行 **Trident**

必须防止应用程序、应用程序管理员、用户和管理应用程序访问 Trident 对象定义或 Pod，以确保可靠的存储并阻止潜在的恶意活动。

要将其他应用程序和用户与 Trident 分离，请始终在自己的 Kubernetes 命名空间中安装 Trident(`trident`）。将 Trident 放在自己的命名空间中可确保只有 Kubernetes 管理人员才能访问 Trident Pod 和存储在命名空间 CRD 对象中的工件（如后端和 CHAP 机密（如果适用））。您应该确保只允许管理员访问 Trident 命名空间，从而访问 `tridentctl`应用程序。

对 **ONTAP SAN** 后端使用 **CHAP** 身份验证

Trident 支持基于 CHAP 的 ONTAP SAN 工作负载身份验证（使用 `ontap-san`和 `ontap-san-economy`驱动程序）。NetApp 建议使用双向 CHAP 与 Trident 进行主机和存储后端之间的身份验证。

对于使用 SAN 存储驱动程序的 ONTAP 后端，Trident 可以设置双向 CHAP 并通过 `tridentctl`管理 CHAP 用户名和机密。请参阅"准备使用 ONTAP SAN 驱动程序配置后端"以了解 Trident 如何在 ONTAP 后端上配置 CHAP。

将 **CHAP** 身份验证与 **NetApp HCI** 和 **SolidFire** 后端一起使用

NetApp 建议部署双向 CHAP，以确保主机与 NetApp HCI 和 SolidFire 后端之间的身份验证。Trident 使用一个秘密对象，其中每个租户包含两个 CHAP 密码。安装 Trident 后，它会管理 CHAP 机密并将其存储在相应 PV 的 `tridentvolume` CR 对象中。创建 PV 时，Trident 使用 CHAP 机密来启动 iSCSI 会话，并通过 CHAP 与 NetApp HCI 和 SolidFire 系统进行通信。

由 Trident 创建的卷不与任何卷访问组关联。

将 **Trident** 与 **NVE** 和 **NAE** 配合使用

NetApp ONTAP 提供静态数据加密，以便在磁盘被盗、退回或重新使用时保护敏感数据。有关详细信息，请参阅 "配置 NetApp 卷加密概述"。

- 如果在后端启用了 NAE，则在 Trident 中配置的任何卷都将启用 NAE。
  - 您可以将 NVE 加密标志设置为 `""`以创建启用 NAE 的卷。
- 如果未在后端启用 NAE，则在 Trident 中配置的任何卷都将启用 NVE，除非在后端配置中将 NVE 加密标志设置为 `false`（默认值）。

在启用 NAE 的后端上在 Trident 中创建的卷必须进行 NVE 或 NAE 加密。

- 您可以在 Trident 后端配置中将 NVE 加密标志设置为 `true`，以覆盖 NAE 加密，并在每个卷的基础上使用特定的加密密钥。
- 在启用 NAE 的后端上将 NVE 加密标志设置为 `false`可创建启用 NAE 的卷。无法通过将 NVE 加密标志设置为 `false`来禁用 NAE 加密。

- 您可以通过显式设置 NVE 加密标志为 `true`在 Trident 中手动创建 NVE 卷。

有关后端配置选项的更多信息，请参阅：

- "ONTAP SAN 配置选项"
- "ONTAP NAS 配置选项"

Linux 统一密钥设置 (LUKS)

您可以启用 Linux Unified Key Setup (LUKS) 来加密 Trident 上的 ONTAP SAN 和 ONTAP SAN ECONOMY 卷。Trident 支持 LUKS 加密卷的密码短语轮换和卷扩展。

在 Trident 中， LUKS 加密的卷使用 aes-xts-plain64 密码和模式，如 "NIST" 所推荐。

ASA r2 系统不支持 LUKS 加密。有关 ASA r2 系统的信息，请参见 "了解 ASA r2 存储系统"。

开始之前

- 工作节点必须安装 cryptsetup 2.1 或更高版本（但低于 3.0）。有关更多信息，请访问 "Gitlab：cryptsetup"。
- 出于性能原因，NetApp 建议工作节点支持高级加密标准新指令 (AES-NI)。要验证 AES-NI 支持，请运行以下命令：

```
grep "aes" /proc/cpuinfo
```

如果没有返回任何内容，则表示您的处理器不支持 AES-NI。有关 AES-NI 的更多信息，请访问："Intel：高级加密标准指令 (AES-NI)"。

启用 LUKS 加密

您可以使用 Linux Unified Key Setup (LUKS) 为 ONTAP SAN 和 ONTAP SAN ECONOMY 卷启用每个卷的主机端加密。

步骤

1. 在后端配置中定义 LUKS 加密属性。有关 ONTAP SAN 后端配置选项的更多信息，请参阅 "ONTAP SAN 配置选项"。

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. 使用 `parameters.selector` 来定义使用 LUKS 加密的存储池。例如：

```
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. 创建一个包含 LUKS 密码短语的密钥。例如：

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

限制

LUKS 加密的卷无法利用 ONTAP 重复数据删除和压缩。

用于导入 LUKS 卷的后端配置

要导入 LUKS 卷，您必须在后端将 `luksEncryption`设置为(`true。 luksEncryption`选项告诉 Trident 卷是否符合 LUKS 标准(`true`）或不符合 LUKS 标准(`false`），如以下示例所示。

```
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'
```

用于导入 LUKS 卷的 PVC 配置

要动态导入 LUKS 卷，请将注释 `trident.netapp.io/luksEncryption` 设置为 `true` 并在 PVC 中包含启用 LUKS 的存储类，如本示例所示。

```
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
```

轮换 LUKS 密码短语

您可以轮换 LUKS 密码短语并确认轮换。

> 在验证密码短语不再被任何卷、快照或密码引用之前，请不要忘记该密码短语。如果引用的密码短语丢失，您可能无法挂载卷，并且数据将保持加密且无法访问。

关于此任务

当在指定新的 LUKS 密码短语后创建挂载卷的 Pod 时，会发生 LUKS 密码短语轮换。创建新 Pod 时，Trident 会将卷上的 LUKS 密码短语与密码中的活动密码短语进行比较。

- 如果卷上的密码与 secret 中的活动密码不匹配，则会发生旋转。
- 如果卷上的密码与 secret 中的活动密码匹配，则此 `previous-luks-passphrase` 参数将被忽略。

步骤

1. 添加 `node-publish-secret-name` 和 `node-publish-secret-namespace` StorageClass 参数。例如：

```
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. 识别卷或快照上的现有密码短语。

   卷

   ```
   tridentctl -d get volume luks-pvc1
   GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

   ...luksPassphraseNames:["A"]
   ```

   Snapshot

   ```
   tridentctl -d get snapshot luks-pvc1
   GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

   ...luksPassphraseNames:["A"]
   ```

3. 更新卷的 LUKS 密钥以指定新的和以前的密码。请确保 `previous-luke-passphrase-name` 和 `previous-luks-passphrase` 匹配之前的密码短语。

   ```
   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. 创建挂载卷的新 pod。这是启动轮换所必需的。
5. 验证密码短语已轮换。

   卷

   ```
   tridentctl -d get volume luks-pvc1
   GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

   ...luksPassphraseNames:["B"]
   ```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

结果

当仅在卷和快照上返回新密码短语时，密码短语已轮换。

> 如果返回两个密码短语，例如 `luksPassphraseNames: ["B", "A"]`，则旋转是不完整的。您可以触发一个新的 pod 来尝试完成旋转。

启用卷扩展

您可以在 LUKS 加密的卷上启用卷扩展。

步骤

1. 启用 `CSINodeExpandSecret`功能门（beta 1.25+）。有关详细信息，请参见 "Kubernetes 1.25：使用 Secrets 进行节点驱动的 CSI 卷扩展"。
2. 添加 `node-expand-secret-name` 和 `node-expand-secret-namespace` StorageClass 参数。例如：

```
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
```

结果

启动在线存储扩展时， kubelet 会将相应的凭据传递给驱动程序。

Kerberos 传输中加密

通过使用 Kerberos 在线加密，您可以为托管群集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。

Trident 支持将 Kerberos 加密用于 ONTAP 作为存储后端：

- 本地 **ONTAP** - Trident 支持通过 NFSv3 和 NFSv4 连接从 Red Hat OpenShift 和上游 Kubernetes 集群到本地 ONTAP 卷的 Kerberos 加密。

可以创建、删除、调整大小、快照、克隆、只读克隆以及导入使用 NFS 加密的卷。

使用本地 **ONTAP** 卷配置运行中的 **Kerberos** 加密

您可以在托管集群与本地 ONTAP 存储后端之间的存储流量上启用 Kerberos 加密。

> 仅使用 `ontap-nas` 存储驱动程序支持针对具有本地 ONTAP 存储后端的 NFS 流量的 Kerberos 加密。

开始之前

- 请确保您拥有该 `tridentctl` 实用程序的访问权限。
- 确保您拥有对 ONTAP 存储后端的管理员访问权限。
- 确保您知道要从 ONTAP 存储后端共享的卷的名称。
- 确保已准备好 ONTAP 存储虚拟机，以支持 NFS 卷的 Kerberos 加密。有关说明，请参阅 "在 dataLIF 上启用 Kerberos"。
- 请确保您使用 Kerberos 加密的任何 NFSv4 卷配置正确。请参阅 NetApp NFSv4 域配置部分（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

您需要向现有的 ONTAP 导出策略添加规则，或创建支持 Kerberos 加密的新导出策略，用于 ONTAP 存储 VM 根卷以及与上游 Kubernetes 集群共享的任何 ONTAP 卷。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

根据您对卷的需求，您可以配置三种不同版本的 Kerberos 加密之一：

- **Kerberos 5** -（身份验证和加密）
- **Kerberos 5i** -（具有身份保护的身份验证和加密）
- **Kerberos 5p** -（具有身份和隐私保护的身份验证和加密）

使用适当的访问权限配置 ONTAP 导出策略规则。例如，如果群集将使用 Kerberos 5i 和 Kerberos 5p 加密的混合方式装载 NFS 卷，请使用以下访问设置：

类型	只读访问	读/写访问权限	超级用户访问
UNIX	已启用	已启用	已启用
Kerberos 5i	已启用	已启用	已启用
Kerberos 5p	已启用	已启用	已启用

有关如何创建 ONTAP 导出策略和导出策略规则，请参阅以下文档：

- "创建导出策略"
- "将规则添加到导出策略"

创建存储后端

您可以创建包含 Kerberos 加密功能的 Trident 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，可以使用 `spec.nfsMountOptions` 参数指定三个不同版本的 Kerberos 加密之一：

- `spec.nfsMountOptions: sec=krb5` （身份验证和加密）
- `spec.nfsMountOptions: sec=krb5i`（具有身份保护的身份验证和加密）
- `spec.nfsMountOptions: sec=krb5p` （具有身份和隐私保护的身份验证和加密）

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，则仅使用第一个选项。

步骤

1. 在托管群集上，使用以下示例创建存储后端配置文件。使用环境中的信息替换括号 <> 中的值：

```
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，则后端配置有问题。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs
```

在识别并更正配置文件的问题后，您可以再次运行 create 命令。

创建存储类

您可以创建存储类，以使用 Kerberos 加密配置卷。

关于此任务

在创建存储类对象时，可以使用 `mountOptions` 参数指定 Kerberos 加密的三个不同版本之一：

- `mountOptions: sec=krb5` （身份验证和加密）
- `mountOptions: sec=krb5i`（具有身份保护的身份验证和加密）
- `mountOptions: sec=krb5p` （具有身份和隐私保护的身份验证和加密）

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，则仅使用第一个选项。如果在存储后端配置中指定的加密级别与在存储类对象中指定的级别不同，则存储类对象优先。

步骤

1. 使用以下示例创建 StorageClass Kubernetes 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
```

此时将显示与以下内容类似的输出：

```
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，现在可以配置卷。有关说明，请参阅 "预配卷"。

使用 **Azure NetApp Files** 卷配置运行中的 **Kerberos** 加密

您可以在托管集群与单个 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上启用 Kerberos 加密。

开始之前

- 请确保已在托管红帽 OpenShift 群集上启用 Trident。
- 请确保您拥有该 `tridentctl` 实用程序的访问权限。
- 通过注意要求并按照 "Azure NetApp Files 文档"中的说明，确保已为 Kerberos 加密准备 Azure NetApp Files 存储后端。
- 请确保您使用 Kerberos 加密的任何 NFSv4 卷配置正确。请参阅 NetApp NFSv4 域配置部分（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

当您创建配置 Kerberos 加密的存储后端配置文件时，您可以将其定义为应在以下两种可能的级别之一应用：

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

在虚拟池级别定义配置时，将使用存储类中的标签选择池。

在任一级别，您都可以指定 Kerberos 加密的三个不同版本之一：

- `kerberos: sec=krb5` （身份验证和加密）
- `kerberos: sec=krb5i`（具有身份保护的身份验证和加密）
- `kerberos: sec=krb5p` （具有身份和隐私保护的身份验证和加密）

步骤

1. 在托管群集上，使用以下示例之一创建存储后端配置文件，具体取决于需要定义存储后端的位置（存储后端级别或虚拟池级别）。使用环境中的信息替换括号 <> 中的值：

存储后端级别示例

```
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
```

虚拟池级别示例

```
---
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. 使用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，则后端配置有问题。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs
```

在识别并更正配置文件的问题后，您可以再次运行 create 命令。

创建存储类

您可以创建存储类，以使用 Kerberos 加密配置卷。

步骤

1. 使用以下示例创建 StorageClass Kubernetes 对象：

```
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. 创建存储类：

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

3. 确保已创建存储类：

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

此时将显示与以下内容类似的输出：

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

配置卷

创建存储后端和存储类后，现在可以配置卷。有关说明，请参阅 "预配卷"。