安全性
Trident

NetApp
January 15, 2026

目录

安全性

安全性

请按照此处列出的建议，确保您的Trident安装安全可靠。

在Trident自身的命名空间中运行它

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 非常重要，以确保可靠的存储并阻止潜在的恶意活动。

为了将其他应用程序和用户与Trident隔离，请始终将Trident安装在其自身的 Kubernetes 命名空间中。(`trident`）。将Trident放在自己的命名空间中，可以确保只有 Kubernetes 管理员才能访问Trident pod 和存储在命名空间 CRD 对象中的工件（例如后端和 CHAP 密钥，如果适用）。您应该确保只有管理员才能访问Trident命名空间，从而获得对以下方面的访问权限： `tridentctl`应用。

使用 CHAP 身份验证与ONTAP SAN 后端配合使用

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`司机）。NetApp建议使用Trident的双向 CHAP 进行主机与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可以通过以下方式设置双向 CHAP 并管理 CHAP 用户名和密钥： `tridentctl`。请参阅"准备配置后端ONTAP SAN 驱动程序"了解Trident如何在ONTAP后端配置CHAP。

使用 CHAP 身份验证连接NetApp HCI和SolidFire后端

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使用一个包含每个租户两个 CHAP 密码的秘密对象。安装Trident后，它会管理 CHAP 密钥并将其存储在一个位置。`tridentvolume`相应 PV 的 CR 对象。创建 PV 时， Trident使用 CHAP 密钥启动 iSCSI 会话，并通过 CHAP 与NetApp HCI和SolidFire系统通信。

Trident创建的卷不与任何卷访问组关联。

将Trident与 NVE 和 NAE 结合使用

NetApp ONTAP提供静态数据加密，以保护敏感数据，防止磁盘被盗、退回或重新利用。有关详细信息，请参阅"配置NetApp卷加密概述"。

- 如果后端启用了 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 `""`创建支持 NAE 的卷。
- 如果后端未启用 NAE，则在Trident中配置的任何卷都将启用 NVE，除非 NVE 加密标志设置为 `false`（后端配置中的默认值）。

在启用 NAE 的后端上使用Trident创建的卷必须使用 NVE 或 NAE 加密。

- 您可以将 NVE 加密标志设置为 `true`在Trident后端配置中，可以覆盖 NAE 加密，并按卷使用特定的加密密钥。
- 将 NVE 加密标志设置为 `false`在启用 NAE 的后端上创建启用 NAE 的卷。您无法通过将 NVE 加密标志设置为 来禁用 NAE 加密。 `false`。

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

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 统一密钥设置 (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 加密卷使用 aes-xts-plain64 密码和模式，这是推荐的。"美国国家标准与技术研究院" 。

ASA r2 系统不支持 LUKS 加密。有关ASA r2 系统的信息，请参阅"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）。欲了解更多信息，请访问"GitLab：加密设置"。
- 出于性能方面的考虑， NetApp建议工作节点支持高级加密标准新指令 (AES-NI)。要验证是否支持 AES-NI，请运行以下命令：

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

如果没有返回任何内容，则说明您的处理器不支持 AES-NI。有关 AES-NI 的更多信息，请访问："英特尔：高级加密标准指令集（AES-NI）"。

启用 LUKS 加密

您可以使用 Linux 统一密钥设置 (LUKS) 为ONTAP SAN 和ONTAP SAN ECONOMY 卷启用按卷主机端加密。

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`使用 LUKS 加密定义存储池。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 加密卷无法利用ONTAP重复数据删除和压缩功能。

导入 LUKS 卷的后端配置

要导入 LUKS 卷，您必须设置 `luksEncryption`到（`true`在后端。这 `luksEncryption`此选项用于告知Trident该卷是否符合 LUKS 标准。（`true`或不符合 LUKS 标准（`false`）如下例所示。

```
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，请设置注释 `trident.netapp.io/luksEncryption`到 `true`并在 PVC 中包含一个支持 LUKS 的存储类，如本例所示。

```
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。

> 在您确认任何卷、快照或密钥不再引用该密码短语之前，请勿忘记该密码短语。如果引用的密码短语丢失，您可能无法挂载卷，数据将保持加密状态且无法访问。

关于此任务

当指定新的 LUKS 密码短语后创建挂载卷的 pod 时，就会发生 LUKS 密码短语轮换。当创建一个新的 pod 时，Trident会将卷上的 LUKS 密码短语与密钥中的活动密码短语进行比较。

- 如果卷上的密码短语与密钥中的活动密码短语不匹配，则会发生轮换。
- 如果卷上的密码短语与密钥中的活动密码短语匹配，则 `previous-luks-passphrase`参数被忽略。

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. 识别卷或快照上已存在的密码短语。

   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短语。确保 `previous-luke-passphrase-name`和 `previous-luks-passphrase`与之前的密码短语匹配。

   ```
   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. 创建一个新的 pod，挂载该卷。这是启动旋转所必需的步骤。
5. 确认密码短语已轮换。

   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

当卷和快照上仅返回新密码短语时，密码短语就会轮换。

> 例如，如果返回两个密码短语。 `luksPassphraseNames: ["B", "A"]`旋转不完整。您可以触发一个新的舱体来尝试完成轮换。

启用卷扩展

您可以对 LUKS 加密卷启用卷扩展。

步骤

1. 启用 `CSINodeExpandSecret`功能门（beta 1.25+）。参考 "Kubernetes 1.25：使用密钥实现 CSI 卷的节点驱动扩展" 了解详情。
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 飞行中加密

使用 Kerberos 传输中加密，您可以对托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。

Trident支持以ONTAP作为存储后端的 Kerberos 加密：

- 本地**ONTAP** - Trident支持通过 NFSv3 和 NFSv4 连接对来自 Red Hat OpenShift 和上游 Kubernetes 集群的本地ONTAP卷进行 Kerberos 加密。

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

配置本地**ONTAP**卷的飞行中 **Kerberos** 加密

您可以为托管集群和本地ONTAP存储后端之间的存储流量启用 Kerberos 加密。

仅使用以下方式支持对本地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

- 确保您可以访问 `tridentctl`公用事业。
- 请确保您拥有ONTAP存储后端的管理员权限。
- 请确保您知道要从ONTAP存储后端共享的卷的名称。
- 请确保您已准备好ONTAP存储 VM，以支持 NFS 卷的 Kerberos 加密。请参阅 "在数据 LIF 上启用 Kerberos"以获取说明。
- 请确保所有与 Kerberos 加密一起使用的 NFSv4 卷都已正确配置。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，或者创建新的导出策略，以支持对ONTAP存储 VM 根卷以及与上游 Kubernetes 集群共享的任何ONTAP卷进行 Kerberos 加密。您添加的导出策略规则或您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

配置导出策略，支持 NFS、NFSv3 和 NFSv4 访问协议。

访问详情

您可以根据卷的需求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** - （身份验证和加密，具有身份保护功能）
- **Kerberos 5p** - （身份验证和加密，提供身份和隐私保护）

配置ONTAP导出策略规则，使其具有适当的访问权限。例如，如果集群将使用 Kerberos 5i 和 Kerberos 5p 混合加密方式挂载 NFS 卷，请使用以下访问设置：

类型	只读访问权限	读/写权限	超级用户访问权限
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"
- "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Trident存储后端配置。

关于此任务

创建配置 Kerberos 加密的存储后端配置文件时，可以使用以下方式指定三种不同版本的 Kerberos 加密之一：`spec.nfsMountOptions`范围：

- `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）

只能指定一个 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. 在托管集群上，使用以下示例创建存储后端配置文件。将方括号 <> 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. 使用上一步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，则后端配置存在问题。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs
```

在您发现并纠正配置文件中的问题后，您可以再次运行创建命令。

创建存储类

您可以创建存储类来配置具有 Kerberos 加密的卷。

关于此任务

创建存储类对象时，可以使用以下方式指定三种不同版本的 Kerberos 加密之一： `mountOptions`范围：

- `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）

只能指定一个 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加密级别不同，则以存储类对象为准。

步骤

1. 创建一个 StorageClass Kubernetes 对象，请参考以下示例：

```
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. 请确保已创建存储类：

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

您应该会看到类似以下内容的输出：

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

供应量

创建存储后端和存储类之后，现在可以配置卷了。有关说明，请参阅 "提供一定量"。

使用Azure NetApp Files卷配置飞行中 Kerberos 加密

您可以为托管群集与单个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

- 请确保已在受管 Red Hat OpenShift 集群上启用Trident 。
- 确保您可以访问 `tridentctl`公用事业。
- 请确保您已按照以下说明准备好Azure NetApp Files存储后端以进行 Kerberos 加密：注意相关要求并按照说明进行操作。 "Azure NetApp Files文档" 。
- 请确保所有与 Kerberos 加密一起使用的 NFSv4 卷都已正确配置。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

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用于以下两个级别之一：

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

在虚拟池级别定义配置时，使用存储类中的标签选择池。

无论在哪个级别，您都可以指定三种不同版本的 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. 在托管集群上，根据您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。将方括号 <> 中的值替换为您环境中的信息：

存储后端示例

```
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，则后端配置存在问题。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs
```

在您发现并纠正配置文件中的问题后，您可以再次运行创建命令。

创建存储类

您可以创建存储类来配置具有 Kerberos 加密的卷。

步骤

1. 创建一个 StorageClass Kubernetes 对象，请参考以下示例：

```
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. 请确保已创建存储类：

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

您应该会看到类似以下内容的输出：

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

供应量

创建存储后端和存储类之后，现在可以配置卷了。有关说明，请参阅 "提供一定量"。