最佳实践和建议
Astra Trident

NetApp
April 04, 2024

目录

最佳实践和建议

部署

在部署 Astra Trident 时，请使用此处列出的建议。

部署到专用命名空间

"命名空间" 在不同应用程序之间实现管理隔离，是资源共享的障碍。例如，一个命名空间中的 PVC 不能从另一个命名空间中使用。Astra Trident 为 Kubernetes 集群中的所有命名空间提供 PV 资源，从而利用具有提升权限的服务帐户。

此外，访问 Trident POD 可能会使用户能够访问存储系统凭据和其他敏感信息。 请务必确保应用程序用户和管理应用程序无法访问 Trident 对象定义或 Pod 本身。

使用配额和范围限制来控制存储消耗

Kubernetes 具有两项功能，这些功能结合使用后，可提供一种功能强大的机制来限制应用程序的资源消耗。 。"存储配额机制" 使管理员能够在每个命名空间基础上实施全局容量和对象计数消耗限制以及特定于存储类的限制。此外，使用 "范围限制" 确保在将 PVC 请求转发给配置程序之前，该请求同时处于最小值和最大值范围内。

这些值是按命名空间定义的，这意味着每个命名空间都应定义符合其资源要求的值。有关信息，请参见此处 "如何利用配额"。

存储配置

NetApp产品组合中的每个存储平台都具有独特的功能、无论应用程序是容器化还是非容器化、都能从中受益。

平台概述

Trident 可与 ONTAP 和 Element 结合使用。没有一个平台比另一个平台更适合所有应用程序和场景，但是，在选择平台时，应考虑应用程序和设备管理团队的需求。

您应遵循使用所使用协议的主机操作系统的基线最佳实践。或者，您也可以考虑将应用程序最佳实践（如果有）与后端，存储类和 PVC 设置结合使用，以便为特定应用程序优化存储。

ONTAP 和 Cloud Volumes ONTAP 最佳实践

了解为 Trident 配置 ONTAP 和 Cloud Volumes ONTAP 的最佳实践。

以下建议是为容器化工作负载配置 ONTAP 的准则，容器化工作负载会占用 Trident 动态配置的卷。应考虑并评估每个问题在您的环境中的适用性。

使用专用于 **Trident** 的 **SVM**

Storage Virtual Machine （ SVM ）可在 ONTAP 系统上的租户之间实现隔离和管理隔离。 通过将 SVM 专用于应用程序，可以委派特权并应用最佳实践来限制资源消耗。

可通过多种方法管理 SVM：

- 在后端配置中提供集群管理接口以及相应的凭据，并指定 SVM 名称。
- 使用 ONTAP 系统管理器或命令行界面为 SVM 创建专用管理接口。
- 与 NFS 数据接口共享管理角色。

在每种情况下，接口都应位于 DNS 中，配置 Trident 时应使用 DNS 名称。这有助于在不使用网络身份保留的情况下实施某些灾难恢复方案，例如 SVM-DR 。

在为 SVM 配置专用管理 LIF 或共享管理 LIF 之间没有任何偏好，但是，您应确保网络安全策略与您选择的方法一致。无论如何，管理 LIF 应可通过 DNS 访问，以实现最大的灵活性 "SVM-DR" 与 Trident 结合使用。

限制最大卷数

ONTAP 存储系统具有最大卷数，具体取决于软件版本和硬件平台。请参见 "NetApp Hardware Universe" 以 ONTAP 确定确切限制。当卷计数用尽时，配置操作不仅会对 Trident 失败，而且会对所有存储请求失败。

Trident的 `ontap-nas` 和 `ontap-san` 驱动程序为创建的每个Kubernetes永久性卷(PV)配置一个FlexVolume。。 `ontap-nas-economy` 驱动程序会为每200个PV创建大约一个FlexVolume (可在50到300之间配置)。。 `ontap-san-economy` 驱动程序大约为每100个PIV创建一个FlexVolume (可在50到200之间配置)。要防止 Trident 占用存储系统上的所有可用卷，您应对 SVM 设置限制。您可以从命令行执行此操作：

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

的值 `max-volumes` 根据您的环境特定的多个条件而有所不同：

- ONTAP 集群中现有卷的数量
- 希望在 Trident 之外为其他应用程序配置的卷数
- Kubernetes 应用程序预期占用的永久性卷数

。 `max-volumes` 值是在ONTAP 集群中的所有节点上配置的总卷数、而不是在单个ONTAP 节点上配置的总卷数。因此，在某些情况下， ONTAP 集群节点所配置的 Trident 卷可能远远多于或少于其他节点。

例如，一个双节点 ONTAP 集群最多可以托管 2000 个 FlexVolume 。将最大卷数设置为 1250 似乎非常合理。但是，如果只是 "聚合" 从一个节点分配给 SVM ，或者从一个节点分配的聚合无法配置（例如，由于容量），则另一个节点将成为所有 Trident 配置卷的目标。这意味着、可能会在之前达到该节点的卷限制 `max-volumes` 达到值后、会同时影响使用该节点的Trident和其他卷操作。* 您可以通过确保将集群中每个节点的聚合分配给 Trident 使用的 SVM 来避免这种情况。 *

限制 **Trident** 创建的卷的最大大小

要为Trident可以创建的卷配置最大大小、请使用 `limitVolumeSize` 中的参数 `backend.json` 定义。

除了控制存储阵列上的卷大小之外，您还应利用 Kubernetes 功能。

配置 **Trident** 以使用双向 **CHAP**

您可以在后端定义中指定 CHAP 启动程序以及目标用户名和密码，并在 SVM 上启用 Trident CHAP 。使用

`useCHAP` 参数在后端配置中、Trident使用CHAP对ONTAP 后端的iSCSI连接进行身份验证。Trident 20.04 及更高版本支持双向 CHAP 。

创建并使用 **SVM QoS** 策略

利用应用于 SVM 的 ONTAP QoS 策略，限制 Trident 配置的卷可使用的 IOPS 数量。 这有助于实现 "防止抢占资源" 或控制不足的容器，以使其不会影响 Trident SVM 以外的工作负载。

您可以通过几个步骤为 SVM 创建 QoS 策略。有关最准确的信息，请参见适用于您的 ONTAP 版本的文档。 以下示例将创建一个 QoS 策略，将 SVM 可用的总 IOPS 限制为 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>
```

此外，如果您的 ONTAP 版本支持此功能，则可以考虑使用最低 QoS 来保证容器化工作负载的吞吐量。自适应 QoS 与 SVM 级别策略不兼容。

专用于容器化工作负载的 IOPS 数量取决于许多方面。其中包括：

- 使用存储阵列的其他工作负载。如果存在与 Kubernetes 部署无关的其他工作负载，则应注意利用存储资源，以确保这些工作负载不会意外受到不利影响。
- 容器中运行的预期工作负载。如果 IOPS 要求较高的工作负载将在容器中运行，则 QoS 策略较低会导致出现不良体验。

请务必记住，在 SVM 级别分配的 QoS 策略会导致配置到 SVM 的所有卷共享同一个 IOPS 池。如果一个或少量容器化应用程序的 IOPS 要求较高，则可能会成为其他容器化工作负载的抢占资源的应用程序。如果是这种情况，您可能需要考虑使用外部自动化来分配每个卷的 QoS 策略。

如果 ONTAP 版本早于 9.8 ，则应将此 QoS 策略组分配给 SVM * 仅 * 。

为 **Trident** 创建 **QoS** 策略组

服务质量（ QoS ）可确保关键工作负载的性能不会因争用工作负载而降级。ONTAP QoS 策略组为卷提供 QoS 选项，并使用户能够为一个或多个工作负载定义吞吐量上限。有关 QoS 的详细信息，请参见 "通过 QoS 保证吞吐量"。
您可以在后端或存储池中指定 QoS 策略组，这些策略组将应用于该池或后端创建的每个卷。

ONTAP 有两种类型的 QoS 策略组：传统和自适应。传统策略组以 IOPS 为单位提供固定的最大（或最小）吞吐量。自适应 QoS 会根据工作负载大小自动扩展吞吐量，并在工作负载大小发生变化时保持 IOPS 与 TBSGB 的比率。如果您要在大型部署中管理数百或数千个工作负载，则这将带来显著优势。

创建 QoS 策略组时，请考虑以下事项：

- 您应设置 `qosPolicy` 输入 `defaults` 后端配置的块。请参见以下后端配置示例：

```
  ---
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
```

- 您应该对每个卷应用策略组，以便每个卷都获得策略组指定的整个吞吐量。不支持共享策略组。

有关 QoS 策略组的详细信息，请参见 "ONTAP 9.8 QoS 命令"。

将存储资源访问限制为 **Kubernetes** 集群成员

限制对 Trident 创建的 NFS 卷和 iSCSI LUN 的访问是 Kubernetes 部署安全状况的重要组成部分。这样可以防止不属于 Kubernetes 集群的主机访问卷并可能意外修改数据。

请务必了解命名空间是 Kubernetes 中资源的逻辑边界。假设同一命名空间中的资源可以共享，但重要的是，没有跨命名空间功能。这意味着，即使 PV 是全局对象，但在绑定到 PVC 时，它们只能由同一命名空间中的 Pod 访问。* 请务必确保使用命名空间在适当时提供分隔。 *

大多数组织在 Kubernetes 环境中的数据安全方面的主要顾虑是，容器中的进程可以访问挂载到主机但不适用于容器的存储。 "命名空间" 旨在防止这种类型的损害。 但是，存在一个例外：特权容器。

有权限的容器是指运行时拥有比正常情况更多主机级别权限的容器。默认情况下，这些选项不会被拒绝，因此请确保使用禁用此功能 "POD 安全策略"。

对于需要从 Kubernetes 和外部主机访问的卷，应采用传统方式管理存储，并由管理员引入 PV ，而不是由 Trident 管理。这样可以确保只有在 Kubernetes 和外部主机断开连接且不再使用此卷时，才会销毁此存储卷。此外，还可以应用自定义导出策略，以便从 Kubernetes 集群节点和 Kubernetes 集群以外的目标服务器进行访问。

对于具有专用基础架构节点(例如OpenShift)或其他无法计划用户应用程序的节点的部署、应使用单独的导出策略进一步限制对存储资源的访问。其中包括为部署到这些基础架构节点的服务（例如 OpenShift 指标和日志记录服务）以及部署到非基础架构节点的标准应用程序创建导出策略。

使用专用导出策略

您应确保每个后端都有一个导出策略，该策略仅允许访问 Kubernetes 集群中的节点。从 20.04 版开始， Trident 可以自动创建和管理导出策略。通过这种方式， Trident 会限制对其配置给 Kubernetes 集群中节点的卷的访问，并简化节点的添加 / 删除。

或者，您也可以手动创建导出策略，并使用一个或多个导出规则来填充此策略，这些导出规则用于处理每个节点访问请求：

- 使用 `vserver export-policy create` 用于创建导出策略的ONTAP 命令行界面命令。
- 使用向导出策略添加规则 `vserver export-policy rule create` ONTAP 命令行界面命令。

通过运行这些命令，您可以限制哪些 Kubernetes 节点可以访问数据。

禁用 `showmount` 用于应用程序**SVM**

。 `showmount` 通过功能、NFS客户端可以向SVM查询可用NFS导出列表。部署到Kubernetes集群的POD可以对进行问题描述 `showmount -e` 对数据LIF执行命令并接收可用挂载列表、包括其无权访问的挂载。虽然这本身并不会影响安全，但它确实会提供不必要的信息，可能有助于未经授权的用户连接到 NFS 导出。

您应禁用 `showmount` 使用SVM级别的ONTAP 命令行界面命令：

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

SolidFire 最佳实践

了解为 Trident 配置 SolidFire 存储的最佳实践。

创建 **SolidFire** 帐户

每个 SolidFire 帐户都代表一个唯一的卷所有者，并接收自己的一组质询握手身份验证协议（ Challenge-Handshake Authentication Protocol ， CHAP ）凭据。您可以使用帐户名称和相对 CHAP 凭据或通过卷访问组访问分配给帐户的卷。一个帐户最多可以分配 2 ， 000 个卷，但一个卷只能属于一个帐户。

创建 **QoS** 策略

如果要创建并保存可应用于多个卷的标准化服务质量设置，请使用 SolidFire 服务质量（ QoS ）策略。

您可以按卷设置 QoS 参数。通过设置三个可配置的参数来定义 QoS ，可以确保每个卷的性能：最小 IOPS ，最大 IOPS 和突发 IOPS 。

以下是 4 KB 块大小的可能最小，最大和突发 IOPS 值。

IOPS参数	定义	最小**value**	默认值	最大值（ **4 KB** ）
最小 IOPS	卷的性能保障级别。	50.	50.	15000
最大 IOPS	性能不会超过此限制。	50.	15000	200,000

IOPS参数	定义	最小value	默认值	最大值（4 KB）
突发 IOPS	在短时突发情形下允许的最大 IOPS 。	50.	15000	200,000

虽然最大 IOPS 和突发 IOPS 可设置为高达 200 ， 000 ，但卷的实际最大性能受集群使用情况和每节点性能的限制。

块大小和带宽会直接影响 IOPS 数量。随着块大小的增加，系统会将带宽增加到处理较大块大小所需的级别。随着带宽的增加，系统能够达到的 IOPS 数量也会减少。请参见 "SolidFire 服务质量" 有关 QoS 和性能的详细信息。

SolidFire 身份验证

Element 支持两种身份验证方法： CHAP 和卷访问组（ VAG ）。CHAP 使用 CHAP 协议向后端对主机进行身份验证。卷访问组控制对其配置的卷的访问。NetApp 建议使用 CHAP 进行身份验证，因为它更简单，并且没有扩展限制。

具有增强型 CSI 配置程序的 Trident 支持使用 CHAP 身份验证。VAG 只能在传统的非 CSI 操作模式下使用。

只有基于帐户的访问控制才支持 CHAP 身份验证（验证启动程序是否为目标卷用户）。如果使用 CHAP 进行身份验证，则可以使用两个选项：单向 CHAP 和双向 CHAP 。单向 CHAP 使用 SolidFire 帐户名称和启动程序密钥对卷访问进行身份验证。双向 CHAP 选项可提供最安全的卷身份验证方法，因为卷会通过帐户名称和启动程序密钥对主机进行身份验证，然后主机通过帐户名称和目标密钥对卷进行身份验证。

但是，如果无法启用 CHAP 且需要使用 VAG ，请创建访问组并将主机启动程序和卷添加到此访问组。添加到访问组的每个 IQN 都可以使用或不使用 CHAP 身份验证访问组中的每个卷。如果将 iSCSI 启动程序配置为使用 CHAP 身份验证，则会使用基于帐户的访问控制。如果 iSCSI 启动程序未配置为使用 CHAP 身份验证，则会使用卷访问组访问控制。

如何查找更多信息

下面列出了一些最佳实践文档。搜索 "NetApp 库" 对于最新版本。

- ONTAP *
- "NFS 最佳实践和实施指南"
- "《 SAN 管理指南》" （对于 iSCSI ）
- "适用于 RHEL 的 iSCSI 快速配置"
- Element 软件 *
- "配置适用于 Linux 的 SolidFire"
- NetApp HCI *
- "NetApp HCI 部署前提条件"
- "访问 NetApp 部署引擎"
- 应用程序最佳实践信息 *

- "基于 ONTAP 的 MySQL 最佳实践"
- "基于 SolidFire 的 MySQL 最佳实践"
- "NetApp SolidFire 和 Cassandra"
- "SolidFire 上的 Oracle 最佳实践"
- "SolidFire 上的 PostgreSQL 最佳实践"

并非所有应用程序都有特定的准则，与您的 NetApp 团队合作并使用非常重要 "NetApp 库" 以查找最新文档。

集成 Astra Trident

要集成Astra Trident、需要集成以下设计和架构要素：驱动程序选择和部署、存储类设计、虚拟池设计、永久性卷声明(PVC)对存储配置的影响、卷操作以及使用Astra Trident部署OpenShift服务。

驱动程序选择和部署

为存储系统选择并部署后端驱动程序。

ONTAP 后端驱动程序

ONTAP 后端驱动程序可通过所使用的协议以及在存储系统上配置卷的方式来区分。因此、在确定要部署的驱动程序时、请仔细考虑。

更高级别的是，如果您的应用程序中的组件需要共享存储（多个 Pod 访问同一个 PVC ），则基于 NAS 的驱动程序将成为默认选项，而基于块的 iSCSI 驱动程序则可满足非共享存储的需求。根据应用程序要求以及存储和基础架构团队的舒适程度选择协议。一般来说，对于大多数应用程序来说，它们之间没有什么区别，因此通常是根据是否需要共享存储（多个 POD 需要同时访问）来决定的。

可用的ONTAP 后端驱动程序包括：

- `ontap-nas`：配置的每个PV都是一个完整的ONTAP FlexVolume。
- `ontap-nas-economy`：配置的每个PV都是一个qtree、每个FlexVolume具有可配置的qtree数量(默认值为200)。
- `ontap-nas-flexgroup`：使用配置为完整ONTAP FlexGroup 的每个PV以及分配给SVM的所有聚合。
- `ontap-san`：配置的每个PV都是其自身FlexVolume中的一个LUN。
- `ontap-san-economy`：配置的每个PV都是一个LUN、每个FlexVolume具有可配置的LUN数量(默认值为100)。

在三个 NAS 驱动程序之间进行选择会对应用程序可用的功能产生一些影响。

请注意，在下表中，并非所有功能都通过 Astra Trident 公开。如果需要某些功能，存储管理员必须在配置后应用这些功能。上标脚注区分了每个功能和驱动程序的功能。

ONTAP NAS 驱动程序	快照	克隆	动态导出策略	多连接	**QoS**	调整大小	**Replication**
`ontap-nas`	是的。	是的。	是脚注：5[]	是的。	是脚注：1[]	是的。	是脚注：1[]
`ontap-nas-economy`	是脚注：3[]	是脚注：3[]	是脚注：5[]	是的。	是脚注：3[]	是的。	是脚注：3[]
`ontap-nas-flexgroup`	是脚注：1[]	否	是脚注：5[]	是的。	是脚注：1[]	是的。	是脚注：1[]

Astra Trident 为 ONTAP 提供了 2 个 SAN 驱动程序，其功能如下所示。

ONTAP SAN驱动程序	快照	克隆	多连接	双向 **CHAP**	**QoS**	调整大小	**Replication**
`ontap-san`	是的。	是的。	是脚注：4[]	是的。	是脚注：1[]	是的。	是脚注：1[]
`ontap-san-economy`	是的。	是的。	是脚注：4[]	是的。	是脚注：3[]	是的。	是脚注：3[]

> 上述表格的脚注：
>
> 是脚注：1[]：不由Astra Trident管理
>
> 是脚注：2[]：由Astra Trident管理，但不是PV粒度
>
> 是脚注：3[]：不由Astra Trident管理，也不是PV粒度
>
> 是脚注：4[]：支持原始块卷
>
> 是脚注：5[]：由CSI三端到支持

非 PV 粒度功能将应用于整个 FlexVolume ，而所有 PV （即共享 FlexVol 中的 qtree 或 LUN ）将共享一个通用计划。

如上表所示、之间的大部分功能 `ontap-nas` 和 `ontap-nas-economy` 相同。但是、因为 `ontap-nas-economy` 驱动程序限制了按PV粒度控制计划的能力、这尤其会影响灾难恢复和备份规划。对于希望在ONTAP存储上使用PVC克隆功能的开发团队、只有在使用时才可能实现这一点 `ontap-nas`， `ontap-san` 或 `ontap-san-economy` 驱动程序。

。 `solidfire-san` 驱动程序还可以克隆PVC。

Cloud Volumes ONTAP 后端驱动程序

Cloud Volumes ONTAP 可为各种使用情形提供数据控制以及企业级存储功能，包括文件共享和为 NAS 和 SAN 协议（ NFS ， SMB/CIFS 和 iSCSI ）提供服务的块级存储。Cloud Volume ONTAP 的兼容驱动程序包括 `ontap-nas`， `ontap-nas-economy`， `ontap-san` 和 `ontap-san-economy`。它们适用于适用于 Azure 的

Cloud Volume ONTAP ，适用于 GCP 的 Cloud Volume ONTAP 。

适用于**ONTAP** 的**Amazon FSX**后端驱动程序

借助适用于 ONTAP 的 Amazon FSX ，客户可以利用他们熟悉的 NetApp 功能，性能和管理功能，同时利用在 AWS 上存储数据的简便性，灵活性，安全性和可扩展性。FSX for ONTAP 支持 ONTAP 的许多文件系统功能和管理 API 。Cloud Volume ONTAP 的兼容驱动程序包括 `ontap-nas`， `ontap-nas-economy`， `ontap-nas-flexgroup`， `ontap-san` 和 `ontap-san-economy`。

NetApp HCI/SolidFire后端驱动程序

。 `solidfire-san` 与NetApp HCI/SolidFire平台结合使用的驱动程序可帮助管理员根据QoS限制为Trident配置Element后端。如果您希望设计后端、以便为Trident配置的卷设置特定的QoS限制、请使用 `type` 参数。管理员还可以使用限制在存储上创建的卷大小 `limitVolumeSize` 参数。目前、不支持通过实现卷大小调整和卷复制等Element存储功能 `solidfire-san` 驱动程序。这些操作应通过 Element Software Web UI 手动完成。

SolidFire 驱动程序	快照	克隆	多连接	**CHAP**	**QoS**	调整大小	**Replication**
`solidfire-san`	是的。	是的。	是脚注：2[]	是的。	是的。	是的。	是脚注：1[]

> 脚注：
>
> 是脚注：1[]：不由Astra Trident管理
>
> 是脚注：2[]：支持原始块卷

Azure NetApp Files 后端驱动程序

Astra Trident使用 `azure-netapp-files` 用于管理的驱动程序 "Azure NetApp Files" 服务

有关此驱动程序及其配置方法的详细信息，请参见 "适用于 Azure NetApp Files 的 Astra Trident 后端配置"。

Azure NetApp Files 驱动程序	快照	克隆	多连接	**QoS**	展开	**Replication**
`azure-netapp-files`	是的。	是的。	是的。	是的。	是的。	是脚注：1[]

> 脚注：
>
> 是脚注：1[]：不由Astra Trident管理

Google Cloud上的**Cloud Volumes Service** 后端驱动程序

Astra Trident使用 `gcp-cvs` 用于链接到Google Cloud上的Cloud Volumes Service 的驱动程序。

。 `gcp-cvs` 驱动程序使用虚拟池抽象化后端、并允许Astra Trident确定卷的放置位置。管理员在中定义虚拟池 `backend.json` 文件。存储类使用选择器按标签标识虚拟池。

- 如果在后端定义了虚拟池、则Astra Trident将尝试在这些虚拟池所受限制的Google Cloud存储池中创建卷。
- 如果未在后端定义虚拟池、则Astra Trident将从该区域的可用存储池中选择Google Cloud存储池。

要在Astra Trident上配置Google Cloud后端、必须指定 `projectNumber`， `apiRegion`，和 `apiKey` 在后端文件中。您可以在Google Cloud控制台中找到项目编号。API密钥来自您在Google Cloud上为Cloud Volumes Service 设置API访问时创建的服务帐户专用密钥文件。

有关Google Cloud上的Cloud Volumes Service 服务类型和服务级别的详细信息、请参见 "了解适用于GCP的CVS的Astra Trident支持"。

适用于**Google Cloud**的**Cloud Volumes Service** 驱动程序	快照	克隆	多连接	**QoS**	展开	**Replication**
`gcp-cvs`	是的。	是的。	是的。	是的。	是的。	仅适用于CVS-Performance服务类型。

复制注释

- 复制不受Astra Trident管理。
- 克隆将在与源卷相同的存储池中创建。

存储类设计

要创建 Kubernetes 存储类对象，需要配置并应用各个存储类。本节讨论如何为您的应用程序设计存储类。

特定后端利用率

可以在特定存储类对象中使用筛选功能来确定要将哪个存储池或一组池与该特定存储类结合使用。可以在存储类中设置三组筛选器： `storagePools`， `additionalStoragePools`和/或 `excludeStoragePools`。

。 `storagePools` 参数有助于将存储限制为与任何指定属性匹配的一组池。。 `additionalStoragePools` 参数用于扩展Astra Trident用于配置的池集以及由属性和选择的池集 `storagePools` parameters您可以单独使用参数，也可以同时使用这两个参数，以确保选择适当的存储池集。

。 `excludeStoragePools` 参数用于明确排除列出的一组与属性匹配的池。

模拟**QoS**策略

如果要设计存储类以模拟服务质量策略、请使用创建存储类 `media` 属性为 `hdd` 或 `ssd`。基于 `media` 属性、Trident将选择提供服务的相应后端 `hdd` 或 `ssd` 聚合以匹配介质属性、然后将卷的配置定向到特定聚合。因此、我们可以创建存储类高级版 `media` 属性设置为 `ssd` 可归类为高级QoS策略。我们可以创建另一个存储类标准，该标准会将介质属性设置为 `HDD'，并可归类为标准 QoS 策略。我们还可以使用存储类中的 ``IOPS' 属性将配置重定向到可定义为 QoS 策略的 Element 设备。

根据特定功能使用后端

存储类可设计为在启用了精简和厚配置，快照，克隆和加密等功能的特定后端直接配置卷。要指定要使用的存储，请创建存储类，以指定启用了所需功能的相应后端。

虚拟池

所有Astra Trident后端均可使用虚拟池。您可以使用Astra Trident提供的任何驱动程序为任何后端定义虚拟池。

通过虚拟池、管理员可以在后端创建一个抽象级别、并可通过存储类进行引用、从而提高卷在后端的灵活性和效率。可以使用相同的服务类定义不同的后端。此外、可以在同一后端创建多个存储池、但其特征不同。如果为存储类配置了具有特定标签的选择器，则 Astra Trident 会选择与所有选择器标签匹配的后端来放置卷。如果存储类选择器标签与多个存储池匹配、则Astra Trident将选择其中一个存储池来配置卷。

虚拟池设计

创建后端时，通常可以指定一组参数。管理员无法使用相同的存储凭据和一组不同的参数创建另一个后端。随着虚拟池的推出、此问题描述 得以缓解。虚拟池是在后端和Kubernetes存储类之间引入的级别抽象、因此管理员可以定义参数以及标签、这些参数和标签可以通过Kubernetes存储类作为选择器进行引用、并且与后端无关。可以使用Astra Trident为所有受支持的NetApp后端定义虚拟池。该列表包括 SolidFire/NetApp HCI ， ONTAP ， GCP 上的 Cloud Volumes Service 以及 Azure NetApp Files 。

> 定义虚拟池时、建议不要尝试在后端定义中重新排列现有虚拟池的顺序。此外，建议不要编辑 / 修改现有虚拟池的属性，而是定义新的虚拟池。

模拟不同的服务级别/**QoS**

可以为模拟服务类设计虚拟池。使用适用于 Azure NetApp Files 的云卷服务的虚拟池实施，让我们来了解一下如何设置不同的服务类。为 ANF 后端配置多个标签，以表示不同的性能级别。设置 `servicelevel` 添加适当的性能级别、并在每个标签下添加其他所需的方面。现在、创建可映射到不同虚拟池的不同Kubernetes存储类。使用 `parameters.selector` 字段中、每个StorageClass都会调用可用于托管卷的虚拟池。

分配特定的方面

可以从一个存储后端设计具有一组特定方面的多个虚拟池。为此，请为后端配置多个标签，并在每个标签下设置所需的方面。现在、使用创建不同的Kubernetes存储类 `parameters.selector` 要映射到不同虚拟池的字段。在后端配置的卷将在选定虚拟池中定义相关方面。

影响存储配置的 **PVC** 特征

在创建PVC时、请求的存储类以外的某些参数可能会影响Astra Trident配置决策过程。

访问模式

通过 PVC 请求存储时，访问模式为必填字段之一。所需的模式可能会影响所选的托管存储请求的后端。

Astra Trident 将尝试与根据下表指定的访问方法所使用的存储协议匹配。这独立于底层存储平台。

	ReadWriteOnce	**ReadOnlyMany**	读取写入任何
iSCSI	是的。	是的。	是（原始块）
NFS	是的。	是的。	是的。

如果在未配置 NFS 后端的情况下向 Trident 部署提交了 ReadWriteMany PVC 请求，则不会配置任何卷。因此，请求者应使用适合其应用程序的访问模式。

卷操作

修改永久性卷

除了两个例外，永久性卷是 Kubernetes 中不可变的对象。创建后，可以修改回收策略和大小。但是，这并不会阻止在 Kubernetes 外部修改卷的某些方面。为了针对特定应用程序自定义卷，确保容量不会意外占用，或者出于任何原因将卷移动到其他存储控制器，这一点可能是理想的。

目前， Kubernetes 树中配置程序不支持对 NFS 或 iSCSI PV 执行卷大小调整操作。Astra Trident 支持扩展 NFS 和 iSCSI 卷。

创建 PV 后，无法修改其连接详细信息。

创建按需卷快照

Astra Trident 支持按需创建卷快照，并使用 CSI 框架从快照创建 PVC 。快照提供了一种维护数据时间点副本的便捷方法，并且生命周期独立于 Kubernetes 中的源 PV 。这些快照可用于克隆 PVC 。

从快照创建卷

Astra Trident 还支持从卷快照创建 PersistentVolumes 。为此、只需创建PersistentVolumeClaim并提及即可 `datasource` 作为需要从中创建卷的所需快照。Astra Trident 将通过创建包含快照上的数据的卷来处理此 PVC 。通过此功能，可以跨区域复制数据，创建测试环境，整体更换损坏或损坏的生产卷，或者检索特定文件和目录并将其传输到另一个连接的卷。

移动集群中的卷

存储管理员可以在 ONTAP 集群中的聚合和控制器之间无中断地将卷移动到存储使用者。此操作不会影响 Astra Trident 或 Kubernetes 集群，只要目标聚合是 Astra Trident 所使用的 SVM 有权访问的聚合即可。重要的是，如果已将聚合新添加到 SVM ，则需要通过将后端重新添加到 Astra Trident 来刷新后端。这将触发 Astra Trident 对 SVM 重新进行清单配置，以便识别新聚合。

但是， Astra Trident 不支持在后端之间自动移动卷。这包括在同一集群中的 SVM 之间，集群之间或不同存储平台上（即使该存储系统是连接到 Astra Trident 的存储系统也是如此）。

如果将卷复制到其他位置，则可以使用卷导入功能将当前卷导入到 Astra Trident 中。

展开卷

Astra Trident 支持调整 NFS 和 iSCSI PV 的大小。这样，用户就可以直接通过 Kubernetes 层调整其卷的大小。所有主要 NetApp 存储平台均可进行卷扩展，包括 ONTAP ， SolidFire/NetApp HCI 和 Cloud Volumes Service 后端。要允许稍后进行扩展、请设置 `allowVolumeExpansion` to `true` 在与卷关联的StorageClass中。每当需要调整持久性卷的大小时、请编辑 `spec.resources.requests.storage` 在永久性卷声明中为所需的卷大小添加标注。Trident会自动调整存储集群上卷的大小。

将现有卷导入到 **Kubernetes** 中

通过卷导入，可以将现有存储卷导入到 Kubernetes 环境中。目前、支持此功能 `ontap-nas`， `ontap-nas-flexgroup`， `solidfire-san`， `azure-netapp-files`，和 `gcp-cvs` 驱动程序。在将现有应用程序移植到 Kubernetes 或在灾难恢复场景中，此功能非常有用。

使用ONTAP 和时 `solidfire-san` 驱动程序、请使用命令 `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` 将现有卷导入到要由Astra Trident管理的Kubernetes中。导入卷命令中使用的 PVC YAML 或 JSON 文件指向将 Astra Trident 标识为配置程序的存储类。使用 NetApp HCI/SolidFire 后端时，请确保卷名称是唯一的。如果卷名称重复，请将卷克隆为唯一名称，以便卷导入功能可以区分它们。

如果 `azure-netapp-files` 或 `gcp-cvs` 使用驱动程序时、请使用命令 `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` 将卷导入到要由Astra Trident管理的Kubernetes中。这样可以确保卷引用是唯一的。

执行上述命令后， Astra Trident 将在后端找到卷并读取其大小。它将自动添加（并在必要时覆盖）已配置的 PVC 卷大小。然后， Astra Trident 会创建新的 PV ， Kubernetes 会将 PVC 绑定到 PV 。

如果部署的容器需要特定的导入 PVC ，则容器将保持待定状态，直到通过卷导入过程绑定 PVC/PV 对为止。在绑定 PVC/PV 对后，如果没有其他问题，应启动容器。

部署 OpenShift 服务

OpenShift 增值集群服务为集群管理员和要托管的应用程序提供了重要功能。这些服务使用的存储可以使用节点本地资源进行配置，但这通常会限制服务的容量，性能，可恢复性和可持续性。利用企业级存储阵列为这些服务提供容量可以显著改善服务，但是，与所有应用程序一样， OpenShift 和存储管理员应密切合作，为每个服务确定最佳选项。应大量利用 Red Hat 文档来确定要求并确保满足规模估算和性能需求。

注册表服务

有关为注册表部署和管理存储的文档，请参见 "netapp.io" 在中 "博客"。

日志记录服务

与其他 OpenShift 服务一样，日志记录服务也是使用清单文件（也称为）提供的配置参数 Ansible 部署的主机，提供给攻略手册。其中包括两种安装方法：在初始OpenShift安装期间部署日志记录、以及在OpenShift完成后部署日志记录
已安装。

自 Red Hat OpenShift 3.9 版开始，官方文档出于对数据损坏的担忧，建议不要对日志记录服务使用 NFS 。这是基于 Red Hat 对其产品的测试得出的。ONTAP 的 NFS 服务器不存在这些问题，可以轻松地备份日志记录部署。最终，您可以选择日志记录服务的协议，只需了解这两种协议在使用 NetApp 平台时都能很好地发挥作用，如果您愿意，也没有理由避免使用 NFS 。

如果选择将NFS与日志记录服务结合使用、则需要设置Ansible变量 `openshift_enable_unsupported_configurations` to `true` 以防止安装程序失败。

入门

可以选择为这两个应用程序以及 OpenShift 集群本身的核心操作部署日志记录服务。如果选择部署操作日志记录、请指定变量 `openshift_logging_use_ops` 作为 `true`、将创建两个服务实例。控制操作日志记录实例的变量包含 "ops" ，而应用程序实例则不包含 "ops" 。

要确保底层服务使用正确的存储，必须根据部署方法配置 Ansible 变量。让我们来了解一下每种部署方法的选项。

下表仅包含与存储配置相关的变量，因为这些变量与日志记录服务相关。您可以在中找到其他选项 "RedHat OpenShift 日志记录文档" 应根据您的部署情况查看，配置和使用。

下表中的变量将导致 Ansible 攻略手册使用提供的详细信息为日志记录服务创建 PV 和 PVC 。与在 OpenShift 安装后使用组件安装攻略手册相比，此方法的灵活性明显降低，但是，如果您有可用的现有卷，则可以选择此方法。

变量	详细信息
`openshift_logging_storage_kind`	设置为 `nfs` 让安装程序为日志记录服务创建NFS PV。
`openshift_logging_storage_host`	NFS 主机的主机名或 IP 地址。此值应设置为虚拟机的数据 LIF 。
`openshift_logging_storage_nfs_directory`	NFS 导出的挂载路径。例如、如果卷接合为 `/openshift_logging`、您将使用该路径作为此变量。
`openshift_logging_storage_volume_name`	名称、例如 `pv_ose_logs`、要创建的PV。
`openshift_logging_storage_volume_size`	NFS导出的大小、例如 `100Gi`。

如果 OpenShift 集群已在运行，因此已部署和配置 Trident ，则安装程序可以使用动态配置来创建卷。需要配置以下变量。

变量	详细信息
`openshift_logging_es_pvc_dynamic`	设置为 true 可使用动态配置的卷。
`openshift_logging_es_pvc_storage_class_name`	要在 PVC 中使用的存储类的名称。
`openshift_logging_es_pvc_size`	在 PVC 中请求的卷大小。
`openshift_logging_es_pvc_prefix`	日志记录服务使用的 PVC 的前缀。
`openshift_logging_es_ops_pvc_dynamic`	设置为 `true` 为操作日志记录实例使用动态配置的卷。
`openshift_logging_es_ops_pvc_storage_class_name`	操作日志记录实例的存储类的名称。
`openshift_logging_es_ops_pvc_size`	操作实例的卷请求大小。
`openshift_logging_es_ops_pvc_prefix`	操作实例 PVC 的前缀。

部署日志记录堆栈

如果要在初始 OpenShift 安装过程中部署日志记录，则只需遵循标准部署过程即可。Ansible 将配置和部署所需的服务和 OpenShift 对象，以便在 Ansible 完成后立即提供此服务。

但是，如果在初始安装后进行部署，则 Ansible 需要使用组件攻略手册。此过程可能会因 OpenShift 的不同版本而略有变化，因此请务必阅读并遵循 "RedHat OpenShift Container Platform 3.11 文档" 适用于您的版本。

指标服务

指标服务可为管理员提供有关 OpenShift 集群的状态，资源利用率和可用性的宝贵信息。此外、POD自动扩展功能也需要使用此功能、许多组织会将来自指标服务的数据用于其成本分摊和/或成本分摊应用程序。

与日志记录服务和 OpenShift 作为一个整体一样， Ansible 用于部署指标服务。此外，与日志记录服务一样，可以在集群初始设置期间或使用组件安装方法运行之后部署指标服务。下表包含在为指标服务配置永久性存储时非常重要的变量。

下表仅包含与存储配置相关的变量，因为这些变量与指标服务相关。文档中还有许多其他选项，应根据您的部署情况进行查看，配置和使用。

变量	详细信息
`openshift_metrics_storage_kind`	设置为 `nfs` 让安装程序为日志记录服务创建NFS PV。
`openshift_metrics_storage_host`	NFS 主机的主机名或 IP 地址。此值应设置为 SVM 的数据 LIF 。
`openshift_metrics_storage_nfs_directory`	NFS 导出的挂载路径。例如、如果卷接合为 `/openshift_metrics`、您将使用该路径作为此变量。
`openshift_metrics_storage_volume_name`	名称、 例如 `pv_ose_metrics`、要创建的PV。
`openshift_metrics_storage_volume_size`	NFS导出的大小、例如 `100Gi`。

如果 OpenShift 集群已在运行，因此已部署和配置 Trident ，则安装程序可以使用动态配置来创建卷。需要配置以下变量。

变量	详细信息
`openshift_metrics_cassandra_pvc_prefix`	用于衡量指标 PVC 的前缀。
`openshift_metrics_cassandra_pvc_size`	要请求的卷的大小。
`openshift_metrics_cassandra_storage_type`	要用于度量指标的存储类型，必须将此类型设置为动态， Ansible 才能创建具有相应存储类的 PVC 。
`openshift_metrics_cassanda_pvc_storage_class_name`	要使用的存储类的名称。

部署指标服务

使用在主机 / 清单文件中定义的适当 Ansible 变量，使用 Ansible 部署服务。如果您在 OpenShift 安装时进行部署，则系统将自动创建和使用 PV 。如果您使用组件攻略手册进行部署，则在 OpenShift 安装之后， Ansible 将创建所需的任何 PVC ，并在 Astra Trident 为其配置存储后部署该服务。

上述变量以及部署过程可能会随 OpenShift 的每个版本而发生变化。确保您查看并遵循 "RedHat 的 OpenShift 部署指南" 为您的版本配置，以便为您的环境进行配置。

数据保护和灾难恢复

了解Asta三元数据以及使用Asta三元数据创建的卷的保护和恢复选项。对于具有持久性要求的每个应用程序，您都应制定一个数据保护和恢复策略。

Astra三元数据复制和恢复

您可以创建备份、以便在发生灾难时还原Asta三端存储。

Astra三项技术复制

Astra Tandent使用Kubnetes CRD存储和管理自己的状态、并使用Kubnetes集群etcd存储其元数据。

步骤

1. 使用备份Kubenetes集群etcd "Kubenetes：备份etcd集群"。
2. 将备份项目放在FlexVol上。

> 建议您保护FlexVol所在的SVM、并将其与另一个SVM建立SnapMirror关系。

Asta三元组恢复

您可以使用Kubenetes CRD和Kubenetes集群etcd快照来恢复Astra Trident。

步骤

1. 从目标SVM中、将包含Kubnetes etcd数据文件和证书的卷挂载到将设置为主节点的主机上。
2. 在下复制与Kubirnetes集群相关的所有必需证书 `/etc/kubernetes/pki` 和下的etcd成员文件 `/var/lib/etcd`。
3. 使用从etcd备份还原Kubenetes集群 "Kubornetes：还原etcd集群"。
4. 运行 `kubectl get crd` 验证所有的三端测试自定义资源是否已启动、并检索三端测试对象以验证所有数据是否可用。

SVM复制和恢复

Asta三端存储无法配置复制关系、但存储管理员可以使用 "ONTAP SnapMirror" 复制SVM。

发生灾难时，您可以激活 SnapMirror 目标 SVM 以开始提供数据。系统还原后、您可以切换回主系统。

关于此任务

使用SnapMirror SVM复制功能时、请考虑以下事项：

- 您应为启用了SVM-DR的每个SVM创建一个不同的后端。
- 将存储类配置为仅在需要时选择复制的后端、以避免将不需要复制的卷配置到支持SVM-DR的后端。
- 应用程序管理员应了解与复制相关的额外成本和复杂性、并在开始此过程之前仔细考虑其恢复计划。

SVM复制

您可以使用 "ONTAP：SnapMirror SVM复制" 以创建SVM复制关系。

使用SnapMirror、您可以设置选项来控制要复制的内容。您需要知道在预成形时选择了哪些选项 使用Asta Trdent恢复SVM。

- "-Identity保留true" 复制整个SVM配置。
- "-discard-configs network" 不包括LIP和相关网络设置。
- "-Identity保留false" 仅复制卷和安全配置。

使用Asta Trdent恢复SVM

Astra Trident 不会自动检测 SVM 故障。如果发生灾难、管理员可以手动启动通过三项功能故障转移到新SVM的操作。

步骤

1. 取消计划的和正在进行的SnapMirror传输、中断复制关系、停止源SVM、然后激活SnapMirror目标SVM。
2. 如果指定了 `-identity-preserve false` 或 `-discard-config network` 配置SVM复制时、请更新 `managementLIF` 和 `dataLIF` 在三端定义文件中。
3. 确认 `storagePrefix` 位于三端定义文件中。无法更改此参数。正在放弃 `storagePrefix` 发生原因后端更新是否会失败。
4. 使用以下命令更新所有必需的后端、以反映新的目标SVM名称：

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. 如果指定了 `-identity-preserve false` 或 `discard-config network`，则必须退回所有应用程序Pod。

> 如果指定了 `-identity-preserve true`、则A作用 是在激活目标SVM后、A作用 是通过Asta Trident配置的所有卷都会开始提供数据。

卷复制和恢复

Asta三端存储无法配置SnapMirror复制关系、但存储管理员可以使用 "ONTAP SnapMirror复制和恢复" 以复制Asta Trdent创建的卷。

然后、您可以使用将恢复的卷导入到Astra Trident中 "tridentctrd卷导入"。

> 不支持导入 `ontap-nas-economy`， `ontap-san-economy`或 `ontap-flexgroup-economy` 驱动程序。

Snapshot数据保护

您可以使用以下方式保护和还原数据：

- 外部快照控制器和CRD、用于为永久性卷(PVs)创建Kubbernetes卷快照。

  "卷快照"

- ONTAP快照、用于还原卷的全部内容或恢复单个文件或LUN。

"ONTAP快照"

Astra Control Center应用程序复制

使用Astra Control、您可以使用SnapMirror的异步复制功能将数据和应用程序更改从一个集群复制到另一个集群。

"Astra Control：使用SnapMirror技术将应用程序复制到远程系统"

安全性

安全性

使用此处列出的建议确保Astra Trident安装安全。

在自己的命名空间中运行 **Astra Trident**

请务必防止应用程序，应用程序管理员，用户和管理应用程序访问 Astra Trident 对象定义或 Pod ，以确保存储可靠并阻止潜在的恶意活动。

要将其他应用程序和用户与Astra Trident分开、请始终在自己的Kubernetes命名空间中安装Astra Trident (`trident`）。将 Astra Trident 置于自己的命名空间中可确保只有 Kubernetes 管理人员才能访问 Astra Trident Pod 以及存储在命名空间 CRD 对象中的项目（如适用，还包括后端和 CHAP 密码）。
您应确保仅允许管理员访问Astra 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

Astra Trident支持对ONTAP SAN工作负载进行基于CHAP的身份验证(使用 `ontap-san` 和 `ontap-san-economy` 驱动程序)。NetApp 建议将双向 CHAP 与 Astra Trident 结合使用，以便在主机和存储后端之间进行身份验证。

对于使用SAN存储驱动程序的ONTAP 后端、Astra Trident可以通过设置双向CHAP并管理CHAP用户名和密码 `tridentctl`。
请参见 "此处" 了解 Astra Trident 如何在 ONTAP 后端配置 CHAP 。

> Trident 20.04 及更高版本支持 ONTAP 后端的 CHAP 。

对 **NetApp HCI** 和 **SolidFire** 后端使用 **CHAP** 身份验证

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。Astra Trident 使用一个机密对象，每个租户包含两个 CHAP 密码。当Trident作为CSI配置程序安装时、它会管理CHAP密码并将其存储在中 `tridentvolume` 相应PV的CR对象。创建 PV 时， CSI Astra Trident 会使用 CHAP 密码启动 iSCSI 会话并通过 CHAP 与 NetApp HCI 和 SolidFire 系统进行通信。

> CSI Trident 创建的卷不与任何卷访问组关联。

在非 CSI 前端中，作为辅助节点上的设备连接卷的操作由 Kubernetes 处理。创建卷后，如果该租户的密钥尚不存在，则 Astra Trident 会对 NetApp HCI/SolidFire 系统进行 API 调用，以检索这些密钥。然后， Astra Trident 将这些机密传递给 Kubernetes 。位于每个节点上的 kubelet 通过 Kubernetes API 访问这些机密，并使用它们在

访问卷的每个节点与卷所在的 NetApp HCI/SolidFire 系统之间运行 / 启用 CHAP 。

将**Astra Trident**与**NVE**和**NAE**结合使用

NetApp ONTAP 提供空闲数据加密、可在磁盘被盗、退回或重新利用时保护敏感数据。有关详细信息，请参见 "配置 NetApp 卷加密概述"。

- 如果在后端启用了NAE、则在Astra Trident中配置的任何卷都将启用NAE。
- 如果后端未启用NAE、则在Astra Trident中配置的任何卷都将启用NVE、除非将NVE加密标志设置为 `false` 在后端配置中。

> 在启用了NAE的后端的Astra Trident中创建的卷必须经过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、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` 在启用了NAE的后端、将创建启用了NAE的卷。您不能通过将NVE加密标志设置为来禁用NAE加密 `false`。

- 您可以通过将NVE加密标志显式设置为来在Astra Trident中手动创建NVE卷 `true`。

有关后端配置选项的详细信息、请参见：

- "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)来对Astra Trident上的ONTAP SAN和ONTAP SAN经济卷进行加密。Astra Trident支持对LUKS加密卷执行密码短语轮换和卷扩展。

在Astra Trident中、LUKS加密的卷会按照建议使用AES-XTS-plain64 Cypher和模式 "NIST"。

开始之前

- 工作节点必须安装加密设置2.1或更高版本(但低于3.0)。有关详细信息，请访问 "Gitlab：密码设置"。
- 出于性能原因、我们建议员工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
```

如果未返回任何内容、则您的处理器不支持AES-NI。有关AES-NI的详细信息、请访问： "Intel：高级加密标准说明(AES-NI)"。

启用**LUKS**加密

您可以对ONTAP SAN和ONTAP SAN经济卷使用Linux统一密钥设置(Unified Key Setup、LUKS)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的后端配置选项的详细信息、请参见 "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: netapp.io/trident
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 重复数据删除和数据压缩功能。

用于导入**LUKS**卷的后端配置

要导入LUKS卷、必须设置 `luksEncryption` to(`true` 在后端。。 `luksEncryption` Option可告知A作用 中的三端磁盘是否符合LUKS (`true`)或不符合LUKS (`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'
```

轮换**LUKS**密码短语

您可以轮换LUKS密码短语并确认轮换。

> 请勿忘记密码短语、除非您确认任何卷、快照或密钥不再引用它。如果引用的密码短语丢失、您可能无法挂载此卷、并且数据将保持加密状态且无法访问。

关于此任务

如果在指定新的LUKS密码短语后创建了挂载卷的POD、则会发生LUKS密码短语轮换。创建新的Pod时、Astra 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、则会显示 `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. 确定卷或快照上的现有密码短语。

Volume

```
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密钥以指定新密码短语和上一密码短语。确保 `previous-luke-passphrase-name` 和 `previous-luks-passphrase` 匹配上一个密码短语。

```
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. 创建一个新的装载卷的POD。这是启动轮换所必需的。
5. 验证密码短语是否已轮换。

Volume

```
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

仅在卷和快照上返回新密码短语时、才会轮换密码短语。

> 如果返回两个密码短语、例如 `luksPassphraseNames: ["B", "A"]`、转出不完整。您可以触发新POD以尝试完成轮换。

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(测试版1.25以上)。请参见 "Kubernetes 1.25：使用机密进行节点驱动型CSI卷扩展" 了解详细信息。
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: netapp.io/trident
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

启动联机存储扩展时、kubelet会将相应的凭据传递给驱动程序。