最佳实践和建议
Trident

NetApp
March 05, 2026

目录

最佳实践和建议

部署

部署Trident时，请遵循此处列出的建议。

部署到专用命名空间

"命名空间"造成不同应用程序之间的管理隔离，并阻碍资源共享。例如，一个命名空间中的 PVC 不能在另一个命名空间中使用。 Trident为 Kubernetes 集群中的所有命名空间提供 PV 资源，因此利用了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。必须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 具有两个特性，这两个特性结合起来，为限制应用程序的资源消耗提供了一种强大的机制。这 "存储配额机制"管理员可以按命名空间实施全局和存储类特定的容量和对象数量消耗限制。此外，使用 "射程限制"确保 PVC 请求在转发给配置器之前，其值既在最小值也在最大值范围内。

这些值是按命名空间定义的，这意味着每个命名空间都应该定义与其资源需求相符的值。请点击此处查看相关信息 "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，无论应用程序是否采用容器化，都能从中受益。

平台概览

Trident可与ONTAP和 Element 配合使用。没有哪个平台比其他平台更适合所有应用和场景，但是，在选择平台时，应该考虑应用的需求以及管理设备的团队的需求。

您应该遵循您所使用协议的主机操作系统的最佳实践。 （可选）您可以考虑将应用程序最佳实践（如果可用）与后端、存储类别和 PVC 设置相结合，以优化特定应用程序的存储。

ONTAP和Cloud Volumes ONTAP最佳实践

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

以下建议是为容器化工作负载配置ONTAP的指导原则，这些工作负载使用由Trident动态配置的卷。每项都应根据您的环境进行考虑和评估，以确定其适用性。

使用专用于**Trident** 的**SVM**。

存储虚拟机 (SVM) 为ONTAP系统上的租户提供隔离和管理分离。将 SVM 专用于应用程序可以实现权限委派，并可以应用限制资源消耗的最佳实践。

SVM的管理有多种选择：

- 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 CLI 为 SVM 创建专用管理界面。
- 与 NFS 数据接口共享管理角色。

在每种情况下，接口都应该在 DNS 中，并且在配置Trident时应该使用 DNS 名称。这有助于实现某些灾难恢复场景，例如无需网络身份保留的 SVM-DR。

对于 SVM 而言，采用专用管理 LIF 还是共享管理 LIF 没有偏好，但是，您应该确保您的网络安全策略与您选择的方法保持一致。无论如何，管理 LIF 应该可以通过 DNS 访问，以实现最大的灵活性。 "SVM-DR"可与Trident配合使用。

限制最大体积数

ONTAP存储系统有最大卷数限制，该限制会根据软件版本和硬件平台而有所不同。请参阅 "NetApp Hardware Universe"请根据您的具体平台和ONTAP版本确定确切的限制。当卷计数耗尽时，不仅Trident的配置操作会失败，而且所有存储请求的配置操作都会失败。

三叉戟的 `ontap-nas`和 `ontap-san`驱动程序为每个创建的 Kubernetes 持久卷 (PV) 提供 FlexVolume。这 `ontap-nas-economy`驱动程序大约每 200 个 PV 创建一个 FlexVolume（可在 50 到 300 之间配置）。这 `ontap-san-economy`驱动程序大约每 100 个 PV 创建一个 FlexVolume（可在 50 到 200 之间配置）。为了防止Trident消耗存储系统上所有可用的卷，您应该对 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>
```

价值 `max-volumes`价格会根据您所在环境的几个具体因素而有所不同：

- ONTAP集群中现有卷的数量
- 您预计在Trident之外为其他应用程序配置的卷数
- Kubernetes 应用程序预计使用的持久卷数量

这 `max-volumes`该值是ONTAP集群中所有节点上配置的总卷数，而不是单个ONTAP节点上的卷数。因此，您可能会遇到某些情况，即ONTAP集群节点的Trident配置卷可能比其他节点多得多或少得多。

例如，一个双节点ONTAP集群最多可以托管 2000 个FlexVol卷。将最大音量计数设置为 1250 似乎非常合理。然而，如果仅仅 "聚合"如果从一个节点分配给 SVM，或者从一个节点分配的聚合无法进行配置（例如，由于容量不足），则另一个节点将成为所有Trident配置卷的目标。这意味着该节点的容量限制可能在达到上限之前就已经达到。 `max-volumes`达到值后，将影响Trident和使用该节点的其他卷操作。 您可以通过确保集群中每个节点的聚合数据以相等的数量分配给**Trident**使用的**SVM**来避免这种情况。

克隆卷

NetApp Trident在使用时支持克隆卷 `ontap-nas`， `ontap-san` ， `solidfire-san` ， 和 `gcp-cvs`存储驱动程序。使用时 `ontap-nas-flexgroup`或者 `ontap-nas-economy`驱动程序不支持克隆。从现有卷创建新卷将创建一个新的快照。

避免克隆与不同存储类关联的PVC。在同一个 StorageClass 内执行克隆操作，以确保兼容性并防止出现意外行为。

限制**Trident**创建的最大体积。

要配置Trident可创建的卷的最大大小，请使用以下方法： `limitVolumeSize`参数 `backend.json`定义。

除了控制存储阵列的卷大小之外，您还应该利用 Kubernetes 功能。

限制**Trident**创建的**FlexVols**的最大尺寸。

要配置用作 ontap-san-economy 和 ontap-nas-economy 驱动程序池的 FlexVols 的最大大小，请使用以下方法： `limitVolumePoolSize`参数 `backend.json`定义。

配置**Trident**使用双向 **CHAP**

您可以在后端定义中指定 CHAP 发起方和目标用户名和密码，并让Trident在 SVM 上启用 CHAP。使用 `useCHAP`在后端配置中设置参数， Trident使用 CHAP 对ONTAP后端的 iSCSI 连接进行身份验证。

创建并使用 **SVM QoS** 策略

利用应用于 SVM 的ONTAP QoS 策略，可以限制Trident已配置卷可消耗的 IOPS 数量。这有助于 "阻止欺凌行为"或者失控的容器影响Trident SVM 之外的工作负载。

只需几个步骤即可为 SVM 创建 QoS 策略。请参阅您所用ONTAP版本的文档以获取最准确的信息。下面的示例创建了一个 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，您可以考虑使用 QoS 最低要求来保证容器化工作负载的吞吐量。自适应QoS与SVM层策略不兼容。

分配给容器化工作负载的 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。如果还有其他与 Kubernetes 部署无关的工作负载正在使用存储资源，则应注意确保这些工作负载不会意外受到不利影响。
- 预期工作负载将在容器中运行。如果对 IOPS 要求高的工作负载将在容器中运行，则低 QoS 策略会导致糟糕的用户体验。

需要注意的是，在 SVM 级别分配的 QoS 策略会导致分配给 SVM 的所有卷共享同一个 IOPS 池。如果一个或少数几个容器化应用程序对 IOPS 有很高的要求，它可能会欺负其他容器化工作负载。如果情况属实，您可能需要考虑使用外部自动化工具来分配按卷划分的 QoS 策略。

只有当您的ONTAP版本低于 9.8 时，才应将 QoS 策略组分配给 SVM。

为**Trident**创建 **QoS** 策略组

服务质量 (QoS) 保证关键工作负载的性能不会因竞争工作负载而降低。 ONTAP QoS 策略组为卷提供 QoS 选项，并允许用户为一个或多个工作负载定义吞吐量上限。有关 QoS 的更多信息，请参阅 "通过服务质量 (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在该池或后端中创建的每个卷。

ONTAP有两种类型的 QoS 策略组：传统型和自适应型。传统策略组在 IOPS 中提供固定的最大（或最小，在后续版本中）吞吐量。自适应 QoS 可根据工作负载大小自动调整吞吐量，随着工作负载大小的变化，保持 IOPS 与 TB|GB 的比率不变。在大型部署中管理成百上千个工作负载时，这提供了显著的优势。

创建QoS策略组时，请考虑以下事项：

- 你应该设置 `qosPolicy`关键在于 `defaults`后端配置块。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
```

- 您应该按卷应用策略组，以便每个卷都能获得策略组指定的全部吞吐量。不支持共享策略组。

有关 QoS 策略组的更多信息，请参阅 "ONTAP命令参考"。

限制对 **Kubernetes** 集群成员的存储资源访问

限制对Trident创建的 NFS 卷、iSCSI LUN 和 FC LUN 的访问是 Kubernetes 部署安全态势的关键组成部分。这样做可以防止不属于 Kubernetes 集群的主机访问卷并可能意外修改数据。

重要的是要理解命名空间是 Kubernetes 中资源的逻辑边界。假设同一命名空间内的资源可以共享，但是，重要的是，没有跨命名空间的功能。这意味着，即使 PV 是全局对象，当绑定到 PVC 时，也只能由同一命名空间中的 pod 访问。 务必确保在适当情况下使用命名空间来实现代码分离。

对于大多数组织而言，在 Kubernetes 环境中数据安全的主要担忧是容器中的进程可以访问挂载到主机上的存储，但该存储并非为容器所预期的。 "命名空间"旨在防止此类妥协。但是，特权容器是一个例外。

特权容器是指拥有比普通容器高得多的主机级权限的容器。这些功能默认情况下不会被禁用，因此请确保使用以下命令禁用该功能： "Pod 安全策略" 。

对于需要从 Kubernetes 和外部主机访问的卷，存储应以传统方式进行管理，PV 由管理员引入，而不是由Trident管理。这样可以确保只有当 Kubernetes 和外部主机都已断开连接并且不再使用该卷时，才会销毁存储卷。此外，还可以应用自定义导出策略，从而允许从 Kubernetes 集群节点和 Kubernetes 集群外部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）或其他无法调度用户应用程序的节点的部署，应使用单独的导出策略来进一步限制对存储资源的访问。这包括为部署到这些基础设施节点的服务（例如，OpenShift Metrics 和 Logging 服务）以及部署到非基础设施节点的标准应用程序创建导出策略。

使用专门的出口政策

您应该确保每个后端都存在导出策略，该策略仅允许访问 Kubernetes 集群中的节点。Trident可以自动创建和管理出口政策。这样， Trident将其提供的卷的访问限制到 Kubernetes 集群中的节点，并简化了节点的添加/删除。

或者，您也可以手动创建导出策略，并向其中填充一条或多条导出规则，以处理每个节点访问请求：

- 使用 `vserver export-policy create`ONTAP CLI 命令创建导出策略。
- 使用以下方式向出口策略添加规则： `vserver export-policy rule create` ONTAP CLI 命令。

运行这些命令可以限制哪些 Kubernetes 节点可以访问数据。

禁用 **`showmount`**对于应用**SVM**

这 `showmount`此功能使 NFS 客户端能够向 SVM 查询可用 NFS 导出列表。部署到 Kubernetes 集群的 pod 可以发出 `showmount -e`对目标执行命令，并接收可用挂载点列表，包括它无法访问的挂载点。虽然这本身并不构成安全隐患，但它确实提供了不必要的信息，可能会帮助未经授权的用户连接到 NFS 导出。

您应该禁用 `showmount`通过使用 SVM 级ONTAP CLI 命令：

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

SolidFire最佳实践

了解配置Trident的SolidFire存储的最佳实践。

创建 **Solidfire** 帐户

每个SolidFire帐户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 (CHAP) 凭证。您可以通过使用帐户名称和相应的 CHAP 凭据或通过卷访问组来访问分配给帐户的卷。一个账户最多可以分配两千卷，但一卷只能属于一个账户。

创建 **QoS** 策略

如果您想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。通过设置定义 QoS 的三个可配置参数，可以保证每个卷的性能：最小 IOPS、最大 IOPS 和突发 IOPS。

以下是 4Kb 块大小的可能最小、最大和突发 IOPS 值。

IOPS 参数	定义	最小值	默认值	最大值（**4Kb**）
最小 IOPS	保证一定容量的性能水平。	50	50	15000
最大 IOPS	性能不会超过此限制。	50	15000	200,000
突发 IOPS	短时突发场景下允许的最大IOPS。	50	15000	200,000

尽管最大 IOPS 和突发 IOPS 可以设置为高达 200,000，但卷的实际最大性能受集群使用情况和每个节点性能的限制。

数据块大小和带宽对 IOPS 数量有直接影响。随着数据块大小的增加，系统会将带宽增加到足以处理更大数据块大小的水平。随着带宽的增加，系统能够达到的IOPS数量会减少。请参阅 "SolidFire服务质量"有关服务质量和性能的更多信息。

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 协议对主机进行后端身份验证。卷访问组控制对其所配置卷的访问。 NetApp建议使用 CHAP 进行身份验证，因为它更简单且没有扩展限制。

带有增强型 CSI 配置器的Trident支持使用 CHAP 身份验证。 VAG 只能在传统的非 CSI 模式下使用。

CHAP 身份验证（验证发起者是否为预期的卷用户）仅在基于帐户的访问控制中受支持。如果您使用 CHAP 进行身份验证，则有两种选择：单向 CHAP 和双向 CHAP。单向 CHAP 通过使用SolidFire帐户名和发起方密钥来验证卷访问。双向 CHAP 选项提供了最安全的卷身份验证方式，因为卷通过帐户名和发起方密钥验证主机身份，然后主机通过帐户名和目标密钥验证卷身份。

但是，如果无法启用 CHAP 并且需要 VAG，则创建访问组并将主机启动器和卷添加到访问组。添加到访问组的每个 IQN 都可以使用或不使用 CHAP 身份验证来访问组中的每个卷。如果 iSCSI 发起程序配置为使用 CHAP 身份验证，则使用基于帐户的访问控制。如果 iSCSI 发起程序未配置为使用 CHAP 身份验证，则使用卷访问组访问控制。

哪里可以找到更多信息？

下面列出了一些最佳实践文档。搜索 "NetApp库"适用于最新版本。

ONTAP

- "NFS 最佳实践和实施指南"
- "SAN 管理"（适用于 iSCSI）
- "RHEL 的 iSCSI Express 配置"

Element软件

- "配置适用于 Linux 的SolidFire"

NetApp HCI

- "NetApp HCI部署先决条件"
- "访问NetApp部署引擎"

应用最佳实践信息

- "ONTAP上 MySQL 的最佳实践"
- "SolidFire上 MySQL 的最佳实践"
- "NetApp SolidFire和 Cassandra"
- "Oracle 在SolidFire的最佳实践"
- "SolidFire上的PostgreSQL最佳实践"

并非所有应用程序都有具体的指导原则，因此与您的NetApp团队合作并使用以下方法至关重要： "NetApp库"查找最新文档。

整合Trident

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

驾驶员选择和部署

为您的存储系统选择并部署后端驱动程序。

ONTAP后端驱动程序

ONTAP后端驱动程序的区别在于所使用的协议以及在存储系统上配置卷的方式。因此，在决定部署哪位驾驶员时要仔细考虑。

从更高的层面来看，如果您的应用程序具有需要共享存储的组件（多个 pod 访问同一个 PVC），则基于 NAS 的驱动程序将是默认选择，而基于块的 iSCSI 驱动程序则满足非共享存储的需求。根据应用程序的要求以及存储和基础设施团队的接受程度来选择协议。一般来说，对于大多数应用来说，它们之间几乎没有区别，因此通常取决于是否需要共享存储（多个 pod 需要同时访问）。

可用的ONTAP后端驱动程序有：

- `ontap-nas`每个已配置的 PV 都是一个完整的ONTAP FlexVolume。
- `ontap-nas-economy`每个已配置的 PV 都是一个 qtree，每个 FlexVolume 的 qtree 数量可配置（默认值为 200）。
- `ontap-nas-flexgroup`每个 PV 都配置为一个完整的ONTAP FlexGroup，并且分配给 SVM 的所有聚合都将被使用。
- `ontap-san`每个已配置的 PV 都是其自身 FlexVolume 中的一个 LUN。
- `ontap-san-economy`每个已配置的 PV 都是一个 LUN，每个 FlexVolume 的 LUN 数量可配置（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Trident公开。如果需要某些功能，则必须由存储管理员在配置完成后应用这些功能。上标脚注区分了每个功能和驱动程序的具体功能。

ONTAP NAS 驱动程序	**Snapshot**	克隆	动态出口策略	多附件	**QoS**	调整大小	复制
`ontap-nas`	是	是	是的脚注：5[]	是	是的脚注：1[]	是	是的脚注：1[]
`ontap-nas-economy`	无脚注：3[]	无脚注：3[]	是的脚注：5[]	是	无脚注：3[]	是	无脚注：3[]
`ontap-nas-flexgroup`	是的脚注：1[]	否	是的脚注：5[]	是	是的脚注：1[]	是	是的脚注：1[]

Trident为ONTAP提供 2 款 SAN 驱动程序，其功能如下所示。

ONTAP SAN 驱动程序	**Snapshot**	克隆	多附件	双向 **CHAP**	**QoS**	调整大小	复制
`ontap-san`	是	是	是的脚注：4[]	是	是的脚注：1[]	是	是的脚注：1[]
`ontap-san-economy`	是	是	是的脚注：4[]	是	无脚注：3[]	是	无脚注：3[]

> 以上表格的脚注：是脚注1[]：非Trident管理；是脚注2[]：由Trident管理，但不支持PV粒度；否脚注3[]：非Trident管理且不支持PV粒度；是脚注4[]：支持原始块卷；是脚注5[]： Trident支持。

非 PV 粒度的功能将应用于整个 FlexVolume，所有 PV（即共享 FlexVol 中的 qtree 或 LUN）将共享一个共同的计划。

如上表所示，大部分功能都体现在以下方面： `ontap-nas`和 `ontap-nas-economy`是一样的。然而，因为 `ontap-nas-economy`驱动程序限制了以 `PV` 粒度控制计划的能力，这可能会特别影响您的灾难恢复和备份计划。对于希望在ONTAP存储上利用 `PVC` 克隆功能的开发团队而言，只有在使用以下配置时才有可能： `ontap-nas` ， `ontap-san`或者 `ontap-san-economy`司机。

这 `solidfire-san`驱动程序还能够克隆PVC。

Cloud Volumes ONTAP后端驱动程序

Cloud Volumes ONTAP提供数据控制以及企业级存储功能，适用于各种用例，包括文件共享和块级存储，支持 NAS 和 SAN 协议（NFS、SMB / CIFS 和 iSCSI）。 Cloud Volume ONTAP的兼容驱动程序有： `ontap-nas` ， `ontap-nas-economy` ， `ontap-san`和 `ontap-san-economy`。这些适用于 Azure 版 Cloud Volume ONTAP和 GCP 版 Cloud Volume ONTAP 。

Amazon FSx for ONTAP后端驱动程序

Amazon FSx for NetApp ONTAP让您能够利用您熟悉的NetApp功能、性能和管理能力，同时享受在 AWS 上存储数据的简单性、敏捷性、安全性和可扩展性。 FSx for ONTAP支持许多ONTAP文件系统功能和管理 API。 Cloud Volume ONTAP的兼容驱动程序有： `ontap-nas` ， `ontap-nas-economy` ， `ontap-nas-flexgroup` ， `ontap-san`和 `ontap-san-economy`。

NetApp HCI/ SolidFire后端驱动程序

这 `solidfire-san`该驱动程序与NetApp HCI/ SolidFire平台配合使用，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设置Trident所配置卷的特定 QoS 限制，请使用以下方法： `type`后端文件中的参数。管理员还可以使用以下方法限制可在存储上创建的卷大小： `limitVolumeSize`范围。目前 ，Element 存储功能（例如卷调整大小和卷复制）尚不支持通过以下方式进行存储： `solidfire-san`司机。这些操作应该通过 Element Software 的 Web 用户界面手动完成。

SolidFire驱动程序	Snapshot	克隆	多附件	CHAP	QoS	调整大小	复制
`solidfire-san`	是	是	是的脚注：2[]	是	是	是	是的脚注：1[]

> 脚注：是脚注1： Trident不管理 是脚注2：支持原始块卷

Azure NetApp Files后端驱动程序

Trident使用 `azure-netapp-files`司机管理"Azure NetApp Files"服务。

有关此驱动程序及其配置方法的更多信息，请参见此处。"用于Azure NetApp Files的Trident后端配置" 。

Azure NetApp Files驱动程序	Snapshot	克隆	多附件	QoS	展开	复制
`azure-netapp-files`	是	是	是	是	是	是的脚注：1[]

> 脚注：是脚注：1[]：并非由Trident管理

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

Trident使用 `gcp-cvs`用于连接 Google Cloud 上的Cloud Volumes Service的驱动程序。

这 `gcp-cvs`驱动程序使用虚拟池来抽象后端，并允许Trident确定卷放置位置。管理员在以下位置定义虚拟池： `backend.json`文件。存储类使用选择器按标签识别虚拟池。

- 如果后端定义了虚拟池， Trident将尝试在 Google Cloud 存储池中创建卷，而这些虚拟池仅限于这些存储池。

- 如果后端未定义虚拟池， Trident将从该区域的可用存储池中选择一个 Google Cloud 存储池。

要在Trident上配置 Google Cloud 后端，您必须指定 `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 密钥取自您在 Google Cloud 上为Cloud Volumes Service设置 API 访问权限时创建的服务帐户私钥文件。

有关 Google Cloud 上的Cloud Volumes Service服务类型和服务级别的详细信息，请参阅："了解Trident对 GCP 版 CVS 的支持" 。

适用于 **Google Cloud Drive** 的**Cloud Volumes Service**	**Snapshot**	克隆	多附件	**QoS**	展开	复制
`gcp-cvs`	是	是	是	是	是	仅适用于 CVS-Performance 服务类型。

复制说明

- 复制功能并非由Trident管理。
- 克隆卷将创建在与源卷相同的存储池中。

存储类设计

要创建 Kubernetes 存储类对象，需要对各个存储类进行配置和应用。本节讨论如何为您的应用程序设计存储类。

具体后端利用

在特定的存储类对象中可以使用过滤来确定要与该特定存储类一起使用的存储池或存储池集。存储类中可以设置三组过滤器： `storagePools` ， `additionalStoragePools`和/或 `excludeStoragePools`。

这 `storagePools`该参数有助于将存储限制在与任何指定属性匹配的存储池集合中。这 `additionalStoragePools`该参数用于扩展Trident用于配置的池集，以及由属性选择的池集。 `storagePools`参数。您可以单独使用其中一个参数，也可以同时使用这两个参数，以确保选择合适的存储池集。

这 `excludeStoragePools`该参数用于专门排除符合属性要求的已列出泳池集合。

模拟 **QoS** 策略

如果您希望设计存储类来模拟服务质量策略，请创建一个具有以下功能的存储类： `media`属性为 `hdd`或者 `ssd`。基于 `media`根据存储类中提到的属性， Trident将选择合适的后端来提供服务。 `hdd`或者 `ssd`将聚合与媒体属性匹配，然后将卷的配置定向到特定的聚合上。因此，我们可以创建一个 PREMIUM 存储类，它将具有 `media`属性集为 `ssd`这可以归类为高级 QoS 策略。我们可以创建另一个存储类 STANDARD，其媒体属性设置为“hdd”，这可以归类为 STANDARD 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 Element 设备，该设备可以定义为 QoS 策略。

根据特定功能使用后端

存储类可以设计为在特定的后端上指导卷配置，其中启用了精简配置和厚配置、快照、克隆和加密等功能。要指

定要使用的存储，请创建存储类，指定启用所需功能的相应后端。

虚拟池

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

虚拟池允许管理员在后端之上创建一个抽象层，可以通过存储类引用该抽象层，从而在后端上更灵活、更高效地放置卷。同一服务类别可以定义不同的后端。此外，可以在同一个后端创建多个具有不同特性的存储池。当使用具有特定标签的选择器配置存储类时， Trident会选择与所有选择器标签匹配的后端来放置卷。如果存储类别选择器标签与多个存储池匹配， Trident将从中选择其中一个来配置卷。

虚拟泳池设计

创建后端时，通常可以指定一组参数。管理员无法创建具有相同存储凭据和不同参数集的另一个后端。随着虚拟池的引入，这个问题得到了缓解。虚拟池是在后端和 Kubernetes 存储类之间引入的级别抽象，以便管理员可以定义参数以及可以通过 Kubernetes 存储类作为选择器引用的标签，以与后端无关的方式。可以使用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，而是定义一个新的虚拟池。

模拟不同的服务级别/**QoS**

可以设计虚拟池来模拟服务类。使用Azure NetApp Files云卷服务的虚拟池实现，让我们来研究如何设置不同的服务类。配置Azure NetApp Files后端，使用多个标签来表示不同的性能级别。放 `servicelevel`将各个方面调整到相应的性能水平，并在每个标签下添加其他所需的方面。现在创建不同的 Kubernetes 存储类，将它们映射到不同的虚拟池。使用 `parameters.selector`字段中，每个 StorageClass 都会指定哪些虚拟池可用于托管卷。

指定一组特定的方面

可以从单个存储后端设计多个具有特定功能的虚拟池。为此，请在后端配置多个标签，并在每个标签下设置所需的方面。现在使用以下方式创建不同的 Kubernetes 存储类： `parameters.selector`该字段将映射到不同的虚拟池。后端配置的卷将具有所选虚拟池中定义的方面。

影响存储供应的**PVC**特性

在创建 PVC 时，除请求的存储类别之外的某些参数可能会影响Trident 的配置决策过程。

访问模式

通过 PVC 请求存储时，必填字段之一是访问模式。所需的模式可能会影响用于托管存储请求的后端选择。

Trident将尝试根据以下矩阵将所使用的存储协议与指定的访问方法进行匹配。这与底层存储平台无关。

	读写一次	只读多	读写多
iSCSI	是	是	是的（原始块）
NFS	是	是	是

如果向未配置 NFS 后端的Trident部署提交 ReadWriteMany PVC 请求，则不会配置任何卷。因此，请求者应该

使用适合其应用的访问模式。

卷操作

修改持久卷

除两种例外情况外，持久卷是 Kubernetes 中的不可变对象。创建完成后，可以修改回收策略和大小。然而，这并不能阻止在 Kubernetes 之外修改卷的某些方面。为了针对特定应用定制卷，确保容量不会意外耗尽，或者出于任何原因将卷移动到不同的存储控制器，这样做可能是可取的。

> ⓘ 目前 Kubernetes 树内配置器不支持对 NFS、iSCSI 或 FC PV 进行卷大小调整操作。 Trident支持扩展 NFS、iSCSI 和 FC 卷。

PV的连接详情创建后无法修改。

创建按需卷快照

Trident支持按需创建卷快照，并支持使用 CSI 框架从快照创建 PVC。快照提供了一种便捷的方法来维护数据在特定时间点的副本，并且在 Kubernetes 中具有独立于源 PV 的生命周期。这些快照可用于克隆PVC。

从快照创建卷

Trident还支持从卷快照创建持久卷。要实现这一点，只需创建一个 PersistentVolumeClaim 并指定 `datasource` 作为创建卷所需的快照。 Trident将通过创建一个包含快照中数据的卷来处理此 PVC。借助此功能，可以跨区域复制数据、创建测试环境、完全替换损坏或已损坏的生产卷，或者检索特定文件和目录并将其传输到另一个附加卷。

在集群中移动卷

存储管理员能够在ONTAP集群中，在聚合和控制器之间移动卷，而不会对存储用户造成任何干扰。只要目标聚合是Trident使用的 SVM 可以访问的目标聚合，此操作就不会影响Trident或 Kubernetes 集群。重要的是，如果聚合体是新添加到 SVM 中的，则需要通过将其重新添加到Trident来刷新后端。这将触发Trident重新清点 SVM ，以便识别新的聚合体。

但是， Trident并不自动支持跨后端移动卷。这包括在同一集群中的 SVM 之间、集群之间，或者到不同的存储平台（即使该存储系统是连接到Trident 的存储系统）。

如果将卷复制到另一个位置，则可以使用卷导入功能将当前卷导入到Trident中。

扩大规模

Trident支持调整 NFS、iSCSI 和 FC PV 的大小。这样用户就可以直接通过 Kubernetes 层调整卷的大小。所有主流NetApp存储平台，包括ONTAP、 SolidFire/ NetApp HCI和Cloud Volumes Service后端，均可进行卷扩展。为了便于日后扩展，请设置 `allowVolumeExpansion`到 `true`在与该卷关联的 StorageClass 中。每当需要调整持久卷的大小时，请编辑以下内容： `spec.resources.requests.storage`在持久卷声明中添加对所需卷大小的注释。 Trident会自动处理存储集群上卷的大小调整。

将现有卷导入 **Kubernetes**

卷导入功能允许将现有存储卷导入 Kubernetes 环境。目前这得到了以下方面的支持： `ontap-nas` ， `ontap-nas-flexgroup` ， `solidfire-san` ， `azure-netapp-files` ， 和 `gcp-cvs`司机。将现有应用程序移植到 Kubernetes 或灾难恢复场景中，此功能非常有用。

使用ONTAP时 `solidfire-san`司机们，请使用该命令 `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml`将现有卷导入 Kubernetes 以便由Trident管理。导入卷命令中使用的 PVC YAML 或 JSON 文件指向一个存储类，该存储类将Trident标识为配置程序。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`或者 `gcp-cvs`使用了驱动程序，请使用命令 `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml`将卷导入 Kubernetes 以便由Trident管理。这样就确保了卷号的唯一性。

执行上述命令后， Trident将在后端找到该卷并读取其大小。它会自动添加（必要时会覆盖）已配置的PVC的容积大小。然后Trident创建新的 PV，Kubernetes 将 PVC 绑定到 PV。

如果部署的集装箱需要特定的进口 PVC，则该集装箱将保持待定状态，直到通过批量进口流程绑定 PVC/PV 对为止。 PVC/PV管对连接好后，如果没有其他问题，容器应该就能升起来。

注册服务

注册表存储的部署和管理已在文档中记录。"netapp.io"在"博客"。

日志服务

与其他 OpenShift 服务一样，日志服务使用 Ansible 进行部署，配置参数由提供给 playbook 的清单文件（又名 hosts）提供。本文将介绍两种安装方法：在 OpenShift 初始安装期间部署日志记录和在 OpenShift 安装完成后部署日志记录。

> 从 Red Hat OpenShift 版本 3.9 开始，官方文档建议不要使用 NFS 作为日志服务，因为存在数据损坏方面的担忧。这是基于红帽公司对其产品的测试结果。 ONTAP NFS 服务器不存在这些问题，并且可以轻松地支持日志部署。最终，日志服务的协议选择取决于您，但要知道，在使用NetApp平台时，这两种协议都能很好地工作，如果您更喜欢 NFS，也没有理由避免使用 NFS。

如果选择将 NFS 与日志服务一起使用，则需要设置 Ansible 变量。 `openshift_enable_unsupported_configurations`到 `true`防止安装程序运行失败。

开始使用

日志服务可以选择性地部署到应用程序以及 OpenShift 集群本身的核心操作中。如果您选择部署操作日志记录，请指定以下变量 `openshift_logging_use_ops`作为 `true`将创建该服务的两个实例。控制操作日志实例的变量中包含“ops”，而控制应用程序日志实例的变量则不包含。

根据部署方法配置 Ansible 变量非常重要，以确保底层服务使用正确的存储。让我们来看看每种部署方法的选项。

> 下表仅包含与日志服务相关的存储配置变量。您还可以找到其他选择"Red Hat OpenShift 日志记录文档"应根据您的部署情况进行审查、配置和使用。

下表中的变量将使 Ansible playbook 使用提供的详细信息为日志服务创建 PV 和 PVC。虽然这种方法不如在 OpenShift 安装后使用组件安装 playbook 灵活，但如果您有现有的卷可用，这也不失为一个选择。

变量	详细信息
`openshift_logging_storage_kind`	设置为 `nfs`让安装程序为日志服务创建 NFS PV。

变量	详细信息
`openshift_logging_storage_host`	NFS主机的主机名或IP地址。这应该设置为虚拟机的dataLIF。
`openshift_logging_storage_nfs_directory`	NFS导出挂载路径。例如，如果体积连接为`/openshift_logging`你会将该路径用于此变量。
`openshift_logging_storage_volume_name`	名称，例如 `pv_ose_logs`，创建 PV。
`openshift_logging_storage_volume_size`	例如，NFS 导出的大小。 `100Gi` 。

如果您的 OpenShift 集群已经运行，因此Trident已经部署和配置，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 以使用动态配置卷。
`openshift_logging_es_pvc_storage_class_name`	PVC 中将使用的存储类别的名称。
`openshift_logging_es_pvc_size`	PVC中要求的体积大小。
`openshift_logging_es_pvc_prefix`	日志记录服务使用的 PVC 前缀。
`openshift_logging_es_ops_pvc_dynamic`	设置为 `true`为运维日志实例使用动态配置的卷。
`openshift_logging_es_ops_pvc_storage_class_name`	运维日志实例的存储类名称。
`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。

部署日志堆栈

如果您将日志记录部署作为 OpenShift 初始安装过程的一部分，那么您只需要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 完成后立即提供该服务。

但是，如果在初始安装之后进行部署，则需要使用 Ansible 的组件 playbook。此过程可能因 OpenShift 版本不同而略有差异，因此请务必阅读并遵循相关说明。"Red Hat OpenShift 容器平台 3.11 文档"适用于您的版本。

指标服务

指标服务为管理员提供有关 OpenShift 集群的状态、资源利用率和可用性的宝贵信息。此外，它对于 pod 自动扩展功能也是必要的，许多组织使用指标服务中的数据进行费用分摊和/或收益展示应用程序。

与日志服务以及整个 OpenShift 一样，Ansible 也用于部署指标服务。此外，与日志服务一样，指标服务可以在集群的初始设置期间或集群运行后使用组件安装方法进行部署。以下表格包含了为指标服务配置持久存储时重要的变量。

下表仅包含与指标服务相关的存储配置变量。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`让安装程序为日志服务创建 NFS PV。
`openshift_metrics_storage_host`	NFS主机的主机名或IP地址。这应该设置为你的 SVM 的 dataLIF。
`openshift_metrics_storage_nfs_directory`	NFS导出挂载路径。例如，如果体积连接为 `/openshift_metrics`你会将该路径用于此变量。
`openshift_metrics_storage_volume_name`	名称，例如 `pv_ose_metrics`，用于创建 PV。
`openshift_metrics_storage_volume_size`	例如，NFS 导出的大小。 `100Gi` 。

如果您的 OpenShift 集群已经运行，因此Trident已经部署和配置，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`	用于指标 PVC 的前缀。
`openshift_metrics_cassandra_pvc_size`	请求的卷册数量。
`openshift_metrics_cassandra_storage_type`	用于指标的存储类型，必须将其设置为动态，以便 Ansible 创建具有适当存储类的 PVC。
`openshift_metrics_cassanda_pvc_storage_class_name`	要使用的存储类的名称。

部署指标服务

在 hosts/inventory 文件中定义了相应的 Ansible 变量后，使用 Ansible 部署服务。如果您在安装 OpenShift 时进行部署，则 PV 将自动创建和使用。如果您使用组件 playbook 进行部署，则在 OpenShift 安装之后，Ansible 会创建所需的任何 PVC，并在Trident为其配置存储之后部署服务。

上述变量以及部署过程可能会随着 OpenShift 的每个版本而改变。请务必查看并遵循"红帽 OpenShift 部署指南"请根据您的环境配置您的版本。

数据保护和灾难恢复

了解Trident的保护和恢复选项以及使用Trident创建的卷。对于每个有持久化需求的应用程序，都应该制定数据保护和恢复策略。

Trident复制和恢复

您可以创建备份，以便在发生灾难时恢复Trident 。

Trident复制

Trident使用 Kubernetes CRD 来存储和管理自己的状态，并使用 Kubernetes 集群 etcd 来存储其元数据。

步骤

1. 使用以下命令备份 Kubernetes 集群 etcd"Kubernetes：备份 etcd 集群" 。

2. 将备份文件放置在FlexVol volume上

> NetApp建议您使用SnapMirror关系将FlexVol所在的 SVM 与其他 SVM 连接起来，从而保护该 SVM。

Trident恢复

使用 Kubernetes CRD 和 Kubernetes 集群 etcd 快照，您可以恢复Trident。

步骤

1. 从目标 SVM 上，将包含 Kubernetes etcd 数据文件和证书的卷挂载到将要设置为主节点的主机上。
2. 复制 Kubernetes 集群所需的所有证书。 `/etc/kubernetes/pki`以及 `etcd` 成员文件 `/var/lib/etcd`。
3. 使用 etcd 备份恢复 Kubernetes 集群"Kubernetes：恢复 etcd 集群"。
4. 跑步 `kubectl get crd`验证所有Trident自定义资源是否已启动，并检索Trident对象以验证所有数据是否可用。

SVM复制和恢复

Trident无法配置复制关系，但是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 开始提供数据服务。系统恢复后，您可以切换回主服务器。

关于此任务

使用SnapMirror SVM 复制功能时，请考虑以下事项：

- 对于启用了 SVM-DR 的每个 SVM，您应该创建一个独立的后端。
- 配置存储类，仅在需要时选择复制后端，以避免将不需要复制的卷配置到支持 SVM-DR 的后端上。
- 应用程序管理员应了解复制带来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"创建 SVM 复制关系。

SnapMirror允许您设置选项来控制要复制的内容。您需要知道您在执行操作时选择了哪些选项。使用Trident进行 SVM 恢复 。

- "-identity-preserve true"复制整个 SVM 配置。
- "-丢弃网络配置"不包括 LIF 和相关网络设置。
- "-identity-preserve false"仅复制卷和安全配置。

使用**Trident**进行 **SVM** 恢复

Trident无法自动检测 SVM 故障。如果发生灾难，管理员可以手动启动Trident故障转移到新的 SVM。

步骤

1. 取消已安排和正在进行的SnapMirror传输，断开复制关系，停止源 SVM，然后激活SnapMirror目标 SVM。
2. 如果您指定 `-identity-preserve false`或者 `-discard-config network`配置 SVM 复制时，请更新以下内容：`managementLIF`和 `dataLIF`在Trident后端定义文件中。
3. 确认 `storagePrefix`存在于Trident后端定义文件中。此参数无法更改。省略 `storagePrefix`这将导致后端更新失败。
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. 如果您指定 `-identity-preserve false`或者 `discard-config network`您必须重启所有应用程序 pod。

> 如果您指定 `-identity-preserve true`当目标 SVM 激活时， Trident提供的所有卷开始提供数据服务。

卷复制和恢复

Trident无法配置SnapMirror复制关系，但是存储管理员可以使用"ONTAP SnapMirror复制和恢复"复制Trident创建的卷。

然后，您可以使用以下方法将恢复的卷导入到Trident中："tridentctl 卷导入" 。

> 不支持导入 `ontap-nas-economy`， `ontap-san-economy` ， 或者 `ontap-flexgroup-economy`司机。

快照数据保护

您可以使用以下方法保护和恢复数据：

- 外部快照控制器和 CRD 用于创建持久卷 (PV) 的 Kubernetes 卷快照。

  "卷快照"

- ONTAP快照用于恢复卷的全部内容，或恢复单个文件或 LUN。

  "ONTAP快照"

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"。

版权信息

商标信息