Red Hat OpenStack Services on OpenShift 18.0

执行存储操作

使用块存储服务、镜像服务、对象存储服务和共享文件系统服务执行操作

Last Updated: 2024-10-01

Red Hat OpenStack Services on OpenShift 18.0 执行存储操作

使用块存储服务、镜像服务、对象存储服务和共享文件系统服务执行操作

OpenStack Team
rhos-docs@redhat.com

法律通告

摘要

在 OpenShift 环境中，使用 Red Hat OpenStack Services 中的卷、镜像、对象和文件共享执行操作。

目录

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。与我们分享您的成功秘诀。

在 JIRA 中提供文档反馈

使用 Create Issue 表单在 OpenShift (RHOSO)或更早版本的 Red Hat OpenStack Platform (RHOSP)上提供有关 Red Hat OpenStack Services 文档的反馈。当您为 RHOSO 或 RHOSP 文档创建问题时，这个问题将在 RHOSO Jira 项目中记录，您可以在其中跟踪您的反馈的进度。

要完成 Create Issue 表单，请确保您已登录到 JIRA。如果您没有红帽 JIRA 帐户，您可以在 https://issues.redhat.com 创建一个帐户。

1. 点击以下链接打开 **Create Issue** 页面： Create Issue
2. 完成 **Summary** 和 **Description** 字段。在 **Description** 字段中，包含文档 URL、章节号以及问题的详细描述。不要修改表单中的任何其他字段。
3. 点 **Create**。

第 1 章 在 RED HAT OPENSTACK SERVICES ON OPENSHIFT 中执行存储操作

Red Hat OpenStack Services on OpenShift (RHOSO)提供以下存储服务：

- Block Storage 服务 (cinder)
- Image 服务 (glance)
- Object Storage 服务 (swift)
- 共享文件系统服务(manila)

您可以使用 RHOSO 控制面板(horizon)或 OpenStack 命令行界面(CLI)来管理云存储。您可以使用任一方法执行大多数步骤，但只能使用 OpenStack CLI 完成一些更高级的步骤。

1.1. 块存储(CINDER)

块存储服务(cinder)允许用户在后端调配块存储卷。用户可以将卷附加到实例，以使用通用持久性存储来增强其临时存储。您可以将卷分离和重新关联到实例，但您只能通过附加的实例访问这些卷。

您还可以配置实例，使其不使用临时存储。您可以配置块存储服务来将镜像写入卷，而不使用临时存储。然后，您可以使用卷作为实例的可引导根卷。卷还通过备份和快照提供固有冗余和灾难恢复。但是，只有在部署可选块存储备份服务时，才会提供备份。另外，您可以加密卷以提高安全性。

1.2. 镜像(GLANCE)

Image 服务(glance)为实例镜像提供发现、注册和交付服务。它还提供存储实例临时磁盘快照以满足克隆或恢复目的的功能。您可以将存储的镜像用作模板，比安装服务器操作系统和单独配置服务，快速、一致地编写新的服务器。

1.3. OBJECT STORAGE (SWIFT)

Object Storage 服务(swift)提供了一个完全分布式的存储解决方案，可用于存储任何类型的静态数据或二进制对象；如介质文件、大型数据集和磁盘镜像。对象存储服务使用对象容器来组织对象，该容器类似于文件系统中的目录，但不能嵌套。您可以使用对象存储服务作为云中各个服务的存储库。

Red Hat Ceph Storage RGW 可用作对象存储服务的替代选择。

1.4. 共享文件系统(MANILA)

共享文件系统服务(manila)提供了置备远程、可共享的文件系统的方法。这些称为共享。共享允许云中的项目共享 POSIX 兼容存储，它们可以被多个实例同时使用。

共享用于实例使用，可通过读/写访问模式同时被多个实例使用。

1.5. 自定义和管理 RED HAT CEPH STORAGE

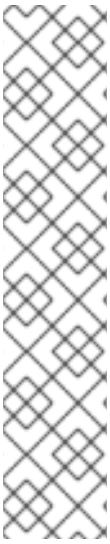
Red Hat OpenStack Services on OpenShift (RHOSO) 18.0 支持 Red Hat Ceph Storage 7。有关自定义和管理 Red Hat Ceph Storage 7 的信息，请参阅 Red Hat Ceph Storage 文档 。以下指南包含这些任务的关键信息和程序：

- 管理指南

- 配置指南
- 操作指南
- 数据安全性和强化指南
- 仪表板指南
- 故障排除指南

第 2 章 使用块存储备份服务执行操作

您可以使用块存储服务(cinder)备份服务来执行、保护、恢复和故障排除备份。



注意

要在云上执行 **openstack** 客户端命令，您必须指定 **clouds.yaml** 文件中详述的云名称。您可以使用以下方法之一指定云的名称：

- 在每个命令 **中使用--os-cloud** 选项：

```
$ openstack flavor list --os-cloud <cloud_name>
```

如果您访问多个云，则使用此选项。

- 在 **bashrc** 文件中为云名称创建一个环境变量：

```
`export OS_CLOUD=<cloud_name>`
```

先决条件

- 管理员已为您创建一个项目，已为您提供了一个 **clouds.yaml** 文件来访问云。
- 已安装 **python-openstackclient** 软件包。
- 只有具有访问权限的管理员和卷所有者才能执行和访问备份。

2.1. 使用块存储备份服务

您可以使用块存储备份服务执行完整或增量备份，以保护备份，并将备份恢复到卷。

2.1.1. 验证卷所有者以访问卷备份

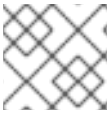
管理员可以备份属于该项目的任何卷。为确保卷所有者也可以访问卷备份，管理员必须提供参数以在备份卷时验证卷所有者。

流程

- 提供以下参数来验证卷备份的访问所有者：

```
$ openstack --os-project-name <projectname> \
--os-username <username> \
--os-password <password> \
volume backup create [--name <backup_name>] <volume>
```

  - 将 **<projectname** > 替换为卷所有者的项目（租户）的名称。
  - 将 **<username &gt**; 和 <password> 替换为此项目中卷所有者的用户的用户名和密码凭证。



注意

[--name <backup_name>] <volume > 是创建卷备份时的典型参数。

2.1.2. 创建备份

创建块存储卷的备份，以便在卷出现问题时保护您的数据丢失。如需更多信息，请参阅创建完整卷备份。您还可以直接从卷快照创建备份。除了卷数据外，备份还存储卷元数据，如名称和描述。

如果为备份启用了数据压缩，则会压缩您的备份，这可以降低备份操作的性能。

完全备份更易于管理，但随着时间增大，它们可能会成为资源密集型。通过增量备份，您可以捕获对卷的定期更改，并最小化资源使用量。

当您创建块存储卷的备份时，此备份的元数据存储在块存储服务数据库中。如果您恢复卷备份，则会使用元数据。为确保备份在块存储服务数据库的灾难性丢失时存下，您可以手动导出和存储此备份的元数据。

2.1.2.1. 创建完整卷备份

您可以创建一个或多个卷的完整备份。

先决条件

- 只有卷所有者和管理员可以备份卷。
- 备份的存储后端必须具有所需的空间。
- 未超过为您的项目指定的备份配额。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出卷以获取您要备份的卷的 ID 或名称：

```
$ openstack volume list
```

注意

通常，您只能备份具有 **available** 状态的卷，但如果需要，您可以使用 **in-use** 状态备份卷。如需更多信息，请参阅 创建使用中的卷的备份。

3. 备份卷：

```
$ openstack volume backup create [--name <backup_name>] <volume>
```

- 将 &**lt;volume** > 替换为您要备份的卷的 ID 或名称。
- 可选：将 **<backup_name** > 替换为此备份的名称。
  这个命令会立即提供此备份的 ID，但卷会在后台异步备份。例如：

```
$ openstack volume backup create --name vol1bu2 vol_1
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
```

```
| id    | 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb |
| name  | vol1bu2                              |
+-------+--------------------------------------+
```

验证

- 列出备份：

```
$ openstack volume backup list
```

此备份具有 **available** 状态时会创建卷备份。例如：

```
+--------------------------------------+---------+-------------+-----------+------+
| ID                                   | Name    | Description | Status    | Size |
+--------------------------------------+---------+-------------+-----------+------+
| 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb | vol1bu2 | None        | available |    1 |
| b604a932-94a5-468e-bf6b-841ac16f69a8 | None    | None        | available |    1 |
+--------------------------------------+---------+-------------+-----------+------+
```

1. 退出 **openstackclient** pod:

```
$ exit
```

其他资源

- 创建备份

2.1.2.2. 创建快照的完整备份

您可以使用与快照关联的卷的 ID 从快照创建完整备份。

通过直接附加到快照来创建备份，这比将快照克隆到卷中快，然后备份这个卷。但是，此功能可能会影响备份性能，因为额外的步骤从快照创建卷。

先决条件

- 您的备份存储库必须具有所需的空间。
- 未超过为您的项目指定的备份配额。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出快照以获取您要备份的快照的名称或 ID：

```
$ openstack volume snapshot list
```

3. 列出此快照的详情，以获取与此快照关联的卷的 ID：

```
$ openstack volume snapshot show <snapshot>
```

- 将 **<snapshot>** 替换为您要备份的快照的名称或 ID。
  volume_id 字段的值是与此快照关联的卷的 ID。例如：

```
$ openstack volume snapshot show snap_1
+-----------------------------------------+--------------------------------------+
| Field                                   | Value                                |
+-----------------------------------------+--------------------------------------+
| created_at                              | 2023-07-18T08:14:16.000000           |
| description                             | None                                 |
| id                                      | 2d95b707-6657-49af-ac8f-f9a7417d4650 |
| name                                    | snap_1                               |
| os-extended-snapshot-attributes:progress | 100%                                 |
| os-extended-snapshot-attributes:project_id | c2c1da89ed1648fc8b4f35a045f8d34c     |
| properties                              |                                      |
| size                                    | 1                                    |
| status                                  | available                            |
| updated_at                              | 2023-07-18T08:14:17.000000           |
| volume_id                               | 6841e3d1-8a1a-4496-bc51-f7c04b787e8f |
+-----------------------------------------+--------------------------------------+
```

4. 备份快照：

```
$ openstack volume backup create [--name <backup_name>] --snapshot <snapshot>
<volume_id>
```

- 将 **<volume_id>** 替换为与此快照关联的卷的 ID。
- 可选：将 **<backup_name>** 替换为此备份的名称。
  这个命令会立即提供此备份的 ID，但快照会在后台异步备份。例如：

```
$ openstack volume backup create --name snap1bu1 --snapshot snap_1 6841e3d1-
8a1a-4496-bc51-f7c04b787e8f
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
| id    | 867e6cfb-9be7-47fa-8a79-221b0e80c757 |
| name  | snap1bu1                             |
+-------+--------------------------------------+
```

验证

- 列出备份：

```
$ openstack volume backup list
```

快照备份在备份具有 **available** 状态时创建。例如：

```
+--------------------------------------+-----------+-------------+-----------+------+
| ID                                   | Name      | Description | Status    | Size |
+--------------------------------------+-----------+-------------+-----------+------+
| 867e6cfb-9be7-47fa-8a79-221b0e80c757 | snap1bu1  | None        | available |    1 |
+--------------------------------------+-----------+-------------+-----------+------+
```

1. 退出 **openstackclient** pod:

```
$ exit
```

2.1.2.3. 创建正在使用的卷的备份

通常，您只能备份具有 **可用** 状态的卷。但是，您可以在创建备份时使用-- **force** 选项备份具有 **in-use** 状态的卷。

当您使用- **force** 卷备份选项时，您可以创建一个崩溃一致性，而不是应用程序一致性，因为卷在执行备份前不会被静止。因此，数据保持不变，但备份不知道在执行备份时运行哪些应用程序。

先决条件

- 只有卷所有者和项目管理员才能备份卷。
- 您的备份存储库必须具有所需的空间。
- 未超过为您的项目指定的备份配额。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出卷以获取您要备份的卷的 ID 或名称：

```
$ openstack volume list
```

例如：

```
+--------------------------------------+---------------+-----------+------+--------------------------------+
| ID                                   | Name          | Status    | Size | Attached to                    |
+--------------------------------------+---------------+-----------+------+--------------------------------+
| 6841e3d1-8a1a-4496-bc51-f7c04b787e8f | vol_1         | available |    1 |                                |
| 92800cf6-82ae-448a-a2bb-872fa4d98099 | Pansible_vol_2 | in-use    |    1 | Attached to inst1 on /dev/vdc  |
+--------------------------------------+---------------+-----------+------+--------------------------------+
```

3. 如果要备份的卷具有 **in-use** 状态，请强制备份：

```
$ openstack volume backup create [--name <backup_name>] --force <volume>
```

- 将 &**lt;volume** > 替换为您要备份的卷的 ID 或名称。
- 可选：将 **<backup_name** > 替换为此备份的名称。
  这个命令会立即提供此备份的 ID，但卷会在后台异步备份。例如：

```
$ openstack volume backup create --name panvol2bu1 --force Pansible_vol_2
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
```

```
| id    | 8c72bbf3-eb8e-4459-83e9-c7654ebe6343 |
| name  | panvol2bu1                           |
+-------+--------------------------------------+
```

验证

- 列出备份：

```
$ openstack volume backup list
```

此备份具有 **available** 状态时会创建卷备份。例如：

```
+--------------------------------------+-----------+-------------+-----------+------+
| ID                                   | Name      | Description | Status    | Size |
+--------------------------------------+-----------+-------------+-----------+------+
| 8c72bbf3-eb8e-4459-83e9-c7654ebe6343 | panvol2bu1    | None        | available |    1 |
+--------------------------------------+-----------+-------------+-----------+------+
```

1. 退出 **openstackclient** pod:

```
$ exit
```

2.1.2.4. 增量备份

如果卷至少有一个完整备份，您可以使用 Block Storage 备份服务来创建增量备份。如需更多信息，请参阅创建增量备份。

完全备份更易于管理，但随着时间增大，它们可能会成为资源密集型。通过增量备份，您可以捕获对卷的定期更改，并尽可能减少资源使用量。

增量备份仅会备份自上次完整或增量备份后，对卷所做的更改。

增量备份会增加管理备份所需的管理开销。例如，如果已经有一个或多个增量备份，则无法删除完整备份，您只能删除最新的增量备份。

增量备份的性能低于完整备份：当您创建增量备份时，必须先读取卷中的所有数据并与完整备份中的数据和后续增量备份进行比较。

2.1.2.5. 创建增量备份

您可以创建一个增量备份，以仅存储自上次或增量备份以来对卷所做的更改。

先决条件

- 至少一个卷的完整备份。如需更多信息，请参阅创建完整卷备份。
- 只有卷所有者和项目管理员才能备份卷。
- 您的备份存储库必须具有所需的空间。
- 未超过为您的项目指定的备份配额。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出卷以获取您要备份的卷的 ID 或名称：

```
$ openstack volume list
```

3. 备份卷并使用- **incremental** 选项：

```
$ openstack volume backup create --incremental [--name <backup_name>] <volume>
```

- 将 & **lt;volume** > 替换为您要备份的卷的 ID 或名称。
- 可选：将 **<backup_name** > 替换为此备份的名称。
  这个命令会立即提供此备份的 ID，但卷会在后台异步备份。例如：

```
$ openstack volume backup create --name vol3incbu1 --incremental vol_3
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
| id    | f1681313-b5ed-4520-9b63-5b533f7cdc11 |
| name  | vol3incbu1                           |
+-------+--------------------------------------+
```

验证

- 列出备份：

```
$ openstack volume backup list
```

此备份具有 **available** 状态时会创建卷备份。例如：

```
+--------------------------------------+---------+-------------+-----------+------+
| ID                                   | Name    | Description | Status    | Size |
+--------------------------------------+---------+-------------+-----------+------+
| f1681313-b5ed-4520-9b63-5b533f7cdc11 | vol3incbu1 | None        | available |    1 |
| f0e9ba67-67e1-4c2c-96ce-221df75bf2c2 | vol3bu1    | None        | available |    1 |
+--------------------------------------+---------+-------------+-----------+------+
```

1. 退出 **openstackclient** pod:

```
$ exit
```

2.1.2.6. 取消备份

您必须在备份上请求强制删除才能取消它。

重要

如果您将 Red Hat Ceph Storage 后端用于备份存储库，则无法取消备份。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出备份以获取您要取消的备份的 ID 或名称：

```
$ openstack volume backup list
```

3. 取消备份：

```
# openstack volume backup delete --force <backup>
```

- 将 &**lt;backup** > 替换为您要取消的卷备份的 ID 或名称。
  备份可能会有小的延迟被成功取消。

验证

- 当这个命令没有列出备份记录时，备份会被取消：

```
$ openstack volume backup show <backup>
```

1. 退出 **openstackclient** pod:

```
$ exit
```

2.1.3. 保护备份

当您创建块存储卷的备份时，此备份的元数据存储在块存储服务数据库中，后者用于恢复这个卷。为确保备份在块存储服务数据库的灾难丢失丢失时存活，您可以手动将此备份的元数据导出并存储在安全的位置，如非站点备份。如需更多信息，请参阅 导出备份元数据。

当块存储服务数据库遇到灾难性丢失时，您无法恢复任何备份，因为此数据库包含恢复备份时使用的备份元数据。但是，如果您手动导出并保存了备份的元数据，您可以将此元数据导入到新的块存储数据库中，以便您可以使用此备份来恢复卷。如需更多信息，请参阅 导入备份元数据。

注意

对于增量备份，您必须导入所有上述备份的元数据，然后才能使用它恢复卷。

2.1.3.1. 导出备份元数据

您可以导出备份的元数据并将其存储在文件中，以便您可以恢复卷备份，即使块存储数据库出现灾难性丢失。

注意

对于增量备份，您必须导出所有上述备份的元数据。

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出备份以获取备份的 ID 或名称：

```
$ openstack volume backup list
```

3. 导出备份的元数据，并将其存储在适当的命名 YAML 文件中：

```
$ openstack volume backup record export -f yaml <backup> > <filename>.yaml
```

- 将 **<backup>** 替换为卷备份的 ID 或名称。
- 将 **<filename** > 替换为 YAML 文件的名称，以保存此备份导出的 **backup_service** 和 **backup_url** 值。
  例如：

```
$ openstack volume backup record export -f yaml vol1bu2 > vol1bu2.yaml
```

4. 将文件复制到安全位置，如非站点备份。

验证

- 编辑该文件，以查看 **backup_service** 和 **backup_url** 的值与此命令提供的值匹配：

```
$ openstack volume backup record export -f yaml <backup>
```

例如：

```
$ openstack volume backup record export -f yaml vol1bu2
backup_service: cinder.backup.drivers.ceph.CephBackupDriver
backup_url: eyJkcml2 … YWxzZX0=
```

1. 退出 **openstackclient** pod:

```
$ exit
```

2.1.3.2. 导入备份元数据

如果您导出并保存卷备份的元数据，则您可以导入此元数据并在块存储服务数据库丢失时使用备份。

您还可以使用此流程重新创建已删除的备份。

注意

对于增量备份，还必须导入所有上述备份的元数据。

先决条件

- 您必须提供这个备份的 **backup_service** 和 **backup_url** 元数据值。如需更多信息，请参阅 导出备份元数据。
- 尚未包含此备份的块存储数据库。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 找到您存储此备份的 **backup_service** 和 **backup_url** 元数据值的文件。

3. 将此卷备份的元数据值导入到块存储数据库中：

```
$ openstack volume backup record import <backup_service> <backup_url>
```

- 将 **<backup_service** > 替换为这个卷备份的 **backup_service** 元数据值。
- 将 **<backup_url** > 替换为这个卷备份的 **backup_url** 元数据值。
  此命令提供此备份的名称和 ID。例如：

```
$ openstack volume backup record import cinder.backup.drivers.ceph.CephBackupDriver
eyJkcml2 … YWxzZX0=
+-------+--------------------------------------+
| Field | Value                                |
+-------+--------------------------------------+
| id    | 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb |
| name  | vol1bu2                              |
+-------+--------------------------------------+
```

4. 退出 **openstackclient** pod:

```
$ exit
```

后续步骤

- 恢复备份

2.1.4. 恢复备份

创建块存储卷备份后，如果需要，您可以恢复这个备份的数据。

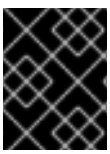
您可以使用以下方法之一恢复备份：

- 将备份恢复到您指定的卷。如需更多信息，请参阅将备份恢复到特定卷。

注意

如果您选择 Red Hat Ceph Storage 作为备份存储库的后端，则您只能将备份的卷恢复到基于 RBD 的块存储后端。

- 将备份恢复到新卷。如需更多信息，请参阅将备份恢复到新卷。



重要

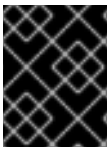
当块存储服务数据库遇到灾难性丢失时，除非已导出并保存它们的元数据，否则您无法恢复任何备份。

只有管理员才能取消恢复卷备份。

2.1.4.1. 将备份恢复到特定卷

您可以将卷备份恢复到已创建的 **可用** 卷。

如果您从加密卷备份中恢复卷，则必须加密目标卷类型。



重要

如果您选择 Red Hat Ceph Storage 作为备份存储库的后端，则您只能将备份的卷恢复到基于 RBD 的块存储后端。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出备份以获取您要恢复的备份的名称或 ID：

```
$ openstack volume backup list
```

例如：

```
+--------------------------------------+---------+-------------+-----------+------+
| ID                                   | Name    | Description | Status    | Size |
+--------------------------------------+---------+-------------+-----------+------+
| 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb | vol1bu2 | None        | available |    1 |
```

3. 列出卷：

```
$ openstack volume list
```

确保所需卷的状态 **可用**，然后获取此卷的名称或 ID。例如：

```
+--------------------------------------+---------------+-----------+------+--------------------------------+
| ID                                   | Name          | Status    | Size | Attached to                    |
+--------------------------------------+---------------+-----------+------+--------------------------------+
| 654e2be8-bc79-4528-96a7-5f773d31c201 | vol_3         | available |    1 |
|
```

4. 将备份恢复到卷：

```
$ openstack volume backup restore <backup> <volume>
```

- 将 **<backup** > 替换为块存储卷备份的名称或 ID。
- 将 **<volume** > 替换为 **可用** 块存储卷的名称或 ID。
  例如：

```
$ openstack volume backup restore vol1bu2 vol_3
+-------------+--------------------------------------+
```

```
| Field       | Value                                |
+-------------+--------------------------------------+
| backup_id   | 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb |
| volume_id   | 654e2be8-bc79-4528-96a7-5f773d31c201 |
| volume_name | vol_3                                |
+-------------+--------------------------------------+
```

5. 验证此命令提供的 **backup_id** 是否与恢复的备份 ID 对应，并且 **volume_name** 和 **volume_id** 值对应于指定卷的名称和 ID。

6. 如果您不再需要备份，请删除它：

```
$ openstack volume backup delete <backup>
```

7. 退出 **openstackclient** pod:

```
$ exit
```

2.1.4.2. 将备份恢复到新卷

您可以在恢复块存储卷的备份时创建新卷。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出备份以获取您要恢复的备份的名称或 ID：

```
$ openstack volume backup list
```

例如：

```
+--------------------------------------+---------+-------------+-----------+------+
| ID                                   | Name    | Description | Status    | Size |
+--------------------------------------+---------+-------------+-----------+------+
| 83dadc43-2aa9-4c0b-bc05-a12203a8f4cb | vol1bu2 | None        | available |    1 |
```

3. 将备份恢复到新卷：

```
$ openstack volume backup restore <backup> [<volume>]
```

- 将 **<backup** > 替换为块存储卷的 ID。
- 可选：将 **<volume** > 替换为块存储卷的 ID。

4. 验证此命令提供的 **backup_id** 是否与恢复的备份 ID 对应。
volume_id 值是所创建的卷的 ID。但是 **volume_name** 可以是使用备份卷的名称替换的临时名称。

5. 列出卷，以验证 ID 为 **volume_id** 的卷是否已创建，并获取这个卷名称：

```
$ openstack volume list
```

例如：

```
+--------------------------------------+--------------+-----------+------+------------------------------+
| ID                                   | Name         | Status    | Size | Attached to                  |
+--------------------------------------+--------------+-----------+------+------------------------------+
| 296c853c-c749-4eb6-857a-57ec182232a6 | vol_1        | available |    1 |
|
```

6. 如果您不再需要备份，请删除它：

```
$ openstack volume backup delete <backup>
```

7. 退出 **openstackclient** pod:

```
$ exit
```

2.1.4.3. 取消恢复备份

您可以通过将备份的状态更改为 **错误** 来取消恢复卷备份。但是，当 Red Hat Ceph Storage 是备份存储库的后端时，您无法取消恢复备份。

> **警告**
>
> 如果您在启动后取消恢复备份，则目标卷会无用，因为无法知道数据量（若有）实际被恢复。

先决条件

- 确保备份存储库的后端不是 Red Hat Ceph Storage。
- 只有管理员才能取消恢复卷备份。

流程

1. 从您的工作站访问 **OpenStackClient** pod 的远程 shell：

```
$ oc rsh -n openstack openstackclient
```

2. 列出备份以获取您要停止恢复的备份的名称或 ID：

```
$ openstack volume backup list
```

3. 将此备份的状态更改为 **error** 以取消其恢复操作：

```
$ openstack volume backup set --state error <backup>
```

- 将 &**lt;backup** > 替换为您要恢复的卷备份的名称或 ID。

取消恢复是一种异步操作，因为备份存储库的后端必须在取消恢复前检测备份状态的更改。

验证

- 列出卷备份，以验证恢复是否已取消：

```
$ openstack volume backup list
```

当备份状态变为 **available** 时，恢复将被取消。

1. 退出 **openstackclient** pod:

```
$ exit
```

2.2. 对块存储备份服务进行故障排除

您可以通过验证块存储服务是否正确运行，然后通过检查日志文件是否有错误消息来诊断许多问题。

2.2.1. 备份故障排除

在收到备份 Block Storage (cinder)卷的请求时，块存储备份服务会执行静态检查。如果这些检查失败，您将收到通知：

- 检查是否有无效的卷引用(**缺少**)。
- 检查卷 **是否在使用** 或附加到实例。**正在使用的情况要求您使用- - force** 选项执行备份。如需更多信息，请参阅 创建使用中的卷的备份。
  当您使用- **force** 卷备份选项时，您可以创建一个崩溃一致性，而不是应用程序一致性，因为卷在执行备份前不会被静止。因此，数据保持不变，但备份不知道在执行备份时运行哪些应用程序。

当这些检查成功时：块存储备份服务接受备份此卷的请求，CLI 备份命令会立即返回，并在后台备份卷。

因此，即使备份失败，CLI backup 命令也会返回。当备份条目的 **Status** 为 **available** 时，您可以使用 **openstack volume backup list** 命令来验证卷备份是否成功。

如果备份失败，请检查块存储备份服务日志文件以了解原因。

2.2.2. 检查块存储备份服务日志文件

当备份或恢复不成功时，您可以检查 Block Storage 备份服务日志文件，以了解可帮助您确定原因的错误消息。

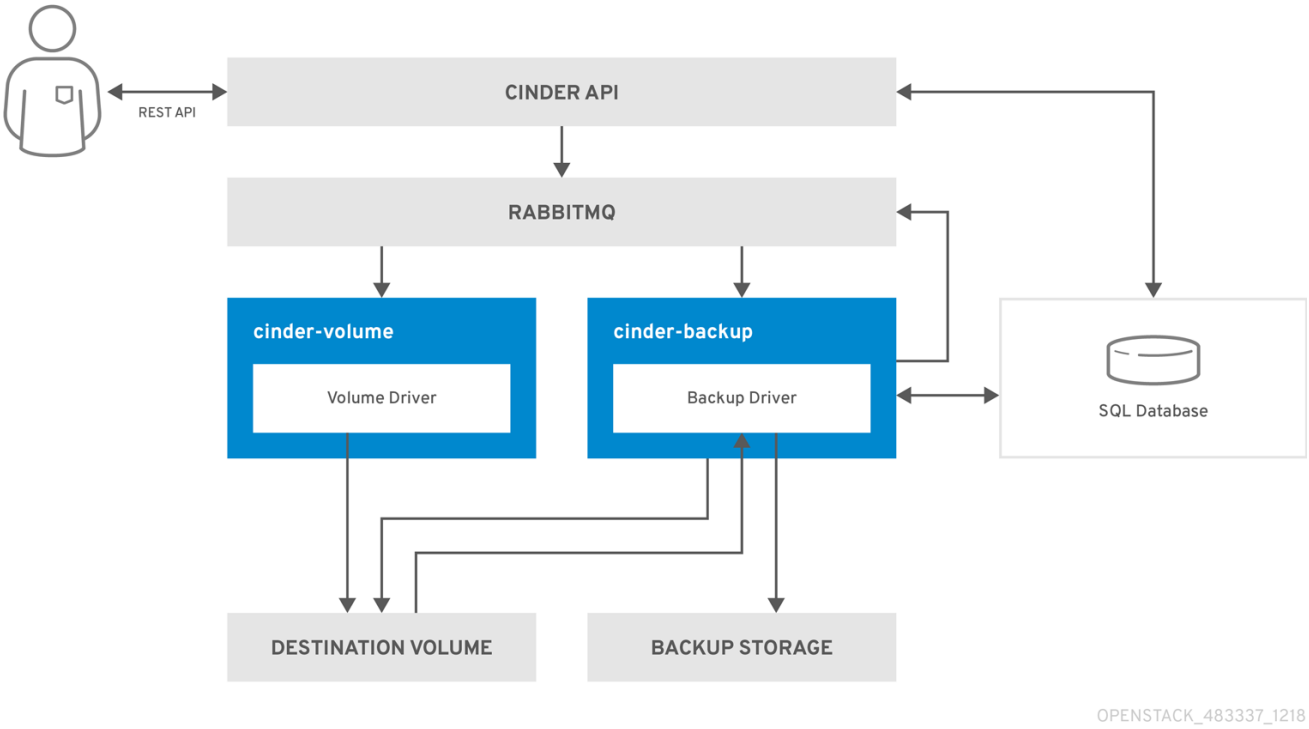
流程

- 在运行备份服务的 Controller 节点上，查找块存储备份服务日志文件。
  此日志文件位于以下路径： **/var/log/containers/cinder/cinder-backup.log**。

2.2.3. 卷备份工作流

下图显示了用户请求 cinder API 备份块存储(cinder)卷时所执行的步骤。

图 2.1. 创建块存储卷的备份



1. 用户向 cinder API 发出请求（即 REST API）以备份块存储卷。
2. cinder API 从 HAProxy 接收请求，并验证请求、用户凭据和其他信息。
3. cinder API 在 SQL 数据库中创建备份记录。
4. cinder API 对 **cinder-scheduler** 发出 RPC 调用。
5. **cinder-scheduler** 通过 AMQP 对 **cinder-backup** 服务发出异步 RPC 调用，以备份卷。
6. cinder API 将当前的备份记录(ID)返回到 API 调用器。
7. RPC 创建消息到达其中一个备份服务。
8. **cinder-backup** 服务执行对 **get_backup_device** 的同步 RPC 调用。
9. **cinder-volume** 服务确保正确的设备返回给调用者。通常，它是同一个卷，但如果卷正在使用，服务会返回临时克隆卷或临时快照，具体取决于配置。
10. **cinder-backup** 服务向 **cinder-volume** 发出另一个同步 RPC，以公开源设备。
11. **cinder-volume** 服务导出并映射源设备（卷或快照），并返回适当的连接信息。
12. **cinder-backup** 服务使用连接信息附加源设备。
13. **cinder-backup** 服务调用备份后端驱动程序，其设备已附加，后者开始传输到备份存储库。
14. 源设备从 Backup 主机分离。
15. **cinder-backup** 服务向 **cinder-volume** 发出同步 RPC，以断开源设备的连接。
16. **cinder-volume** 服务取消映射并删除设备的导出。

17. 如果创建了临时卷或临时快照，cinder-backup 调用 **cinder-backup** 调用 **cinder-volume** 将其移除。

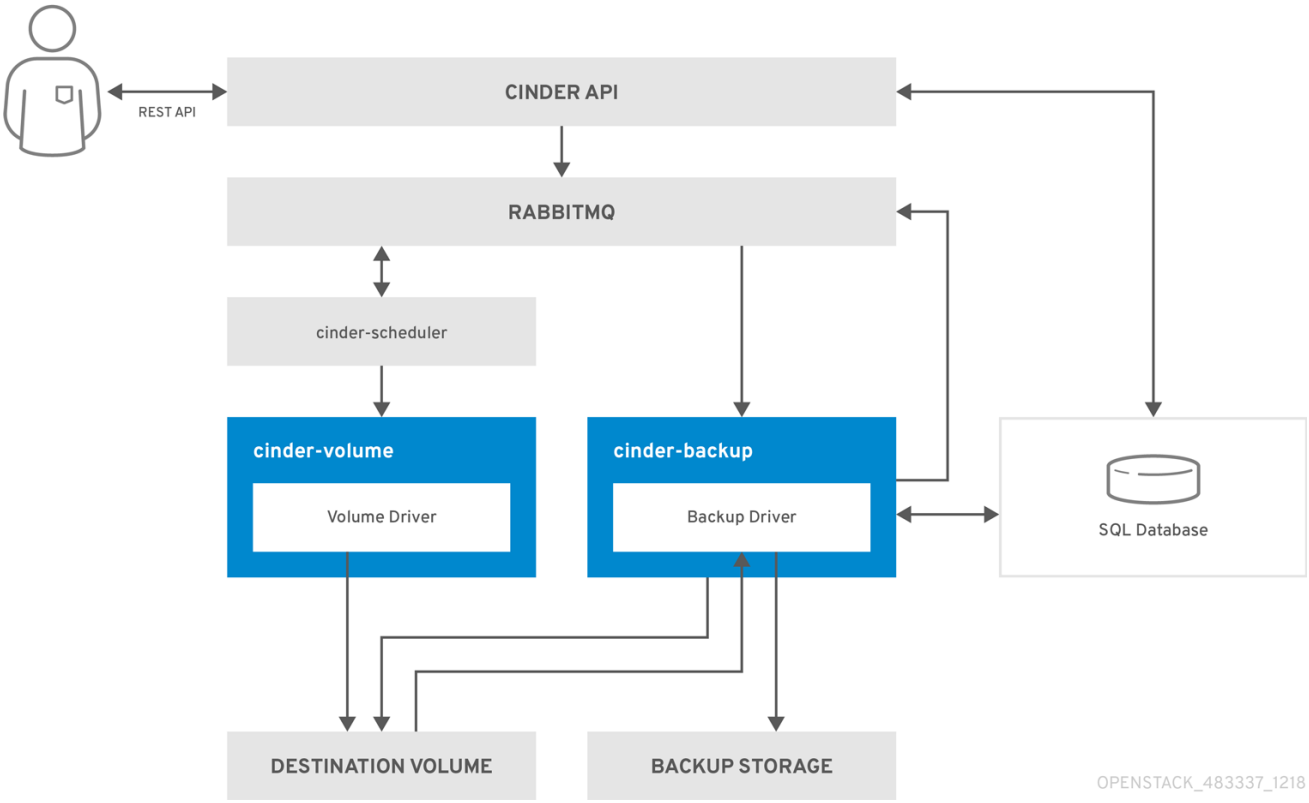
18. **cinder-volume** 服务移除临时卷。

19. 备份完成后，会在数据库中更新备份记录。

2.2.4. 卷恢复工作流

下图显示了用户请求 cinder API 恢复块存储服务(cinder)备份时所执行的步骤。

图 2.2. 恢复块存储备份



1. 用户向 cinder API 发出请求（即 REST API），以恢复块存储备份。

2. cinder API 从 HAProxy 接收请求，并验证请求、用户凭据和其他信息。

3. 如果请求不包含现有卷作为目的地，则 cinder API 将发出异步 RPC 调用来创建新卷并轮询卷的状态，直到卷可用为止。

4. **cinder-scheduler** 选择一个卷服务，并发出 RPC 调用来创建卷。

5. 所选 **cinder-volume** 服务创建卷。

6. 当 cinder API 检测到卷可用时，会在数据库中更新备份记录。

7. cinder API 通过 AMQP 对备份服务发出异步 RPC 调用，以恢复备份。

8. cinder API 将当前卷 ID、备份 ID 和卷名称返回到 API 调用者。

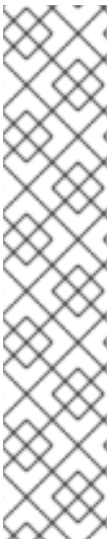
9. RPC 创建消息到达其中一个备份服务。

10. **cinder-backup** 服务对 **cinder-volume** 执行同步 RPC 调用来公开卷。

11. **cinder-volume** 服务导出并映射卷返回适当的连接信息。
12. **cinder-backup** 服务使用连接信息附加卷。
13. **cinder-backup** 服务使用已附加的卷调用后端驱动程序，这将开始对卷进行数据恢复。
14. 卷从备份主机分离。
15. **cinder-backup** 服务向 **cinder-volume** 发出同步 RPC，以断开卷的连接。
16. **cinder-volume** 服务取消映射并删除卷的导出。
17. 恢复卷后，会在数据库中更新备份记录。

第 3 章 使用 IMAGE 服务(GLANCE)执行操作

您可以在 OpenShift (RHOSO)镜像服务(glance)上的 Red Hat OpenStack Services 中创建和管理镜像。



注意

要在云上执行 **openstack** 客户端命令，您必须指定 **clouds.yaml** 文件中详述的云名称。您可以使用以下方法之一指定云的名称：

- 在每个命令 **中使用--os-cloud** 选项：

```
$ openstack flavor list --os-cloud <cloud_name>
```

如果您访问多个云，则使用此选项。

- 在 **bashrc** 文件中为云名称创建一个环境变量：

```
`export OS_CLOUD=<cloud_name>`
```

先决条件

- 管理员已为您创建一个项目，已为您提供了一个 **clouds.yaml** 文件来访问云。
- 已安装 **python-openstackclient** 软件包。

3.1. 创建操作系统镜像

要创建可在镜像服务(glance)中管理的操作系统镜像，您可以使用 Red Hat Enterprise Linux (RHEL)基于内核的虚拟机(KVM)实例镜像，或者您可以使用 RHEL ISO 文件或 Windows ISO 文件手动创建与 QCOW2 格式的 RHOSO 兼容镜像。

3.1.1. 虚拟机镜像格式

虚拟机(VM)镜像是包含安装可引导操作系统的虚拟磁盘的文件。Red Hat OpenStack Services on OpenShift (RHOSO)支持以不同格式的虚拟机镜像。

VM 镜像的磁盘格式是底层磁盘镜像的格式。容器格式指示虚拟机镜像是否为包含虚拟机元数据的文件格式。

将镜像添加到镜像服务(glance)时，您可以使用 **openstack image create、glance image-create -via-import 和 openstack image set 命令选项将您的镜像的磁盘** 或容器格式设置为下表中的任何值。**如果您不确定虚拟机镜像的容器格式，您可以将其设置为 bare。**

表 3.1. 磁盘镜像格式

格式	描述
aki	表示存储在镜像服务中的 Amazon 内核镜像。
ami	表示存储在镜像服务中的 Amazon 机器镜像。

格式	描述
ari	表示存储在镜像服务中的 Amazon ramdisk 镜像。
iso	磁盘上数据的扇区副本，存储在二进制文件中。虽然 ISO 文件通常不被视为虚拟机镜像格式，但这些文件包含有安装操作系统的可引导文件系统，并且您可以使用与其他虚拟机镜像文件的方式相同。
PLOOP	Virtuozzo 支持并用来运行 OS 容器的磁盘格式。
qcow2	QEMU 模拟器支持。此格式包括 QCOW2v3 （有时称为 QCOW3），这需要 QEMU 1.1 或更高版本。
raw	非结构化磁盘镜像格式.
vdi	VirtualBox VM 监控和 QEMU 模拟器支持。
vhd	虚拟硬盘.虚拟机监控器来自 VMware、VirtualBox 等使用。
vhdx	虚拟硬盘 v2.存储容量大于 VHD 的磁盘镜像格式。
vmdk	虚拟机磁盘.允许从上次备份时间对数据进行增量备份的磁盘镜像格式。

表 3.2. 容器镜像格式

格式	描述
aki	表示存储在镜像服务中的 Amazon 内核镜像。
ami	表示存储在镜像服务中的 Amazon 机器镜像。
ari	表示存储在镜像服务中的 Amazon ramdisk 镜像。
裸机	表示镜像没有容器或元数据信封。
docker	表示存储在镜像服务中的 Docker 容器的 TAR 归档。
ova	指明存储在镜像服务中的开放虚拟设备(OVA) TAR 归档文件。此文件存储在 Open Virtualization Format (OVF)容器文件中。
OVF	OVF 容器文件格式。用于打包并分发虚拟设备或软件在虚拟机上运行的开放式标准。

3.1.2. 创建 RHEL KVM 镜像

使用 Red Hat Enterprise Linux (RHEL)基于内核的虚拟机(KVM)实例镜像创建可在 OpenShift

(RHOSO)镜像服务(glance)上的 Red Hat OpenStack Services (RHOSO)镜像服务(glance)中管理的镜像。

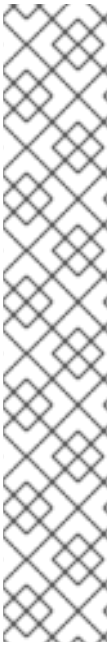
3.1.2.1. 使用 RHEL KVM 实例镜像

您可以将以下 Red Hat Enterprise Linux (RHEL)基于内核的虚拟机(KVM)实例镜像与 OpenShift (RHOSO 上的 Red Hat OpenStack Services)搭配使用：

- Red Hat Enterprise Linux 9 KVM Guest Image

QCOW2 镜像使用 cloud-init 配置，且必须具有 EC2 兼容元数据服务，以便置备 Secure Shell (SSH)密钥才能正常工作。

无法使用 QCOW2 格式的 Windows KVM 实例镜像。



注意

对于 KVM 实例镜像：

- 镜像中的 root 帐户被取消激活，但 sudo 访问权限被赋予一个名为 cloud-user 的特殊用户。
- 此映像没有设置 root 密码。

通过将 !! 放置到第二个字段中，将 root 密码锁定在 /etc/shadow 中。

对于 RHOSO 实例，从 RHOSO 控制面板或命令行生成 SSH 密钥对，并使用该密钥组合以 root 用户身份对实例执行 SSH 公钥身份验证。

当您启动实例时，此公钥会注入到其中。然后，您可以使用创建密钥对时下载的私钥进行身份验证。

3.1.2.2. 为裸机实例创建基于 RHEL 的 root 分区镜像

要为裸机实例创建自定义根分区镜像，请下载基本 Red Hat Enterprise Linux KVM 实例镜像，然后将镜像上传到镜像服务(glance)。

流程

1. 从客户门户网站下载 基本 Red Hat Enterprise Linux KVM 实例镜像。

2. 定义 DIB_LOCAL_IMAGE 作为下载的镜像：

```
$ export DIB_LOCAL_IMAGE=rhel-<ver>-x86_64-kvm.qcow2
```

   - 将 <ver > 替换为镜像的 RHEL 版本号。

3. 根据您的注册方法设置注册信息：

   - 红帽客户门户网站：

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_AUTO_ATTACH=true
$ export REG_METHOD=portal
$ export https_proxy='<IP_address:port>' (if applicable)
$ export http_proxy='<IP_address:port>' (if applicable)
```

   - Red Hat Satellite：

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_SAT_URL='<satellite-url>'
$ export REG_ORG='<satellite-org>'
$ export REG_ENV='<satellite-env>'
$ export REG_METHOD=<method>
```

   - 将尖括号 & lt;> 中的值替换为您的红帽客户门户网站或 Red Hat Satellite 注册的正确值。

4. 可选：如果您有任何离线存储库，您可以将 DIB_YUM_REPO_CONF 定义为本地存储库配置：

```
$ export DIB_YUM_REPO_CONF=<file-path>
```

- 将 **<file-path >** 替换为本地存储库配置文件的路径。

5. 使用 **diskimage-builder** 工具将内核提取为 **rhel-image.vmlinuz**，初始 **RAM** 磁盘作为 **rhel-image.initrd** ：

```
$ export DIB_RELEASE=<ver>
$ disk-image-create rhel baremetal \
  -o rhel-image
```

6. 将镜像上传到镜像服务：

```
$ KERNEL_ID=$(openstack image create \
  --file rhel-image.vmlinuz --public \
  --container-format aki --disk-format aki \
  -f value -c id rhel-image.vmlinuz)
$ RAMDISK_ID=$(openstack image create \
  --file rhel-image.initrd --public \
  --container-format ari --disk-format ari \
  -f value -c id rhel-image.initrd)
$ openstack image create \
  --file rhel-image.qcow2   --public \
  --container-format bare \
  --disk-format qcow2 \
  --property kernel_id=$KERNEL_ID \
  --property ramdisk_id=$RAMDISK_ID \
  rhel-root-partition-bare-metal-image
```

3.1.2.3. 为裸机实例创建基于 RHEL 的整个磁盘用户镜像

要为裸机实例创建整个磁盘用户镜像，请下载基本 **Red Hat Enterprise Linux KVM** 实例镜像，然后将镜像上传到镜像服务(glance)。

流程

1. 从客户门户网站下载 基本 **Red Hat Enterprise Linux KVM** 实例镜像。

2. 定义 **DIB_LOCAL_IMAGE** 作为下载的镜像：

```
$ export DIB_LOCAL_IMAGE=rhel-<ver>-x86_64-kvm.qcow2
```

- 将 <ver > 替换为镜像的 RHEL 版本号。

3. 根据您的注册方法设置注册信息：

- 红帽客户门户网站：

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_AUTO_ATTACH=true
$ export REG_METHOD=portal
$ export https_proxy='<IP_address:port>' (if applicable)
$ export http_proxy='<IP_address:port>' (if applicable)
```

- Red Hat Satellite：

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_SAT_URL='<satellite-url>'
$ export REG_ORG='<satellite-org>'
$ export REG_ENV='<satellite-env>'
$ export REG_METHOD=<method>
```

- 将尖括号 & lt;> 中的值替换为您的红帽客户门户网站或 Red Hat Satellite 注册的正确值。

4. 可选：如果您有任何离线存储库，您可以将 DIB_YUM_REPO_CONF 定义为本地存储库配置：

```
$ export DIB_YUM_REPO_CONF=<file-path>
```

- 将 <file-path > 替换为本地存储库配置文件的路径。

5. 将镜像上传到镜像服务：

```
$ openstack image create \
  --file rhel-image.qcow2 --public \
  --container-format bare \
  --disk-format qcow2 \
  rhel-whole-disk-bare-metal-image
```

3.1.3. 使用 RHEL 或 Windows ISO 文件创建实例镜像

您可以从 ISO 文件以 QCOW2 格式创建自定义 Red Hat Enterprise Linux (RHEL)或 Windows 镜像，并将这些镜像上传到 OpenShift (RHOSO)镜像服务(glance)，以便在创建实例时使用。

3.1.3.1. 先决条件

- 创建镜像的 Linux 主机。这可以是您可以在其中安装和运行 Linux 软件包的任何机器，但 undercloud 或 overcloud 除外。
- advanced-virt 存储库已启用：

```
$ sudo subscription-manager repos --enable=advanced-virt-for-rhel-<ver>-x86_64-rpms
```

- 安装 virt-manager 应用程序以拥有创建客户机操作系统所需的所有软件包：

```
$ sudo dnf module install -y virt
```

- 已安装 libguestfs-tools 软件包，以便具有访问和修改虚拟机镜像的一组工具：

```
$ sudo dnf install -y libguestfs-tools-c
```

- RHEL 9 ISO 文件或 Windows ISO 文件。有关 RHEL ISO 文件的更多信息，请参阅 RHEL 9.0 Binary DVD。如果您没有 Windows ISO 文件，请参阅 Microsoft Evaluation Center 以下载评估镜像。
- 文本编辑器（如果您想更改 kickstart 文件（仅限 RHEL）。

重要

如果您在 undercloud 上安装 libguestfs-tools 软件包，请停用 iscsid.socket，以避免与 undercloud 上的 tripleo_iscsid 服务的端口冲突：

```
$ sudo systemctl disable --now iscsid.socket
```

当您满足先决条件时，您可以继续创建 RHEL 或 Windows 镜像：

- **创建 Red Hat Enterprise Linux 9 镜像**
- **创建 Windows 镜像**

3.1.3.2. 创建 Red Hat Enterprise Linux 9 镜像

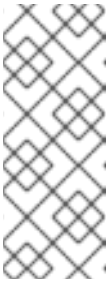
您可以使用 Red Hat Enterprise Linux (RHEL) 9 ISO 文件以 QCOW2 格式在 OpenShift (RHOSO) 镜像创建 Red Hat OpenStack Services。

流程

1. **以 root 用户身份登录您的主机计算机。**
2. **使用 virt-install 开始安装：**

```
[root@host]# virt-install \
  --virt-type kvm \
  --name <rhel9-cloud-image> \
  --ram <2048> \
  --cdrom </var/lib/libvirt/images/rhel-9.0-x86_64-dvd.iso> \
  --disk <rhel9.qcow2>,format=qcow2,size=<10> \
  --network=bridge:virbr0 \
  --graphics vnc,listen=127.0.0.1 \
  --noautoconsole \
  --os-variant=<rhel9.0>
```

- **将尖括号 &lt ;& gt; 中的值替换为 RHEL 9 镜像的正确值。**



注意

```
[root@host]# virt-viewer <rhel9-cloud-image>
```

3.

a.

b.

选择**相应的 Language 和 Keyboard 选项。**

c.

d.

e.

f.

g.

4.

5.

```
TYPE=Ethernet
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

6.

重启机器。

7.

```
# sudo subscription-manager register
# sudo subscription-manager attach \
```

```
  --pool=<pool-id>
# sudo subscription-manager repos \
  --enable rhel-9-for-x86_64-baseos-rpms \
  --enable rhel-9-for-x86_64-appstream-rpms
```

-

8.

```
# dnf -y update
```

9.

```
# dnf install -y cloud-utils-growpart cloud-init
```

10.

```
- resolv-conf
```

11.

```
NOZEROCONF=yes
```

12. **要确保控制台信息会在仪表板的 Log 标签页和 nova console-log 输出中显示，请在 /etc/default/grub 文件中添加以下引导选项：**

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8"
```

13.

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

```
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-229.9.2.el9.x86_64
Found initrd image: /boot/initramfs-3.10.0-229.9.2.el9.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-121.el9.x86_64
Found initrd image: /boot/initramfs-3.10.0-121.el9.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-b82a3044fb384a3f9aeacf883474428b
Found initrd image: /boot/initramfs-0-rescue-b82a3044fb384a3f9aeacf883474428b.img
done
```

14.

```
# subscription-manager repos --disable=*
# subscription-manager unregister
# dnf clean all
```

15.

```
# poweroff
```

16.

```
[root@host]# virt-sysprep -d <rhel9-cloud-image>
```

17.

```
[root@host]# virt-sparsify \
  --compress <rhel9.qcow2> <rhel9-cloud.qcow2>
```

注意

3.1.3.3.

流程

1.

2.

```
[root@host]# virt-install \
    --name=<windows-image> \
    --disk size=<size> \
    --cdrom=<file-path-to-windows-iso-file> \
    --os-type=windows \
    --network=bridge:virbr0 \
    --graphics spice \
    --ram=<ram>
```

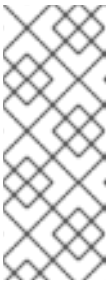
-

注意

3.

```
--disk path=<file-name>,size=<size>
```

-

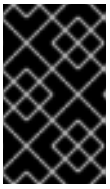


注意

```
[root@host]# virt-viewer <windows-image>
```

4.

5.



重要

3.1.4. 为 UEFI 安全引导创建镜像

流程

1.

```
$ openstack image create \
--file <base_image_file> \
--container-format <container_format> \
--disk-format <disk_format> \
uefi_secure_boot_image
```

-
-
-

2.

```
$ openstack image set --property hw_machine_type=q35 uefi_secure_boot_image
```

3.

```
$ openstack image set \
 --property hw_firmware_type=uefi \
 --property os_secure_boot=required \
 uefi_secure_boot_image
```

3.1.5.

- **os_distro**
- **os_version**
- **hw_cdrom_bus**
- **hw_disk_bus**
- **hw_scsi_model**
- **hw_vif_model**
- **hw_video_model**
- **hypervisor_type**

3.2.

3.2.1.

流程

-

  例如：

  ```
  $ openstack image create --name <name> \
      --is-public true --disk-format <qcow2> \
      --container-format <bare> \
      --file </path/to/image> \
      --property <os_version>=<11.10>
  ```

  -
  -
  -
  -
  -

3.2.2.

-
-

3.2.2.1.

1.

2.

流程

-

```
$ glance image-create-via-import \
    --container-format <container_format> \
    --disk-format <disk_format> \
    --name <name> \
    --import-method web-download \
    --uri <uri>
```

  -

  -

  -

-

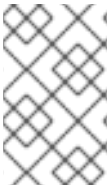
验证

-

```
$ openstack image show <image-id>
```

-

3.2.2.2.



注意

-
-
-

流程

1.

```
$ glance image-create-via-import \
```

```
    --container-format <container-format> \
    --disk-format <disk-format> \
    --name <name> \
    --file </path/to/image>
```

-
-
-
-

验证

-

```
$ openstack image show <image-id>
```

  -

3.2.3.

流程

-
  -

```
$ glance image-create-via-import \
    --disk-format <qcow2> \
    --container-format <bare> \
    --name <name> \
    --visibility public \
    --import-method web-download \
    --uri __<http://server/image.qcow2>__
```

    -
    -
    -
    -

  -

```
$ glance image-create-via-import \
    --disk-format <qcow2> \
    --container-format <bare>
    --name <name>
    --visibility public
    --file <local_file.qcow2>
```

    -

3.2.3.1.

流程

1.

```
$ qemu-img info <image_id>.qcow2
```

-

2.

```
$ qemu-img convert -p -f qcow2 -O raw <image_id>.qcow2 <image_id>.raw
```

3.2.3.2.

```
$ glance image-create-via-import \
    --disk-format qcow2 \
    --container-format bare \
    --name <name> \
    --visibility public \
    --import-method web-download \
    --uri <http://server/image.qcow2>
```

-

-

注意

3.2.4.

流程

-

  例如：

  ```
  $ openstack image set <image-id> \
      --property <architecture>=<x86_64>
  ```

  -

  -

3.2.5.

流程

-

  ```
  $ openstack image set <image_id> --hidden 'true'
  ```

-

  ```
  $ openstack image set <image_id> --hidden 'false'
  ```

-

  ```
  $ openstack image list --hidden 'true'
  ```

3.2.6.

流程

-

```
$ openstack image delete <image-id> [<image-id> ...]
```

  -

警告

3.3.

重要

3.3.1.

流程

1.

```
$ glance image-create-via-import \
--container-format bare \
--name <image-name> \
```

```
--import-method web-download \
--uri <uri> \
--store <store>
```

-

-

-

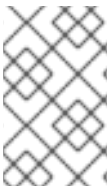
注意

2.

```
$ openstack image show <image-id> | grep stores
```

-

3.3.2.



注意

流程

-

```
$ glance image-create-via-import \
```

```
--container-format bare \
--name <image-name> \
--import-method web-download \
--uri <uri> \
--stores <store-1>,<store-2>,<store-3>
```

-
-
-
-

3.3.3.

流程

-

```
$ glance image-create-via-import \
--container-format bare \
--name <image-name> \
--import-method web-download \
--uri <uri> \
--stores <store-1>,<store-2>,<store-3>
```

-
-
-

-

注意

如需更多信息，请参阅 第 3.3.4 节 “”。

验证

-

```
$ openstack image show <image-id> | grep stores
```

3.3.4.

-

-

注意

流程

1.

```
$ openstack image show <image-id>
```

-

```
| os_glance_failed_import       |
| os_glance_importing_to_stores | central,dcn0,dcn1
| status                        | importing
```

2.

```
$ watch openstack image show <image-id>
```

-

```
| os_glance_failed_import       |
| os_glance_importing_to_stores | dcn0,dcn1
| status                        | importing
```

```
| os_glance_failed_import       | dcn0
| os_glance_importing_to_stores | dcn1
| status                        | importing
```

```
| os_glance_failed_import       | dcn0
| os_glance_importing_to_stores |
| status                        | active
```

3.3.5.

- 这是默认设置。
-

3.3.6.

流程

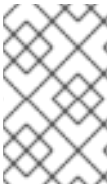
1.

- ```
  $ openstack image import <image_id> \
  --store <store_id>\
  --import-method copy-image
  ```
  -
  -
- ```
  $ openstack image import <image-id> \
  --stores <store-1>,<store-2> \
  --import-method copy-image
  ```
  -

2.

```
$ openstack image list --include-stores
```

3.3.7.



注意

-
-

重要

3.3.8.

流程

1.

```
$ openstack image import <image-id> \
--all-stores true \
--import-method copy-image
```

-

2.

```
$ openstack image list --include-stores
```

3.3.9.

流程

-

```
$ openstack image delete --store <store-id> <image-id>
```

-

-

警告

3.3.10.

流程

1.

```
$ openstack image show ID | grep "stores"

| stores |  default_backend,dcn1,dcn2
```

2.

```
$ openstack image list --include-stores

| ID                                   | Name    | Stores

| 2bd882e7-1da0-4078-97fe-f1bb81f61b00 | cirros | default_backend,dcn1,dcn2
```

3.

```
$ openstack image show ID -c properties

| properties |

(--- cut ---)
locations='[{'url': 'rbd://79b70c32-df46-4741-93c0-8118ae2ae284/images/2bd882e7-1da0-
4078-97fe-f1bb81f61b00/snap', 'metadata': {'store': 'default_backend'}}, {'url': 'rbd://63df2767-
8ddb-4e06-8186-8c155334f487/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap',
'metadata': {'store': 'dcn1'}}, {'url': 'rbd://1b324138-2ef9-4ef9-bd9e-
aa7e6d6ead78/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap', 'metadata': {'store':
'dcn2'}}]',
(--- cut --)
```

```
rbd://79b70c32-df46-4741-93c0-8118ae2ae284/images/2bd882e7-1da0-4078-97fe-
f1bb81f61b00/snap', 'metadata': {'store': 'default_backend'}}
```

- 每个 Ceph 集群都有唯一的 FSID。
-
-
-

3.4.

3.4.1.

表 3.3. 命令选项

选项		描述
All		
All	**--protected [True_False]**	
All	**--name <NAME>**	
All		
All	**--min-disk <MIN_DISK>**	
All		
All	**--kernel-id <KERNEL_ID>**	
All		

	选项	描述
All	**--disk-format <DISK_FORMAT>**	
All	**--os-distro <OS_DISTRO>**	
All		
All	**--ramdisk-id <RAMDISK_ID>**	
All	**--min-ram <MIN_RAM>**	
All	**--container-format <CONTAINER_FORMAT>**	
All	**--property <key=value>**	
	--tags <TAGS> [<TAGS> ...]	
	--id <ID>	
	--remove-property	

3.4.2.

表 3.4.

键	描述	支持的值

	键	描述	支持的值
All	**架构**		• • • **armv7l** - ARM Cortex-A7 MPCore • • **i686** - Intel 6th-generation x86（P6 微架构） • • • • • • • • • • **openrisc** - OpenCores RISC • • **parisc64** - HP Precision Architecture 64-bit RISC • **ppc** - PowerPC 32-bit • **ppc64** - PowerPC 64-bit • • • • • • • • • • • •

	键	描述	支持的值
All	**hypervisor_type**		
All	**instance_uuid**	对于快照镜像，这是用于创建此镜像的服务器的 UUID。	
All	**kernel_id**		
All	**os_distro**		• • • • • • • • • • • • • • • • • **rhel** - Red Hat Enterprise Linux. • • •
All	**os_version**		
All	**ramdisk_id**		
All	**vm_mode**	这代表了用于虚拟机的主机/guest ABI （应用程序二进制接口）。	**Hvm**-完全虚拟化.这是 QEMU 和 KVM 使用的模式。

	键	描述	支持的值
libvirt API 驱动程序	**hw_cdrom_bus**	指定要将 CD-ROM 设备附加到的磁盘控制器类型。	**SCSI**,**virtio**,**ide**, 或 **usb**.如果指定了 **iscsi**，您必须将 **hw_scsi_model** 参数设置为 **virtio-scsi**。
libvirt API 驱动程序	**hw_disk_bus**	指定要附加磁盘设备的磁盘控制器类型。	**SCSI**,**virtio**,**ide**, 或 **usb**.请注意，如果使用 **iscsi**，则 **hw_scsi_model** 需要设置为 **virtio-scsi**。
libvirt API 驱动程序	**hw_firmware_type**	指定用于引导实例的固件类型。	设置为以下有效值之一： • **BIOS** • **uefi**
libvirt API 驱动程序	**hw_machine_type**	启用使用指定的机器类型引导 ARM 系统。如果使用 ARM 镜像并且未明确指定其机器类型，则 Compute 将使用 virt 机器类型作为 ARMv7 和 AArch64 的默认。	可以使用 **virsh capabilities** 命令查看有效的类型。机器类型显示在机器标签中。
libvirt API 驱动程序	**hw_numa_nodes**	向实例公开的 NUMA 节点数量（不会覆盖类别定义）。	整数.
libvirt API 驱动程序	**hw_numa_cpus.0**	vCPU N-M 到 NUMA 节点 0 的映射（不会覆盖类别定义）。	以逗号分隔的整数列表。
libvirt API 驱动程序	**hw_numa_cpus.1**	vCPU N-M 到 NUMA 节点 1 的映射（不要覆盖类别定义）。	以逗号分隔的整数列表。
libvirt API 驱动程序	**hw_numa_mem.0**	将 N MB RAM 映射到 NUMA 节点 0 （不会覆盖类别定义）。	整数
libvirt API 驱动程序	**hw_numa_mem.1**	将 N MB RAM 映射到 NUMA 节点 1（不要覆盖类别定义）。	整数

	键	描述	支持的值

libvirt API 驱动程序	**hw_pci_numa_affinity_policy**	指定 PCI 透传设备和 SR-IOV 接口的 NUMA 关联性策略。	设置为以下有效值之一：
			• **必需**：计算服务创建一个实例，该实例仅当实例的其中一个 NUMA 节点与 PCI 设备关联时才会请求 PCI 设备。这个选项提供最佳性能。
			• **preferred**：计算服务会尝试根据 NUMA 关联性选择 PCI 设备。如果无法关联性，则计算服务将实例调度到没有与 PCI 设备关联性的 NUMA 节点上。
			• **传统**：（默认）计算服务在以下情况下创建请求 PCI 设备的实例：
			  ◦ PCI 设备与至少一个 NUMA 节点的关联。
			  ◦ PCI 设备不提供有关其 NUMA 关联性的信息。

libvirt API 驱动程序	**hw_qemu_guest_agent**	客户机代理支持。如果设置为 **yes**，如果也安装了 **qemu-ga**，则可以自动静止(frozen)和快照。	**是 / no**
libvirt API 驱动程序	**hw_rng_model**	向使用此镜像启动的实例添加一个随机数生成器(RNG)设备。 实例类别默认启用 RNG 设备。要禁用 RNG 设备，管理员必须在类别上将 **hw_rng:allowed** 设置为 **False**。 默认熵源是 **/dev/random**。要指定硬件 RNG 设备，请在 Compute 环境文件中将 **rng_dev_path** 设置为 **/dev/hwrng**。	**VirtIO**，或其他支持的设备。

	键	描述	支持的值
libvirt API 驱动程序	**hw_scsi_model**	启用 VirtIO SCSI (virtio-scsi)来为计算实例提供块设备访问；默认情况下，实例使用 VirtIO 块(virtio-blk)。VirtIO SCSI 是一个半虚拟化 SCSI 控制器设备，它提高了可扩展性和性能，并支持高级 SCSI 硬件。	**virtio-scsi**
libvirt API 驱动程序	**hw_tpm_model**	设置为要使用的 TPM 设备的型号。如果没有配置 **hw:tpm_version**，则忽略。	• **TPM-tis** ：（默认） TPM 接口规格。 • **TPM-crb**: Command-Response Buffer.仅与 TPM 版本 2.0 兼容。
libvirt API 驱动程序	**hw_tpm_version**	设置为要使用的 TPM 版本。TPM 版本 **2.0** 是唯一支持的版本。	**2.0**

	键	描述	支持的值
libvirt API 驱动程序	**hw_video_model**	在虚拟机实例中使用的显示设备的视频设备驱动程序。	设置为以下值之一，以指定要使用的支持的驱动程序： • **VirtIO** -（默认）虚拟机显示设备 **的建议** 驱动程序，由大多数架构支持。VirtIO GPU 驱动程序包含在 RHEL-7 及更新的版本中，Linux 内核版本 4.4 及更新的版本。如果实例内核具有 VirtIO GPU 驱动程序，则实例可以使用所有 VirtIO GPU 功能。如果实例内核没有 VirtIO GPU 驱动程序，则 VirtIO GPU 设备可以安全地回退到 VGA 兼容性模式，后者为实例提供正常工作的显示。 • **QXL** - 不再维护的 Spice 或 noVNC 环境已弃用的驱动程序。 • **Cirrus** - Legacy 驱动程序，只支持向后兼容。不要将 用于新实例。 • **VGA** - 将此驱动程序用于 IBM Power 环境。 • **Gop** - 不支持 QEMU/KVM 环境。 • **Xen** - 不支持 KVM 环境。 • **vmvga** - 旧的驱动程序，不使用。 • **none** - 使用这个值在单独配置驱动程序的虚拟 GPU (vGPU)实例中禁用模拟图形或视频。
libvirt API 驱动程序	**hw_video_ram**	视频镜像的最大 RAM。仅在类别的 **extra_specs** 中设置 **hw_video:ram_max_mb** 值并且该值高于 **hw_video_ram** 中设置的值时使用。	以 MB 为单位的整数(*例如*64)

	键	描述	支持的值
libvirt API 驱动程序	**hw_watchdog_action**	启用虚拟硬件 watchdog 设备，在服务器挂起时执行指定操作。watchdog 使用 i6300esb 设备（模拟 PCI Intel 6300ESB）。如果没有指定 **hw_watchdog_action**，则禁用 watchdog。	• Disabled - 设备没有附加。允许用户禁用镜像的 watchdog，即使已使用镜像的类别启用了它。这个参数的默认值被禁用。 • 重置客户机。 • poweroff-Forcefully 关闭客户机。 • 暂停 guest。 • none-Only 启用 watchdog；如果服务器挂起，什么都不做。
libvirt API 驱动程序	**os_command_line**	**libvirt** 驱动程序使用的内核命令行，而不是默认值。对于 Linux 容器(LXC)，该值用作初始化的参数。这个密钥只适用于 Amazon 内核、ramdisk 或机器镜像 (aki、ari 或 ami)。	
libvirt API 驱动程序	**os_secure_boot**	使用 创建使用 UEFI 安全引导保护的实例。	设置为以下有效值之一： • **必需** ：为使用此镜像启动的实例启用安全引导。只有 Compute 服务找到可以支持安全引导的主机时，才会启动该实例。如果没有找到主机，计算服务会返回 "No valid host" 错误。 • **禁用** ：为使用此镜像启动的实例禁用安全引导。默认禁用此选项。 • **可选** ：仅在计算服务决定主机可以支持安全引导时，为使用此镜像启动的实例启用安全引导。

	键	描述	支持的值
libvirt API 驱动程序和 VMware API 驱动程序	**hw_vif_model**	指定要使用的虚拟网络接口设备的型号。	有效选项取决于配置的虚拟机监控程序。 • KVM 和 QEMU：e1000、ne2k_pci、pcnet、rtl8139 和 virtio. • VMware：e1000、e1000e、VirtualE1000、VirtualE1000e、VirtualPCNet32, VirtualSriovEthernetCard 和 VirtualVmxnet。 • Xen: e1000, netfront, ne2k_pci, pcnet, and rtl8139.
VMware API 驱动程序	**vmware_adaptertype**	管理程序使用的虚拟 SCSI 或 IDE 控制器。	**lsiLogic**,**busLogic**, 或 **ide**
VMware API 驱动程序	**vmware_ostype**	VMware GuestID，用于描述在镜像中安装的操作系统。这个值在创建虚拟机时传递给虚拟机监控程序。如果没有指定，则密钥默认为 **otherGuest**。	如需更多信息，请参阅使用 VMware vSphere 的镜像。
VMware API 驱动程序	**vmware_image_version**	当前未使用。	**1**
XenAPI driver	**auto_disk_config**	如果为 true，则在实例引导前，磁盘上的根分区会自动调整大小。只有使用带有 XenAPI 驱动程序的基于 Xen 的 hypervisor 时，计算服务才会考虑该值。只有映像上只有一个分区，并且仅当分区采用 **ext3** 或 **ext4** 格式时，计算服务才会尝试调整大小。	**true** / **false**

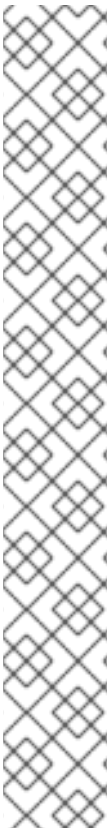
	键	描述	支持的值
libvirt API 驱动程序和 XenAPI 驱动程序	**os_type**	在镜像上安装的操作系统。XenAPI 驱动程序包含根据镜像的 **os_type** 参数的值来采取不同操作的逻辑。例如，对于 **os_type=windows** 镜像，它会创建一个基于 FAT32 的交换分区而不是 Linux swap 分区，并将注入的主机名限制为小于 16 个字符。	**linux** 或 **windows**

第 4 章 使用 OBJECT STORAGE 服务(SWIFT)执行操作

Object Storage 服务(swift)将其对象或数据存储在容器中。容器与文件系统中的目录类似，但您无法嵌套它们。您可以在容器中存储任何类型的非结构化数据。例如，对象可以包含 photos、文本文件或镜像。存储的对象不会被压缩。

您可以在容器中创建伪文件夹来组织数据。伪文件夹是包含对象和在容器中创建嵌套结构的逻辑设备。例如，您可以创建一个 *Images* 文件夹来存储图片和用于存储视频的 *Media* 文件夹。

您可以在各个项目中创建一个或多个容器，以及每个容器中的一个或多个对象或伪文件夹。



注意

要在云上执行 openstack 客户端命令，您必须指定 clouds.yaml 文件中详述的云名称。您可以使用以下方法之一指定云的名称：

- 在每个命令 中使用--os-cloud 选项：

```
$ openstack flavor list --os-cloud <cloud_name>
```

如果您访问多个云，则使用此选项。

- 在 bashrc 文件中为云名称创建一个环境变量：

```
`export OS_CLOUD=<cloud_name>`
```

先决条件

- 管理员已为您创建一个项目，已为您提供了一个 clouds.yaml 文件来访问云。
- 已安装 python-openstackclient 软件包。

4.1. 创建私有和公共容器

您可以创建私有或公共容器，以在对象存储服务(swift)中存储数据：

- private ：对项目成员的限制访问权限。
- Public：允许访问具有公共 URL 的任何人。

新容器使用默认的存储策略。如果您的 Red Hat OpenStack Services on OpenShift (RHOSO)部署定义了多个存储策略，例如，默认策略以及启用纠删代码的另一个策略，您可以将容器配置为使用非默认存储策略。

流程

1. 创建私有或公共容器：
   - 创建私有容器，以允许项目的成员列出容器中的对象，上传和下载对象。项目成员在其请求中包含项目的 Identity 服务(keystone)令牌：

```
$ openstack container create <container> \
    --read-acl "<project_id>":*" \
    --write-acl "<project_id>:*"
```

     - 将 <container > 替换为容器的名称。
     - 将 <project_id > 替换为项目的 ID。
   - 创建一个公共容器，以允许具有公共 URL 的任何人列出容器中的对象并从容器中下载对象：

```
$ openstack container create <container> \
    --read-acl ".r:*,.rlistings"
```

2. 将容器配置为使用非默认存储策略：

```
$ openstack container set -H "X-Storage-Policy:<policy>" <container>
```

- 将 & lt;policy > 替换为您要用于容器的策略的名称或别名。

4.2. 在容器中创建伪文件夹

您可以创建伪文件夹来组织 OpenStack Object Storage 服务(swift)中的容器中的数据。您可以创建一个伪文件夹，使用伪文件夹名称和正斜杠字符(/)的前缀来创建伪文件夹。

例如，如果您有一个名为 容器 的容器，并且您想要在名为 folder 的伪文件夹中组织对象，您可以在对象数据文件名称的开头添加 folder/ ： folder/object.ext。您可以以同样的方式创建嵌套伪文件夹，方法是在对象名称的开头包括嵌套文件夹的名称和正斜杠，如 folder/nested_folder/object.ext。

对象的 URL 将以 container/folder/object.ext 或 container/folder/nested_folder/object.ext 结尾。您可以使用带有 prefix 和 delimiter 参数的 GET 方法来导航伪文件夹。

流程

1. 上传对象并在容器中创建伪文件夹：

```
$ openstack object create <container> <pseudo_folder>/<object_filename>
```

   - 将 <container > 替换为容器的名称。
   - 将 < pseudo_folder > 替换为您要创建的伪文件夹的名称。
   - 将 <object_filename > 替换为对象数据文件的名称。

2. 上传对象并创建嵌套伪文件夹：

```
$ openstack object create <container> <pseudo_folder>/<nested_folder>/<object_filename>
```

   - 将 <nested_folder > 替换为嵌套伪文件夹的名称。

3. 在伪文件夹中查看对象列表，包括嵌套伪文件夹：

```
$ curl -X GET -i -H "X-Auth-Token: $token" \
 $publicurl/v1/<account>/<container>?prefix=<folder>&delimiter=/
```

- 将 <account > 替换为您的容器的命名空间，如 OpenShift (RHOSO)项目或租户上的 Red Hat OpenStack Services。

4.3. 从对象存储服务删除容器

如果要从 Object Storage 服务(swift)中删除容器，请确保首先删除容器中的所有对象。如需更多信息，请参阅从对象存储服务中删除对象。

流程

- 删除容器：

```
$ openstack container delete <container>
```

  - 将 <container> 替换为您要删除的容器的名称。

4.4. 将对象上传到容器

您可以将对象数据文件上传到对象存储服务(swift)中的容器或伪文件夹。或者，您可以创建对象作为容器或伪文件夹中的占位符，稍后将该文件上传到对象。

流程

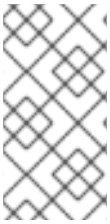
- 将对象上传到容器：

```
$ openstack object create <container> <object_filename>
```

  - 将 <container > 替换为容器的名称。
  - 将 <object_filename > 替换为对象数据文件的名称。

4.5. 在容器间复制对象

您可以将对象从源容器或伪文件夹复制到对象存储服务(swift)中的目标容器或伪文件夹。



注意

如果没有为目标对象指定唯一名称，它会保留与源对象相同的名称。如果您使用目标中已存在的名称，新对象会覆盖上一个对象的内容。

流程

- 将对象从一个容器复制到目标容器：

```
$ openstack copy --destination </container/object> \
        <container> <object> \
        [<object>] [...]
```

  - 将 **</container/object >** 替换为目标对象的容器和名称。
  - 将 **<container>** 替换为您要从中复制对象的容器的名称。
  - 将 **<object >** 替换为您要复制的对象的名称。您可以指定要复制的多个对象。

4.6. 从对象存储服务中删除对象

从 **Object Storage** 服务(swift)中的容器中删除对象。

流程

- 从容器中删除对象：

```
$ openstack object delete [--all] <container> <object> [...]
```

  - 将 **<container>** 替换为您要从中删除对象的容器的名称。
  - 将 **<object >** 替换为您要删除的对象名称。您可以指定要删除的多个对象。

- 可选： 要删除容器中的所有对象，请使用-- all 命令选项。

第 5 章 使用共享文件系统服务(MANILA)执行操作

您可以从共享文件系统服务(manila)中的可用共享类型创建和管理共享。

注意

要在云上执行 **openstack** 客户端命令，您必须指定 **clouds.yaml** 文件中详述的云名称。您可以使用以下方法之一指定云的名称：

- 在每个命令 中使用**--os-cloud** 选项：

```
$ openstack flavor list --os-cloud <cloud_name>
```

如果您访问多个云，则使用此选项。

- 在 **bashrc** 文件中为云名称创建一个环境变量：

```
`export OS_CLOUD=<cloud_name>`
```

先决条件

- 管理员已为您创建一个项目，已为您提供了一个 **clouds.yaml** 文件来访问云。
- 已安装 **python-openstackclient** 软件包。

5.1. 列出共享类型

在创建共享时，您必须指定一个共享类型，您只能创建与可用共享类型匹配的共享。配置的共享类型定义了共享文件系统服务调度程序用来做出调度决策以及驱动程序用来控制共享创建的服务类型。

流程

- 列出可用的共享类型：

```
$ openstack share type list
```

命令输出列出了可用共享类型的名称和 ID。

5.2. 创建 NFS、CEPHFS 或 CIFS 共享

您可以创建 CephFS-NFS、原生 CephFS 或 CIFS 共享来读取和写入数据。

在创建共享时，您必须指定共享协议和共享的大小（以 GB 为单位）。您还可以包含 share-type、share-network 和 name 命令选项：

```
$ openstack share create [--share-type <share_type>] \
      [--share-network <share_network>]  \
      [--name <share_name>] <share_protocol> <GB>
```

在命令示例中，替换以下值：

value	描述	必需或可选
<share_type>	应用与指定共享类型关联的设置	可选。如果没有指定共享类型，则使用默认共享类型。
<share_network>	共享网络的名称	• 如果共享类型将 **driver_handles_share_servers** 设置为 **true**，则需要此项。 • 如果共享类型将 **driver_handles_share_servers** 设置为 **false**，则不支持。 • CephFS-NFS 和原生 CephFS 不支持。这些协议不支持将 **driver_handles_share_servers** 设置为 **true** 的共享类型。
<share_name>	共享的名称	可选。共享不需要名称，并且名称不需要唯一。

value	描述	必需或可选
<share_protocol>	要使用的共享协议	• 对于 CephFS-NFS，将 <**share_protocol>** 替换为 **nfs**。 • 对于原生 CephFS，将 <**share_protocol>** 替换为 **cephfs**。 • 对于支持 NFS 或 CIFS 协议的其他存储后端，例如 NetApp 或 Dell EMC 存储后端，请将 < **share_protocol&** gt; 替换为 **nfs** 或 **cifs**。
<GB>	以 GB 为单位的共享大小	必需。

5.2.1. 使用 DHSS=true 创建 NFS 或 CIFS 共享

当共享类型额外规格时，driver_handles_share_servers 被设置为 true，您可以在共享网络中添加自己的安全服务来创建和导出 NFS 或 CIFS 共享。原生 CephFS 协议不支持共享网络。

要添加安全服务，您必须首先创建一个共享网络。如果您要创建 CIFS 共享，还必须创建一个安全服务资源来代表您的 Active Directory 服务器。然后，您将安全服务与共享网络关联。

如果您要创建 NFS 共享，则不需要安全服务，除非您想在共享中使用 Kerberos 或 LDAP 授权。

流程

1. **创建共享网络：**

```
$ openstack share network create --name <network_name> \
   --neutron-net-id <25d1e65c-d961-4f22-9476-1190f55f118f> \
   --neutron-subnet-id <8ba20dce-0ca5-4efd-bf1c-608d6bceffe1>
```

   - **将 < network_name > 替换为您要用于 NFS 或 CIFS 共享的共享网络名称。**
   - **将 neutron-net-id 和 neutron-subnet-id 替换为您的共享网络的正确值。**

2. **创建一个安全服务资源来代表您的活动目录服务器：**

```
$ openstack share security service create <active_directory> \
        --dns-ip <192.02.12.10> \
        --domain <domain_name.com> \
        --user <administrator> \
        --password <password> \
        --name <AD_service>
```

- 将尖括号 &lt ;& gt; 中的值替换为您的安全服务资源的正确详情。

3. 将安全服务资源与共享网络关联：

```
$ openstack share network set --new-security-service \
 <AD_service> <network_name>
```

4. 创建 NFS 或 CIFS 共享：

- 10 GB NFS 示例：

```
$ openstack share create --name <nfs_share> --share-type <netapp> \
   --share-network <nfs_network> nfs 10
```

- 20 GB CIFS 示例：

```
$ openstack share create --name <cifs_share> --share-type dhss_true \
   --share-network <cifs_network> cifs 20
```

  - 将尖括号 <& gt; 中的值替换为您的 NFS 或 CIFS 共享的正确详情。

5.2.2. 使用 DHSS=false 创建 NFS、CephFS 或 CIFS 共享

当共享类型额外规格时，driver_handles_share_servers 被设为'false'，您无法使用自定义安全服务，因为安全服务已在存储系统上直接配置。由于 CIFS 共享需要 Active Directory 服务器以及存储系统来管理访问控制，您的管理员必须预先创建 Active Directory 服务器并将其与存储系统关联以使用 CIFS 共享。

当 DHSS=false 时，您可以创建共享而无需使用 share-network 命令选项，因为共享存储网络被预先配置。

流程

- 当 DHSS=false 时，创建 NFS、原生 CephFS 或 CIFS 共享。这些示例 指定名称，但不指定 share-type 或 share-network。它们使用 默认共享类型 和配置的共享存储网络：
  - 创建一个名为 share-01 的 10 GB NFS 共享。

```
$ openstack share create --name share-01 nfs 10
```

  - 创建名为 share-02 的 15 GB 原生 CephFS 共享：

```
$ openstack share create --name share-02 cephfs 15
```

  - 创建名为 share-03 的 20 GB CIFS 共享：

```
$ openstack share create --name share-03 cifs 20
```

5.3. 列出共享和导出信息

要验证您在共享文件系统服务(manila)中成功创建了 NFS、CephFS 或 CIFS 共享，您可以列出共享并查看它们的导出位置和参数。

流程

1. 列出共享：

```
$ openstack share list
```

2. 查看共享的导出位置：

```
$ openstack share export location list <share>
```

   - 将 <share > 替换为共享名称或共享 ID。

3. 查看共享的参数：

```
$ openstack share export location show <share_id>
```

- 将 <share_id& gt; 替换为共享 ID。

注意

您可以使用导出位置来挂载共享，如 第 5.8.2 节 “挂载 NFS、原生 CephFS 或 CIFS 共享” 所述。

5.4. 在共享文件系统上创建数据快照

快照是共享上数据的只读点副本。您可以使用快照恢复通过意外删除或文件系统损坏而丢失的数据。快照比备份更高效，它们不会影响共享文件系统服务(manila)的性能。

先决条件

- 在父共享上，snapshot_support 参数必须是 true。您可以运行以下命令来验证：

```
$ openstack share show | grep snapshot_support
```

流程

1. 创建共享快照：

```
$ openstack share snapshot create [--name <snapshot_name>] <share>
```

- 将 <share> 替换为您要创建快照的共享的名称或 ID。
- 可选：将 <snapshot_name > 替换为快照的名称。

2. 确认您创建了快照：

```
$ openstack share snapshot list --share <share>
```

将 <share > 替换为您从其创建快照的共享 ID。

5.4.1. 从快照创建共享

您可以从快照创建共享。如果从 中创建快照的父共享具有 driver_handles_share_servers 的共享类型，则新共享会在与父级相同的共享网络上创建，您无法更改新共享的网络。

先决条件

- create_share_from_snapshot_support 共享属性设为 true。
- 快照的 status 属性设置为 available。

流程

1. 检索包含新共享所需数据的共享快照 ID：

```
$ openstack share snapshot list
```

2. 从快照创建的共享可能会大于快照，但不能小于快照。检索快照的大小：

```
$ openstack share snapshot show <snapshot_id>
```

   - 将 < snapshot_id > 替换为您要用来创建共享的快照 ID。

3. 从快照创建共享：

```
$ openstack share create <share_protocol> <size> \
  --snapshot-id <snapshot_id> \
  --name <name>
```

   - 将 <share_protocol > 替换为协议，如 NFS。
   - 将 < size> 替换为要创建的共享的大小，以 GiB 为单位。
   - 将 <name > 替换为新共享的名称。

4. 列出共享，以确认共享已创建成功：

```
$ openstack share list
```

5. 查看新共享的属性：

```
$ openstack share show <name>
```

验证

创建快照后，确认快照是否可用。

- 列出快照以确认它们可用：

```
$ openstack share snapshot list
```

5.4.2. 删除快照

当您创建共享的快照时，您无法删除共享，直到您删除从该共享创建的所有快照。

流程

1. 找到您要删除的快照并检索其 ID：

```
$ openstack share snapshot list
```

2. 删除快照：

```
$ share snapshot delete <snapshot>
```

- 将 & lt;snapshot > 替换为您要删除的快照的名称或 ID。

注意

对您要删除的每个快照重复此步骤。

3. 删除快照后，运行以下命令确认删除了快照：

```
$ share snapshot list
```

5.5. 连接到共享网络以访问共享

当 driver_handles_share_servers 参数(DHSS)等于 false 时，共享将导出到管理员提供的共享提供商网络。您必须将客户端（如计算实例）连接到共享提供商网络，以访问您的共享。

在本例中，共享提供商网络名为 StorageNFS。当使用 CephFS-NFS 后端部署共享文件系统服务(manila)时，StorageNFS 会被配置。按照以下步骤，连接到 OpenShift (RHOSO)部署中的 Red Hat OpenStack Services 中的可用网络。

注意

示例流程中的步骤使用 IPv4 寻址，但步骤与 IPv6 相同。

流程

1. 为 StorageNFS 端口创建一个安全组，允许数据包出口端口，但不允许来自未建立连接的入口数据包：

```
$ openstack security group create no-ingress -f yaml
created_at: '2018-09-19T08:19:58Z'
description: no-ingress
id: 66f67c24-cd8b-45e2-b60f-9eaedc79e3c5
name: no-ingress
project_id: 1e021e8b322a40968484e1af538b8b63
revision_number: 2
rules: 'created_at="2018-09-19T08:19:58Z", direction="egress", ethertype="IPv4",
 id="6c7f643f-3715-4df5-9fef-0850fb6eaaf2", updated_at="2018-09-19T08:19:58Z"
 created_at="2018-09-19T08:19:58Z", direction="egress", ethertype="IPv6",
 id="a8ca1ac2-fbe5-40e9-ab67-3e55b7a8632a", updated_at="2018-09-19T08:19:58Z"
```

2. 在 StorageNFS 网络中创建一个端口，其安全性由 no-ingress 安全组强制使用。

```
$ openstack port create nfs-port0 \
  --network StorageNFS \
  --security-group no-ingress -f yaml

admin_state_up: UP
allowed_address_pairs: ''
```

```
binding_host_id: null
binding_profile: null
binding_vif_details: null
binding_vif_type: null
binding_vnic_type: normal
created_at: '2018-09-19T08:03:02Z'
data_plane_status: null
description: ''
device_id: ''
device_owner: ''
dns_assignment: null
dns_name: null
extra_dhcp_opts: ''
fixed_ips: ip_address='198.51.100.160', subnet_id='7bc188ae-aab3-425b-a894-
863e4b664192'
id: 7a91cbbc-8821-4d20-a24c-99c07178e5f7
ip_address: null
mac_address: fa:16:3e:be:41:6f
name: nfs-port0
network_id: cb2cbc5f-ea92-4c2d-beb8-d9b10e10efae
option_name: null
option_value: null
port_security_enabled: true
project_id: 1e021e8b322a40968484e1af538b8b63
qos_policy_id: null
revision_number: 6
security_group_ids: 66f67c24-cd8b-45e2-b60f-9eaedc79e3c5
status: DOWN
subnet_id: null
tags: ''
trunk_details: null
updated_at: '2018-09-19T08:03:03Z'
```

注意

在本例中，StorageNFS 网络上的 StorageNFS 子网将 IP 地址 198.51.100.160 分配给 nfs-port0。

3. **将 nfs-port0 添加到 Compute 实例。**

```
$ openstack server add port instance0 nfs-port0
$ openstack server list -f yaml
- Flavor: m1.micro
  ID: 0b878c11-e791-434b-ab63-274ecfc957e8
  Image: manila-test
  Name: demo-instance0
  Networks: demo-network=198.51.100.4, 10.0.0.53; StorageNFS=198.51.100.160
  Status: ACTIVE
```

除了其专用和浮动地址外，计算实例还分配一个具有 StorageNFS 网络上 IP 地址

198.51.100.160 的端口。当向共享授予该地址的访问权限时，您可以使用此 IP 地址挂载 NFS 共享。

注意

您可能需要调整计算实例上的网络配置，然后重启 Compute 实例的服务来激活具有此地址的接口。

5.6. 在网络和实例之间配置 IPV6 接口

当导出的共享共享使用 IPv6 寻址时，您可能会在二级接口上使用 DHCPv6 遇到问题。如果出现这个问题，请在实例上手动配置 IPv6 接口。

先决条件

- 连接到共享网络以访问共享

流程

1. 登录实例。
2. 配置 IPv6 接口地址：

```
$ sudo ip address add fd00:fd00:fd00:7000::c/64 dev eth1
```

3. 激活接口：

```
$ sudo ip link set dev eth1 up
```

4. 在共享的导出位置 Ping IPv6 地址，以测试接口连接：

```
$ ping -6  fd00:fd00:fd00:7000::21
```

5. 或者，验证您可以通过 Telnet 访问 NFS 服务器：

```
$ sudo dnf install -y telnet
$ telnet fd00:fd00:fd00:7000::21 2049
```

5.7. 为最终用户客户端授予共享访问权限

在客户端上挂载共享（如计算实例）之前，您可以向最终用户授予共享访问权限，以便用户可以从客户端读取和写入数据到共享。

访问类型取决于共享的协议：

- 对于 CIFS 共享，请使用 CIFS 用户或组名称。
- 对于 NFS 共享，请使用您要挂载共享的 Compute 实例的 IP 地址。
- 对于原生 CephFS 共享，请使用 Ceph 客户端用户名进行 cephx 身份验证。

您可以使用以下命令授予对共享的访问权限：

```
$ openstack share access create <share> <access_type> \
  --access-level <access_level>  <client_identifier>
```

- 将 <share> 替换为您创建的共享的共享名称或 ID。
- 将 <access_type > 替换为您要授予共享的访问权限类型，例如，用于 CIFS 的用户，ip 用于 NFS，或用于原生 CephFS 的 cephfx。
- 可选：将 <access_level > 替换为 ro 以进行只读访问。对于读写访问，默认值为 rw。
- 将 client_identifier 替换为用于 NFS 的 NFS、用户或组名称的 IP 地址或原生 CephFS 的 Ceph 客户端用户名。对于 CIFS 和原生 CephFS，您可以在多个客户端间使用相同的 client_identifier。

5.7.1. 授予对 NFS 共享的访问权限

您可以使用计划挂载共享的客户端 Compute 实例的 IP 地址来提供对 NFS 共享的访问。

注意

您可以将以下步骤用于 IPv4 或 IPv6 地址。

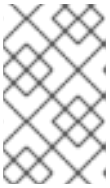
流程

- 检索您要挂载共享的客户端计算实例的 IP 地址。请确定您选择与可以访问共享的网络对应的 IP 地址。在本例中，它是 StorageNFS 网络的 IP 地址：

```
$ openstack server list -f yaml
- Flavor: m1.micro
  ID: 0b878c11-e791-434b-ab63-274ecfc957e8
  Image: manila-test
  Name: demo-instance0
  Networks: demo-network=198.51.100.4, 10.0.0.53;
  StorageNFS=198.51.100.160
  Status: ACTIVE

$ openstack share access create <share> ip 198.51.100.160
```

- 将 & lt;share > 替换为您要授予访问权限的共享的名称或 ID。



注意

对共享的访问权限有自己的 ID id。

```
+-----------------+--------------------------------------+
| Property        | Value                                |
+-----------------+--------------------------------------+
| access_key      | None
| share_id        | db3bedd8-bc82-4100-a65d-53ec51b5cba3
| created_at      | 2018-09-17T21:57:42.000000
| updated_at      | None
| access_type     | ip
| access_to       | 198.51.100.160
| access_level    | rw
| state           | queued_to_apply
| id              | 875c6251-c17e-4c45-8516-fe0928004fff
+-----------------+--------------------------------------+
```

验证

- 验证访问配置是否成功：

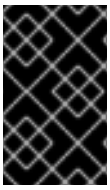
```
$ openstack share access list <share>

+--------------+------------+--------------+--------------+--------+ ...
| id           | access_type | access_to    | access_level | state  | ...
+--------------+------------+--------------+--------------+--------+
| 875c6251-... | ip          | 198.51.100.160 | rw         | active | ...
+--------------+------------+--------------+--------------+---------+ ...
```

5.7.2. 授予对原生 CephFS 共享的访问权限

您可以使用 Ceph 客户端用户名 cephx 身份验证来提供原生 CephFS 共享的访问权限。共享文件系统服务(manila)可防止使用预先存在的 Ceph 用户，因此您必须创建唯一的 Ceph 客户端用户名。

要挂载共享，您需要 Ceph 客户端用户名和访问密钥。您可以使用共享文件系统服务 API 检索访问密钥。默认情况下，访问密钥对项目命名空间中的所有用户可见。您可以提供对项目命名空间中的不同共享的访问权限。然后，用户可以使用客户端计算机上的 CephFS 内核客户端访问共享。



重要

仅将原生 CephFS 驱动程序与可信客户端一起使用。

流程

1. 授予用户对原生 CephFS 共享的访问权限：

```
$ openstack share access create <share> cephx <user>
```

   - 将 <share > 替换为共享名称或共享 ID。
   - 将 <user > 替换为 Ceph 客户端用户名。

2. 收集用户的访问密钥：

```
$ openstack share access list <share>
```

5.7.3. 授予对 CIFS 共享的访问权限

您可以使用 Active Directory 服务中的用户名授予对 CIFS 共享的访问权限。共享文件系统服务 (manila)不会在 Active Directory 服务器上创建新用户。它只通过安全服务验证用户名，使用无效的用户名访问规则 会导致错误 状态。

如果 driver_handles_share_servers (DHSS)参数的值被设置为 true，您可以通过添加安全服务来配置 Active Directory 服务。如果 DHSS 参数设置为 false，则您的管理员已配置了 Active Directory 服务，并将其与存储网络相关联。

要挂载共享，您必须指定用户的 Active Directory 用户名和密码。您无法通过共享文件系统服务获取此密码。

流程

- 授予用户对 CIFS 共享的访问权限：

```
$ openstack share access create <share> user <user>
```

  - 将 <share > 替换为共享名称或共享 ID。
  - 将 <user > 替换为 Active Directory 用户的用户名。

5.7.4. 撤销对共享的访问权限

共享的所有者可以撤销对共享的访问权限。完成以下步骤，撤销之前授予共享的访问权限。

流程

1. 查看共享的访问权限列表以检索访问 ID：

```
$ openstack share access list <share_01>
```

   - 将 <share_01 > 替换为共享名称或共享 ID。

2. 撤销对共享的访问权限：

```
$ openstack share access delete <share_01> <875c6251-c17e-4c45-8516-fe0928004fff>
```

- 将 <875c6251-c17e-4c45-8516-fe0928004fff > 替换为共享的访问 ID。

3. 再次查看共享的访问权限列表以验证共享已被删除：

```
$ openstack share access list <share_01>
```

注意

如果您有一个对共享具有读写访问权限的客户端，您必须撤销对共享的访问权限，如果您希望客户端具有只读访问权限，则必须添加只读规则。

5.8. 在 COMPUTE 实例上挂载共享

当您向客户端授予共享访问权限时，客户端可以挂载并使用共享。只要客户端有网络连接，任何类型的客户端访问共享。

用于在虚拟计算实例上挂载 NFS 共享的步骤与在裸机计算实例上挂载 NFS 共享的步骤类似。有关如何在 OpenShift 容器上挂载共享的更多信息，请参阅 Red Hat OpenShift Container Platform 产品文档。

注意

必须在挂载共享的 Compute 实例上安装不同协议的客户端软件包。例如，对于通过 NFS 使用 CephFS 的共享文件系统服务，NFS 客户端软件包必须支持 NFS 4.1。

5.8.1. 列出共享导出位置

检索共享的导出位置，以便您可以挂载共享。

流程

- 检索共享的导出位置：

```
$ openstack share export location list <share_01>
```

- 将 **<share_01 >** 替换为共享名称或共享 **ID**。

如果存在多个导出位置，选择一个 **preferred metadata** 字段的值等于 **True** 的位置。如果没有首选位置，您可以使用任何导出位置。

5.8.2. 挂载 NFS、原生 CephFS 或 CIFS 共享

当您创建 **NFS**、原生 **CephFS** 或 **CIFS** 共享并授予最终用户客户端的共享访问权限时，您可以在客户端上挂载共享以启用对数据的访问，只要有网络连接。

先决条件

- 要挂载 **NFS** 共享，必须在客户端计算机上安装 **nfs-utils** 软件包。
- 要挂载原生 **CephFS** 共享，必须在客户端计算机上安装 **ceph-common** 软件包。用户可以通过在客户端计算机上使用 **CephFS** 内核客户端来访问原生 **CephFS** 共享。
- 要挂载 **CIFS** 共享，必须在客户端计算机上安装 **cifs-utils** 软件包。

流程

1. 登录到实例：

```
$ openstack server ssh demo-instance0 --login user
```

2. 挂载 **NFS** 共享。如需示例语法，请参考以下示例：

```
$ mount -t nfs \
 -v <198.51.100.13:/volumes/_nogroup/e840b4ae-6a04-49ee-9d6e-67d4999fbc01>  \
 /mnt
```

- 将 **<198.51.100.13:/volumes/_nogroup/e840b4ae-6a04-49ee-9d6e-67d4999fbc01 >** 替换为共享的导出位置。

- 检索导出位置，如 第 5.8.1 节 “列出共享导出位置” 所述。

3. 挂载原生 CephFS 共享。如需示例语法，请参考以下示例：

```
$ mount -t ceph \
 <192.0.2.125:6789,192.0.2.126:6789,192.0.2.127:6789:/volumes/_nogroup/4c55ad20-9c55-4a5e-9233-8ac64566b98c> \
 -o name=<user>,secret='<AQA8+ANW/<4ZWNRAAOtWJMFPEihBA1unFImJczA==>'
```

- 将 <192.0.2.125:6789,192.0.2.126:6789,192.0.2.127:6789:6789:6789:/volumes/_nogroup/4c55ad20-9c5e-9233-8ac64566b98c > 替换为共享的导出位置。
- 检索导出位置，如 第 5.8.1 节 “列出共享导出位置” 所述。
- 将 <user > 替换为有权访问共享的 cephx 用户。
- 将 secret 值替换为您在 第 5.7.2 节 “授予对原生 CephFS 共享的访问权限” 中收集的访问密钥。

4. 挂载 CIFS 共享.如需示例语法，请参考以下示例：

```
$ mount -t cifs \
 -o user=<user>,pass=<password> \
 <\\192.0.2.128/share_11265e8a_200c_4e0a_a40f_b7a1117001ed>
```

- 将 <user > 替换为有权访问共享的 Active Directory 用户。
- 将 <password > 替换为用户的 Active Directory 密码。
- 将 <\\192.0.2.128/share_11265e8a_200c_4e0a_a40f_b7a1117001ed > 替换为共享的导出位置。
- 检索导出位置，如 第 5.8.1 节 “列出共享导出位置” 所述。

验证

- 验证 **mount** 命令是否成功：

```
$ df -k
```

5.9. 删除共享

共享文件系统服务(manila)提供保护以防止您删除数据。该服务不会检查客户端是否已连接，还是工作负载正在运行。当您删除共享时，您无法检索它。

警告

在删除共享前备份您的数据。

先决条件

- 如果从共享创建快照，您必须删除所有快照和副本，然后才能删除共享。如需更多信息，请参阅 删除快照。

流程

- 删除共享：

```
$ openstack share delete <share>
```

  - 将 **<share >** 替换为共享名称或共享 ID。

5.10. 列出共享文件系统服务的资源限值

您可以列出项目中共享文件系统服务(manila)的当前资源限值，以规划工作负载并根据资源消耗准备任何操作。

流程

- 列出项目的资源限值和当前资源消耗：

```
$ openstack share limits show --absolute
```

5.11. 操作失败的故障排除

如果在创建或挂载共享时出现错误，您可以从命令行运行查询以获取有关错误的更多信息。

5.11.1. 查看共享的错误消息

如果共享显示错误状态，可以使用命令行来检索用户支持消息。

流程

1. 创建共享时，运行以下命令来查看共享的状态：

```
$ openstack share list
```

2. 如果共享的状态显示错误，请运行 **share message list** 命令。您可以使用 the **-resource-id** 选项来过滤您要查找的特定共享：

```
$ openstack share message list [--resource-id]
```

3. 检查 **share message list** 命令的输出中的 **User Message** 列，以查找错误摘要。

4. 要查看有关错误的更多详细信息，请运行 **message show** 命令，后跟消息 列表命令的输出中的消息 **ID**：

```
$ openstack share message show <id>
```

- 将 **<id >** 替换为 **消息列表** 命令输出的消息 **ID**。

5.11.2. 调试共享挂载失败

您可以使用这些验证步骤来识别挂载共享时错误的根本原因。

流程

1. 验证共享的访问控制列表，以确保与您的客户端对应的规则正确且已成功应用：

```
$ openstack share access list <share_01>
```

- 将 <share_01 > 替换为共享名称或共享 ID。

  在成功规则中，state 属性等于 active。

2. 如果共享类型参数配置为 driver_handles_share_servers=false，请从导出位置复制主机名或 IP 地址，并 ping 它确认到 NAS 服务器的连接：

   Example:

```
$ ping -c 1 198.51.100.13
PING 198.51.100.13 (198.51.100.13) 56(84) bytes of data.
64 bytes from 198.51.100.13: icmp_seq=1 ttl=64 time=0.048 ms--- 198.51.100.13 ping
statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 7.851/7.851/7.851/0.000 ms
```

3. 如果您使用 NFS 协议，您可以验证 NFS 服务器是否已准备好响应正确端口上的 NFS RPC 调用：

```
$ rpcinfo -T tcp -a 198.51.100.13.8.1 100003 4
program 100003 version 4 ready and waiting
```

注意

IP 地址以通用地址格式(uaddr)编写，它添加了两个额外的 octets (8.1)来代表 NFS 服务端口 2049。

如果这些验证步骤失败，则可能会存在网络连接问题或共享文件系统服务(manila)的后端存储的问题。收集日志文件并联系红帽支持。