

## ► 解决方案

# 实施高级存储功能

在本练习中，您将使用 Stratis 存储管理解决方案来创建可根据数据需求的增加而增长的文件系统，并使用虚拟数据优化器创建卷，以高效利用存储空间。

## 成果

您应能够：

- 使用 Stratis 存储管理解决方案来创建精简配置的文件系统。
- 验证 Stratis 卷是否会动态增长，以支持实时数据增加。
- 通过精简配置的文件系统的快照访问数据。
- 使用虚拟数据优化器创建卷并将其挂载到文件系统上。
- 调查删除重复数据和压缩对虚拟数据优化器卷的影响。

## 在你开始之前

以 student 身份并使用 student 作为密码登录 workstation。

在 workstation 上，运行 **lab advstorage-review start** 以开始实验。此脚本可正确设置环境并确保将 serverb 上的其他磁盘清理干净。

```
[student@workstation ~]$ lab advstorage-review start
```

1. 从 workstation，以 student 用户身份打开连接 serverb 的 SSH 会话。

```
[student@workstation ~]$ ssh student@serverb
...output omitted...
[student@serverb ~]$
```

2. 切换到 root 用户。

```
[student@serverb ~]$ sudo -i
[sudo] password for student:
[root@serverb ~]#
```

3. 使用 yum 安装 stratisd 和 stratis-cli 软件包。

```
[root@serverb ~]# yum install stratisd stratis-cli
...output omitted...
Is this ok [y/N]: y
...output omitted...
Complete!
```

4. 使用 **systemctl** 命令启动并启用 stratisd 服务。

```
[root@serverb ~]# systemctl enable --now stratisd
```

5. 创建包含块设备 /dev/vdb 的 Stratis 池 labpool。

5.1. 使用 **stratis pool create** 命令创建 Stratis 池 labpool。

```
[root@serverb ~]# stratis pool create labpool /dev/vdb
```

5.2. 使用 **stratis pool list** 命令验证 labpool 的可用性。

```
[root@serverb ~]# stratis pool list
Name          Total Physical Size  Total Physical Used
labpool        5 GiB                 52 MiB
```

请注意上述输出中池的大小。

6. 利用系统中可用的磁盘 /dev/vdc 扩展 labpool 的容量。

6.1. 使用 **stratis pool add-data** 命令将块设备 /dev/vdc 添加到 labpool 中。

```
[root@serverb ~]# stratis pool add-data labpool /dev/vdc
```

6.2. 使用 **stratis pool list** 命令验证 labpool 的大小。

```
[root@serverb ~]# stratis pool list
Name          Total Physical Size  Total Physical Used
labpool        10 GiB                56 MiB
```

上述输出显示，将新磁盘添加到池中后，labpool 的大小已增加。

6.3. 使用 **stratis blockdev list** 命令列出现在是 labpool 成员的块设备。

```
[root@serverb ~]# stratis blockdev list labpool
Pool Name   Device Node   Physical Size   State   Tier
labpool    /dev/vdb       5 GiB           In-use  Data
labpool    /dev/vdc       5 GiB           In-use  Data
```

7. 在 labpool 池中创建精简配置的文件系统 labfs。将此文件系统挂载于 /labstratisvol，以实现在重启后持久保留。在 labfs 文件系统上创建一个名为 Labfile1 的文件，其中包含文本 Hello World!。别忘了在 /etc/fstab 中使用 x-systemd.requires=stratisd.service 挂载选项。

7.1. 使用 **stratis filesystem create** 命令，在 labpool 中创建精简配置的文件系统 labfs。此命令最多可能需要一分钟完成。

```
[root@serverb ~]# stratis filesystem create labpool labfs
```

7.2. 使用 **stratis filesystem list** 命令验证 labfs 的可用性。

```
[root@serverb ~]# stratis filesystem list
Pool Name      Name          Used     Created        Device
          UUID
labpool  labfs  546 MiB  Mar 29 2019 07:48  /stratis/labpool/labfs  9825...d6ca
```

请注意 **labfs** 的当前使用情况。在以下步骤中，文件系统的这种使用会按需增加。

7.3. 使用 **lsblk** 命令确定 **labfs** 的 UUID。

```
[root@serverb ~]# lsblk --output=UUID /stratis/labpool/labfs
UUID
9825e289-fb08-4852-8290-44d1b8f0d6ca
```

7.4. 编辑 **/etc/fstab**，以便在启动时挂载精简配置的文件系统 **labfs**。使用在上一步中确定的 UUID。以下显示了要在 **/etc/fstab** 中添加的行。您可以使用 **vi /etc/fstab** 命令来编辑文件。

```
UUID=9825...d6ca /labstratisvol xfs defaults,x-systemd.requires=stratisd.service
0 0
```

7.5. 使用 **mkdir** 命令创建名为 **/labstratisvol** 的目录。

```
[root@serverb ~]# mkdir /labstratisvol
```

7.6. 使用 **mount** 命令挂载精简配置的文件系统 **labfs**，确认 **/etc/fstab** 文件中是否包含相应的条目。

```
[root@serverb ~]# mount /labstratisvol
```

如果上述命令发生任何错误，请重新访问 **/etc/fstab** 文件并确保其中包含相应的条目。

7.7. 使用 **echo** 命令创建名为 **/labstratisvol/labfile1** 的文本文件。

```
[root@serverb ~]# echo "Hello World!" > /labstratisvol/labfile1
```

8. 验证精简配置的文件系统 **labfs** 是否随着文件系统上数据的增加而动态增长。

8.1. 使用 **stratis filesystem list** 命令查看 **labfs** 的当前使用情况。

```
[root@serverb ~]# stratis filesystem list
Pool Name      Name          Used     Created        Device
          UUID
labpool  labfs  546 MiB  Mar 29 2019 07:48  /stratis/labpool/labfs  9825...d6ca
```

8.2. 使用 **dd** 命令在 **labfs** 中创建一个 2 GiB 文件。此命令最多可能需要一分钟完成。

```
[root@serverb ~]# dd if=/dev/urandom of=/labstratisvol/labfile2 bs=1M count=2048
```

8.3. 使用 **stratis filesystem list** 命令，验证 labfs 的使用是否已增加。

```
[root@serverb ~]# stratis filesystem list
Pool Name      Name          Used       Created        Device
                UUID
labpool  labfs  2.53 GiB  Mar 29 2019 07:48  /stratis/labpool/labfs  9825...d6ca
```

9. 创建 labfs 文件系统的快照 labfs-snap。此快照允许您访问从 labfs 中删除的任何文件。

9.1. 使用 **stratis filesystem snapshot** 命令创建 labfs 的一个快照。此命令最多可能需要一分钟完成。

```
[root@serverb ~]# stratis filesystem snapshot labpool \
labfs labfs-snap
```

9.2. 使用 **stratis filesystem list** 命令验证快照的可用性。

```
[root@serverb ~]# stratis filesystem list
...output omitted...
labpool  labfs-snap  2.53 GiB  Mar 29 2019 10:28  /stratis/labpool/labfs-snap
291d...8a16
```

9.3. 删除 /labstratisvol/labfile1 文件。

```
[root@serverb ~]# rm /labstratisvol/labfile1
rm: remove regular file '/labstratisvol/labfile1'? y
```

9.4. 使用 **mkdir** 命令创建 /labstratisvol-snap 目录。

```
[root@serverb ~]# mkdir /labstratisvol-snap
```

9.5. 使用 **mount** 命令，将快照 labfs-snap 挂载于 /labstratisvol-snap。

```
[root@serverb ~]# mount /stratis/labpool/labfs-snap \
/labstratisvol-snap
```

9.6. 确认是否仍能使用快照 labfs-snap 访问从 labfs 中删除的文件。

```
[root@serverb ~]# cat /labstratisvol-snap/labfile1
Hello World!
```

10. 使用设备 /dev/vdd 创建 VDO 卷 labvdo。将其逻辑大小设置为 50 GB。

10.1. 使用 **vdo create** 命令创建卷 labvdo。

```
[root@serverb ~]# vdo create --name=labvdo --device=/dev/vdd -vdoLogicalSize=50G
...output omitted...
```

10.2. 使用 **vdo list** 命令验证卷 labvdo 的可用性。

```
[root@serverb ~]# vdo list  
labvdo
```

11. 采用 XFS 文件系统将卷 labvdo 挂载于 **/labvdovol**，以实现在重启后持久保留。别忘了在 **/etc/fstab** 中使用 **x-systemd.requires=vdo.service** 挂载选项。

11.1. 使用 **mkfs** 命令将 labvdo 卷格式化为 XFS 文件系统。

```
[root@serverb ~]# mkfs.xfs -K /dev/mapper/labvdo  
...output omitted...
```

11.2. 使用 **udevadm** 命令注册新设备节点。

```
[root@serverb ~]# udevadm settle
```

11.3. 使用 **mkdir** 命令创建 **/labvdovol** 目录。

```
[root@serverb ~]# mkdir /labvdovol
```

11.4. 使用 **lsblk** 命令确定 labvdo 的 UUID。

```
[root@serverb ~]# lsblk --output=UUID /dev/mapper/labvdo  
UUID  
ef8cce71-228a-478d-883d-5732176b39b1
```

11.5. 编辑 **/etc/fstab**，以便在启动时挂载 labvdo。使用在上一步中确定的 UUID。以下显示了要在 **/etc/fstab** 中添加的行。您可以使用 **vi /etc/fstab** 命令来编辑文件。

```
UUID=ef8c...39b1 /labvdovol xfs defaults,x-systemd.requires=vdo.service 0 0
```

11.6. 使用 **mount** 命令挂载 labvdo 卷，确认 **/etc/fstab** 文件中是否包含相应的条目。

```
[root@serverb ~]# mount /labvdovol
```

如果上述命令发生任何错误，请重新访问 **/etc/fstab** 文件并确保其中包含相应的条目。

12. 在 labvdo 卷上创建文件 **/root/install.img** 的三个副本。比较卷的统计信息，验证卷上是否发生数据重复删除和压缩的情况。

12.1. 使用 **vdostats** 命令查看卷的初始统计信息和状态。

```
[root@serverb ~]# vdostats --human-readable  
Device          Size     Used Available Use% Space saving%  
/dev/mapper/labvdo      5.0G    3.0G      2.0G  60%   99%
```

请注意，卷中有 3 GB 的空间已被占用，因为 VDO 卷在创建时会为自己保留 3-4 GB。此外还要注意，**Space saving%** 字段中的值 **99%** 表示您尚未在卷中创建任何内容，构成了所有节省的卷空间。

12.2. 将 **/root/install.img** 复制到 **/labvdovol/install.img.1** 并验证卷统计信息。  
复制文件最多可能需要一分钟。

```
[root@serverb ~]# cp /root/install.img /labvdovol/install.img.1
[root@serverb ~]# vdostats --human-readable
Device           Size     Used Available Use% Space saving%
/dev/mapper/labvdo    5.0G    3.4G    1.6G  68%      5%
```

请注意，**Used** 字段的值从 **3.0G** 增加至 **3.4G**，因为您向卷中复制了一个文件，这会占用一些空间。此外还要注意，**Space saving%** 字段的值从 **99%** 减少至 **5%**，因为最初卷中并没有内容，由此导致在其中创建文件前，卷空间利用率较低，而卷空间节省率较高。现在，由于您在卷中创建了该文件的唯一副本，里面没有要删除的重复数据，因此卷空间的节省率非常低。

12.3. 将 **/root/install.img** 复制到 **/labvdovol/install.img.2** 并验证卷统计信息。  
复制文件最多可能需要一分钟。

```
[root@serverb ~]# cp /root/install.img /labvdovol/install.img.2
[root@serverb ~]# vdostats --human-readable
Device           Size     Used Available Use% Space saving%
/dev/mapper/labvdo    5.0G    3.4G    1.6G  68%      51%
```

请注意，已用的卷空间并未发生改变。相反，节省的卷空间百分比还有所增加，这说明发生了数据重复删除以减少同一文件的冗余副本所占用的空间。上述输出中的 **Space saving%** 值可能在您的系统上有所不同。

12.4. 重新启动 serverb 计算机。

```
[root@serverb ~]# systemctl reboot
```



### 注意

注意：如果在重新启动时，serverb 无法启动到常规登录提示，而是显示“提供维护用的 root 密码（或按 **Control-D** 继续）：”，则说明 **/etc/fstab** 中可能存在错误。提供 **redhat** 的 root 密码后，您需要通过下列命令以读-写模式重新挂载 root 文件系统：

```
[root@serverb ~]# 安装 -o 重新装入, RW /
```

验证 **/etc/fstab** 是否按照解决方案中指定的方式进行了正确配置。请特别注意与 **/labstratisvol** 和 **/labvdovol** 有关的行的挂载选项。

## 评估

在 **workstation** 上，运行 **lab advstorage-review grade** 命令以确认本练习是否成功。

```
[student@workstation ~]$ lab advstorage-review grade
```

## 完成

在 workstation 上，运行 **lab advstorage-review finish** 来完成本练习。此脚本将删除在实验过程中创建的分区和文件，并确保环境清理干净。

```
[student@workstation ~]$ lab advstorage-review finish
```

本实验到此结束。

## 总结

---

在本章中，您学到了：

- Stratis 存储管理解决方案可实施灵活的文件系统，使之随数据动态增长。
- Stratis 存储管理解决方案支持精简配置、快照和监控功能。
- 虚拟数据优化器 (VDO) 旨在降低数据存储成本。
- 虚拟数据优化器采用零块消除、重复数据删除和数据压缩来优化磁盘空间使用效率。

## 章 9

# 访问网络附加存储

### 目标

使用 NFS 协议访问网络附加存储。

### 培训目标

- 启动时，从命令行挂载、使用和卸载 NFS 导出。
- 为自动挂载器配置直接和间接映射，以根据需要自动挂载 NFS 文件系统，并在不再使用时将其卸载。
- 借助新的 **nfsconf** 工具，配置 NFS 客户端以使用 NFSv4。

### 章节

- 通过 NFS 挂载网络附加存储（及引导式练习）
- 自动挂载网络附加存储（及引导式练习）

### 实验

访问网络附加存储

# 通过 NFS 挂载网络附加存储

## 培训目标

学完本节后，您应能够：

- 识别 NFS 共享信息。
- 创建要用作挂载点的目录。
- 使用 **mount** 命令或通过配置 **/etc/fstab** 文件来挂载 NFS 共享。
- 使用 **umount** 命令卸载 NFS 共享。
- 借助新的 **nfsconf** 工具，配置 NFS 客户端以使用 NFSv4。

## 挂载和卸载 NFS 共享

NFS（网络文件系统）是由 Linux、UNIX 及类似操作系统使用的互联网标准协议，可作为它们的本地网络文件系统。它是一种仍在积极增强的开放标准，可支持本地 Linux 权限和文件系统功能。

红帽企业 Linux 8 中的默认 NFS 版本为 4.2。支持 NFSv4 和 NFSv3 的主要版本。NFSv2 已不再受支持。NFSv4 仅使用 TCP 协议与服务器进行通信；较早的 NFS 版本可使用 TCP 或 UDP。

NFS 服务器导出共享（目录）。NFS 客户端将导出的共享挂载到本地挂载点（目录），该挂载点必须存在。可以通过多种方式挂载 NFS 共享：

- 使用 **mount** 命令手动挂载。
- 使用 **/etc/fstab** 条目在启动时自动挂载。
- 按需挂载：使用 **autofs** 服务或 **systemd.automount** 功能。

## 挂载 NFS 共享

要挂载 NFS 共享，请执行以下三个步骤：

1. **识别：**NFS 客户端系统的管理员可以通过各种方式识别可用的 NFS 共享：

NFS 服务器的管理员可以提供导出详细信息，包括安全性要求。

或者，客户端管理员可以通过挂载 NFS 服务器的根目录并浏览已导出目录来识别 NFSv4 共享。以 **root** 用户身份执行该操作。对使用 Kerberos 安全性的共享的访问将被拒绝，但共享（目录）名称将可见。可以浏览其他共享目录。

```
[user@host ~]$ sudo mkdir mountpoint  
[user@host ~]$ sudo mount server:/ mountpoint  
[user@host ~]$ sudo ls mountpoint
```

2. **挂载点：**使用 **mkdir** 在合适的位置创建挂载点。

```
[user@host ~]$ mkdir -p mountpoint
```

3. **挂载：**与分区上的文件系统一样，NFS 共享必须先进行挂载才可用。要挂载 NFS 共享，请从以下选项中进行选择。无论在哪种情况下，您都必须作为 **root** 用户登录，或使用 **sudo** 命令，以超级用户身份运行这些命令。

- 临时挂载：使用 **mount** 命令挂载 NFS 共享：

```
[user@host ~]$ sudo mount -t nfs -o rw,sync server:/share mountpoint
```

**-t nfs** 选项是 NFS 共享的文件系统类型（未严格要求，为完整性而显示）。**-o sync** 选项使 **mount** 立即与 NFS 服务器同步写操作（默认值为异步）。

此命令将立即但并不持久挂载共享；下一次系统启动时，此 NFS 共享将不可用。对于一次性访问数据的情况，该选项很有用。它也可用于在持久提供共享之前对挂载 NFS 共享进行测试。

- 持久挂载：为了确保在启动时挂载 NFS 共享，请编辑 **/etc/fstab** 文件以添加挂载条目。

```
[user@host ~]$ sudo vim /etc/fstab
...
server:/share /mountpoint nfs rw,soft 0 0
```

接着，挂载 NFS 共享：

```
[user@host ~]$ sudo mount /mountpoint
```

由于 NFS 客户端服务会在 **/etc/fstab** 文件中查找 NFS 服务器和挂载选项，因此您无需在命令行中指定这些内容。

## 卸载 NFS 共享

以 **root** 用户身份（或使用 **sudo**），使用 **umount** 命令卸载 NFS 共享。

```
[user@host ~]$ sudo umount mountpoint
```



### 注意

卸载共享不会删除它的 **/etc/fstab** 条目。除非删除或注释掉该条目，否则在下一次系统启动或重启 NFS 客户端服务时将重新挂载 NFS 共享。

## nfsconf 工具

红帽企业 Linux 8 引入了 **nfsconf** 工具，用于管理 NFSv4 与 NFSv3 下的 NFS 客户端和服务器配置文件。使用 **/etc/nfs.conf** 配置 **nfsconf** 工具（早期版本的操作系统中的 **/etc/sysconfig/nfs** 文件现已被弃用）。使用 **nfsconf** 工具来获取、设置或取消设置 NFS 配置参数。

**/etc/nfs.conf** 配置文件由多个部分组成，它的开头是一个位于方括号中的关键字 (**[keyword]**) 以及这一部分的值分配。对于 NFS 服务器，请配置 **[nfsd]** 部分。值的分配（或键）由值的名称等号和值的设置组成，例如 **vers4.2=y**。以 “#” 或 “;” 开头的行将被忽略，就像空白行一样。

```
[user@host ~]$ sudo cat /etc/nfs.conf
...output omitted...
[nfsd]
# debug=0
# threads=8
# host=
# port=0
# grace-time=90
# lease-time=90
# tcp=y
# vers2=n
# vers3=y
# vers4=y
# vers4.0=y
# vers4.1=y
# vers4.2=y
# rdma=n
#
...output omitted...
```

默认情况下，**[nfsd]** 部分的键值对会被注释掉。不过，注释中会显示在不做更改的情况下将会生效的默认选项。这为您配置 NFS 提供了一个很好的起点。

使用 **nfsconf --set section key value** 来设置指定部分的键值。

```
[user@host ~]$ sudo nfsconf --set nfsd vers4.2 y
```

此命令将更新 **/etc/nfs.conf** 配置文件：

```
[user@host ~]$ sudo cat /etc/nfs.conf
...output omitted...
[nfsd]
vers4.2 = y
# debug=0
# threads=8
# host=
# port=0
# grace-time=90
# lease-time=90
# tcp=y
# vers2=n
# vers3=y
# vers4=y
# vers4.0=y
# vers4.1=y
# vers4.2=y
# rdma=n
#
...output omitted...
```

使用 **nfsconf --get section key** 来检索指定部分的键值：

```
[user@host ~]$ sudo nfsconf --get nfssd vers4.2  
y
```

使用 **nfsconf --unset section key** 来取消设置指定部分的键值：

```
[user@host ~]$ sudo nfsconf --unset nfssd vers4.2
```

## 配置一个仅限使用 NFSv4 的客户端

通过在 **/etc/nfs.conf** 配置文件中设置以下值，可以配置一个仅限使用 NFSv4 的客户端。

首先禁用 UDP 以及其他与 NFSv2 和 NFSv3 有关的键：

```
[user@host ~]$ sudo nfsconf --set nfssd udp n  
[user@host ~]$ sudo nfsconf --set nfssd vers2 n  
[user@host ~]$ sudo nfsconf --set nfssd vers3 n
```

启用 TCP 和 NFSv4 相关键。

```
[user@host ~]$ sudo nfsconf --set nfssd tcp y  
[user@host ~]$ sudo nfsconf --set nfssd vers4 y  
[user@host ~]$ sudo nfsconf --set nfssd vers4.0 y  
[user@host ~]$ sudo nfsconf --set nfssd vers4.1 y  
[user@host ~]$ sudo nfsconf --set nfssd vers4.2 y
```

和以前一样，**/etc/nfs.conf** 配置文件中会显示所做的更改：

```
[[user@host ~]$ cat /etc/nfs.conf  
[nfssd]  
udp = n  
vers2 = n  
vers3 = n  
tcp = y  
vers4 = y  
vers4.0 = y  
vers4.1 = y  
vers4.2 = y  
...output omitted...
```



### 参考文献

**mount(8)**、**umount(8)**、**fstab(5)**、**mount.nfs(8)**、**nfs.conf(8)** 和 **nfsconf(8)**  
man page

## ► 指导练习

# 通过 NFS 管理网络附加存储

### 任务执行清单

在本练习中，您将修改 **/etc/fstab** 文件，以便在启动时持久挂载 NFS 导出。

### 成果

您应能够：

- 使用 **mount** 命令测试 NFS 服务器。
- 在 **/etc/fstab** 配置文件中配置 NFS 共享，即便在系统重新启动后也保存所做的更改。
- 借助新的 **nfsconf** 工具，配置 NFS 客户端以使用 NFSv4。

### 在你开始之前

以 **student** 身份并使用 **student** 作为密码登录 **workstation**。

在 **workstation** 上，运行 **lab netstorage-nfs start** 命令。该命令将运行一个起始脚本，它将确定 **servera** 和 **serverb** 计算机是否可从网络访问。如果不可用，脚本会提醒您。该起始脚本会将 **serverb** 配置为 NFSv4 服务器、设置权限并导出目录。它将创建 **servera** 和 **serverb** 所需的用户和组。

```
[student@workstation ~]$ lab netstorage-nfs start
```

航运公司使用中央服务器 **serverb** 来托管大量的共享文档和目录。**servera** 上的用户（均为 **admin** 组的成员）需要访问持久挂载的 NFS 共享。

重要信息：

- **serverb** 共享 **/shares/public** 目录，其中包含一些文本文件。
- **admin** 组的成员（**admin1**、**sysmanager1**）对 **/shares/public** 共享目录有读写访问权限。
- **servera** 的主要挂载点是 **/public**。
- 所有的用户密码都设置为 **redhat**。

#### ► 1. 以 **student** 用户身份登录 **servera** 并切换到 **root** 用户。

1.1. 以 **student** 用户身份登录 **servera**。

```
[student@workstation ~]$ ssh student@servera  
...output omitted...  
[student@servera ~]$
```

1.2. 使用 **sudo -i** 命令，切换为 **root** 用户。**student** 用户的密码为 **student**。

```
[student@servera ~]$ sudo -i
[sudo] password for student: student
[root@servera ~]#
```

- 2. 使用 **nfsconf** 工具配置 **/etc/nfs.conf**, 使 NFS 客户端启用为仅以版本 4.X 工作, 同时确保启用 TCP 模式并禁用 UDP 模式。

- 2.1. 使用 **nfsconf** 工具禁用键 **udp**、**vers2**、**vers3**。

```
[root@servera ~]# nfsconf --set nfssd udp n
[root@servera ~]# nfsconf --set nfssd vers2 n
[root@servera ~]# nfsconf --set nfssd vers3 n
```

- 2.2. 使用 **nfsconf** 工具启用键 **tcp**、**vers4**、**vers4.0**、**vers4.1**、**vers4.2**。

```
[root@servera ~]# nfsconf --set nfssd tcp y
[root@servera ~]# nfsconf --set nfssd vers4 y
[root@servera ~]# nfsconf --set nfssd vers4.0 y
[root@servera ~]# nfsconf --set nfssd vers4.1 y
[root@servera ~]# nfsconf --set nfssd vers4.2 y
```

- 3. 将 **servera** 用作 NFS 客户端, 在 **serverb** 上测试 NFS 服务器。

- 3.1. 在 **servera** 上创建 **/public** 挂载点。

```
[root@servera ~]# mkdir /public
```

- 3.2. 在 **servera** 上, 使用 **mount** 命令验证由 **serverb** 导出的 **/share/public** NFS 共享是否正确挂载到 **/public** 挂载点上。

```
[root@servera ~]# mount -t nfs serverb.lab.example.com:/shares/public /public
```

- 3.3. 列出已挂载的 NFS 共享的内容。

```
[root@servera ~]# ls -l /public
total 16
-rw-r--r--. 1 root admin 42 Apr  8 22:36 Delivered.txt
-rw-r--r--. 1 root admin 46 Apr  8 22:36 NOTES.txt
-rw-r--r--. 1 root admin 20 Apr  8 22:36 README.txt
-rw-r--r--. 1 root admin 27 Apr  8 22:36 Trackings.txt
```

- 3.4. 探索 NFS 挂载共享的 **mount** 选项。

```
[root@servera ~]# mount | grep public
serverb.lab.example.com:/shares/public on /public type nfs4
(rw,relatime,vers=4.2,rsiz...=262144,wsize=262144,namlen=255,sync,proto=tcp,timeo=600,
retrans=2,sec=sys,clientaddr=172.25.250.10,local_lock=none,addr=172.25.250.11)
```

- 3.5. 卸载 NFS 共享。

```
[root@servera ~]# umount /public
```

► 4. 配置 servera，确保上面使用的共享被持久挂载。

4.1. 打开 **/etc/fstab** 文件进行编辑。

```
[root@servera ~]# vim /etc/fstab
```

将下列行添加到该文件的末尾：

```
serverb.lab.example.com:/shares/public /public nfs rw,sync 0 0
```

4.2. 使用 **mount** 命令挂载共享目录。

```
[root@servera ~]# mount /public
```

4.3. 列出该共享目录的内容。

```
[root@servera ~]# ls -l /public
total 16
-rw-r--r--. 1 root    admin 42 Apr  8 22:36 Delivered.txt
-rw-r--r--. 1 root    admin 46 Apr  8 22:36 NOTES.txt
-rw-r--r--. 1 root    admin 20 Apr  8 22:36 README.txt
-rw-r--r--. 1 root    admin 27 Apr  8 22:36 Trackings.txt
```

4.4. 重新启动 servera 计算机。

```
[root@servera ~]# systemctl reboot
```

► 5. 在 servera 完成重启后，以 **admin1** 用户身份登录 servera 并测试持久挂载的 NFS 共享。

5.1. 以 **admin1** 用户身份登录 servera。

```
[student@workstation ~]$ ssh admin1@servera
[admin1@servera ~]$
```

5.2. 测试 **/public** 上挂载的 NFS 共享

```
[admin1@servera ~]$ ls -l /public
total 16
-rw-r--r--. 1 root    admin 42 Apr  8 22:36 Delivered.txt
-rw-r--r--. 1 root    admin 46 Apr  8 22:36 NOTES.txt
-rw-r--r--. 1 root    admin 20 Apr  8 22:36 README.txt
-rw-r--r--. 1 root    admin 27 Apr  8 22:36 Trackings.txt
[admin1@servera ~]$ cat /public/NOTES.txt
###In this file you can log all your notes###
```

```
[admin1@servera ~]$ echo "This is a test" > /public/Test.txt  
[admin1@servera ~]$ cat /public/Test.txt  
This is a test
```

5.3. 从 servera 注销。

```
[admin1@servera ~]$ exit  
logout  
Connection to servera closed.
```

## 完成

在 workstation 上，运行 **lab netstorage-nfs finish** 脚本来完成本练习。

```
[student@workstation ~]$ lab netstorage-nfs finish
```

本引导式练习到此结束。

# 自动挂载网络附加存储

## 培训目标

学完本节后，您应能够：

- 描述使用自动挂载器的优势。
- 使用直接映射和间接映射（包括通配符）自动挂载 NFS 共享。

## 使用自动挂载器挂载 NFS 共享

自动挂载器是一种服务 (**autofs**)，它可以“根据需要”自动挂载 NFS 共享，并将在不再使用 NFS 共享时自动卸载这些共享。

### 自动挂载器优势

- 用户无需具有 root 特权就可以运行 **mount** 和 **umount** 命令。
- 自动挂载器中配置的 NFS 共享可供计算机上的所有用户使用，受访问权限约束。
- NFS 共享不像 **/etc/fstab** 中的条目一样永久连接，从而可释放网络和系统资源。
- 自动挂载器在客户端配置，无需进行任何服务器端配置。
- 自动挂载器与 **mount** 命令使用相同的选项，包括安全性选项。
- 自动挂载器支持直接和间接挂载点映射，在挂载点位置方面提供了灵活性。
- **autofs** 可创建和删除间接挂载点，从而避免了手动管理。
- NFS 是默认的自动挂载器网络文件系统，但也可以自动挂载其他网络文件系统。
- **autofs** 是一种服务，其管理方式类似于其他系统服务。

## 创建自动挂载

配置自动挂载的过程包括多个步骤：

1. 安装 **autofs** 软件包。

```
[user@host ~]$ sudo yum install autofs
```

此软件包包含使用自动挂载器挂载 NFS 共享所需的所有内容。

2. 向 **/etc/auto.master.d** 添加一个主映射文件。此文件确定用于挂载点的基础目录，并确定用于创建自动挂载的映射文件。

```
[user@host ~]$ sudo vim /etc/auto.master.d/demo.autofs
```

主映射文件的名称是任意的（尽管通常会有一定的含意），但为了让子系统能够识别，它必须以 **.autofs** 作为扩展名。您可以在一个主映射文件中放置多个条目；或者，您可以创建多个主映射文件，且每个文件的条目都进行逻辑分组。

在此例中，为间接映射的挂载添加主映射条目：

```
/shares /etc/auto.demo
```

此条目将使用 **/shares** 目录作为间接自动挂载的基础目录。**/etc/auto.demo** 文件中包含挂载详细信息。请使用绝对文件名。需要在启动 **autofs** 服务之前创建 **auto.demo** 文件。

3. 创建映射文件。每个映射文件确定一组自动挂载的挂载点、挂载选项及挂载的源位置。

```
[user@host ~]$ sudo vim /etc/auto.demo
```

映射文件的命名规则是 **/etc/auto.name**，其中 name 反映了映射内容。

```
work -rw, sync serverb:/shares/work
```

条目的格式为挂载点、挂载选项和源位置。此示例显示基本的间接映射条目。本节稍后部分将讨论直接映射和使用通配符的间接映射。

- 挂载点在 man page 中被称为“密钥”，它由 **autofs** 服务自动创建和删除。在此例中，完全限定挂载点是 **/shares/work**（请参阅主映射文件）。**autofs** 服务将根据需要创建和删除 **/shares** 目录和 **/shares/work** 目录。

在此例中，本地挂载点将镜像服务器的目录结构，但这不是必需的；本地挂载点可以随意命名。**autofs** 服务不会在客户端上强制执行特定的命名结构。

- 挂载选项以短划线字符 (-) 开头，并使用逗号分隔，不带空格。自动挂载时，可以使用相应的挂载选项来手动挂载文件系统。在此例中，自动挂载器将挂载具有读/写访问权限的共享内容 (**rw** 选项)，并且在写入操作期间服务器会立即同步 (**sync** 选项)。

有用的自动挂载器特定选项包括 **-fstype=** 和 **-strict**。使用 **fstype** 指定文件系统类型，如 **nfs4** 或 **xfs**；挂载文件系统时，使用 **strict** 可将错误视为严重。

- NFS 共享的源位置遵循 **host:/pathname** 模式；在此示例中为 **&serverb;:/shares/work**。为了确保此次自动挂载成功，NFS 服务器 **serverb** 必须以读/写访问权限导出目录，而请求访问的用户必须对该目录具有标准 Linux 文件权限。如果 **serverb** 以只读访问权限导出目录，那么客户端也仅获得只读访问权限，即使它要求读/写访问权限。

4. 启动并启用自动挂载器服务。

使用 **systemctl** 启动并启用 **autofs** 服务。

```
[user@host ~]$ sudo systemctl enable --now autofs  
Created symlink /etc/systemd/system/multi-user.target.wants/autofs.service → /usr/  
lib/systemd/system/autofs.service.
```

## 直接映射

直接映射用于将 NFS 共享映射到现有的绝对路径挂载点。

要使用直接映射的挂载点，主映射文件可能如下所示：

```
/- /etc/auto.direct
```

所有直接映射条目都使用 `/-` 作为基础目录。在此例中，包含挂载详细信息的映射文件是 `/etc/auto.direct`。

`/etc/auto.direct` 文件的内容可能如下所示：

```
/mnt/docs -rw, sync serverb:/shares/docs
```

挂载点（或密钥）始终为绝对路径。映射文件的其余部分使用相同的结构。

在此例中，存在 `/mnt` 目录，它由 `autofs` 管理。`autofs` 服务将自动创建和删除整个 `/mnt/docs` 目录。

## 间接通配符映射

当 NFS 服务器导出一个目录中的多个子目录时，可将自动挂载程序配置为使用单个映射条目访问这些子目录其中的任何一个。

继续前面的示例，如果 `&serverb;:/shares` 导出两个或多个子目录，并且能够使用相同的挂载选项访问这些子目录，则 `/etc/auto.demo` 文件的内容可能如下所示：

```
* -rw, sync serverb:/shares/&
```

挂载点（或密钥）是星号字符 (\*)，而源位置上的子目录是 & 符号。条目中的所有其他内容都相同。

当用户尝试访问 `/shares/work` 时，密钥 \*（此例中为 `work`）将代替源位置中的 & 符号，并挂载 `&serverb;:/shares/work`。对于间接示例，`autofs` 将自动创建和删除 `work` 目录。



### 参考文献

`autofs(5)`、`automount(8)`、`auto.master(5)` 和 `mount.nfs(8)` man page

## ► 指导练习

# 自动挂载网络附加存储

### 任务执行清单

在本练习中，您将创建由自动挂载管理的直接映射和间接映射挂载点，上面挂载 NFS 文件系统。

### 成果

您应能够：

- 安装自动挂载器所需的软件包。
- 配置直接和间接自动挂载器映射，从预配置的 NFSv4 服务器获取资源。
- 了解直接和间接自动挂载器映射之间的区别。

### 在你开始之前

以 student 身份并使用 student 作为密码登录 workstation。

在 workstation 上，运行 **lab netstorage-autofs start** 命令。该起始脚本将确定 servera 和 serverb 是否可从网络访问。如果不可用，脚本会提醒您。该起始脚本会将 serverb 配置为 NFSv4 服务器、设置权限并导出目录。此外，它还将创建 servera 和 serverb 上所需的用户和组。

```
[student@workstation ~]$ lab netstorage-autofs start
```

互联网服务提供商使用中央服务器 serverb 来托管包含需要按需提供的共享目录。当用户登录 servera 时，他们需要访问自动挂载的共享目录。

重要信息：

- serverb 将作为 NFS 共享导出 **/shares/indirect** 目录，其中包含 **west**、**central** 和 **east** 子目录。
- serverb 也将作为 NFS 共享导出 **/shares/direct/external** 目录。
- operators 组由用户 operator1 和 operator2 组成。他们对共享目录 **/shares/indirect/west**、**/shares/indirect/central** 和 **/shares/indirect/east** 具有读写访问权限。
- contractors 组由用户 contractor1 和 contractor2 组成。他们对共享目录 **/shares/direct/external** 具有读写访问权限。
- servera 预期的挂载点是 **/external** 和 **/internal**。
- 应使用 **/external** 上的直接映射将 **/shares/direct/external** 共享目录自动挂载到 servera 上。

- 应使用 **/internal/west** 上的间接映射将 **/shares/indirect/west** 共享目录自动挂载到 **servera** 上。
- 应使用 **/internal/central** 上的间接映射将 **/shares/indirect/central** 共享目录自动挂载到 **servera** 上。
- 应使用 **/internal/east** 上的间接映射将 **/shares/indirect/east** 共享目录自动挂载到 **servera** 上。
- 所有的用户密码都设置为 **redhat**。

► 1. 登录 **servera** 并安装所需的软件包。

1.1. 以 **student** 用户身份登录 **servera**。

```
[student@workstation ~]$ ssh student@servera
...output omitted...
[student@servera ~]$
```

1.2. 使用 **sudo -i** 命令，切换为 **root** 用户。**student** 用户的密码为 **student**。

```
[student@servera ~]$ sudo -i
[sudo] password for student: student
[root@servera ~]#
```

1.3. 安装 **autofs** 软件包。

```
[root@servera ~]# yum install autofs
...output omitted...
Is this ok [y/N]: y
...output omitted...
```

► 2. 利用 **serverb** 共享，配置 **servera** 上的自动挂载器直接映射。使用名为 **/etc/auto.master.d/direct.autofs** 的文件为主映射创建直接映射，同时为映射文件创建 **/etc/auto.direct**。使用 **/external** 目录作为 **servera** 上的主挂载点。

2.1. 在继续配置自动挂载器之前，测试 NFS 服务器和共享。

```
[root@servera ~]# mount -t nfs \
serverb.lab.example.com:/shares/direct/external /mnt
[root@servera ~]# ls -l /mnt
total 4
-rw-r--r-- 1 root contractors 22 Apr  7 23:15 README.txt
[root@servera ~]# umount /mnt
```

2.2. 创建一个名为 **/etc/auto.master.d/direct.autofs** 的主映射文件，插入以下内容并保存更改。

```
[root@servera ~]# vim /etc/auto.master.d/direct.autofs
/- /etc/auto.direct
```

2.3. 创建一个名为 **/etc/auto.direct** 的直接映射文件，插入以下内容并保存更改。

```
[root@servera ~]# vim /etc/auto.direct
/external -rw, sync, fstype=nfs4 serverb.lab.example.com:/shares/direct/external
```

- 3. 利用 serverb 共享，配置 servera 上的自动挂载器间接映射。使用名为 **/etc/auto.master.d/indirect.autofs** 的文件为主映射创建间接映射，同时为映射文件创建 **/etc/auto.indirect**。使用 **/internal** 目录作为 servera 上的主挂载点。

3.1. 在继续配置自动挂载器之前，测试 NFS 服务器和共享。

```
[root@servera ~]# mount -t nfs serverb.lab.example.com:/shares/indirect /mnt
[root@servera ~]# ls -l /mnt
total 0
drwxrws---. 2 root operators 24 Apr  7 23:34 central
drwxrws---. 2 root operators 24 Apr  7 23:34 east
drwxrws---. 2 root operators 24 Apr  7 23:34 west
[root@servera ~]# umount /mnt
```

3.2. 创建一个名为 **/etc/auto.master.d/indirect.autofs** 的主映射文件，插入以下内容并保存更改。

```
[root@servera ~]# vim /etc/auto.master.d/indirect.autofs
/internal /etc/auto.indirect
```

3.3. 创建一个名为 **/etc/auto.indirect** 的间接映射文件，插入以下内容并保存更改。

```
[root@servera ~]# vim /etc/auto.indirect
* -rw, sync, fstype=nfs4 serverb.lab.example.com:/shares/indirect/&
```

- 4. 在 servera 上启动并启用 autofs 服务，使其在启动时自动启动。重新启动 servera，以确定 autofs 服务是否已自动启动。

4.1. 在 servera 上启动并启用 autofs 服务。

```
[root@servera ~]# systemctl enable --now autofs
Created symlink /etc/systemd/system/multi-user.target.wants/autofs.service → /usr/
lib/systemd/system/autofs.service.
```

4.2. 重新启动 servera 计算机。

```
[root@servera ~]# systemctl reboot
```

- 5. 以 contractor1 用户身份测试直接自动挂载器映射。完成后，退出 servera 上的 contractor1 用户会话。

5.1. 在 servera 计算机完成启动后，以 student 用户身份登录 servera。

```
[student@workstation ~]$ ssh student@servera
...output omitted...
[student@servera ~]$
```

5.2. 切换到 contractor1 用户。

```
[student@servera ~]$ su - contractor1  
Password: redhat
```

5.3. 列出 /external 挂载点。

```
[contractor1@servera ~]$ ls -l /external  
total 4  
-rw-r--r--. 1 root contractors 22 Apr 7 23:34 README.txt
```

5.4. 查看内容并测试对 /external 挂载点的访问权限。

```
[contractor1@servera ~]$ cat /external/README.txt  
###External Folder###  
[contractor1@servera ~]$ echo testing-direct > /external/testing.txt  
[contractor1@servera ~]$ cat /external/testing.txt  
testing-direct
```

5.5. 退出 contractor1 用户会话。

```
[contractor1@servera ~]$ exit  
logout
```

► 6. 以 operator1 用户身份测试间接自动挂载器映射。完成后，从 servera 注销。

6.1. 切换到 operator1 用户。

```
[student@servera ~]$ su - operator1  
Password: redhat
```

6.2. 列出 /internal 挂载点。

```
[operator1@servera ~]$ ls -l /internal  
total 0
```



### 注意

您会注意到，在自动挂载器间接映射中，即使位于映射的挂载点，也需要按需调用每个共享子目录或文件才能访问它们。在自动挂载器直接映射中，打开映射的挂载点后，便可以访问共享目录中配置的目录和内容。

6.3. 测试 /internal/west 自动挂载器共享目录的访问权限。

```
[operator1@servera ~]$ ls -l /internal/west/  
total 4  
-rw-r--r--. 1 root operators 18 Apr 7 23:34 README.txt  
[operator1@servera ~]$ cat /internal/west/README.txt  
###West Folder###
```

```
[operator1@servera ~]$ echo testing-1 > /internal/west/testing-1.txt
[operator1@servera ~]$ cat /internal/west/testing-1.txt
testing-1
[operator1@servera ~]$ ls -l /internal
total 0
drwxrws---. 2 root operators 24 Apr  7 23:34 west
```

6.4. 测试 **/internal/central** 自动挂载器共享目录的访问权限。

```
[operator1@servera ~]$ ls -l /internal/central
total 4
-rw-r--r--. 1 root operators 21 Apr  7 23:34 README.txt
[operator1@servera ~]$ cat /internal/central/README.txt
###Central Folder###
[operator1@servera ~]$ echo testing-2 > /internal/central/testing-2.txt
[operator1@servera ~]$ cat /internal/central/testing-2.txt
testing-2
[operator1@servera ~]$ ls -l /internal
total 0
drwxrws---. 2 root operators 24 Apr  7 23:34 central
drwxrws---. 2 root operators 24 Apr  7 23:34 west
```

6.5. 测试 **/internal/east** 自动挂载器共享目录的访问权限。

```
[operator1@servera ~]$ ls -l /internal/east
total 4
-rw-r--r--. 1 root operators 18 Apr  7 23:34 README.txt
[operator1@servera ~]$ cat /internal/east/README.txt
###East Folder###
[operator1@servera ~]$ echo testing-3 > /internal/east/testing-3.txt
[operator1@servera ~]$ cat /internal/east/testing-3.txt
testing-3
[operator1@servera ~]$ ls -l /internal
total 0
drwxrws---. 2 root operators 24 Apr  7 23:34 central
drwxrws---. 2 root operators 24 Apr  7 23:34 east
drwxrws---. 2 root operators 24 Apr  7 23:34 west
```

6.6. 测试 **/external** 自动挂载器共享目录的访问权限。

```
[operator1@servera ~]$ ls -l /external
ls: cannot open directory '/external': Permission denied
```

6.7. 从 servera 注销。

```
[operator1@servera ~]$ exit
logout
[student@servera ~]$ exit
logout
Connection to servera closed.
```