

```
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size   File system  Name      Flags
 1       1049kB  1000MB  999MB  xfs          usersdata
```

3. 删除分区。

```
(parted) rm 1
```

**rm** 子命令会立即从磁盘上的分区表中删除该分区。

4. 退出 **parted**。

```
(parted) quit
Information: You may need to update /etc/fstab.

[root@host ~]#
```

## 创建文件系统

创建块设备后，下一步是向其中添加文件系统。红帽企业 Linux 支持许多不同的文件系统类型，其中两种常见的类型是 XFS 和 ext4。红帽企业 Linux 的安装程序 Anaconda 默认使用 XFS。

以 **root** 用户身份，使用 **mkfs.xfs** 命令为块设备应用 XFS 文件系统。对于 ext4，请使用 **mkfs.ext4**。

```
[root@host ~]# mkfs.xfs /dev/vdb1
meta-data=/dev/vdb1              isize=512    agcount=4, agsize=60992 blks
                                =           sectsz=512  attr=2, projid32bit=1
                                =           crc=1        finobt=1, sparse=1, rmapbt=0
                                =           reflink=1
data     =           bsize=4096   blocks=243968, imaxpct=25
                                =           sunit=0        swidth=0 blks
naming  =version 2               bsize=4096   ascii-ci=0, ftype=1
log      =internal log          bsize=4096   blocks=1566, version=2
                                =           sectsz=512  sunit=0 blks, lazy-count=1
realtime =none                  extsz=4096   blocks=0, rtextents=0
```

## 挂载文件系统

添加完文件系统后，最后一步是将文件系统挂载到目录结构中的目录上。将文件系统挂载到目录层次结构上后，用户空间实用程序可以访问设备上的文件或在设备上写入文件。

### 手动挂载文件系统

管理员可以使用 **mount** 命令将设备手动附加到目录位置（或挂载点）。**mount** 命令预期的参数包括设备、挂载点和文件系统选项（可选）。文件系统选项将自定义文件系统的行为。

```
[root@host ~]# mount /dev/vdb1 /mnt
```

您还可以使用 **mount** 命令来查看当前已挂载的文件系统、挂载点和选项。

```
[root@host ~]# mount | grep vdb1
/dev/vdb1 on /mnt type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

## 持久挂载文件系统

手动挂载文件系统是一种验证已格式化的设备是否可访问以及是否按预期方式工作的好方法。但是，当服务器重启时，系统不会再次将文件系统自动挂载到目录树上；文件系统上的数据将完好无损，但用户却无法访问。

为了确保系统在启动时自动挂载文件系统，请在 **/etc/fstab** 文件中添加一个条目。此配置文件列出了在系统启动时要挂载的文件系统。

**/etc/fstab** 是以空格分隔的文件，每行具有六个字段。

```
[root@host ~]# cat /etc/fstab
#
# /etc/fstab
# Created by anaconda on Wed Feb 13 16:39:59 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
UUID=a8063676-44dd-409a-b584-68be2c9f5570    /      xfs    defaults    0 0
UUID=7a20315d-ed8b-4e75-a5b6-24ff9e1f9838    /dbdata  xfs    defaults    0 0
```

在 **/etc/fstab** 文件中添加或删除条目时，请运行 **systemctl daemon-reload** 命令或重启服务器，以便让 **systemd** 注册新配置。

```
[root@host ~]# systemctl daemon-reload
```

第一个字段指定设备。本示例使用 UUID 来指定设备。创建时，文件系统会在其超级块中创建和存储 UUID。或者，您可以使用设备文件，例如 **/dev/vdb1**。

**注意**

使用 UUID 更为可取，因为块设备标识符在特定情况下可能会变化，例如当云提供商更改虚拟机的基础存储层或在每次系统启动会以不同顺序检测磁盘时。块设备文件名可能会发生改变，但 UUID 在文件系统的超级块中会保持不变。

使用 **lsblk --fs** 命令可扫描连接到计算机的块设备并检索文件系统 UUID。

```
[root@host ~]# lsblk --fs
NAME   FSTYPE LABEL UUID                                     MOUNTPOINT
sr0
vda
└─vda1 xfs      a8063676-44dd-409a-b584-68be2c9f5570 /
vdb
└─vdb1 xfs      7a20315d-ed8b-4e75-a5b6-24ff9e1f9838 /dbdata
```

第二个字段是目录挂载点，通过它可以访问目录结构中的块设备。挂载点必须存在；如果不存在，请使用 **mkdir** 命令进行创建。

第三个字段包含文件系统类型，如 **xfs** 或 **ext4**。

第四个字段是以逗号分隔的、应用于设备的选项列表。**defaults** 是一组常用选项。**mount(8)** man page 中还记录了其他可用的选项。

**dump** 命令使用第五个字段来备份设备。其他备份应用通常不使用此字段。

最后一个字段（**fsck** 顺序字段）决定了在系统启动时是否应运行 **fsck** 命令，以验证文件系统是否干净。该字段中的值指示了 **fsck** 的运行顺序。对于 XFS 文件系统，请将该字段设为 **0**，因为 XFS 并不使用 **fsck** 来检查自己的文件系统状态。对于 ext4 文件系统，如果是根文件系统，请将该字段设为 **1**；如果是其他 ext4 文件系统，则将该字段设为 **2**。这样，**fsck** 就会先处理根文件系统，然后同步检查不同磁盘上的文件系统，并按顺序检查同一磁盘上的文件系统。

**注意**

**/etc/fstab** 中存在错误的条目可能会导致计算机无法启动。管理员应卸载新文件系统，然后使用 **mount /mountpoint**（该命令会读取 **/etc/fstab**）重新挂载该文件系统，以验证条目是否有效。如果 **mount** 命令返回错误，请在重新启动计算机之前纠正该错误。

作为替代方法，您可以使用 **findmnt --verify** 命令来控制 **/etc/fstab** 文件。

**参考文献**

**info parted** (GNU Parted 用户手册)

**parted(8)**、**mkfs(8)**、**mount(8)**、**lsblk(8)** 和 **fstab(5)** man page

如需更多信息，请参阅《文件系统配置与管理指南》，网址为：  
[https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/8/html-single/configuring\\_and\\_managing\\_file\\_systems/](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/configuring_and_managing_file_systems/)

## ► 指导练习

# 添加分区、文件系统和持久挂载

在本练习中，您将在新存储设备上创建一个分区，将它格式化为 XFS 文件系统，配置为在启动时挂载该分区，然后进行挂载以供使用。

## 成果

您应能够使用 **parted**、**mkfs.xfs** 及其他命令在新磁盘上创建分区、对其格式化并进行持久挂载。

## 在你开始之前

在 **workstation** 上，以 **student** 用户身份并使用密码 **student** 进行登录。

在 **workstation** 上，运行 **lab storage-partitions start** 命令。此命令将运行一个起始脚本，它将确定 **servera** 计算机是否可从网络访问。此外，它还在 **servera** 上准备了另一个磁盘，供练习使用。

```
[student@workstation ~]$ lab storage-partitions start
```

- 1. 使用 **ssh** 命令，以 **student** 用户身份登录 **servera**。系统已配置为使用 SSH 密钥来进行身份验证，因此不需要提供密码。

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

- 2. 使用 **sudo -i** 命令，切换为 **root** 用户。收到提示时，使用 **student** 作为密码。

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

- 3. 使用 **parted** 在 **/dev/vdb** 磁盘上创建类型为 **msdos** 的新磁盘标签，以便为 MBR 分区方案准备新的磁盘。

```
[root@servera ~]# parted /dev/vdb mklabel msdos
Information: You may need to update /etc/fstab.
```

- 4. 添加一个大小为 1 GB 的新主分区。为了确保正确对齐，从扇区 2048 开始分区。将分区的文件系统类型设为 XFS。

- 4.1. 使用 **parted** 交互模式帮助您创建分区。

```
[root@servera ~]# parted /dev/vdb
GNU Parted 3.2
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mkpart
Partition type? primary/extended? primary
File system type? [ext2]? xfs
Start? 2048s
End? 1001MB
(parted) quit
Information: You may need to update /etc/fstab.
```

由于分区从扇区 2048 开始，因此上一个命令将结束位置设为 1001MB 可获得大小为 1000MB (1 GB) 的分区。

或者，您可以使用以下非交互式命令执行同一操作：**parted /dev/vdb mkpart primary xfs 2048s 1001MB**

#### 4.2. 通过列出 **/dev/vdb** 上分区，验证您的工作。

```
[root@servera ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type      File system  Flags
 1       1049kB  1001MB  1000MB  primary
```

#### 4.3. 运行 **udevadm settle** 命令。此命令会等待系统注册新分区，并在完成后返回。

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

### ► 5. 将新分区格式化为 XFS 文件系统。

```
[root@servera ~]# mkfs.xfs /dev/vdb1
meta-data=/dev/vdb1              isize=512    agcount=4, agsize=61056 blks
                                =          sectsz=512  attr=2, projid32bit=1
                                =          crc=1        finobt=1, sparse=1, rmapbt=0
                                =          reflink=1
data     =          bsize=4096   blocks=244224, imaxpct=25
                                =          sunit=0    swidth=0 blks
naming   =version 2             bsize=4096   ascii-ci=0, ftype=1
log      =internal log          bsize=4096   blocks=1566, version=2
                                =          sectsz=512  sunit=0 blks, lazy-count=1
realtime =none                  extsz=4096   blocks=0, rtextents=0
```

### ► 6. 将新的文件系统配置为持久挂载于 **/archive**。

#### 6.1. 使用 **mkdir** 创建 **/archive** 目录挂载点。

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

6.2. 使用 **lsblk** 命令及 **--fs** 选项以发现设备 **/dev/vdb1** 的 UUID。

```
[root@servera ~]# lsblk --fs /dev/vdb  
NAME   FSTYPE LABEL UUID                                     MOUNTPOINT  
vdb  
└─vdb1  xfs   e3db1abe-6d96-4faa-a213-b96a6f85dcc1
```

上一个输出中的 UUID 可能与您的系统的 UUID 有所不同。

6.3. 向 **/etc/fstab** 添加一个条目。在以下命令中，将 UUID 替换为您在上一步中发现的 UUID。

```
[root@servera ~]# vim /etc/fstab  
...output omitted...  
UUID=e3db1abe-6d96-4faa-a213-b96a6f85dcc1  /archive  xfs  defaults  0  0
```

6.4. 更新系统的 **systemd** 以注册新的 **/etc/fstab** 配置。

```
[root@servera ~]# systemctl daemon-reload  
[root@servera ~]#
```

6.5. 使用向 **mount /archive** 添加的新条目，执行 **/etc/fstab** 命令以挂载新文件系统。

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

6.6. 验证新文件系统是否已挂载于 **/archive**。

```
[root@servera ~]# mount | grep /archive  
/dev/vdb1 on /archive type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

► 7. 重启 **servera**。服务器重新启动后，登录并验证 **/dev/vdb1** 是否已挂载于 **/archive**。完成后，从 **servera** 注销。

7.1. 重启 **servera**。

```
[root@servera ~]# systemctl reboot  
Connection to servera closed by remote host.  
Connection to servera closed.  
[student@workstation ~]$
```

7.2. 等待几分钟，让 **servera** 重新启动，然后以 **student** 用户身份登录。

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

7.3. 验证 **/dev/vdb1** 是否已挂载于 **/archive**。

```
[student@servera ~]$ mount | grep /archive  
/dev/vdb1 on /archive type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

7.4. 从 **servera** 注销。

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

## 完成

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

```
[student@workstation ~]$ lab storage-partitions finish
```

本引导式练习到此结束。

# 管理交换空间

## 培训目标

学完本节后，您应能够创建和管理交换空间以补充物理内存。

## 交换空间概念简介

交换空间是受 Linux 内核内存管理子系统控制的磁盘区域。内核使用交换空间，通过保存不活动的内存页来补充系统 RAM。系统 RAM 与交换空间组合在一起称为虚拟内存。

当系统上的内存使用量超过定义的限制时，内核将搜索 RAM，寻找已分配给进程但空闲的内存页。内核将空闲的内存页写入到交换区，并向其他进程重新分配 RAM 页面。如果某个程序需要访问磁盘上的页面，则内核会找到另一个空闲的内存页，将其写入到磁盘，然后从交换区重新调用所需的页面。

由于交换区位于磁盘上，所以与 RAM 相比，交换会比较慢。虽然是用于增加系统 RAM，但对于 RAM 不足以满足工作负载需求的问题，不应将交换空间视为可持续性的解决方案。

## 调整交换空间大小

管理员应根据系统的内存工作负载来调整交换空间大小。应用供应商有时会提供这方面的建议。根据物理内存总量，下表提供了一些指导。

### RAM 和交换空间建议

RAM	交换空间	允许 HIBERNATE 时的交换空间
2 GiB 或以下	两倍的 RAM	三倍的 RAM
介于 2 GiB 和 8 GiB 之间	同等的 RAM	两倍的 RAM
介于 8 GiB 和 64 GiB 之间	至少 4 GiB	1.5 倍的 RAM
64 GiB 以上	至少 4 GiB	不建议 Hibernate

笔记本电脑和台式机的 Hibernate 功能会在关闭系统电源之前使用交换空间来保存 RAM 内容。重新打开系统时，内核将从交换空间恢复 RAM 内容，无需完全启动。对于这些系统而言，交换空间需要超过 RAM 量。

本节末尾“参考资料”部分中的知识库文章提供了有关调整交换空间大小的更多指导。

## 创建交换空间

要创建交换空间，您需要执行以下操作：

- 创建文件系统类型为 **linux-swap** 的分区。
- 为设备放置交换签名。

## 创建交换分区

使用 **parted** 可以创建所需大小的分区并将其文件系统类型设置为 **linux-swap**。在过去，工具会根据分区文件系统类型来确定是否应激活设备，但现在情况已不再如此。即使实用程序不再使用分区文件系统类型，设置此类型也可以使管理员快速确定该分区的用途。

以下示例将创建一个 256 MB 分区。

```
[root@host ~]# parted /dev/vdb
GNU Parted 3.2
Using /dev/vdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name  Flags
 1      1049kB  1001MB  1000MB          data

(parted) mkpart
Partition name? []? swap1
File system type? [ext2]? linux-swap
Start? 1001MB
End? 1257MB
(parted) print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system        Name  Flags
 1      1049kB  1001MB  1000MB          data
 2      1001MB  1257MB  256MB   linux-swap(v1)  swap1

(parted) quit
Information: You may need to update /etc/fstab.

[root@host ~]#
```

创建分区后，运行 **udevadm settle** 命令。此命令会等待系统检测新分区并在 **/dev** 中创建关联的设备文件。只有在完成上述操作后，它才会返回。

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

## 格式化设备

**mkswap** 命令向设备应用交换签名。与其他格式化实用程序不同，**mkswap** 在设备开头写入单个数据块，而将设备的其余部分保留为未格式化，这样内核就可以使用它来存储内存页。

```
[root@host ~]# mkswap /dev/vdb2
Setting up swap space version 1, size = 244 MiB (255848448 bytes)
no label, UUID=39e2667a-9458-42fe-9665-c5c854605881
```

## 激活交换空间

您可以使用 **swapon** 命令激活已格式化的交换空间。

使用 **swapon** 并将设备作为参数，或者使用 **swapon -a** 来激活 **/etc/fstab** 文件中列出的所有交换空间。使用 **swapon --show** 和 **free** 命令检查可用的交换空间。

```
[root@host ~]# free
              total        used         free      shared  buff/cache   available
Mem:       1873036      134688      1536436          0      16748      201912      1576044
Swap:            0          0          0
[root@host ~]# swapon /dev/vdb2
[root@host ~]# free
              total        used         free      shared  buff/cache   available
Mem:       1873036      135044      1536040          0      16748      201952      1575680
Swap:      249852          0      249852
```

您可以使用 **swapoff** 命令停用交换空间。如果交换空间具有写入的页面，**swapoff** 会尝试将这些页面移动到其他活动交换空间或将其写回到内存中。如果无法将数据写入到其他位置，则 **swapoff** 命令会失败，并显示错误，而交换空间将仍保持活动。

## 持久激活交换空间

要想在每次启动时都激活交换空间，请在 **/etc/fstab** 文件中放置一个条目。基于上面创建的交换空间，以下示例显示了 **/etc/fstab** 中比较典型的一行。

```
UUID=39e2667a-9458-42fe-9665-c5c854605881    swap    swap    defaults    0 0
```

该示例使用 UUID 作为第一个字段。格式化设备时，**mkswap** 命令会显示该 UUID。如果丢失了 **mkswap** 的输出，请使用 **lsblk --fs** 命令。作为替代方法，您也可以在第一个字段中使用设备名称。

第二个字段通常为 mount point 保留。但是，由于交换设备无法通过目录结构访问，因此该字段取占位符值 **swap**。

第三个字段是文件系统类型。交换空间的文件系统类型是 **swap**。

第四个字段是选项。本示例中使用了 **defaults** 选项。**defaults** 选项包括挂载选项 **auto**，它用于在系统启动时自动激活交换空间。

最后两个字段是 **dump** 标志和 **fsck** 顺序。交换空间既不需要备份，也不需要文件系统检查，因此应将这些字段设为 0。

在 **/etc/fstab** 文件中添加或删除条目时，请运行 **systemctl daemon-reload** 命令或重启服务器，以便让 **systemd** 注册新配置。

```
[root@host ~]# systemctl daemon-reload
```

## 设置交换空间优先级

默认情况下，系统会按顺序使用交换空间，即内核先使用第一个已激活交换空间，直至其空间已满，然后开始使用第二个交换空间。不过，您也可以为每个交换空间定义一个优先级，从而强制按该顺序使用交换空间。

要设置优先级，请在 **/etc/fstab** 中使用 **pri** 选项。内核会首先使用优先级最高的交换空间。默认优先级为 -2。

以下示例显示了 **/etc/fstab** 中定义的三个交换空间。内核首先使用最后一个条目，其优先级为 **pri=10**。当该空间已满时，它将使用第二个条目，其优先级为 **pri=4**。最后，它将使用第一个条目，其默认优先级为 -2。

```
UUID=af30cbb0-3866-466a-825a-58889a49ef33    swap      swap      defaults  0 0
UUID=39e2667a-9458-42fe-9665-c5c854605881    swap      swap      pri=4      0 0
UUID=fb7fa60-b781-44a8-961b-37ac3ef572bf     swap      swap      pri=10     0 0
```

使用 **swapon --show** 可以显示交换空间的优先级。

当交换空间具有相同的优先级时，内核会以轮循方式向其中写入。



### 参考文献

**mkswap(8)**、**swapon(8)**、**swapoff(8)**、**mount(8)** 和 **parted(8)** man page

**知识库：对于红帽平台，建议的交换空间大小是多少？**

<https://access.redhat.com/solutions/15244>

## ► 指导练习

# 管理交换空间

在本练习中，您将创建并格式化用作交换空间的分区，将其格式化为交换分区，并持久激活。

## 成果

您应能够使用 GPT 分区方案在磁盘上创建分区和交换空间。

## 在你开始之前

在 workstation 上，以 student 用户身份并使用密码 student 进行登录。

在 workstation 上，运行 **lab storage-swap start** 命令。此命令将运行一个起始脚本，它将确定 servera 计算机是否可从网络访问。此外，它还在 servera 上准备了另一个磁盘，供练习使用。

```
[student@workstation ~]$ lab storage-swap start
```

- 1. 使用 **ssh** 命令，以 student 用户身份登录 servera。系统已配置为使用 SSH 密钥来进行身份验证，因此不需要提供密码。

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

- 2. 使用 **sudo -i** 命令，切换为 root 用户。收到提示时，使用 student 作为密码。

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

- 3. 使用 **parted** 命令检查 /dev/vdb 磁盘。

```
[root@servera ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Partition Flags:

Number  Start   End     Size    File system  Name  Flags
 1      1049kB  1001MB  1000MB          data
```

请注意，磁盘已经有分区表，并使用 GPT 分区方案。同样，1 GB 分区也已存在。

- 4. 添加一个大小为 500 MB 的新分区，用作交换空间。将分区类型设置为 Linux swap。

- 4.1. 使用 **parted** 创建分区。由于磁盘使用了 GPT 分区方案，因此需要为分区指定名称。将它命名为 **myswap**。

```
[root@servera ~]# parted /dev/vdb mkpart myswap linux-swap 1001MB 1501MB
Information: You may need to update /etc/fstab.
```

请注意，在上一个命令中，起始位置 1001 MB 就是现有第一个分区的结束位置。这样，**parted** 便可确保新分区紧跟在前一个分区的后面，中间没有任何间隔。

由于分区的起始位置为 1001 MB，因此该命令将结束位置设为 1501 MB 可获得大小为 500 MB 的分区。

- 4.2. 通过列出 **/dev/vdb** 上分区，验证您的工作。

```
[root@servera ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name     Flags
 1      1049kB  1001MB  1000MB          data
 2      1001MB  1501MB  499MB          myswap  swap
```

新分区的大小并不是恰好为 500 MB。这是因为 **parted** 必须让分区与磁盘布局对齐。

- 4.3. 运行 **udevadm settle** 命令。此命令会等待系统注册新分区，并在完成后返回。

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

#### ► 5. 将新创建分区初始化为交换空间。

```
[root@servera ~]# mkswap /dev/vdb2
Setting up swap space version 1, size = 476 MiB (499118080 bytes)
no label, UUID=cb7f71ca-ee82-430e-ad4b-7dda12632328
```

#### ► 6. 启用新创建的交换空间。

- 6.1. 使用 **swapon --show** 命令可显示创建和初始化交换空间并未将它启用。

```
[root@servera ~]# swapon --show
[root@servera ~]#
```

#### 6.2. 启用新创建的交换空间。

```
[root@servera ~]# swapon /dev/vdb2
[root@servera ~]#
```

#### 6.3. 验证新创建的交换空间现在是否可用。

```
[root@servera ~]# swapon --show
NAME      TYPE      SIZE USED PRIO
/dev/vdb2 partition 476M   0B   -2
```

6.4. 禁用该交换空间。

```
[root@servera ~]# swapoff /dev/vdb2
[root@servera ~]#
```

6.5. 确认该交换空间是否已禁用。

```
[root@servera ~]# swapon --show
[root@servera ~]#
```

## ► 7. 将新交换空间配置为在系统启动时启用。

7.1. 使用 **lsblk** 命令及 **--fs** 选项以发现设备 **/dev/vdb2** 的 UUID。

```
[root@servera ~]# lsblk --fs /dev/vdb2
NAME FSTYPE LABEL UUID                                     MOUNTPOINT
vdb2 swap          cb7f71ca-ee82-430e-ad4b-7dda12632328
```

上一个输出中的 UUID 可能与您的系统的 UUID 有所不同。

7.2. 向 **/etc/fstab** 添加一个条目。在以下命令中，将 UUID 替换为您在上一步中发现的 UUID。

```
[root@servera ~]# vim /etc/fstab
...output omitted...
UUID=cb7f71ca-ee82-430e-ad4b-7dda12632328    swap    swap    defaults    0    0
```

7.3. 更新系统的 **systemd** 以注册新的 **/etc/fstab** 配置。

```
[root@servera ~]# systemctl daemon-reload
[root@servera ~]#
```

7.4. 使用刚添加到 **/etc/fstab** 的条目启用交换空间。

```
[root@servera ~]# swapon -a
[root@servera ~]#
```

7.5. 验证新交换空间是否已启用。

```
[root@servera ~]# swapon --show
NAME      TYPE      SIZE USED PRIO
/dev/vdb2 partition 476M   0B   -2
```

## ► 8. 重启 **servera**。服务器重新启动后，登录并验证该交换空间是否已启用。完成后注销 **servera**。

8.1. 重启 servera。

```
[root@servera ~]# systemctl reboot
Connection to servera closed by remote host.
Connection to servera closed.
[student@workstation ~]$
```

8.2. 等待几分钟，让 servera 重新启动，然后以 student 用户身份登录。

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

8.3. 验证该交换空间是否已启用。

```
[root@servera ~]# swapon --show
NAME      TYPE      SIZE USED PRIO
/dev/vdb2  partition 476M   0B   -2
```

8.4. 从 servera 注销。

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

## 完成

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

```
[student@workstation ~]$ lab storage-swap finish
```

本引导式练习到此结束。

## ► 开放研究实验

### 管理基本存储

#### 任务执行清单

在本实验中，您将在新磁盘上创建多个分区，将其中一些分区格式化为相应的文件系统且予以挂载，并激活其他分区作为交换空间。

#### 成果

您应能够：

- 使用 **parted** 命令显示和创建分区。
- 在分区上创建新文件系统并进行持久挂载。
- 创建交换空间并在启动时将其激活。

#### 在你开始之前

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

在 **workstation** 上，运行 **lab storage-review start** 命令。此命令将运行一个起始脚本，它将确定 **serverb** 计算机是否可从网络访问。此外，它还在 **serverb** 上准备了另一个磁盘，供练习使用。

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

1. 在 **serverb** 上有新的磁盘可用。在第一个新磁盘上，创建一个名为 **backup** 的 2 GB GPT 分区。由于可能难以设置确切的大小，因此介于 1.8 GB 和 2.2 GB 之间的大小都是可以接受的。为该分区设置正确的文件系统类型，以托管 XFS 文件系统。  
**serverb** 上的 **student** 用户帐户的密码为 **student**。通过 **sudo**，该用户具有完全 **root** 访问权限。
2. 将 2 GB 分区格式化为 XFS 文件系统并持久挂载于 **/backup**。
3. 在同一新磁盘上创建两个 512 MB GPT 分区，分别命名为 **swap1** 和 **swap2**。介于 460 MB 和 564 MB 之间的大小是可以接受的。为这些分区设置正确的文件系统类型，以托管交换空间。
4. 将两个 512 MiB 分区初始化为交换空间，并将它们配置为在启动时激活。将 **swap2** 分区上的交换空间设置为优先于另一个交换空间。
5. 要验证您的工作，请重启 **serverb**。确认系统是否自动将第一个分区挂载于 **/backup**。同时，确认系统是否激活了这两个交换空间。

完成后，从 **serverb** 注销。

#### 评估

在 **workstation** 上，运行 **lab storage-review grade** 脚本来确认是否成功完成本练习。

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

## 完成

在 workstation 上，运行 **lab storage-review finish** 脚本来完成实验。

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

本实验到此结束。

## ► 解决方案

# 管理基本存储

## 任务执行清单

在本实验中，您将在新磁盘上创建多个分区，将其中一些分区格式化为相应的文件系统且予以挂载，并激活其他分区作为交换空间。

## 成果

您应能够：

- 使用 **parted** 命令显示和创建分区。
- 在分区上创建新文件系统并进行持久挂载。
- 创建交换空间并在启动时将其激活。

## 在你开始之前

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

在 **workstation** 上，运行 **lab storage-review start** 命令。此命令将运行一个起始脚本，它将确定 **serverb** 计算机是否可从网络访问。此外，它还在 **serverb** 上准备了另一个磁盘，供练习使用。

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

1. 在 **serverb** 上有新的磁盘可用。在第一个新磁盘上，创建一个名为 **backup** 的 2 GB GPT 分区。由于可能难以设置确切的大小，因此介于 1.8 GB 和 2.2 GB 之间的大小都是可以接受的。为该分区设置正确的文件系统类型，以托管 XFS 文件系统。  
**serverb** 上的 **student** 用户帐户的密码为 **student**。通过 **sudo**，该用户具有完全 **root** 访问权限。

- 1.1. 使用 **ssh** 命令，以 **student** 用户身份登录 **serverb**。系统已配置为使用 SSH 密钥来进行身份验证，因此不需要提供密码。

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

- 1.2. 由于创建分区和文件系统需要 **root** 访问权限，因此请使用 **sudo -i** 命令切换到 **root** 用户。收到提示时，使用 **student** 作为密码。

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

- 1.3. 使用 **lsblk** 命令来标识新磁盘。这些磁盘应该还没有任何分区。

```
[root@serverb ~]# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sr0     11:0    1 1024M  0 rom
vda    252:0    0   10G  0 disk
└─vda1 252:1    0   10G  0 part /
vdb   252:16   0    5G  0 disk
vdc    252:32   0    5G  0 disk
vdd    252:48   0    5G  0 disk
```

注意第一个新磁盘 **vdb** 没有任何分区。

1.4. 确认磁盘没有标签。

```
[root@serverb ~]# parted /dev/vdb print
Error: /dev/vdb: unrecognised disk label
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: unknown
Disk Flags:
```

1.5. 使用 **parted** 和 **mklabel** 子命令，定义 GPT 分区方案。

```
[root@serverb ~]# parted /dev/vdb mklabel gpt
Information: You may need to update /etc/fstab.
```

1.6. 创建 2 GB 分区。将其命名为 **backup**，将其类型设为 **xfs**。从扇区 2048 开始分区。

```
[root@serverb ~]# parted /dev/vdb mkpart backup xfs 2048s 2GB
Information: You may need to update /etc/fstab.
```

1.7. 确认是否已正确创建新分区。

```
[root@serverb ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size   File system  Name   Flags
 1      1049kB  2000MB  1999MB          backup
```

1.8. 运行 **udevadm settle** 命令。此命令会等待系统检测新分区并创建 **/dev/vdb1** 设备文件。只有在完成上述操作后，它才会返回。

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

2. 将 2 GB 分区格式化为 XFS 文件系统并持久挂载于 **/backup**

2.1. 使用 **mkfs.xfs** 命令格式化 **/dev/vdb1** 分区。

```
[root@serverb ~]# mkfs.xfs /dev/vdb1
meta-data=/dev/vdb1              isize=512    agcount=4, agsize=121984 blks
                                =          sectsz=512  attr=2, projid32bit=1
                                =          crc=1     finobt=1, sparse=1, rmapbt=0
                                =          reflink=1
data     =           bsize=4096   blocks=487936, imaxpct=25
                                =          sunit=0   swidth=0 blks
naming   =version 2             bsize=4096   ascii-ci=0, ftype=1
log      =internal log          bsize=4096   blocks=2560, version=2
                                =          sectsz=512  sunit=0 blks, lazy-count=1
realtime =none                 extsz=4096   blocks=0, rtextents=0
```

2.2. 创建 **/backup** 挂载点。

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

2.3. 在向 **/etc/fstab** 中添加新文件系统之前，检索其 UUID。

```
[root@serverb ~]# lsblk --fs /dev/vdb1
NAME FSTYPE LABEL UUID                                     MOUNTPOINT
vdb1 xfs      a3665c6b-4bfb-49b6-a528-74e268b058dd
```

您的系统的 UUID 可能有所不同。

2.4. 编辑 **/etc/fstab** 并定义新文件系统。

```
[root@serverb ~]# vim /etc/fstab
...output omitted...
UUID=a3665c6b-4bfb-49b6-a528-74e268b058dd  /backup  xfs  defaults  0 0
```

2.5. 强制 **systemd** 重新读取 **/etc/fstab** 文件。

```
[root@serverb ~]# systemctl daemon-reload
[root@serverb ~]#
```

2.6. 手动挂载 **/backup** 以验证您的工作。确认挂载是否成功。

```
[root@serverb ~]# mount /backup
[root@serverb ~]# mount | grep /backup
/dev/vdb1 on /backup type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

3. 在同一新磁盘上创建两个 512 MB GPT 分区，分别命名为 **swap1** 和 **swap2**。介于 460MB 和 564 MB 之间的大小是可以接受的。为这些分区设置正确的文件系统类型，以托管交换空间。

- 3.1. 通过在 **/dev/vdb** 上显示当前分区表来检索第一个分区的结束位置。在下一步中，将使用该值作为 **swap1** 分区的起始位置。

```
[root@serverb ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size   File system  Name   Flags
 1      1049kB  2000MB  1999MB  xfs          backup
```

- 3.2. 创建第一个 512 MB 分区并命名为 **swap1**。将该分区的类型设为 **linux-swap**。使用第一个分区的结束位置作为起始点。结束位置为  $2000\text{ MB} + 512\text{ MB} = 2512\text{ MB}$

```
[root@serverb ~]# parted /dev/vdb mkpart swap1 linux-swap 2000MB 2512M
Information: You may need to update /etc/fstab.
```

- 3.3. 创建第二个 512 MB 分区并命名为 **swap2**。将该分区的类型设为 **linux-swap**。使用上一个分区的结束位置 **2512M** 作为起始点。结束位置为  $2512\text{ MB} + 512\text{ MB} = 3024\text{ MB}$

```
[root@serverb ~]# parted /dev/vdb mkpart swap2 linux-swap 2512M 3024M
Information: You may need to update /etc/fstab.
```

- 3.4. 显示分区表以验证您的工作。

```
[root@serverb ~]# parted /dev/vdb print
Model: Virtio Block Device (virtblk)
Disk /dev/vdb: 5369MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size   File system  Name   Flags
 1      1049kB  2000MB  1999MB  xfs          backup
 2      2000MB  2512MB  513MB    swap1        swap
 3      2512MB  3024MB  512MB    swap2        swap
```

- 3.5. 运行 **udevadm settle** 命令。此命令会等待系统注册新分区并创建设备文件。

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

4. 将两个 512 MiB 分区初始化为交换空间，并将它们配置为在启动时激活。将 **swap2** 分区上的交换空间设置为优先于另一个交换空间。

- 4.1. 使用 **mkswap** 命令初始化交换分区。

```
[root@serverb ~]# mkswap /dev/vdb2
Setting up swap space version 1, size = 489 MiB (512749568 bytes)
no label, UUID=87976166-4697-47b7-86d1-73a02f0fc803
[root@serverb ~]# mkswap /dev/vdb3
Setting up swap space version 1, size = 488 MiB (511700992 bytes)
no label, UUID=4d9b847b-98e0-4d4e-9ef7-dfaaf736b942
```

记下两个交换空间的 UUID。您会在下一步中使用此信息。如果再也看不到 **mkswap** 输出，则使用 **lsblk --fs** 命令来检索 UUID。

- 4.2. 编辑 **/etc/fstab** 并定义新的交换空间。要将 **swap2** 分区上的交换空间设置为优先于 **swap1**，请通过 **pri** 选项为它赋予一个更高的优先级。

```
[root@serverb ~]# vim /etc/fstab
...output omitted...
UUID=a3665c6b-4bfb-49b6-a528-74e268b058dd  /backup xfs  defaults  0 0
UUID=87976166-4697-47b7-86d1-73a02f0fc803  swap    swap  pri=10  0 0
UUID=4d9b847b-98e0-4d4e-9ef7-dfaaf736b942  swap    swap  pri=20  0 0
```

- 4.3. 强制 **systemd** 重新读取 **/etc/fstab** 文件。

```
[root@serverb ~]# systemctl daemon-reload
[root@serverb ~]#
```

- 4.4. 使用 **swapon -a** 命令激活新交换空间。使用 **swapon --show** 命令确认交换空间是否已正确激活。

```
[root@serverb ~]# swapon -a
[root@serverb ~]# swapon --show
NAME      TYPE      SIZE USED PRIO
/dev/vdb2  partition 489M   0B   10
/dev/vdb3  partition 488M   0B   20
```

5. 要验证您的工作，请重启 **serverb**。确认系统是否自动将第一个分区挂载于 **/backup**。同时，确认系统是否激活了这两个交换空间。

完成后，从 **serverb** 注销。

- 5.1. 重启 **serverb**。

```
[root@serverb ~]# systemctl reboot
[root@serverb ~]#
Connection to serverb closed by remote host.
Connection to serverb closed.
[student@workstation ~]$
```

- 5.2. 等待几分钟，让 **serverb** 重新启动，然后以 **student** 用户身份登录。

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

5.3. 验证系统是否自动将 **/dev/vdb1** 挂载于 **/backup**。

```
[student@serverb ~]$ mount | grep /backup  
/dev/vdb1 on /backup type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

5.4. 使用 **swapon --show** 命令确认系统是否激活了这两个交换空间。

```
[student@serverb ~]$ swapon --show  
NAME      TYPE      SIZE USED PRIO  
/dev/vdb2 partition 489M   0B   10  
/dev/vdb3 partition 488M   0B   20
```

5.5. 从 **serverb** 注销。

```
[student@serverb ~]$ exit  
logout  
Connection to serverb closed.  
[student@workstation ~]$
```

## 评估

在 **workstation** 上，运行 **lab storage-review grade** 脚本来确认是否成功完成本练习。

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

## 完成

在 **workstation** 上，运行 **lab storage-review finish** 脚本来完成实验。

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

本实验到此结束。

## 总结

---

在本章中，您学到了：

- 使用 **parted** 命令在采用 MBR 或 GPT 分区方案的磁盘上添加、修改和删除分区。
- 使用 **mkfs.xfs** 命令在磁盘分区上创建 XFS 文件系统。
- 要使这些挂载持久保留，需要为 **/etc/fstab** 添加文件系统挂载命令。
- 使用 **mkswap** 命令初始化交换空间。

海量视频题库 myitpub.com QQ:5565462  
www.52myit.com