CIS 444 - Mounting Report

Lucas Hasting

October 7, 2024

Contents

1 Partitioning Purpose

1.1 Why Partition?

Partitions are done to avoid filling up the disk, along with segmentation.

So what happens when the disk is filled? There would be no space for caching files, and if files are constantly being created (such as logs), anomalies can occur, and it can bring the system down to a halt and it may not be possible to recover the system. This can be damaging for servers in corporations but does not affect as nearly as large on personal computers.

1.2 Important Partitions with Linux

The two biggest partitions that need to be made in a Linux environment are the /var directory and the /home directory, /var stands for variable and is where logs and other data are stored, the folder is constantly changing in size. The home folder stores the user files on the machine. Since users are constantly downloading items, the content in the directory is changing as more users download files.

When creating partitions, it is important to know how much storage should be allocated to the partition and consider how much it could/would grow.

1.3 Linux System Partitions

The partitions below are made by the system:

```
1. /boot/efi: this is the partition used for booting the operating system
   and other utilities
2. /: this is the partition used for the root user
```

1.4 Other important directories in Linux

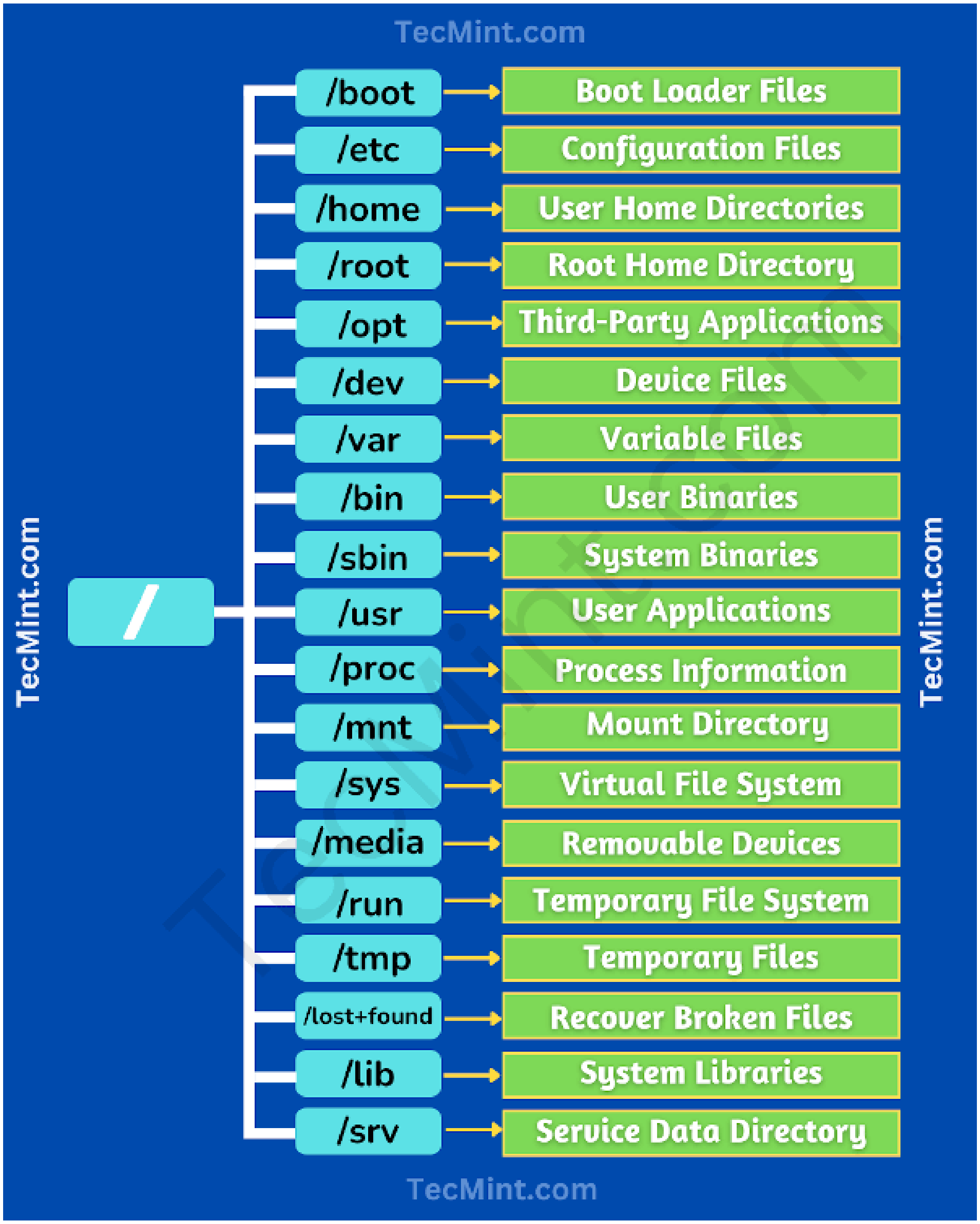


Figure 1:

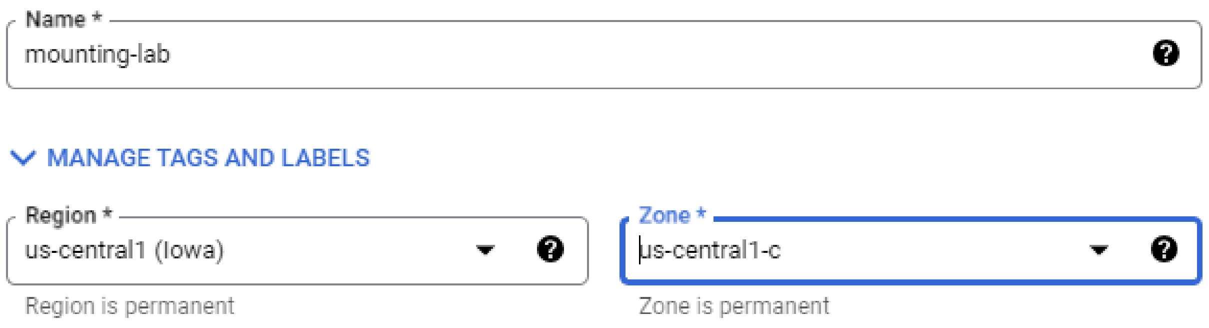
2 Partitioning Process

2.1 Setting up the machine - Google Cloud VM

For this section, the purpose is to create the virtual machine in google cloud to be used for the lab.

Process:

```
1. create a new VM instance
2. select a name
3. change the zone to us-central-c
4. choose the N1 machine configuration
5. select the shared core, f1-micro machine type
6. go to advanced options -> disks and backups
7. add a new disk with 15GB storage and select the delete disk, deletion
   rule
8. (optional) in advanced options, select security -> managae access, and
   then add ssh keys for authentication
```


Name * mounting-lab

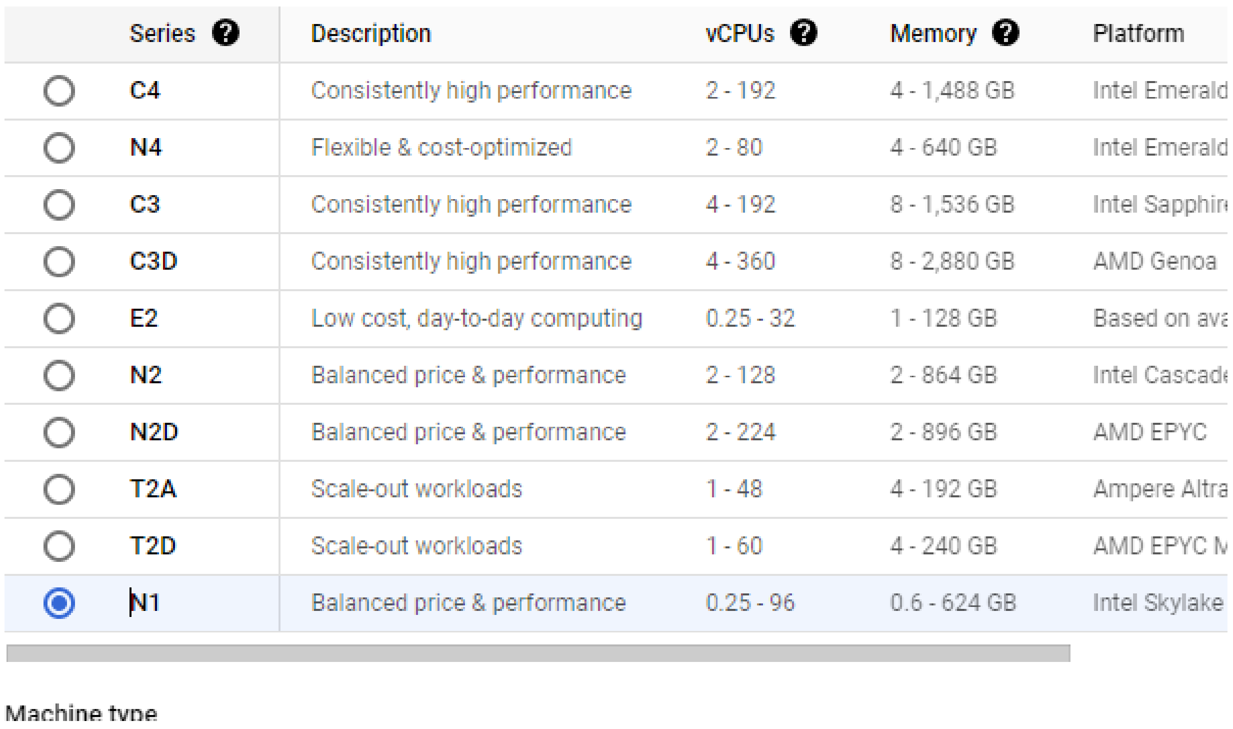
Region * us-central1 (Iowa)

Region is permanent

Zone * us-central1-c

Zone is permanent

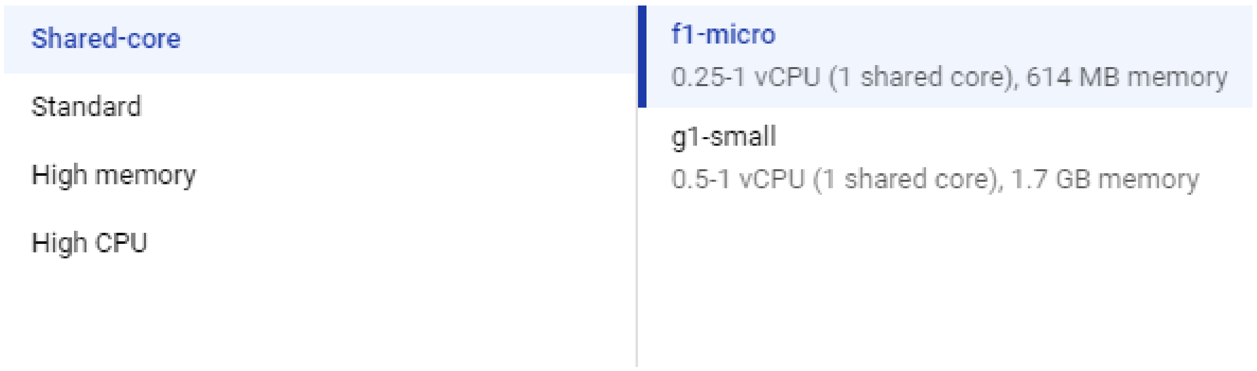
Figure 2:



	Series	Description	vCPUs	Memory	Platform
○	C4	Consistently high performance	2 - 192	4 - 1,488 GB	Intel Emerald
○	N4	Flexible & cost-optimized	2 - 80	4 - 640 GB	Intel Emerald
○	C3	Consistently high performance	4 - 192	8 - 1,536 GB	Intel Sapphir
○	C3D	Consistently high performance	4 - 360	8 - 2,880 GB	AMD Genoa
○	E2	Low cost, day-to-day computing	0.25 - 32	1 - 128 GB	Based on ava
○	N2	Balanced price & performance	2 - 128	2 - 864 GB	Intel Cascad
○	N2D	Balanced price & performance	2 - 224	2 - 896 GB	AMD EPYC
○	T2A	Scale-out workloads	1 - 48	4 - 192 GB	Ampere Altra
○	T2D	Scale-out workloads	1 - 60	4 - 240 GB	AMD EPYC M
●	N1	Balanced price & performance	0.25 - 96	0.6 - 624 GB	Intel Skylake

Machine type

Figure 3:

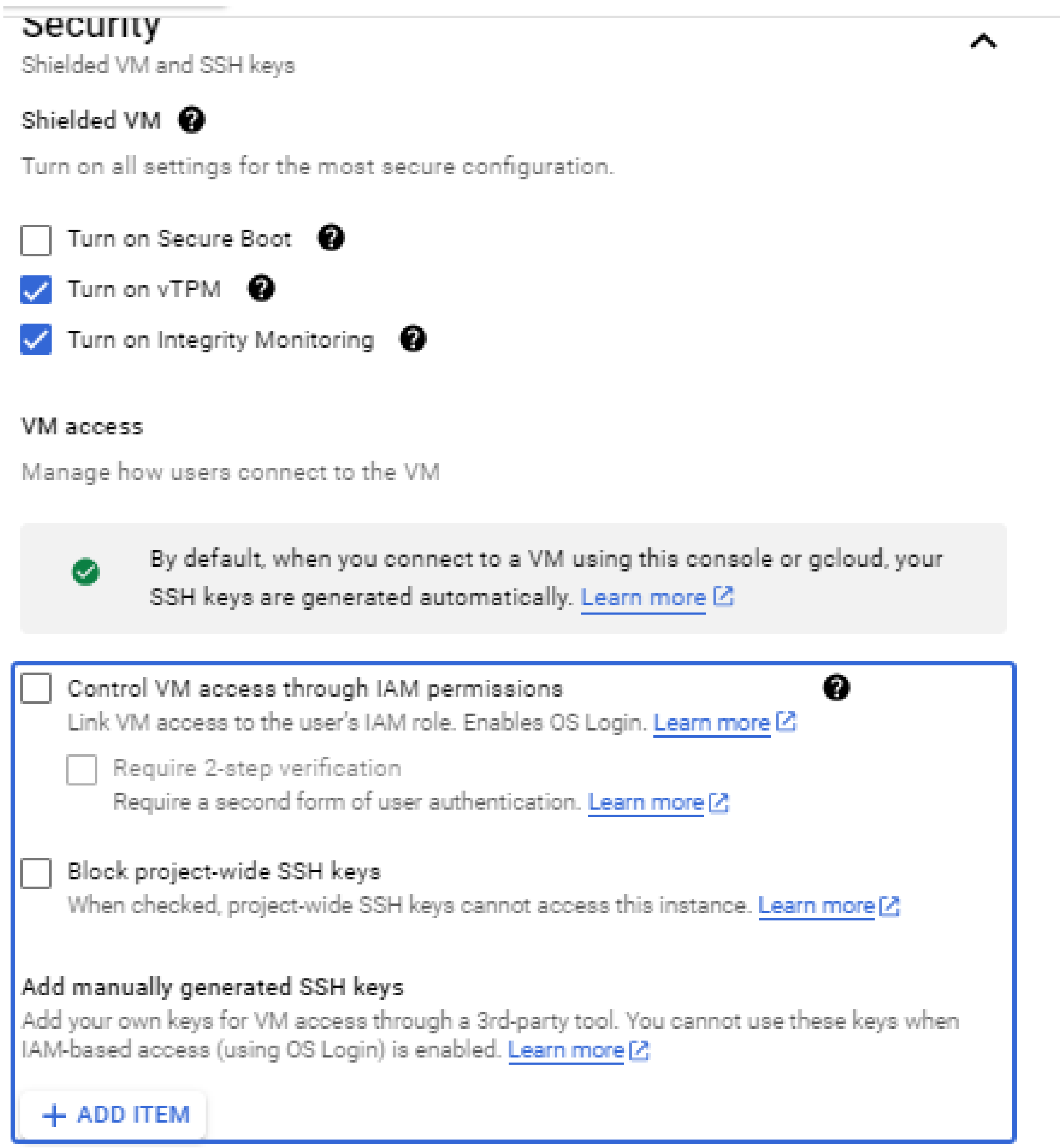


Shared-core
Standard
High memory
High CPU

f1-micro
0.25-1 vCPU (1 shared core), 614 MB memory

g1-small
0.5-1 vCPU (1 shared core), 1.7 GB memory

Figure 4:



Security

Shielded VM and SSH keys

Shielded VM

Turn on all settings for the most secure configuration.

- [ ] Turn on Secure Boot
- [✓] Turn on vTPM
- [✓] Turn on Integrity Monitoring

VM access

Manage how users connect to the VM

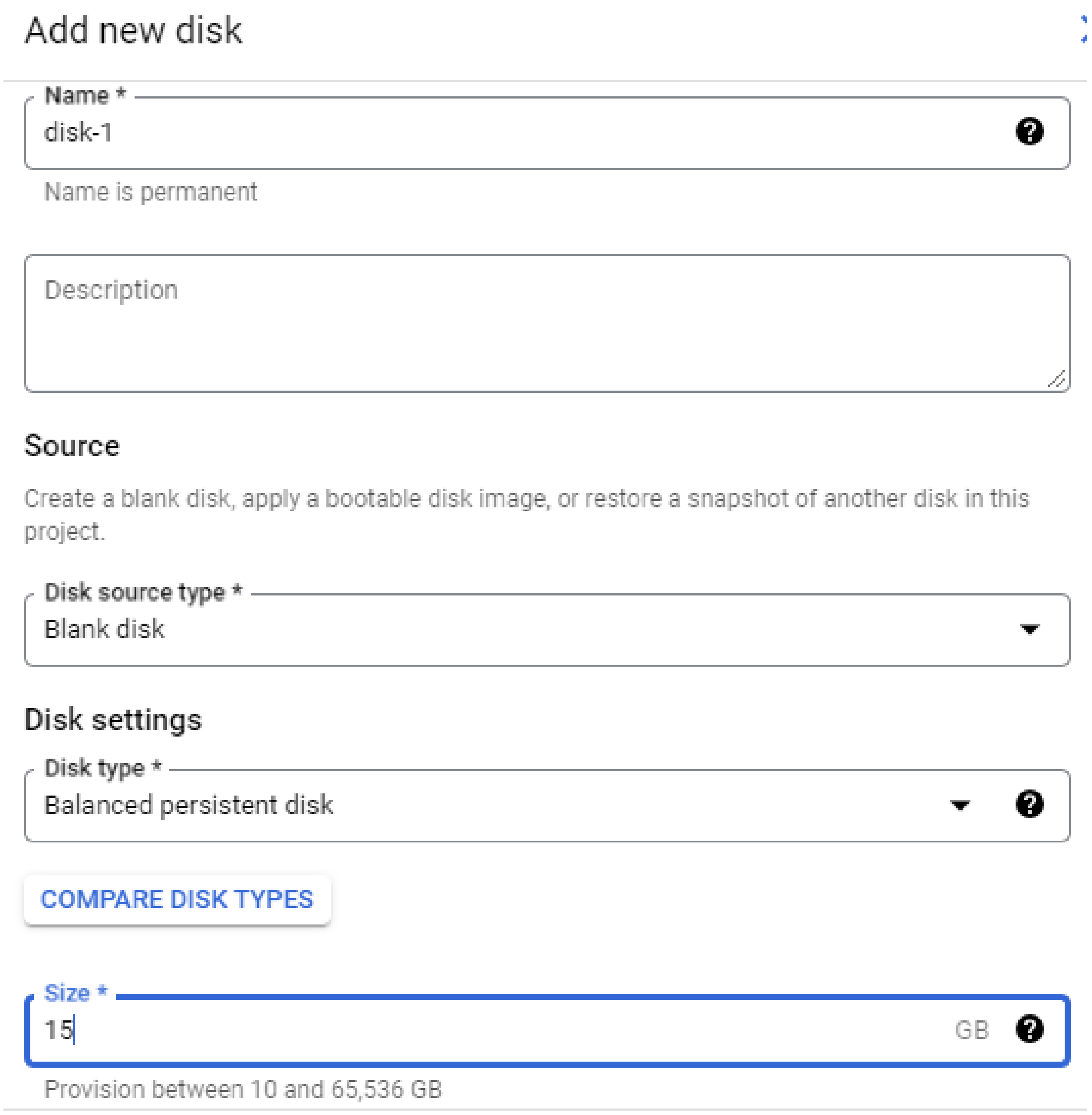
By default, when you connect to a VM using this console or gcloud, your SSH keys are generated automatically. Learn more

- [ ] Control VM access through IAM permissions
  Link VM access to the user's IAM role. Enables OS Login. Learn more
  - [ ] Require 2-step verification
    Require a second form of user authentication. Learn more
- [ ] Block project-wide SSH keys
  When checked, project-wide SSH keys cannot access this instance. Learn more

Add manually generated SSH keys

Add your own keys for VM access through a 3rd-party tool. You cannot use these keys when IAM-based access (using OS Login) is enabled. Learn more

Figure 5:


Add new disk

Name *
disk-1
Name is permanent

Description

Source

Create a blank disk, apply a bootable disk image, or restore a snapshot of another disk in this project.

Disk source type *
Blank disk

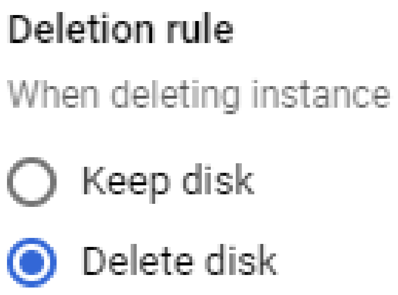
Disk settings

Disk type *
Balanced persistent disk

COMPARE DISK TYPES

Size *
15 GB
Provision between 10 and 65,536 GB

Figure 6:



Deletion rule

When deleting instance

Keep disk [✓ Delete disk]

Figure 7:

2.2 Know what to partition

Recall from section 1, that two important partitions are home and var, the size of these partitions can vary. A way to determine the size is to use the internet to make a decision. After looking it up on Google, the result is that var should be at least 3 GB and home should be as much as possible. In this lab, var will be close to 6GB, while home will be the rest of the avaliable space.

2.3 Creating the partitions

For this section, the purpose is to create the blkvar and blkhome partitions which will be used to partition the var and home directories.

Process:

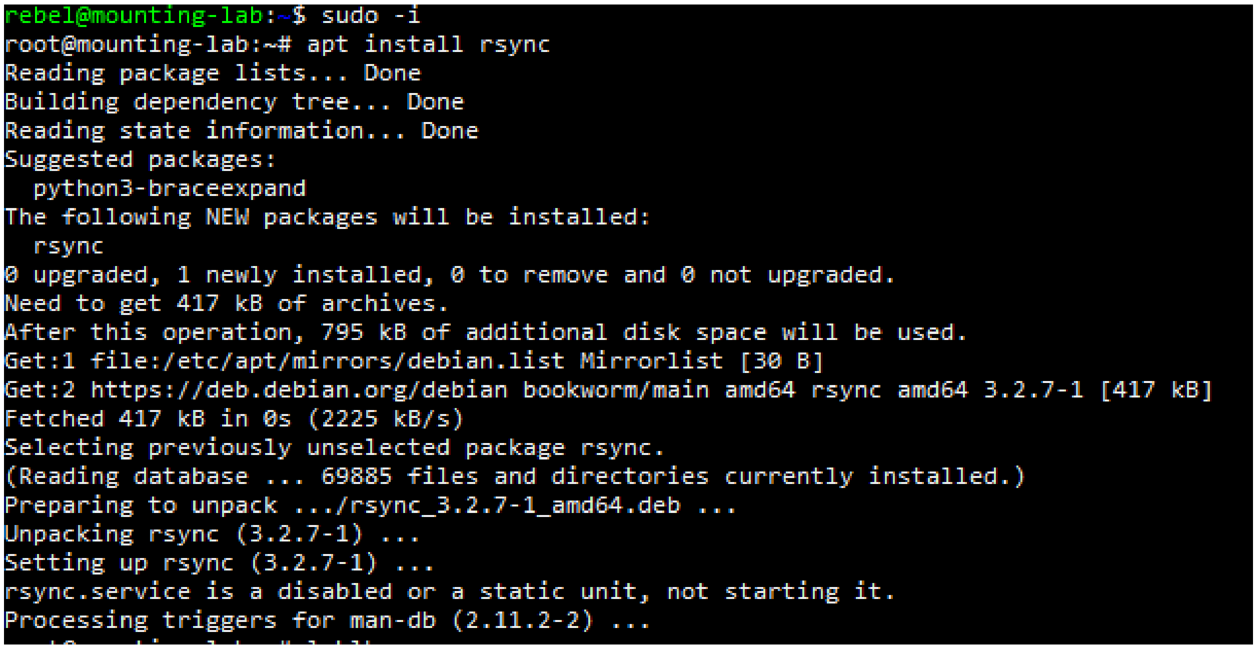
```
1. Become root user
2. install rsync which will be used to copy files and retain permissions
3. list out the devices
4. use parted on the disk that is empty
5. make the partition label on the disk, linux uses gpt
6. make the partition for var and home, sizes can vary
7. use blkid to see where each partition is stored
8. format the partitions with a file system, ext4 will be used
9. list out the devices to ensure the partitions exist
10. use blkid to show partition UUID
    varuuid = the UUID for the blkvar partition
    homeuuid = the UUID for the homeuuid partition
```

Commands used:

```
sudo -i
apt install rsync
lsblk
parted <device>
(parted) mklabel gpt
(parted) mkpart blkvar 1MiB 6GiB
(parted) mkpart blkhome 6GiB 100%
(parted) quit
blkid
mkfs.ext4 -L blkvar /dev/sda1
mkfs.ext4 -L blkhome /dev/sda2
lsblk
blkid
```

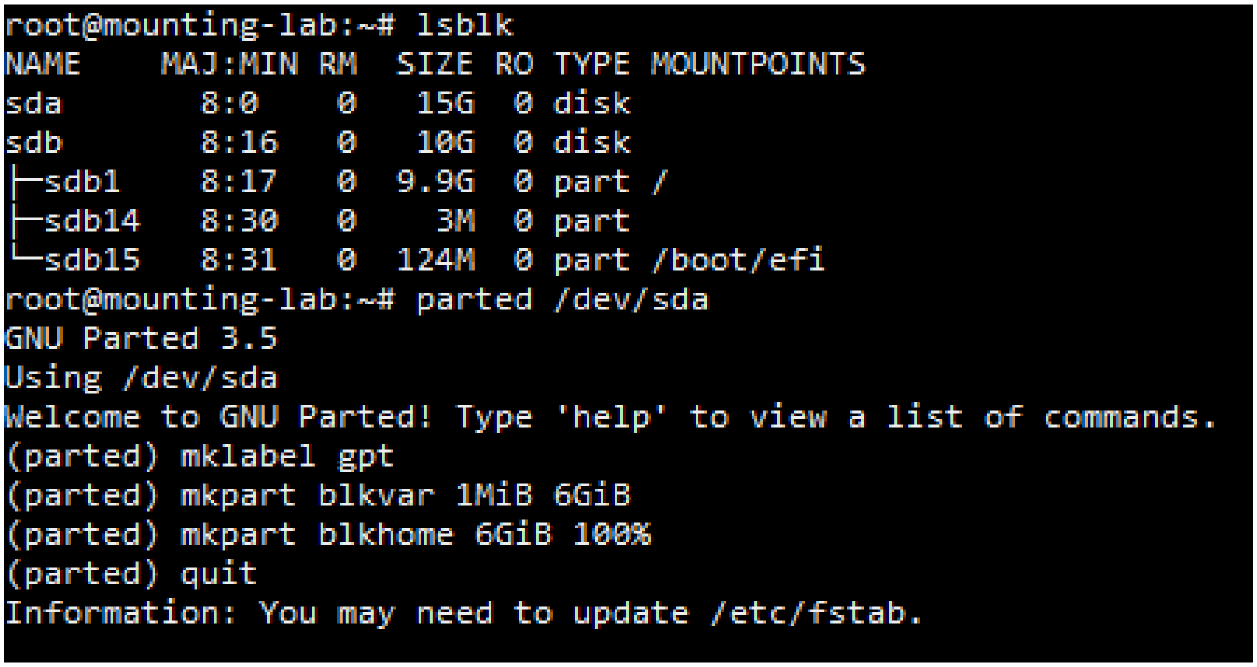
Some notes about the commands: blkvar is located at 1MB through 6GB and blkhome is located at 6GB through 15GB which is a total of 9GB.

Pictures:



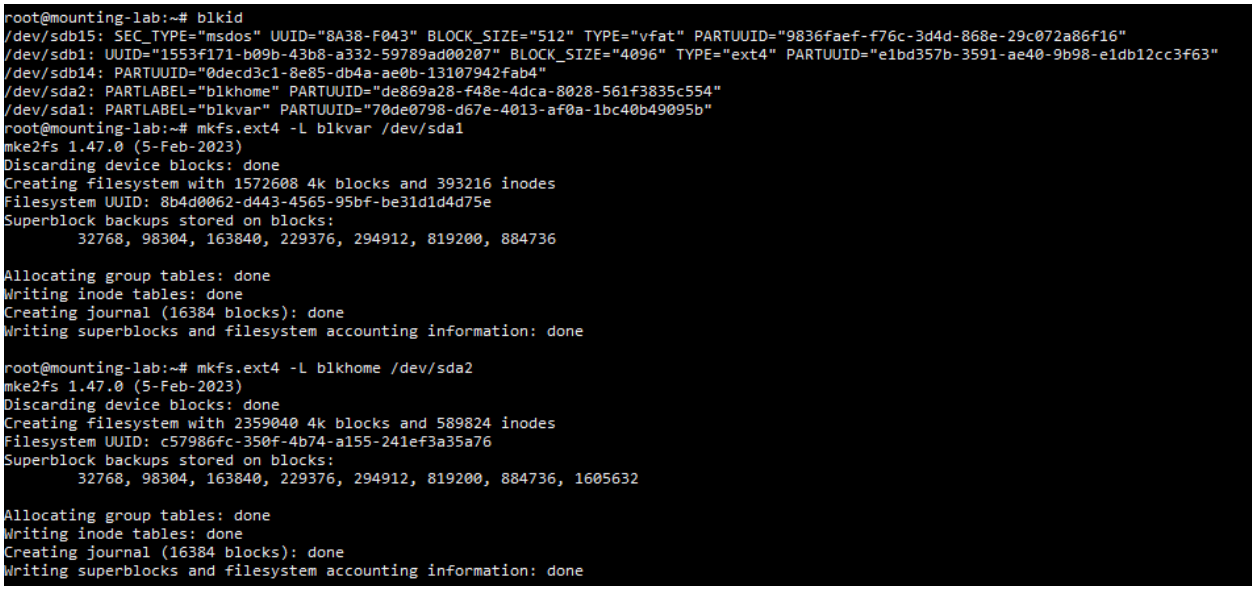
```
rebel@mounting-lab:~$ sudo -i
root@mounting-lab:~# apt install rsync
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  python3-braceexpand
The following NEW packages will be installed:
  rsync
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 417 kB of archives.
After this operation, 795 kB of additional disk space will be used.
Get:1 file:/etc/apt/mirrors/debian.list Mirrorlist [30 B]
Get:2 https://deb.debian.org/debian bookworm/main amd64 rsync amd64 3.2.7-1 [417 kB]
Fetched 417 kB in 0s (2225 kB/s)
Selecting previously unselected package rsync.
(Reading database ... 69885 files and directories currently installed.)
Preparing to unpack .../rsync_3.2.7-1_amd64.deb ...
Unpacking rsync (3.2.7-1) ...
Setting up rsync (3.2.7-1) ...
rsync.service is a disabled or a static unit, not starting it.
Processing triggers for man-db (2.11.2-2) ...
```

Figure 8:



```
root@mounting-lab:~# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   15G  0 disk
sdb       8:16   0   10G  0 disk
├─sdb1    8:17   0  9.9G  0 part /
├─sdb14   8:30   0    3M  0 part
└─sdb15   8:31   0  124M  0 part /boot/efi
root@mounting-lab:~# parted /dev/sda
GNU Parted 3.5
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mklabel gpt
(parted) mkpart blkvar 1MiB 6GiB
(parted) mkpart blkhome 6GiB 100%
(parted) quit
Information: You may need to update /etc/fstab.
```

Figure 9:



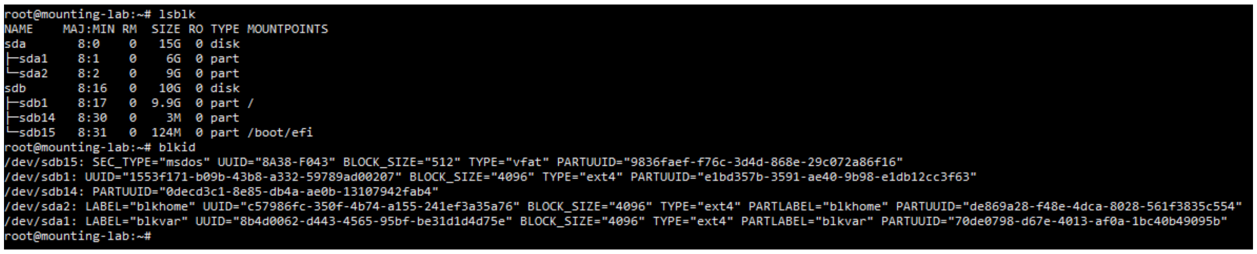
```
root@mounting-lab:~# blkid
/dev/sdb15: SEC_TYPE="msdos" UUID="8A38-F043" BLOCK_SIZE="512" TYPE="vfat" PARTUUID="9836faef-f76c-3d4d-868e-29c072a86f16"
/dev/sdb1: UUID="1553f171-b09b-43b8-a332-59789ad00207" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="e1bd357b-3591-ae40-9b98-e1db12cc3f63"
/dev/sdb14: PARTUUID="0decd3c1-8e85-db4a-ae0b-13107942fab4"
/dev/sda2: PARTLABEL="blkhome" PARTUUID="de869a28-f48e-4dca-8028-561f3835c554"
/dev/sda1: PARTLABEL="blkvar" PARTUUID="70de0798-d67e-4013-af0a-1bc40b49095b"
root@mounting-lab:~# mkfs.ext4 -L blkvar /dev/sda1
mke2fs 1.47.0 (5-Feb-2023)
Discarding device blocks: done
Creating filesystem with 1572608 4k blocks and 393216 inodes
Filesystem UUID: 8b4d0062-d443-4565-95bf-be31d1d4d75e
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

root@mounting-lab:~# mkfs.ext4 -L blkhome /dev/sda2
mke2fs 1.47.0 (5-Feb-2023)
Discarding device blocks: done
Creating filesystem with 2359040 4k blocks and 589824 inodes
Filesystem UUID: c57986fc-350f-4b74-a155-241ef3a35a76
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Figure 10:



```
root@mounting-lab:~# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   15G  0 disk
├─sda1    8:1    0    6G  0 part
└─sda2    8:2    0    9G  0 part
sdb       8:16   0   10G  0 disk
├─sdb1    8:17   0  9.9G  0 part /
├─sdb14   8:30   0    3M  0 part
└─sdb15   8:31   0  124M  0 part /boot/efi
root@mounting-lab:~# blkid
/dev/sdb15: SEC_TYPE="msdos" UUID="8A38-F043" BLOCK_SIZE="512" TYPE="vfat" PARTUUID="9836faef-f76c-3d4d-868e-29c072a86f16"
/dev/sdb1: UUID="1553f171-b09b-43b8-a332-59789ad00207" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="e1bd357b-3591-ae40-9b98-e1db12cc3f63"
/dev/sdb14: PARTUUID="0decd3c1-8e85-db4a-ae0b-13107942fab4"
/dev/sda2: LABEL="blkhome" UUID="c57986fc-350f-4b74-a155-241ef3a35a76" BLOCK_SIZE="4096" TYPE="ext4" PARTLABEL="blkhome" PARTUUID="de869a28-f48e-4dca-8028-561f3835c554"
/dev/sda1: LABEL="blkvar" UUID="8b4d0062-d443-4565-95bf-be31d1d4d75e" BLOCK_SIZE="4096" TYPE="ext4" PARTLABEL="blkvar" PARTUUID="70de0798-d67e-4013-af0a-1bc40b49095b"
root@mounting-lab:~#
```

Figure 11:

2.4 Mounting/Setting the partitions

For this section, the purpose is to mount the partitions onto the directories var and home

Process:

```
1. create temporary directories tempvar and temp home in the /mnt/
   directory
2. mount the partitions on the temporary directories
3. copy the contents of the directories to the temporary directories
4. delete the content in the direcotries
5. unmount from the temporary directories
6. mount the partitions to the directories
7. ensure the commands were successfull
```

Commands used:

```
mkdir -p /mnt/tempvar
mkdir -p /mnt/temphome
mount UUID={varuuid} /mnt/tempvar
mount UUID={homeuuid} /mnt/temphome
rsync -a /var/ /mnt/tempvar
rsync -a /home/ /mnt/temphome
rm -r /var/*
rm -r /home/*
umount /mnt/tempvar
umount /mnt/temphome
mount UUID={varuuid} /var
mount UUID={homeuuid} /home
ls /var
ls /home
lsblk
```

Note about some commands: mount makes a filesystem/device accessible to the system allowing the system to make changes to it.

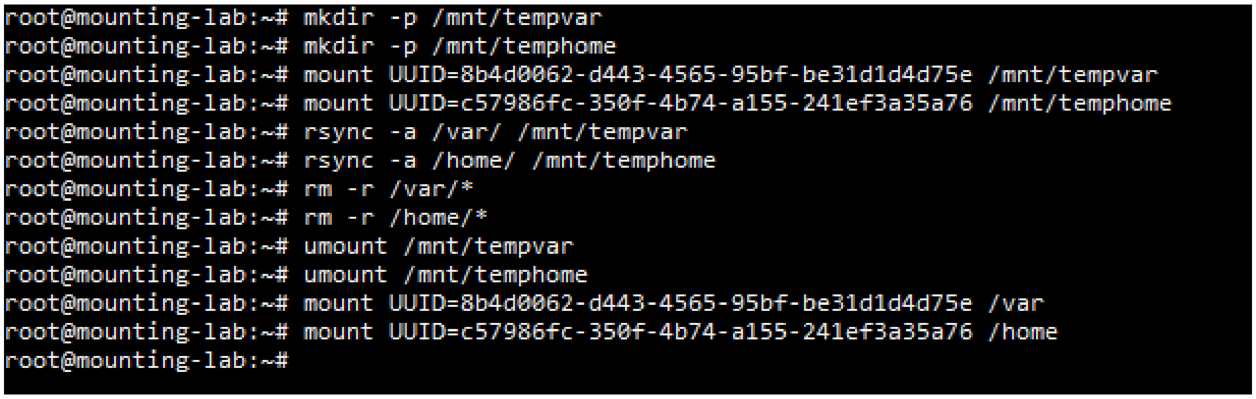
unmounting a filesystem/device removes the ability for the system to make changes to the filesystem/device, but the content is not deleted.

when a mount for a partition is created, that allows us to make changes to the parition, in this lab we copy the content from a directory (/var and /home) to its temp mount point (/mnt/tempvar) and (/mnt/temphome) which represents the partitions. We then unmount the partition, delete everything in the directories the partitions are for, and mount the partition to the directories they are intended for.

Below is pseudocode on what we are accomplishing:

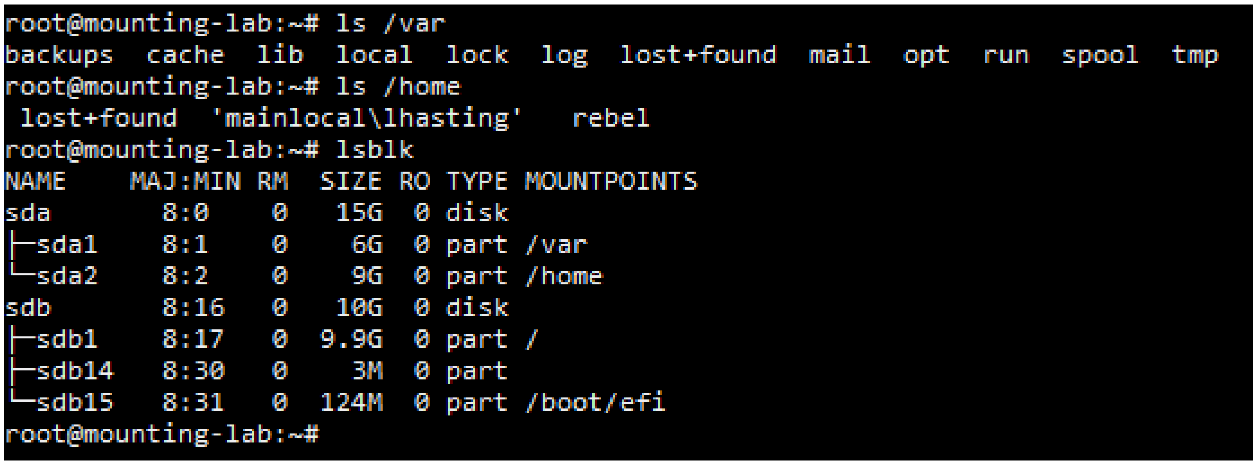
```
mount partition as tempdir
dir -> tempdir (partition)
del dir
unmount partition as tempdir
mount partition as dir
```

Pictures:



```
root@mounting-lab:~# mkdir -p /mnt/tempvar
root@mounting-lab:~# mkdir -p /mnt/temphome
root@mounting-lab:~# mount UUID=8b4d0062-d443-4565-95bf-be31d1d4d75e /mnt/tempvar
root@mounting-lab:~# mount UUID=c57986fc-350f-4b74-a155-241ef3a35a76 /mnt/temphome
root@mounting-lab:~# rsync -a /var/ /mnt/tempvar
root@mounting-lab:~# rsync -a /home/ /mnt/temphome
root@mounting-lab:~# rm -r /var/*
root@mounting-lab:~# rm -r /home/*
root@mounting-lab:~# umount /mnt/tempvar
root@mounting-lab:~# umount /mnt/temphome
root@mounting-lab:~# mount UUID=8b4d0062-d443-4565-95bf-be31d1d4d75e /var
root@mounting-lab:~# mount UUID=c57986fc-350f-4b74-a155-241ef3a35a76 /home
root@mounting-lab:~#
```

Figure 12:



```
root@mounting-lab:~# ls /var
backups  cache  lib  local  lock  log  lost+found  mail  opt  run  spool  tmp
root@mounting-lab:~# ls /home
 lost+found  'mainlocal\lhasting'   rebel
root@mounting-lab:~# lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   15G  0 disk
├─sda1    8:1    0    6G  0 part /var
└─sda2    8:2    0    9G  0 part /home
sdb       8:16   0   10G  0 disk
├─sdb1    8:17   0  9.9G  0 part /
├─sdb14   8:30   0    3M  0 part
└─sdb15   8:31   0  124M  0 part /boot/efi
root@mounting-lab:~#
```

Figure 13:

2.5 Updating the fstab file

For this section, the purpose is to update the fstab file so the partitions remain once the machine is restarted.

Process:

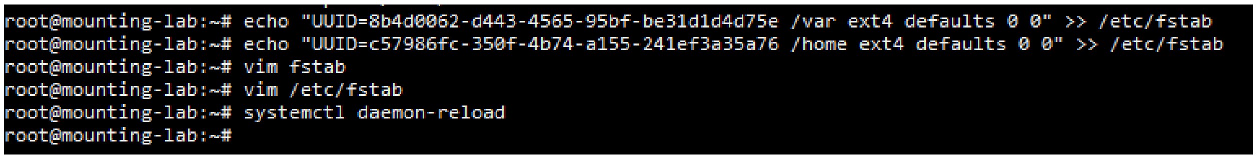
```
1. append the UUID, directory, file system to the fstab file
2. view the fstab file to make sure everything is correct
3. reload the daemons
```

Commands used:

```
echo "UUID={varuuid} /var ext4 defaults 0 0" >> /etc/fstab
echo "UUID={homeuuid} /home ext4 defaults 0 0" >> /etc/fstab
vim /etc/fstab
systemctl daemon-reload
```

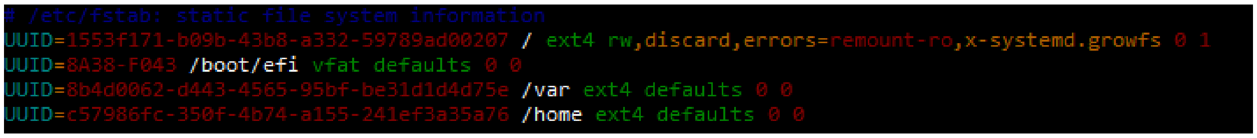
Note about fstab: /etc/fstab represents the file system table and describes how the filesystems are mounted on the machine.

Pictures:



```
root@mounting-lab:~# echo "UUID=8b4d0062-d443-4565-95bf-be31d1d4d75e /var ext4 defaults 0 0" >> /etc/fstab
root@mounting-lab:~# echo "UUID=c57986fc-350f-4b74-a155-241ef3a35a76 /home ext4 defaults 0 0" >> /etc/fstab
root@mounting-lab:~# vim fstab
root@mounting-lab:~# vim /etc/fstab
root@mounting-lab:~# systemctl daemon-reload
root@mounting-lab:~#
```

Figure 14:



```
# /etc/fstab: static file system information
UUID=1553f171-b09b-43b8-a332-59789ad00207 / ext4 rw,discard,errors=remount-ro,x-systemd.growfs 0 1
UUID=8A38-F043 /boot/efi vfat defaults 0 0
UUID=8b4d0062-d443-4565-95bf-be31d1d4d75e /var ext4 defaults 0 0
UUID=c57986fc-350f-4b74-a155-241ef3a35a76 /home ext4 defaults 0 0
```

Figure 15:

2.6 Wrapping up

For this section, the purpose is to ensure that everything done in the lab was done correctly.

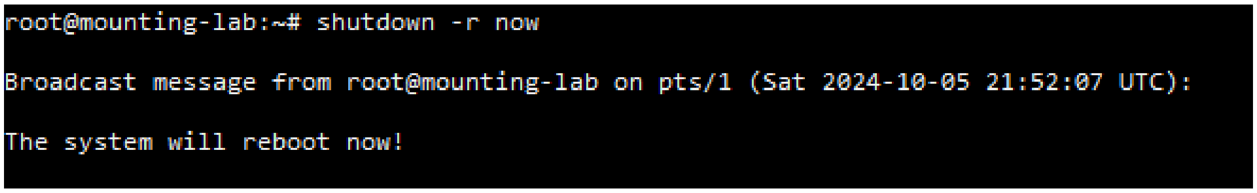
Process:

```
1. restart the machine
2. ensure nothing went wrong and the partitions still remain
```

Commands used:

```
shutdown -r now
lsblk
```

Pictures:

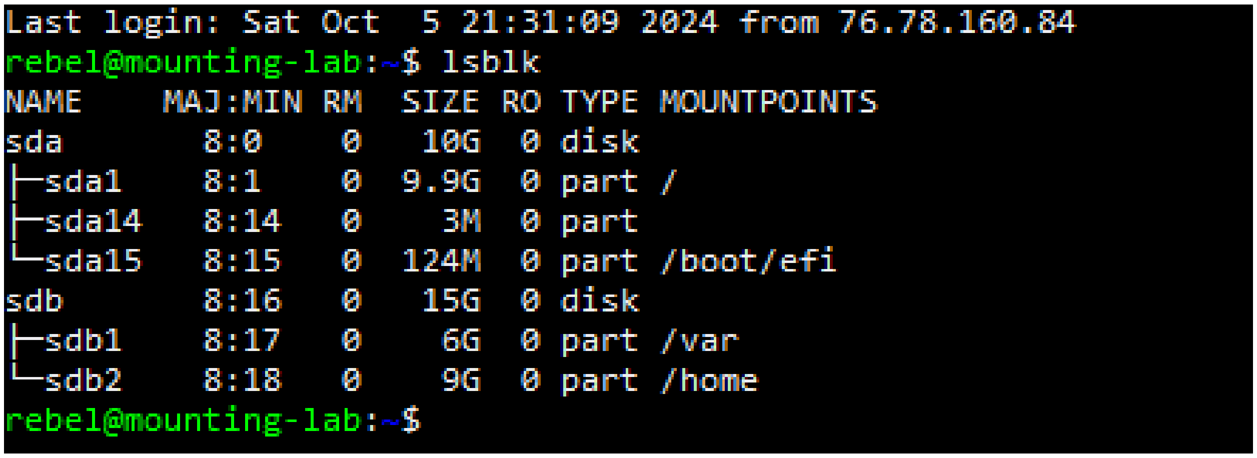


```
root@mounting-lab:~# shutdown -r now

Broadcast message from root@mounting-lab on pts/1 (Sat 2024-10-05 21:52:07 UTC):

The system will reboot now!
```

Figure 16:



```
Last login: Sat Oct  5 21:31:09 2024 from 76.78.160.84
rebel@mounting-lab:~$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda       8:0    0   10G  0 disk
├─sda1    8:1    0  9.9G  0 part /
├─sda14   8:14   0    3M  0 part
└─sda15   8:15   0  124M  0 part /boot/efi
sdb       8:16   0   15G  0 disk
├─sdb1    8:17   0    6G  0 part /var
└─sdb2    8:18   0    9G  0 part /home
rebel@mounting-lab:~$
```

Figure 17:

2.7 The Script

```
#!/bin/bash

apt install rsync #install rsync

#create gpt partitions for home and var along with their sizes on /dev/sda
parted -s /dev/sda -- mklabel gpt \
            mkpart blkvar 1MiB 6GiB \
            mkpart blkhome 6GiB 100%

#format the partitions files system - using ext4
mkfs.ext4 -L blkvar /dev/sda1
mkfs.ext4 -L blkhome /dev/sda2

#get the uuid for the two partitions
varuuid=$(blkid -s UUID -o value /dev/sda1)
homeuuid=$(blkid -s UUID -o value /dev/sda2)

#make the temp directories for mounting
mkdir -p /mnt/${varuuid}
mkdir -p /mnt/${homeuuid}

#mount the files in the folders that will be in the partitions
mount UUID=${varuuid} /mnt/${varuuid}
mount UUID=${homeuuid} /mnt/${homeuuid}

#copy the files into the partition
rsync -a /var/ /mnt/${varuuid}/
rsync -a /home/ /mnt/${homeuuid}/

#update the fstab file with the var partition
echo "UUID=${varuuid} /var ext4 defaults 0 0" >> /etc/fstab

#remove all files in the /var/ directory
rm -r /var/*

#unmount the partition
umount /mnt/${varuuid}

#mount the partitions uuid to the /var directory
mount UUID=${varuuid} /var

#repeat the last 4 actions on the /home directory
echo "UUID=${homeuuid} /home ext4 defaults 0 0" >> /etc/fstab
rm -r /home/*
umount /mnt/${homeuuid}
mount UUID=${homeuuid} /home

#reload the deamons
systemctl daemon-reload

#disable google startup scripts
systemctl disable google-startup-scripts
```

2.8 Using the script

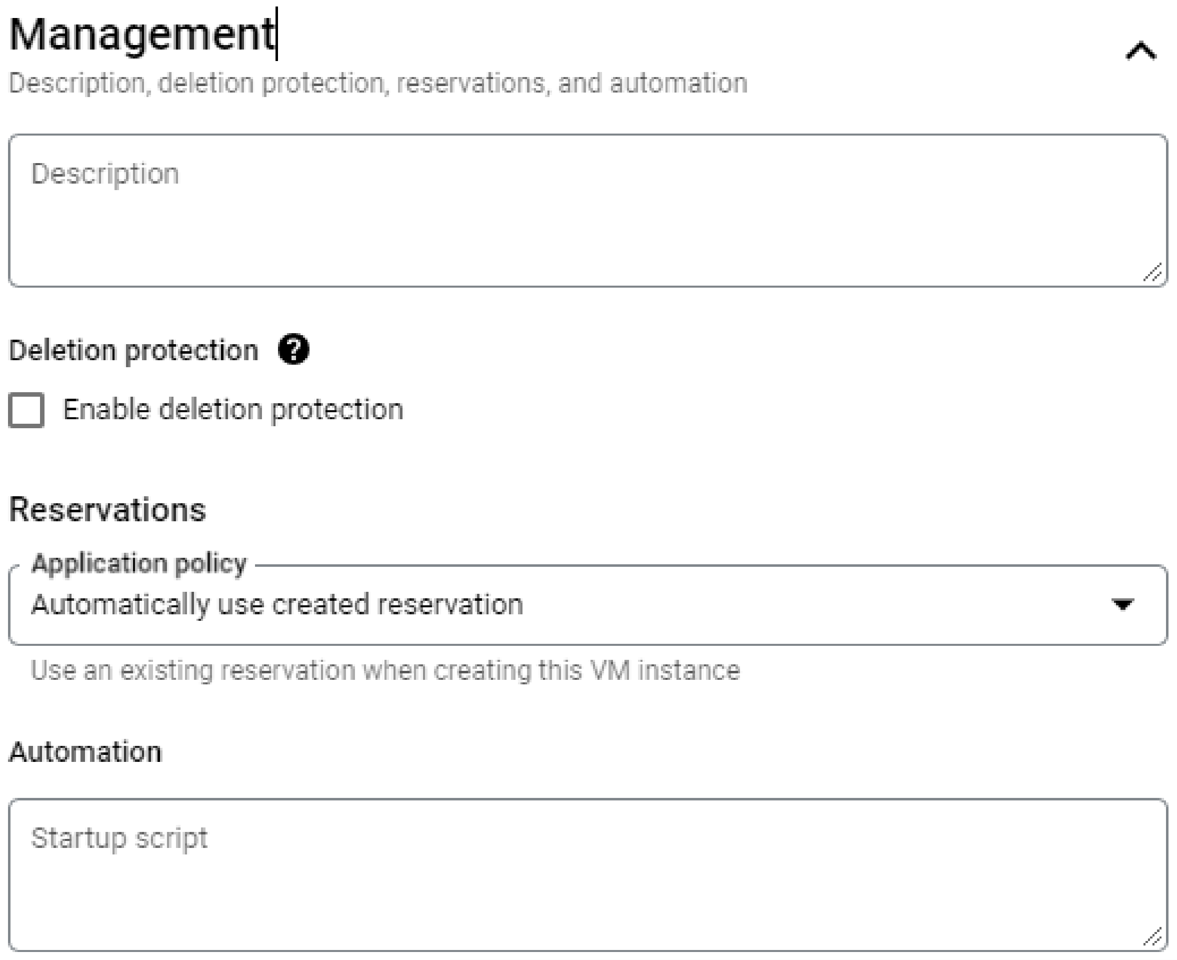
The script shown in the previous section is an automated version of what was done throughout the lab, and it can be used to automatically make the partitions, the script can be copied and executed on the machine, or, in the case that will be shown in this section, the script will be used as a startup script which executes when the machine starts.

Process:

```
in the creation of the VM, go to advanced options -> management, and paste
    the script into the text box below Automation

if the machine is already created, make sure it is off, edit the machine,
   and do the same step as mentioned above
```

Picture:



Management

Description, deletion protection, reservations, and automation

Description

Deletion protection

Enable deletion protection

Reservations

Application policy: Automatically use created reservation

Use an existing reservation when creating this VM instance

Automation

Startup script

Figure 18:

3 Linux Commands

```
1. ls <optional:directory path> - The most frequently used command in
   Linux to list directories
2. pwd - Print working directory command in Linux
3. cd <directory path> - Linux command to navigate through directories
4. mkdir <name> - Command used to create directories in Linux
5. mv <source> <destination> - Move or rename files in Linux
6. cp <source> <destination> - Similar usage as mv but for copying files
   in Linux
7. rm <file> - Delete files or directories
8. touch <file> - Create blank/empty files
9. ln -s <source> <destination> - Create symbolic links (shortcuts) to
   other files
10. clear - Clear the terminal display
11. cat <file> - Display file contents on the terminal
12. echo <text> - Print any text that follows the command
13. less <file> - Linux command to display paged outputs in the terminal
14. man <command> - Access manual pages for all Linux commands
15. uname - Linux command to get basic information about the OS
16. whoami - Get the active username
17. tar - Command to extract and compress files in linux
    -cf <archive> <files>: puts files into single tar file
    -xf <archive>: extracts tar file
18. grep <text> - Search for a string within an output
19. head <file> - Return the specified number of lines from the top
20. tail <file> - Return the specified number of lines from the bottom
21. diff <file1> <file2> - Find the difference between two files
22. cmp <file1> <file2> - Allows you to check if two files are identical
23. comm <file1> <file2> - Combines the functionality of diff and cmp
24. sort <file> - Linux command to sort the content of a file while
   outputting
    -n for sorting numbers
25. export <variable>=<value> - Export environment variables in Linux
    -  environment variable = variable that can be used in shell (the
   environment)
26. zip <archive-name> <files> - Zip files in Linux
27. unzip <archive-name> - Unzip files in Linux
28. ssh <username>@location- Secure Shell command in Linux
    - many more ways to authenticate using ssh - had a whole lab on it
29. service <service> <operation> - Linux command to start and stop
   services
    - service can be a protocol
    - operation can be start/stop/status
30. ps - Display active processes
31. kill <process ID (PID)> and killall <process name> - Kill active
   processes by process ID or name
    - pkill <process name> - same description as above
32. df - Display disk filesystem information
33. mount <filesystem/device> <mount point> - Mount file systems in Linux
34. chmod <permission> <file> - Command to change file permissions
35. chown <own> <file> - Command for granting ownership of files or
   folders
36. ifconfig - Display network interfaces and IP addresses
```

```
37. traceroute <destination> - Trace all the network hops to reach the
   destination
38. wget <link> - Direct download files from the internet
39. ufw <rule> - Firewall command
40. iptables <rule> - Base firewall for all other firewall utilities to
   interface with
41. apt, pacman, yum, rpm - Package managers depending on the distribution
    - <package manager> <-S with packman> install <package>
42. sudo <command> - Command to escalate privileges in Linux
    -i to become root user
43. cal - View a command-line calendar
44. alias <alias name>=<command> - Create custom shortcuts for your
   regularly used commands
45. dd - Majorly used for creating bootable USB sticks, can copy from one
   file type/device to another
    - if=<input file>
    - of=<output file>
46. whereis <program> - Locate the binary, source, and manual pages for a
   command
47. whatis <program> - Find what a command is used for
48. top - View active processes live with their system usage
49. useradd <user> <options> - Add a new user
50. usermod <user> <options> - change existing user data
51. passwd - Create or update passwords for existing users
```