Linux Storage Setup and Management Lab
By Michael Ambeguia

Purpose:

Storage is vital for any operating system to function. Storage devices are where the operating system and user files are stored. Without storage a computing device will not be able to function since ram memory needs to receive data from somewhere, data does not come out of thin air. Not having computer storage is like a car not having an engine, the computer simply won't run. Thus, knowing how to manage storage on Linux OS computers is a vital skill to have. In this lab I will be applying Linux storage management skills on my Ubuntu Server VM. I will be adding virtual disks to my Ubuntu VM, partitioning the disks, putting file systems on the disks, mounting the newly created partitions onto my Linux VM, and finally I will explore storage management and monitoring on Linux systems.

Lab Sections:

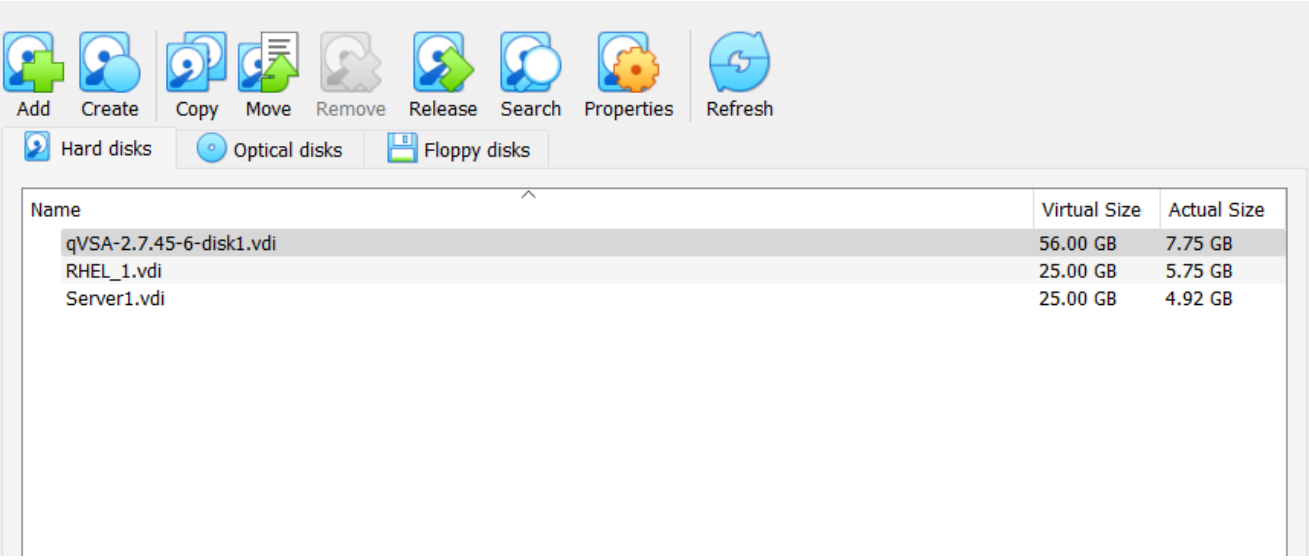
1. Adding storage devices to a Linux server
2. Partition storage devices
3. Put file system on partitions/ device
4. Mount filesystem and make it persistent
5. Removing storage devices from a Linux system
6. Storage Management and Monitoring

Section #1 Adding Storage Devices onto a Linux Server

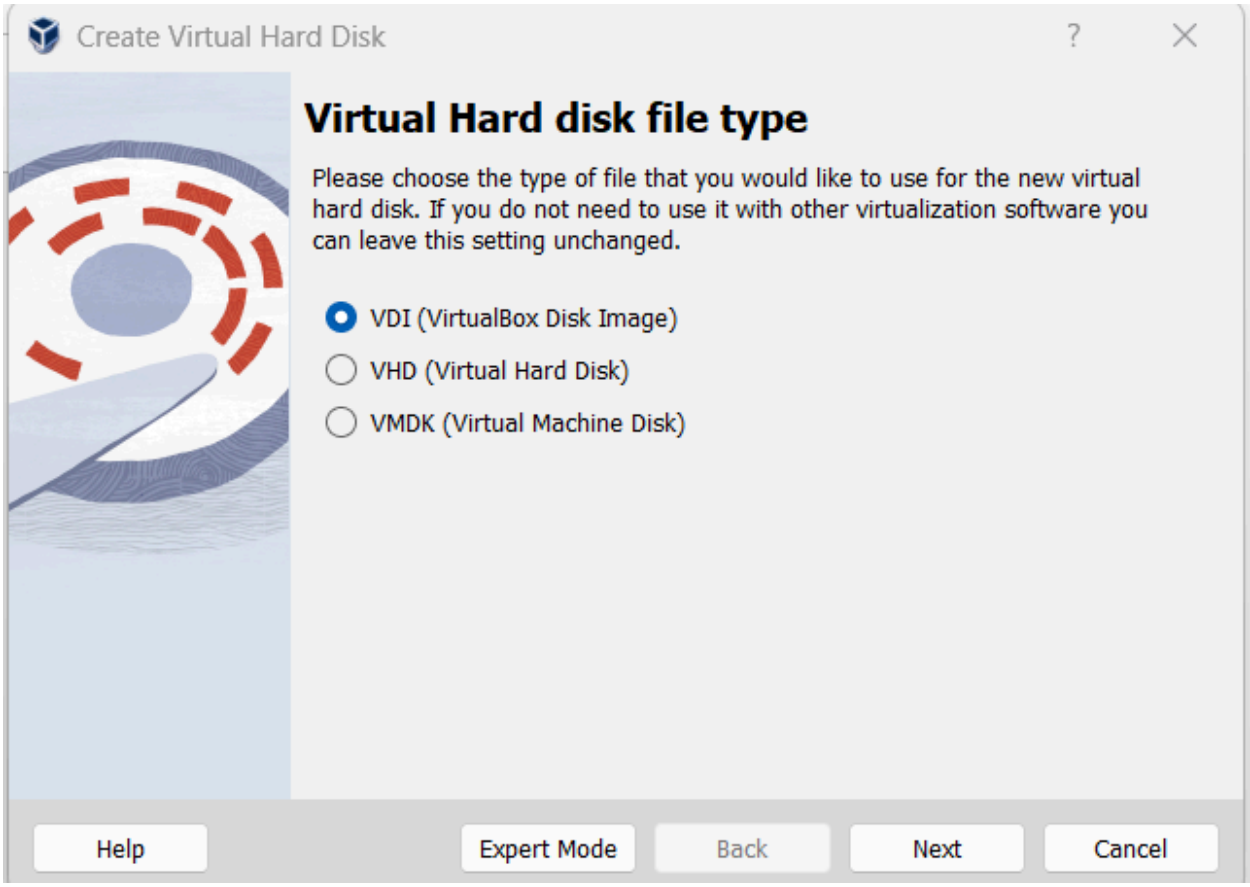
Intro: Since I will not be adding any physical storage devices like SSDs or HDDs to my VM, the storage devices in question will be virtual disks. Virtual disks emulate physical storage devices like SSDs and HDDs and are software defined, meaning that they do not exist in the physical world.

Step 1. I will create a virtual disk for the purpose of this lab on my VirtualBox type 2 hypervisor. This can be accomplished by clicking ion file on the top left hand corner, then clicking on tools, then media manager.

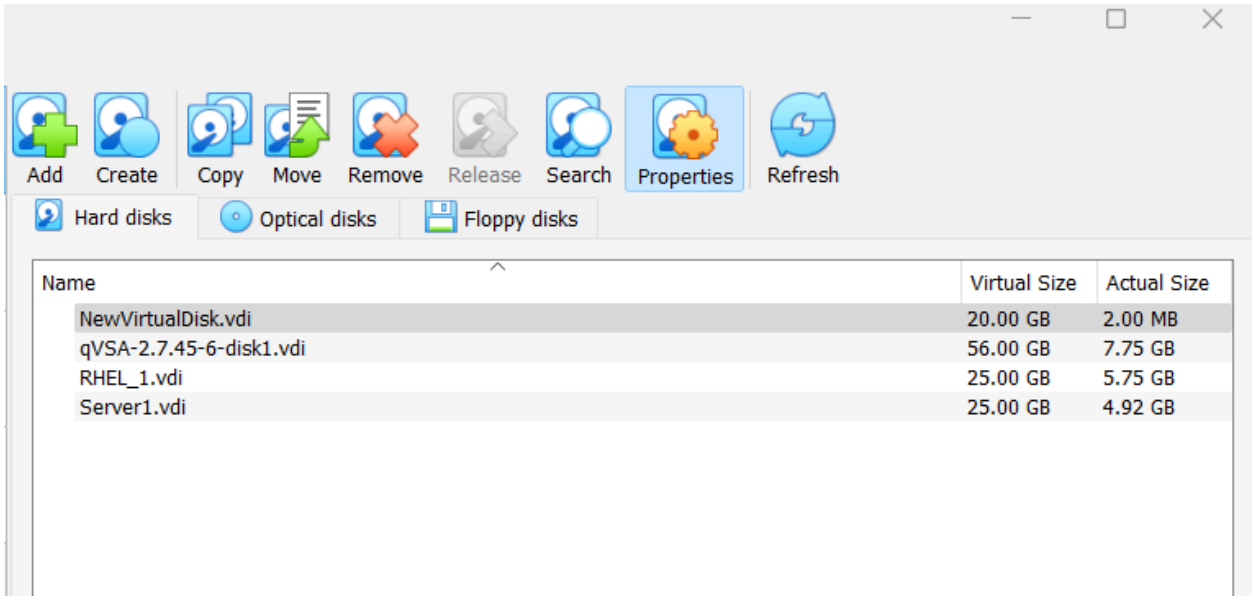
- This is what media manager will look like:



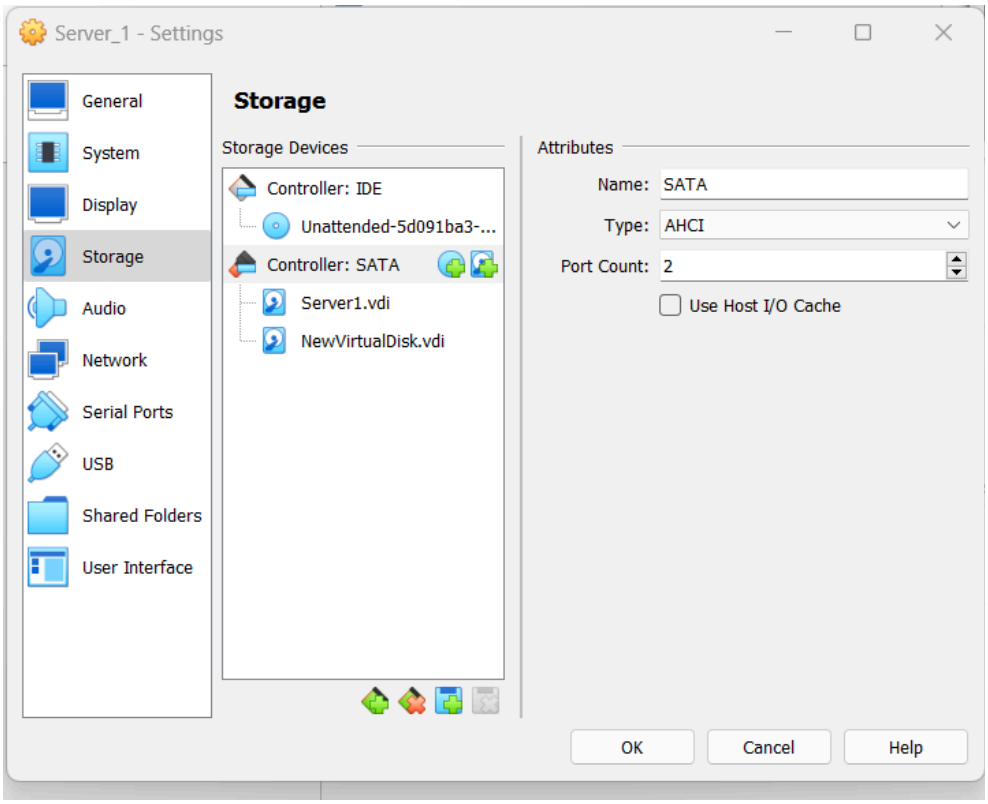
- I will now click on create on the top left corner as seen in the previous screenshot

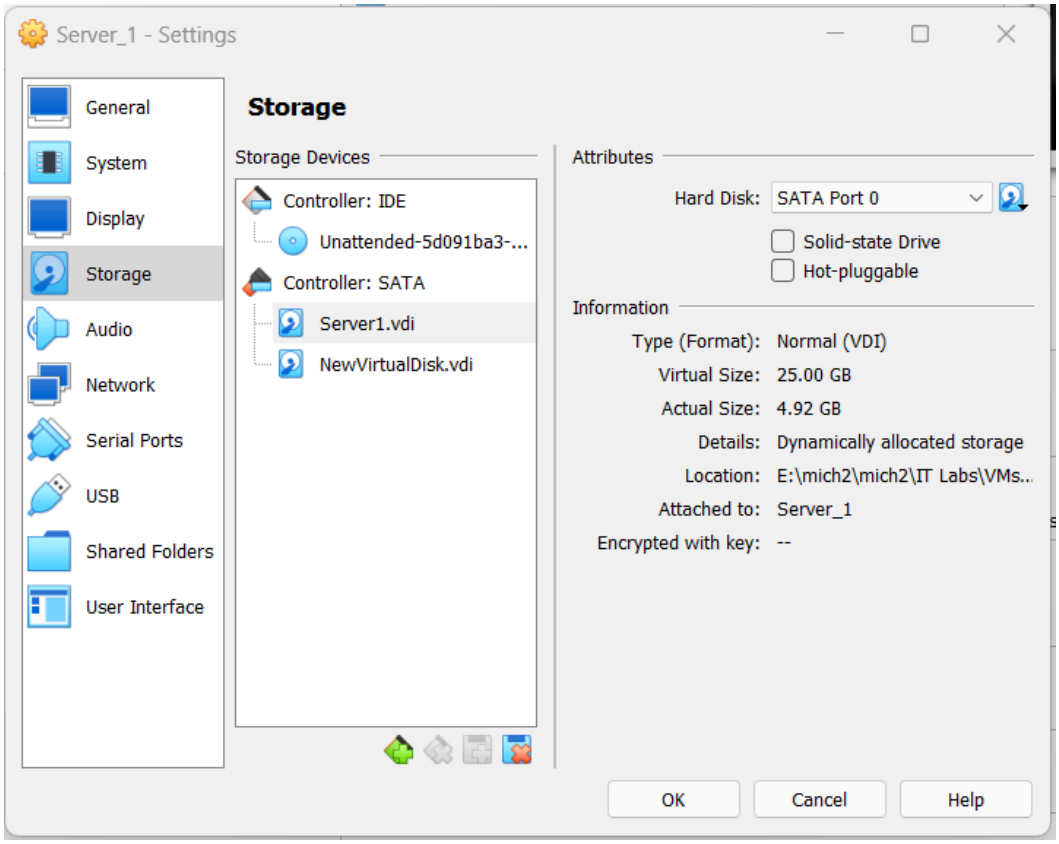


I followed the steps provided. I chose to create a VDI. I thin provisioned the storage so that the full storage mount won't be used right away. The storage amount I chose for the purposes of the lab is 20 GB. The new disk is called NewVirtualDisk.vdi.



Step 2. I attached NewVirtualDisk.vdi to my Ubuntu Server VM. This was done by clicking on the settings for the VM. Then I clicked on storage, then I clicked on the SATA controller then on green add hard disk icon.





Step 3. Now I will log into my Ubuntu Server and check if the new virtual disk has been attached to it. The whole concept of adding a virtual disk is essentially the same as if I put a new HDD or SSD into my laptop where my VM runs on.

```
spy3@spyserver1:~$ lsblk
NAME                      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0                       7:0    0 63.9M  1 loop /snap/core20/2105
loop1                       7:1    0 63.9M  1 loop /snap/core20/2264
loop2                       7:2    0   87M  1 loop /snap/lxd/27037
loop3                       7:3    0   87M  1 loop /snap/lxd/27948
loop4                       7:4    0 40.4M  1 loop /snap/snapd/20671
loop5                       7:5    0 39.1M  1 loop /snap/snapd/21184
sda                         8:0    0   25G  0 disk
├─sda1                      8:1    0    1M  0 part
├─sda2                      8:2    0    2G  0 part /boot
└─sda3                      8:3    0   23G  0 part
  └─ubuntu--vg-ubuntu--lv 253:0    0 11.5G  0 lvm  /
sdb                         8:16   0   20G  0 disk
sr0                        11:0    1    2G  0 rom
spy3@spyserver1:~$
```

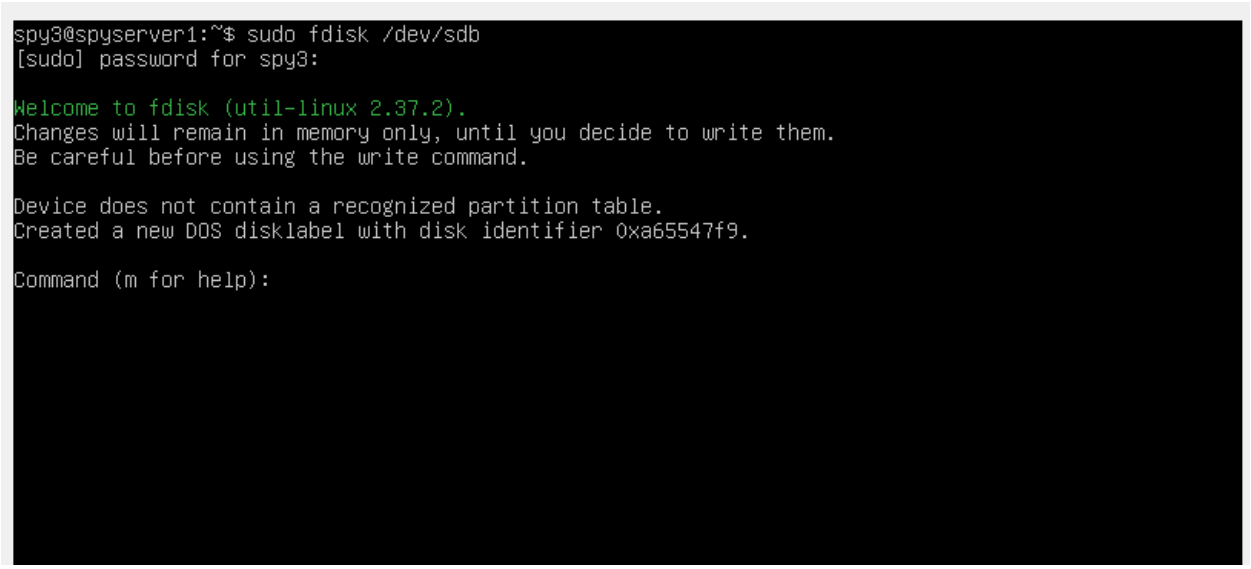
My new disk is the sdb (20GB) device. As you can see I have an sda device as well. This is the first storage device that my Linux VM used, my new disk attached is called sdb since it is the

second storage device attached to the VM. The Linux OS names storage devices in alphabetical order. The sda1, sda2, and sda3 devices represent the partitions made out of the sda device.

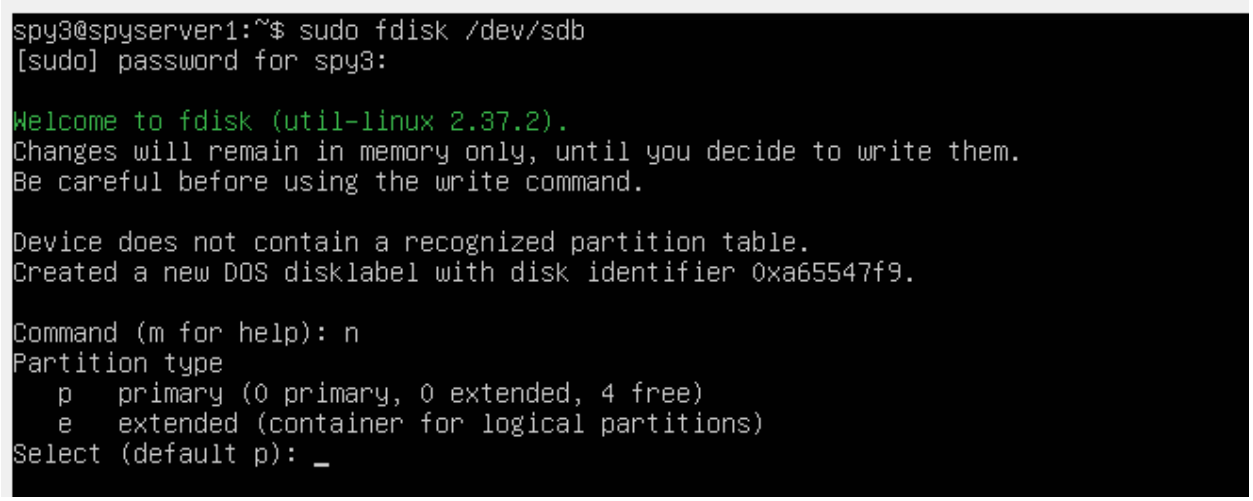
Section #2 Partition storage devices

Now I will partition my new sdb into two partitions, each will be 10 GB in size.

Step 1. I will be using fdisk to partition the sdb device. I used sudo fdisk, and the full path to my sdb storage device.



Step 2. I entered n to start creating a new partition.



Step 3. Entered p for primary partition.

```
spy3@spyserver1:~$ sudo fdisk /dev/sdb
[sudo] password for spy3:

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa65547f9.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): _
```

Step 4. Enter 1 for the default partition.

```
spy3@spyserver1:~$ sudo fdisk /dev/sdb
[sudo] password for spy3:

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa65547f9.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
```

Step 5. Press enter to choose the default first sector.

```
spy3@spyserver1:~$ sudo fdisk /dev/sdb
[sudo] password for spy3:

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa65547f9.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): _
```

Step 6. Choose the last sector. This can be determined by doing some math. One gb is 1024 mb and I want the partition to be 10 GBs in size. 1024 mbs in a gb * 10 gbs * 1024 kb /mb * 1024 b /kb/ 512 bytes result in the last sector being 20971520.

```
spy3@spyserver1:~$ sudo fdisk /dev/sdb
[sudo] password for spy3:

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa65547f9.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): 20971520

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): _
```

Now I have a 10 gb partition. I will complete the same process for the second partition. To exit and save this new partition to the partition table enter w.

```
spy3@spyserver1:~$ sudo fdisk /dev/sdb
[sudo] password for spy3:

Welcome to fdisk (util-linux 2.37.2).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xa65547f9.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-41943039, default 41943039): 20971520

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

spy3@spyserver1:~$ _
```

To further verify that the partition was created I can use the lsblk command. The partition is sdb1.

```
spy3@spyserver1:~$ lsblk
NAME                      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0                       7:0    0 63.9M  1 loop /snap/core20/2105
loop1                       7:1    0 63.9M  1 loop /snap/core20/2264
loop2                       7:2    0   87M  1 loop /snap/lxd/27037
loop3                       7:3    0   87M  1 loop /snap/lxd/27948
loop4                       7:4    0 40.4M  1 loop /snap/snapd/20671
loop5                       7:5    0 39.1M  1 loop /snap/snapd/21184
sda                         8:0    0   25G  0 disk
├─sda1                      8:1    0    1M  0 part
├─sda2                      8:2    0    2G  0 part /boot
└─sda3                      8:3    0   23G  0 part
  └─ubuntu--vg-ubuntu--lv 253:0    0 11.5G  0 lvm  /
sdb                         8:16   0   20G  0 disk
└─sdb1                      8:17   0   10G  0 part
sr0                        11:0    1    2G  0 rom
spy3@spyserver1:~$
```

Now here are both the partitions created.

```
spy3@spyserver1:~$ lsblk
NAME                      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
loop0                       7:0    0 63.9M  1 loop /snap/core20/2105
loop1                       7:1    0 63.9M  1 loop /snap/core20/2264
loop2                       7:2    0   87M  1 loop /snap/lxd/27037
loop3                       7:3    0   87M  1 loop /snap/lxd/27948
loop4                       7:4    0 40.4M  1 loop /snap/snapd/20671
loop5                       7:5    0 39.1M  1 loop /snap/snapd/21184
sda                         8:0    0   25G  0 disk
├─sda1                      8:1    0    1M  0 part
├─sda2                      8:2    0    2G  0 part /boot
└─sda3                      8:3    0   23G  0 part
  └─ubuntu--vg-ubuntu--lv 253:0    0 11.5G  0 lvm  /
sdb                         8:16   0   20G  0 disk
├─sdb1                      8:17   0   10G  0 part
└─sdb2                      8:18   0   10G  0 part
sr0                        11:0    1    2G  0 rom
spy3@spyserver1:~$ _
```

Section #3 Put file system on partitions/ device

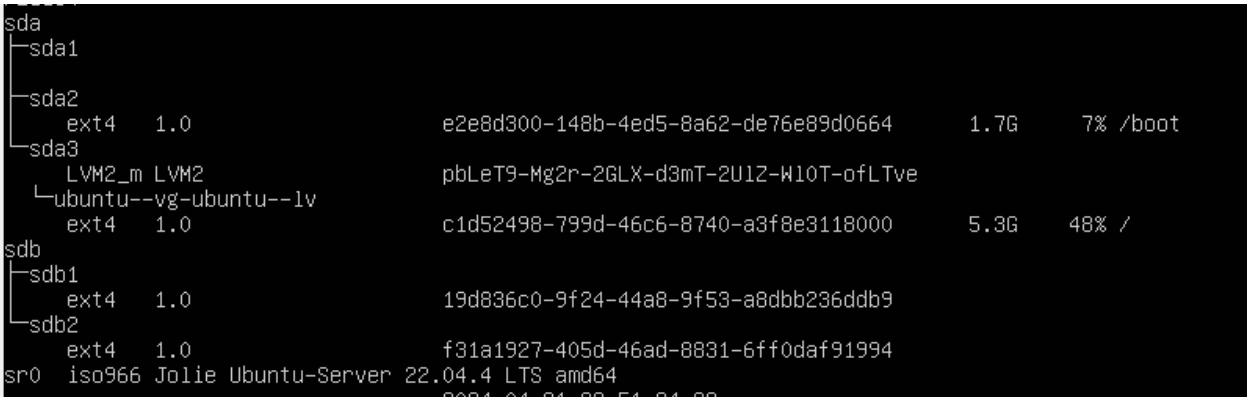
I now have two partitions, but they can't be used yet since a filesystem has not been put onto them. I will now have to put a filesystem over the two partitions I created.

I created a filesystem for the partitions using mkfs. I will choose ext4 since it is commonly used, stable, and great for general purpose use. It can support large file sizes and has journaling to facilitate data integrity as well.

```
spy3@spyserver1:~$ sudo mkfs.ext4 /dev/sdb1
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: 19d836c0-9f24-44a8-9f53-a8dbb236ddb9
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

spy3@spyserver1:~$ sudo mkfs.ext4 /dev/sdb2
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 2621184 4k blocks and 655360 inodes
Filesystem UUID: f31a1927-405d-46ad-8831-6ff0daf91994
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

spy3@spyserver1:~$
```

Note how each partition has a certain number of inodes. Inodes are what are used by the filesystem to track the location of data like a file on the disk. There is a set number of inodes that can be used so if I make more files than there are indies, I will run into issues.

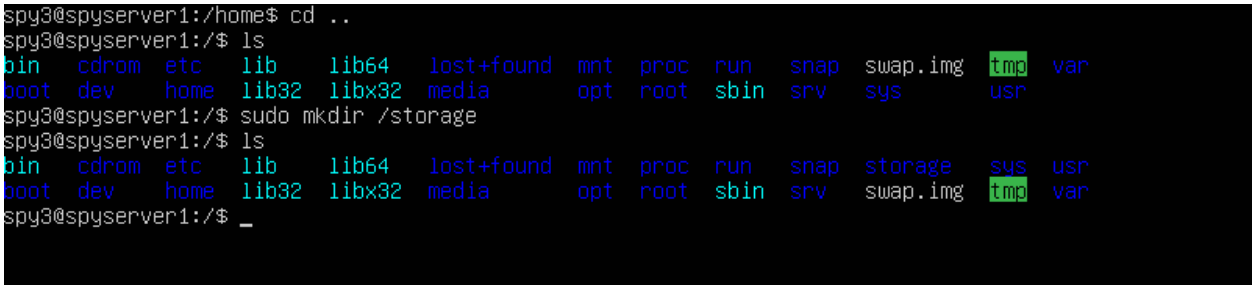
Verify that the partitions have a filesystem on them.

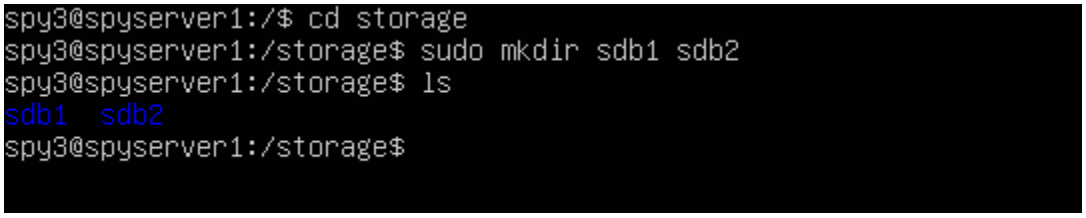


Section #4 Mount filesystem and make it persistent

The last step I will need to do is mount the file system.For Linux devices this is vital since mounting allows the storage device to be accessible from the OS. Mounting tells the OS where to find the data on a storage device so that applications and the OScan use data.

Step 1. Under the root directory I created a new directory called storage. This new directory will serve as the mount point. Within it I will create two more directories called sdb1 and sdb2 to directly map the partitions to the new mount points.





Step 2. I will edit the /etc/fstab file to make the mounting of the partitions persistent.

```
 GNU nano 6.2                                 /etc/fstab *
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/ubuntu-vg/ubuntu-lv during curtin installation
/dev/disk/by-id/dm-uuid-LVM-UPc5C3fdARpNrqa5Pnwrrv8FPXGU84rE9tNZD5QAFOzFz2cNpiD7BBSrbrWkVFf2 / ext4
# /boot was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/e2e8d300-148b-4ed5-8a62-de76e89d0664 /boot ext4 defaults 0 1
/swap.img       none    swap    sw      0       0

/dev/sdb1      /storage/sdb1   ext4 defaults 0 0
/dev/sdb2      /storage/sdb2   ext4 defaults 0 0
```

Step 3. I will test to see if the partitions are mounted after shutting down my VM and turning it back on.

```
spy3@spyserver1:/storage/sdb1$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                              297M  1.1M  296M   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv   12G  5.4G  5.3G  51% /
tmpfs                              1.5G     0  1.5G   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sdb1                          9.8G   24K  9.3G   1% /storage/sdb1
/dev/sdb2                          9.8G   24K  9.3G   1% /storage/sdb2
/dev/sda2                          2.0G  129M  1.7G   8% /boot
tmpfs                              297M  4.0K  297M   1% /run/user/1001
spy3@spyserver1:/storage/sdb1$
```

Step 4. Create files on the two partitions to show that the filesystem works.

- I ran into an issue since the /storage directory and its subdirectories are owned by root. I could not create files on the sdb1 mount point since my user spy3 is not root. To get around this issue I will create a group called storageusers and add users to the group. Then, I will change the ownership of the /storage directory to include the group, and root.

```
spy3@spyserver1:/storage/sdb1$ sudo groupadd storageusers
spy3@spyserver1:/storage/sdb1$
```

```
spy3@spyserver1:/storage/sdb1$ sudo usermod -aG storageusers spy3
spy3@spyserver1:/storage/sdb1$ _
```

```
spy3@spyserver1:/storage/sdb1$ sudo usermod -aG storageusers spy3
spy3@spyserver1:/storage/sdb1$ groups spy3
spy3 : spy3 sudo storageusers
spy3@spyserver1:/storage/sdb1$
```

Add the storageusers group as owners of the /storage directory.

```
spy3@spyserver1:/$ sudo chown :storageusers /storage
spy3@spyserver1:/$ ls
bin   cdrom  etc   lib    lib64   lost+found  mnt  proc  run   snap  storage   sys  usr
boot  dev    home  lib32  libx32  media       opt  root  sbin  srv   swap.img  tmp  var
spy3@spyserver1:/$ ls -l
total 2222156
lrwxrwxrwx   1 root root                7 Feb 16 18:37 bin -> usr/bin
drwxr-xr-x   4 root root             4096 Mar 23 22:27 boot
dr-xr-xr-x   2 root root             4096 Mar 23 22:07 cdrom
drwxr-xr-x  20 root root             4160 Apr 21 23:57 dev
drwxr-xr-x  97 root root             4096 Apr 22 00:10 etc
drwxr-xr-x   5 root root             4096 Mar 23 23:14 home
lrwxrwxrwx   1 root root                7 Feb 16 18:37 lib -> usr/lib
lrwxrwxrwx   1 root root                9 Feb 16 18:37 lib32 -> usr/lib32
lrwxrwxrwx   1 root root                9 Feb 16 18:37 lib64 -> usr/lib64
lrwxrwxrwx   1 root root               10 Feb 16 18:37 libx32 -> usr/libx32
drwx------   2 root root            16384 Mar 23 22:22 lost+found
drwxr-xr-x   2 root root             4096 Feb 16 18:37 media
drwxr-xr-x   2 root root             4096 Feb 16 18:37 mnt
drwxr-xr-x   2 root root             4096 Feb 16 18:37 opt
dr-xr-xr-x 194 root root                0 Apr 21 23:57 proc
drwx------   5 root root             4096 Mar 28 23:09 root
drwxr-xr-x  29 root root              840 Apr 21 23:57 run
lrwxrwxrwx   1 root root                8 Feb 16 18:37 sbin -> usr/sbin
drwxr-xr-x   6 root root             4096 Feb 16 18:52 snap
drwxr-xr-x   2 root root             4096 Feb 16 18:37 srv
drwxrwxrwx   4 root storageusers     4096 Apr 21 23:45 storage
-rw-------   1 root root       2275409920 Mar 23 22:26 swap.img
dr-xr-xr-x  13 root root                0 Apr 21 23:57 sys
drwxrwxrwt  12 root root             4096 Apr 22 00:02 tmp
drwxr-xr-x  14 root root             4096 Feb 16 18:37 usr
drwxr-xr-x  13 root root             4096 Feb 16 18:46 var
spy3@spyserver1:/$ _
```

Then create a file in the /storage/sdb1 directory.

```
spy3@spyserver1:/storage/sdb2$ ls
lost+found  test.txt
spy3@spyserver1:/storage/sdb2$ _
```

- Make sure you restart the system to have the group permissions set

Section #5 Removing storage devices from a Linux system

To remove storage on a Linux system you need to use umount. Umount unmounts the storage device from the file system. If the storage devices are on the fstab file, you simply delete them from it.

Step 1. Use mount to identify the mount points that the storage devices/ partitions are mounted on.

```
/var/lib/snapd/snaps/core20_2264.snap on /snap/core20/2264 type squashfs (ro,nodev,relatime,errors=c
ontinue,x-gdu.hide)
/var/lib/snapd/snaps/core20_2105.snap on /snap/core20/2105 type squashfs (ro,nodev,relatime,errors=c
ontinue,x-gdu.hide)
/var/lib/snapd/snaps/lxd_27948.snap on /snap/lxd/27948 type squashfs (ro,nodev,relatime,errors=conti
nue,x-gdu.hide)
/var/lib/snapd/snaps/lxd_28373.snap on /snap/lxd/28373 type squashfs (ro,nodev,relatime,errors=conti
nue,x-gdu.hide)
/var/lib/snapd/snaps/snapd_21184.snap on /snap/snapd/21184 type squashfs (ro,nodev,relatime,errors=c
ontinue,x-gdu.hide)
/var/lib/snapd/snaps/snapd_21465.snap on /snap/snapd/21465 type squashfs (ro,nodev,relatime,errors=c
ontinue,x-gdu.hide)
/dev/sda2 on /boot type ext4 (rw,relatime)
/dev/sdb2 on /storage/sdb2 type ext4 (rw,relatime)
/dev/sdb1 on /storage/sdb1 type ext4 (rw,relatime)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,nosuid,nodev,noexec,relatime)
tmpfs on /run/snapd/ns type tmpfs (rw,nosuid,nodev,noexec,relatime,size=303952k,mode=755,inode64)
nsfs on /run/snapd/ns/lxd.mnt type nsfs (rw)
tmpfs on /run/user/1001 type tmpfs (rw,nosuid,nodev,relatime,size=303948k,nr_inodes=75987,mode=700,u
id=1001,gid=1001,inode64)
spy3@spyserver1:~$
```

Step 2. Use sudo umount mountpoint to unmount the storage devices from the Linux system.

```
spy3@spyserver1:~$ sudo umount /storage/sdb1
spy3@spyserver1:~$ sudo umount /storage/sdb2
spy3@spyserver1:~$
```

Step 3. Edit the /etc/fstab and remove the storage devices you unmounted.

```
 GNU nano 6.2                                 /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/ubuntu-vg/ubuntu-lv during curtin installation
/dev/disk/by-id/dm-uuid-LVM-UPc5C3fdARpNrqa5Pnwrrv8FPXGU84rE9tNZD5QAFOzFz2cNpiD7BBSrbrWkVFf2 / ext4
# /boot was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/e2e8d300-148b-4ed5-8a62-de76e89d0664 /boot ext4 defaults 0 1
/swap.img       none    swap    sw      0       0

/dev/sdb1      /storage/sdb1   ext4 defaults 0 0
/dev/sdb2      /storage/sdb2   ext4 defaults 0 0
```

I will be removing sdb1 and sdb2 from the /etc/fstab file.

```
 GNU nano 6.2                                 /etc/fstab *
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/ubuntu-vg/ubuntu-lv during curtin installation
/dev/disk/by-id/dm-uuid-LVM-UPc5C3fdARpNrqa5Pnwrrv8FPXGU84rE9tNZD5QAFOzFz2cNpiD7BBSrbrWkVFf2 / ext4
# /boot was on /dev/sda2 during curtin installation
/dev/disk/by-uuid/e2e8d300-148b-4ed5-8a62-de76e89d0664 /boot ext4 defaults 0 1
/swap.img       none    swap    sw      0       0
```

Now they are gone. So they won't be mounted when the system boots.

Step 4. Delete the mount points you used.

```
spy3@spyserver1:~$ cd /storage/
spy3@spyserver1:/storage$ ls
sdb1  sdb2
spy3@spyserver1:/storage$ rm -r sdb1
rm: remove write-protected directory 'sdb1'? y
spy3@spyserver1:/storage$ rm -r sdb2
rm: remove write-protected directory 'sdb2'? y
spy3@spyserver1:/storage$
```

Note that you will be prompted about whether or not you want to remove the write-protected directory if you are deleting a directory that was used as a mount point. This is because the system wants to prevent the accidental deletion of a mount point. Also, I used cd to enter the two mount points, and they were empty, so this demonstrates that the storage device has been removed.

Section #6 Storage Management and Monitoring

After you manage storage devices on a Linux system by partitioning, formatting, and mounting them to the system, you will need to monitor them. There are multiple storage metrics you will want to monitor on Linux systems. Here are some the metrics you should watch:

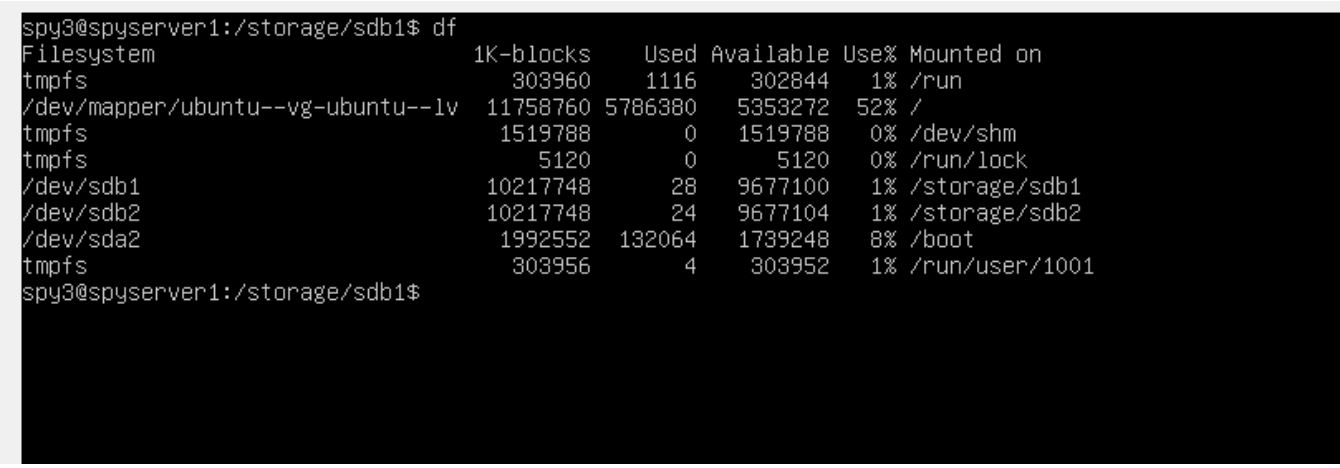
Metric	Reasoning	Tool used to monitor
1. Disk utilization	Disk utilization determines how much storage is being used in bytes, kbs, mbs, and gbs, etc… . It is important to monitor disk utilization since running out of available storage can slow a system down and make it unstable.	df/ df -h
2. Inode utilization	For Linux systems storage is measured in two ways, how much physical storage is being used (like KBs and GBS) and how many inodes are used. Inodes are used by Linux filesystems like ext4 and others to track files. File systems provide you with a set number of inodes, and if you go over the limit you will not be able to create any more files.	Df - i
3. I/O wait time	I/O wait time refers to the time spent by the CPU waiting for reads and writes from a storage device. If a CPU has to wait for the hard drive the system will feel slower since the cpu can’t do anything.	iostat
4. Disk Throughput	Disk throughput measures how fast data can be sent from your storage device to your computer.	Iostat -x 1

5. Disk Latency	Disk latency is the time it takes a hard drive to respond to read and write requests. Lower latency is better than high latency.	Iostat - x 1
6. File System fragmentation	Fragmentation is when a file's data is spread around a disk randomly, and not close together. Since the data is not all in the same sector, it takes longer for the storage device to read and write to the data.	filefrag
7. Mount Point Availability	Monitoring mount point availability is important since mount points are used to access a storage device on a Linux system. If a mount point is unavailable (disconnected) then users on the Linux system will not be able to access their data, and the system will not be able to use the data.	Mount or cat /etc/fstab

Now that I have gone over the various metrics, why they should be monitored, and what you can use to watch them, I will demonstrate some of them.

1. Disk Utilization

- I will use df first to demonstrate its output.



Df displays the storage used in a very unfriendly manner. As you can see, for the two partitions I created in this lab (sdb1 and sdb2) only 1% of their storage capacity has been used. But we can't determine how much we used in finer detail. What is 28 and 24? What do the other numbers mean?

- Now I will use df -h to figure out the storage used in more practical terms.

```
spy3@spyserver1:/storage/sdb1$ df -h
Filesystem                         Size  Used Avail Use% Mounted on
tmpfs                              297M  1.1M  296M   1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv   12G  5.6G  5.2G  52% /
tmpfs                              1.5G     0  1.5G   0% /dev/shm
tmpfs                              5.0M     0  5.0M   0% /run/lock
/dev/sdb1                          9.8G   28K  9.3G   1% /storage/sdb1
/dev/sdb2                          9.8G   24K  9.3G   1% /storage/sdb2
/dev/sda2                          2.0G  129M  1.7G   8% /boot
tmpfs                              297M  4.0K  297M   1% /run/user/1001
spy3@spyserver1:/storage/sdb1$ _
```

Now we can see that I used 28K or kilobytes (1000 bytes * 28 for sdb1). I have 9.3 GB available to use for sdb1 as well. For sdb2 I have used 24K or kilobytes (24 * 1000 bytes). For sdb2 I have 9.3 GB available to use as well. You can see that df -h produces a more interpretable output than just using df.

2. Inode utilization

- Now I will use df -i to display the inode utilization.

```
spy3@spyserver1:/storage/sdb1$ df -i | grep /dev/sdb1
/dev/sdb1                          655360    13 655347    1% /storage/sdb1
spy3@spyserver1:/storage/sdb1$ df -i | grep /dev/sdb2
/dev/sdb2                          655360    12 655348    1% /storage/sdb2
spy3@spyserver1:/storage/sdb1$ df -i
Filesystem                         Inodes IUsed  IFree IUse% Mounted on
tmpfs                              379947   863 379084    1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv 753664 83749 669915   12% /
tmpfs                              379947     1 379946    1% /dev/shm
tmpfs                              379947     3 379944    1% /run/lock
/dev/sdb1                          655360    13 655347    1% /storage/sdb1
/dev/sdb2                          655360    12 655348    1% /storage/sdb2
/dev/sda2                          131072   316 130756    1% /boot
tmpfs                               75989    25  75964    1% /run/user/1001
spy3@spyserver1:/storage/sdb1$
```

First I used df -i and piped its output to grep. I used grep to look specifically for /dev/sdb1 and /dev/sdb2. Then I ran df -i without piping its output. For sdb1 I have used only 13 inodes out of 655360 available. For sdb2 I have used 12 out of 655360. Note that the number of inodes is

allocated during file system allocation and is based on the size of the partition or disk the filesystem has been created on. I have plenty of inodes available to create files and soft/ hard links.

3. I/O Wait Time, Latency, and Throughput

- For monitoring I/O wait time I will use iostat.

```
spy3@spyserver1:/storage/sdb1$ iostat
Command 'iostat' not found, but can be installed with:
sudo apt install sysstat
spy3@spyserver1:/storage/sdb1$ sudo apt install sysstat
[sudo] password for spy3:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libsensors-config libsensors5
Suggested packages:
  lm-sensors isag
The following NEW packages will be installed:
  libsensors-config libsensors5 sysstat
0 upgraded, 3 newly installed, 0 to remove and 42 not upgraded.
Need to get 519 kB of archives.
After this operation, 1,649 kB of additional disk space will be used.
Do you want to continue? [Y/n] _
```

Note that you might need to install iostat onto your Linux system. Apparently I do not have it installed, but I can install it by installing the sysstat package.

Here is the output of running iostat.

```
spy3@spyserver1:/storage/sdb1$ iostat
Linux 5.15.0-101-generic (spyserver1)   04/25/2024      _x86_64_        (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.25    0.02    0.84    0.30    0.00   98.58

Device             tps    kB_read/s    kB_wrtn/s    kB_dscd/s    kB_read    kB_wrtn    kB_dscd
dm-0             16.87       281.02        82.94         0.00     354649     104672          0
loop0             0.03         0.27         0.00         0.00        346          0          0
loop1             0.17         1.71         0.00         0.00       2161          0          0
loop2             0.04         0.86         0.00         0.00       1080          0          0
loop3             0.06         0.87         0.00         0.00       1099          0          0
loop4             0.03         0.27         0.00         0.00        343          0          0
loop5             0.45        16.59         0.00         0.00      20934          0          0
loop6             0.01         0.01         0.00         0.00         14          0          0
sda              11.48       288.76        83.05         0.00     364414     104812          0
sdb               0.32         9.57         0.02         0.00      12077         28          0
sr0               0.28         9.97         0.00         0.00      12577          0          0


spy3@spyserver1:/storage/sdb1$
```

Here is the output of running iostat 1 which causes the iostat command to output the change in the metrics every one second.

```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0.00    0.00    0.25    0.00    0.00   99.75

Device             tps    kB_read/s    kB_wrtn/s    kB_dscd/s    kB_read    kB_wrtn    kB_dscd
dm-0              0.00         0.00         0.00         0.00          0          0          0
loop0             0.00         0.00         0.00         0.00          0          0          0
loop1             0.00         0.00         0.00         0.00          0          0          0
loop2             0.00         0.00         0.00         0.00          0          0          0
loop3             0.00         0.00         0.00         0.00          0          0          0
loop4             0.00         0.00         0.00         0.00          0          0          0
loop5             0.00         0.00         0.00         0.00          0          0          0
loop6             0.00         0.00         0.00         0.00          0          0          0
sda               0.00         0.00         0.00         0.00          0          0          0
sdb               0.00         0.00         0.00         0.00          0          0          0
sr0               0.00         0.00         0.00         0.00          0          0          0
```

Now what does this all mean? How can I determine the I/O wait time, throughput, and latency by looking at these metrics?

1. Determining I/O wait time:
   Under the avg-cpu section of the iostat output there is a %iowait metric. Currently the system has a 0.00% iowait. This means that 0% of the cpu time is spent waiting for i/o operations to occur.
2. Determining throughput: Throughput determines how much data is written or read from the storage device.You can find this by using iostat -x on a specific device and looking for the rKB/s and wKB/s metric.

   ```
   spy3@spyserver1:~$ iostat -hx /dev/sda
   Linux 5.15.0-105-generic (spyserver1)   05/23/2024      _x86_64_        (4 CPU)

   avg-cpu:  %user   %nice %system %iowait  %steal   %idle
              0.3%    0.0%    1.4%    0.4%    0.0%   97.9%

       r/s     rkB/s   rrqm/s  %rrqm r_await rareq-sz Device
      9.25    454.1k     3.27  26.1%    2.29    49.1k sda

       w/s     wkB/s   wrqm/s  %wrqm w_await wareq-sz Device
      3.64    138.4k     4.58  55.7%    6.12    38.0k sda

       d/s     dkB/s   drqm/s  %drqm d_await dareq-sz Device
      0.00     0.0k      0.00   0.0%    0.00     0.0k sda

       f/s f_await  aqu-sz  %util Device
      0.72    2.21    0.05   2.1% sda
   ```

3. Determining latency: Latency determines how long it takes an I/O device to complete a read/write operation. You can find this metric by using iostat -x on a specific storage device and looking for r_await and w_await. The w_await is 2.29

and the r_await is 6.12 for /dev/sda so that means that it takes only 2.29 milliseconds to perform read operations, but it takes longer to do write operations.

```
spy3@spyserver1:~$ iostat -hx /dev/sda
Linux 5.15.0-105-generic (spyserver1)   05/23/2024      _x86_64_        (4 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          0.3%    0.0%    1.4%    0.4%    0.0%   97.9%

     r/s     rkB/s   rrqm/s  %rrqm r_await rareq-sz Device
    9.25    454.1k     3.27  26.1%    2.29    49.1k sda

     w/s     wkB/s   wrqm/s  %wrqm w_await wareq-sz Device
    3.64    138.4k     4.58  55.7%    6.12    38.0k sda

     d/s     dkB/s   drqm/s  %drqm d_await dareq-sz Device
    0.00     0.0k     0.00   0.0%    0.00     0.0k sda

     f/s f_await  aqu-sz  %util Device
    0.72    2.21    0.05   2.1% sda
```

4. File Fragmentation
   File fragmentation results when data is scrambled around a disk. This results in longer times to write and read data and can even lead to data being lost and corrupted.

   - I will use filefrag to check the fragmentation of sdb1 and sdb2 files. Note that this can't be used on a partition itself but rather the files on a partition/ disk.

```
spy3@spyserver1:/storage/sdb1$ ls
lost+found  test  text1.txt
spy3@spyserver1:/storage/sdb1$ filefrag -v test
Filesystem type is: ef53
File size of test is 72 (1 block of 4096 bytes)
 ext:     logical_offset:        physical_offset: length:   expected: flags:
   0:        0..       0:      34305..     34305:      1:             last,eof
test: 1 extent found
spy3@spyserver1:/storage/sdb1$
```

5. Mount Point Availability
   - To check mount point availability I will use the mount command and grep the output to look for a specific storage device.

```
spy3@spyserver1:/storage/sdb1$ mount | grep /dev/sdb1
/dev/sdb1 on /storage/sdb1 type ext4 (rw,relatime)
spy3@spyserver1:/storage/sdb1$ mount | grep /dev/sdb2
/dev/sdb2 on /storage/sdb2 type ext4 (rw,relatime)
spy3@spyserver1:/storage/sdb1$ _
```

6. Inode Utilization

To view the used inodes I will use df -i. Inodes are the basis of many Linux file systems and serve as a pointer to space on a storage device for a file. They also contain important metadata related to the files they point to such as file type, file size, permissions, and ownership of the file. Linux partitions are granted a limited number of inodes, and it is possible to go over your inode limit and not even use up all of your physical storage. This occurs if you create too many small files with different names and forget to delete some of them. If you run out of inodes you won't be able to create any new files until the old inodes are freed up.

```
spy3@spyserver1:~$ df -i
Filesystem                         Inodes  IUsed  IFree IUse% Mounted on
tmpfs                              379938    857 379081    1% /run
/dev/mapper/ubuntu--vg-ubuntu--lv 753664 119468 634196   16% /
tmpfs                              379938      1 379937    1% /dev/shm
tmpfs                              379938      3 379935    1% /run/lock
/dev/sda2                          131072    320 130752    1% /boot
tmpfs                               75987     25  75962    1% /run/user/1001
spy3@spyserver1:~$ df -i /dev/sda2
Filesystem     Inodes IUsed  IFree IUse% Mounted on
/dev/sda2      131072   320 130752    1% /boot
spy3@spyserver1:~$ _
```

Only 1% of my inodes have been used up for the file systems used on my storage devices and partitions. This is great, I have plenty of inodes available for file creation.