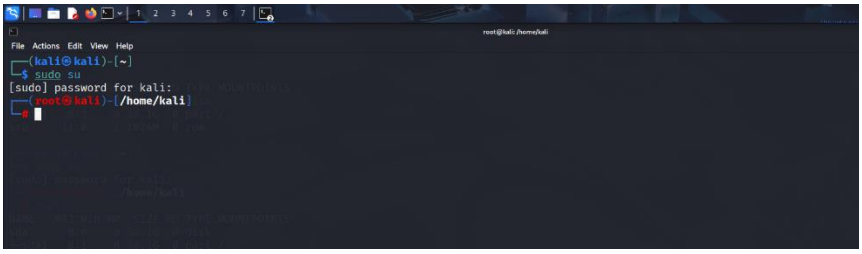
Navigating the Linux Filesystem and Permission Settings.

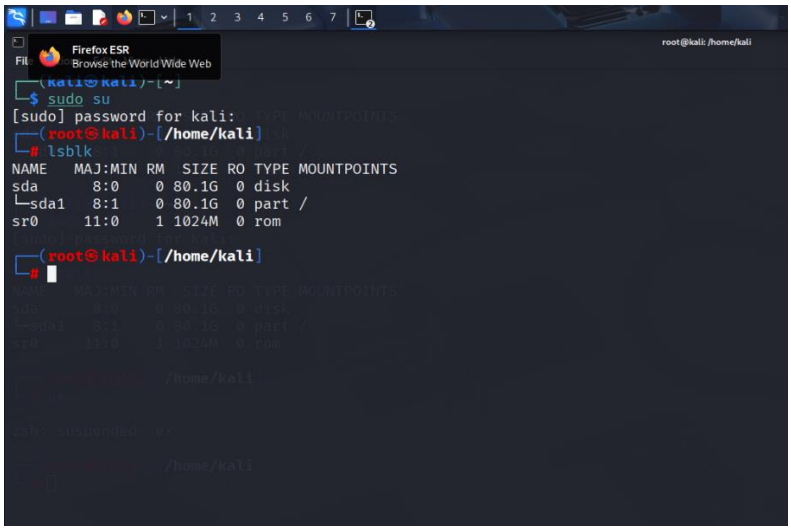
Part 1: Exploring Filesystems in Linux:

Step 1: Access the command line and launch the VM Workstation and open a terminal.



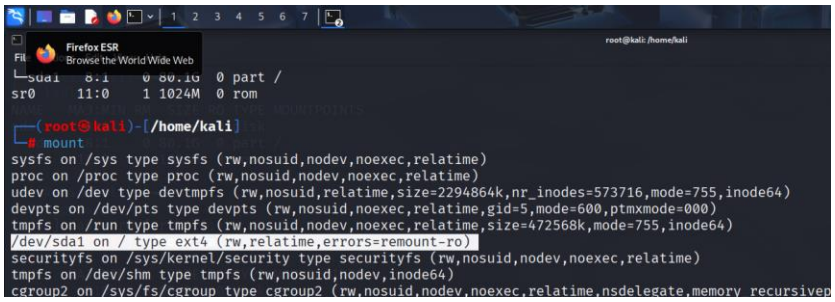
Step 2: Display the filesystems currently mounted: A filesystem must be mounted before use. Mounting attaches a storage device's partition to a directory, making its contents accessible to the operating system. This directory, which becomes the root of the attached filesystem.

i. Use the **lsblk** command to display all block devices:



The output above shows that the Workstation has two block devices installed: sr0 and sda.

ii. Use the **mount** command to display more detailed information on the currently mounted filesystems in the Workstation.



Focusing on the root filesystem, the filesystem stored in **/dev/sda1.** The root filesystem is where the Linux operating system itself is stored; all the programs, tools, configuration files are stored in root filesystem by default.

iii. Run the **mount** command again with the pipe | to send the output of mount to **grep** to filter the output and display only the root filesystem:

```
┌──(root㉿kali)-[/home/kali]
└─# mount | grep sda1
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro)

┌──(root㉿kali)-[/home/kali]
└─#
```

The `mount` output indicates that the root filesystem resides on the first partition of the `sda` device (`/dev/sda1`). We identify it as the root filesystem by its mount point `/`. The output also specifies the partition's filesystem type, `ext4` here, and lists its mount options in parentheses.

iv. Issue the following two commands below on the **Security Workstation VM**:

Step 3: Manually mounting and unmounting filesystems.

i. Use the **ls -l** command to verify that the directory **second_drive** is in the analyst's home directory.

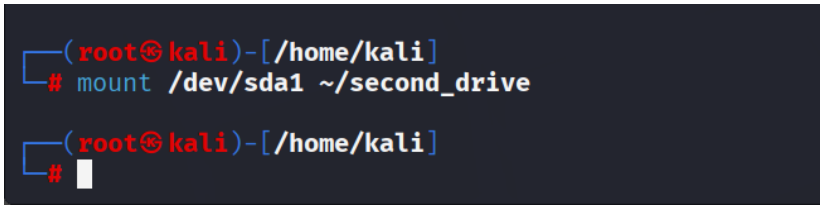
Note: If the `second_drive` directory does not already exist, create it using `mkdir second_drive`.

```
┌──(root㉿kali)-[/home/kali]
└─# cd ~

┌──(root㉿kali)-[~]
└─# ls -l
total 760
drwxr-xr-x 2 root root   4096 Aug 29 13:00 NewFolder
-rw-r--r-- 1 root root     91 May  4  2025 pass.txt
-rw-r--r-- 1 root root 252928 Aug  8  2025 Payload.exe
-rw-r--r-- 1 root root    673 Feb 10  2025 practice.txt
-rw-r--r-- 1 root root   3745 Dec 17 07:17 special.txt
-rw-r--r-- 1 root root 252928 Aug 30 02:27 testpaylad1.exe
-rw-r--r-- 1 root root 252928 Aug 29 13:01 testpayload.exe

┌──(root㉿kali)-[~]
└─# mkdir second_drive

┌──(root㉿kali)-[~]
└─# ls -l
total 764
drwxr-xr-x 2 root root   4096 Aug 29 13:00 NewFolder
-rw-r--r-- 1 root root     91 May  4  2025 pass.txt
-rw-r--r-- 1 root root 252928 Aug  8  2025 Payload.exe
-rw-r--r-- 1 root root    673 Feb 10  2025 practice.txt
drwxr-xr-x 2 root root   4096 Feb 14 04:35 second_drive
-rw-r--r-- 1 root root   3745 Dec 17 07:17 special.txt
-rw-r--r-- 1 root root 252928 Aug 30 02:27 testpaylad1.exe
-rw-r--r-- 1 root root 252928 Aug 29 13:01 testpayload.exe

┌──(root㉿kali)-[~]
└─#
```

ii. Use **ls -l** again to list the contents of the newly created second_drive directory.

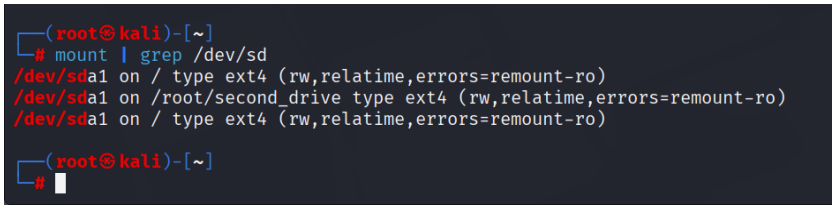
Note that the directory is empty.

iii. Use the `mount` command to attach `/dev/sda1` to the newly created `second_drive` directory.



No output is provided which means the mounting process was successful.

iv. Run the `mount` command again without options to view the detailed information for `/dev/sda1`. As before, pipe the output through `grep` to show only the `/dev/sd` entries.

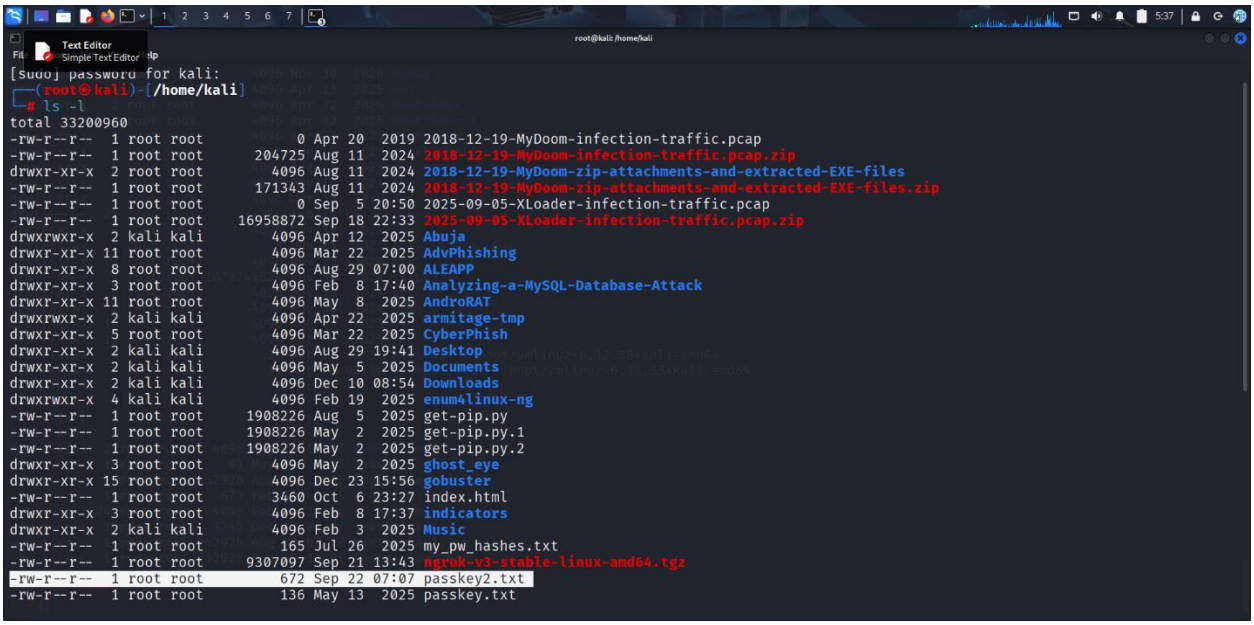


Part 2: File Permissions

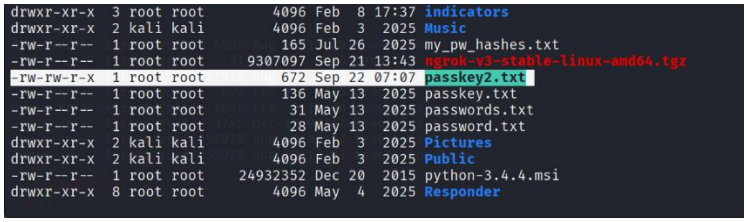
Linux filesystems include built-in permissions that control how users can view, modify, navigate, and execute files. Each file has its own set of permissions defining what actions are allowed for specific users and groups.

Step 1: Visualize and Change the File Permissions.

a. Use the ls -l command to display file permissions.



b. Use the **chmod** command to change the permissions of **passkey2.txt**.



Note that the permissione was changed from **-rw-r- -r- -** to **-rw-rw-r-x**

The **chmod** command takes permissions in the octal format. In that way, a breakdown of the 665 is as follows:

6 in octal is 110 in binary. Assuming each position of the permissions of a file can be 1 or 0, 110 means rw- (read=1, write=1 and execute=0).

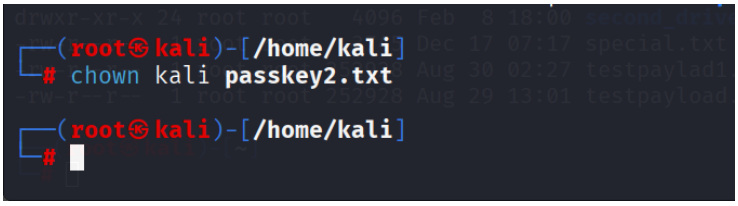
Therefore, the **chmod 665 passkey2.txt** command changes the permissions to:

Owner: rw- (6 in octal or 110 in binary)

Group: rw- (6 in octal or 110 in binary)

Other: r-x (5 in octal or 101 in binary)

c. The **chown** command is used to change ownership of a file or directory. Issue the command to make root the owner of the **passkey2.txt**



Note, the ownership is being changed from **root** to **kali**