Unit - 4

Performing Basic Linux Tasks : Identify the linux design philosophy, Enter shell commands, get help with linux

Managing Users and Groups : Assume superuser privileges, Create, modify and delete users, Create, modify and delete groups, Query users and groups, Configure account profiles

Managing Permissions and Ownership : Modify file and directory permissions, Modify file and directory ownership, Configure special permissions and attributes, Troubleshoot permissions issues

Managing Storage : Create partitions, Manage logical volumes, Mount file systems, Manage file systems, Navigate the linux directory structure, Troubleshoot storage issues

Identify the linux design philosophy

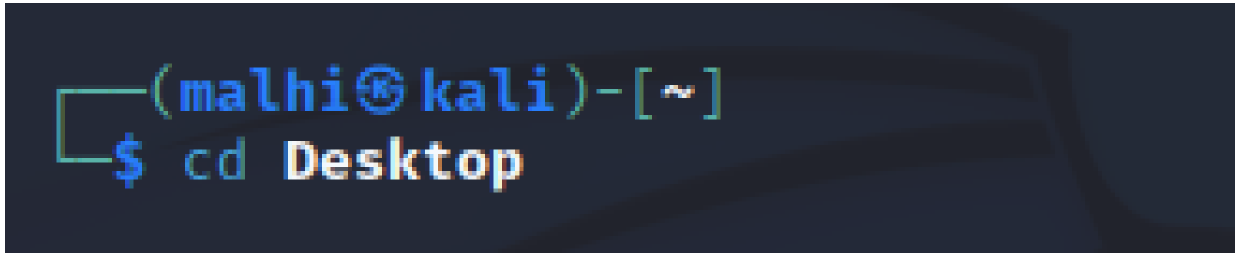
- The following quote from the section *Introduction* of the Unix philosophy of the book *Linux and the Unix Philosophy*, by Mike Gancarz, quite informative.
- *“An operating system, by its nature, embodies the philosophy of its creators. The creators of the Unix operating system started with a radical concept: they assumed that the user of their operating system would be computer literate from the start. The entire Unix philosophy revolves around the idea that the user knows what he or she is doing.”*

Enter shell commands

- A shell is a special user program that provides an interface to the user to use operating system services.
- Shell accepts human-readable commands from the user and converts them into something which the kernel can understand.
- It is a command language interpreter that executes commands read from input devices such as keyboards or from files.
- The shell gets started when the user logs in or starts the terminal.

cd

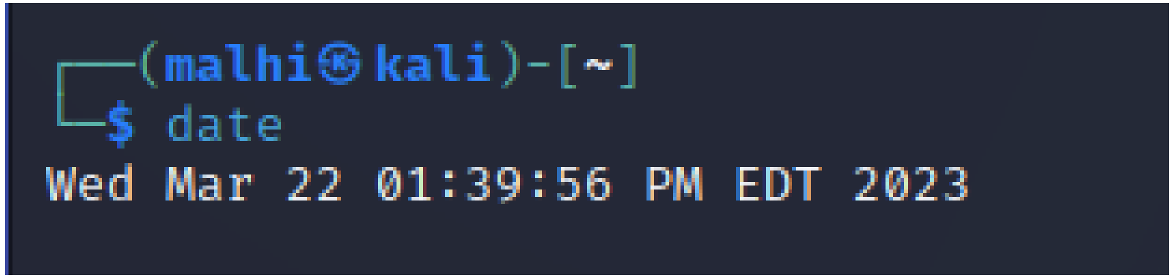
- Change directory



```
┌──(malhi㉿kali)-[~]
└─$ cd Desktop
```

date

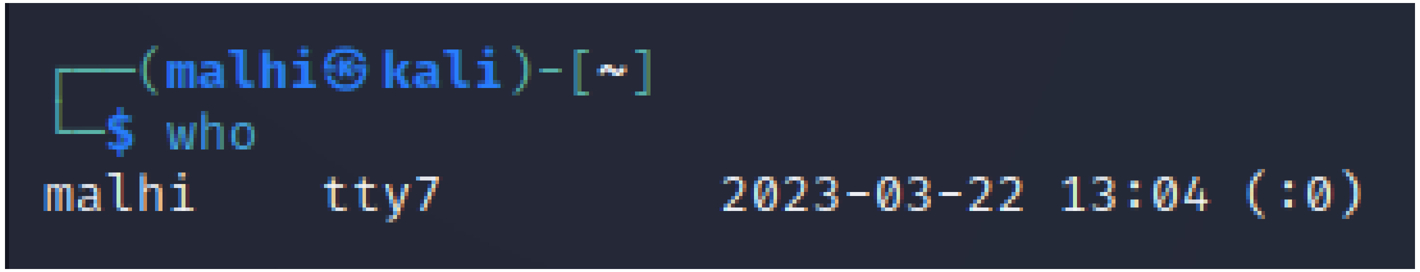
- Displays current day date time



```
┌──(malhi㉿kali)-[~]
└─$ date
Wed Mar 22 01:39:56 PM EDT 2023
```

who

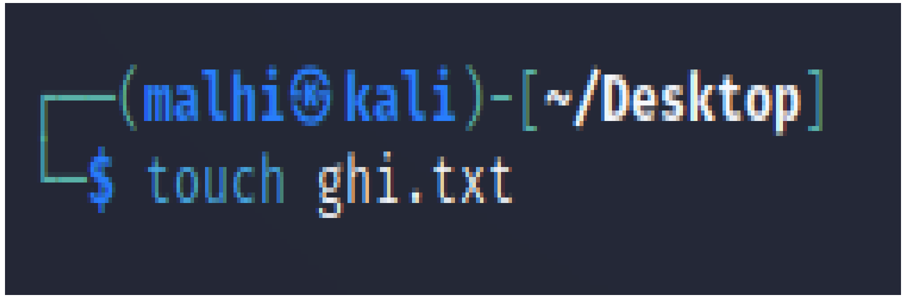
- Displays current login user



```
┌──(malhi㉿kali)-[~]
└─$ who
malhi    tty7         2023-03-22 13:04 (:0)
```

touch

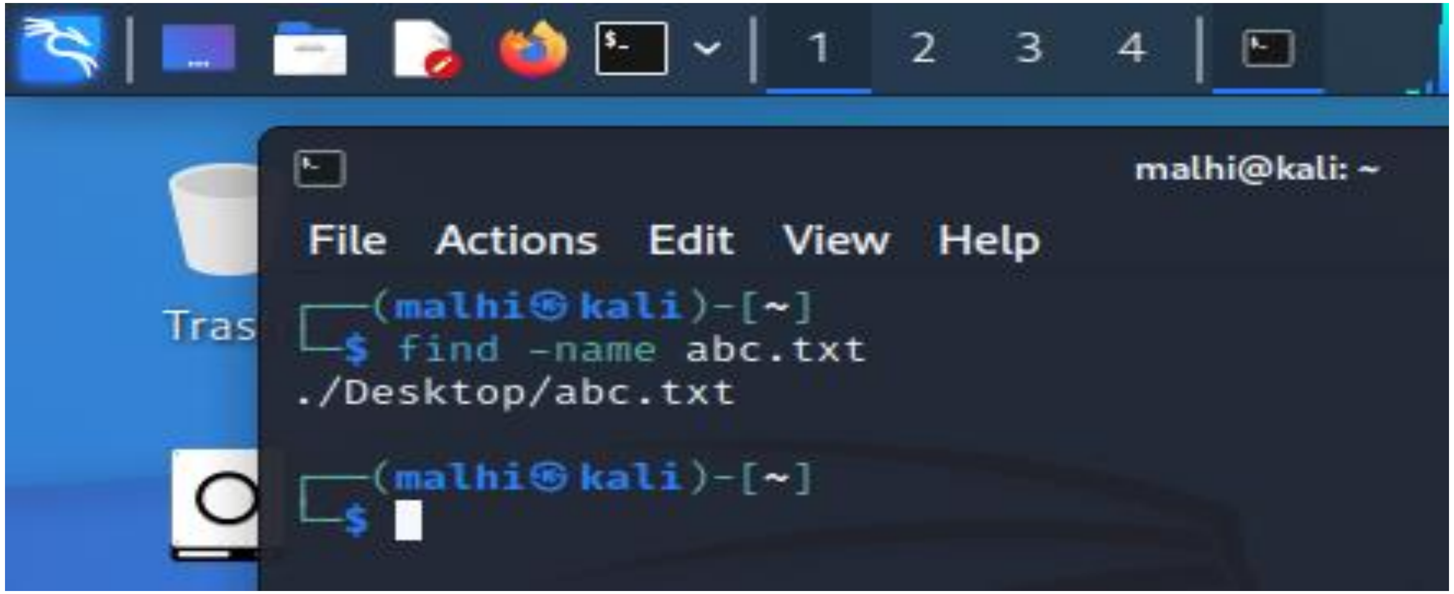
- To create any file



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ touch ghi.txt
```

find

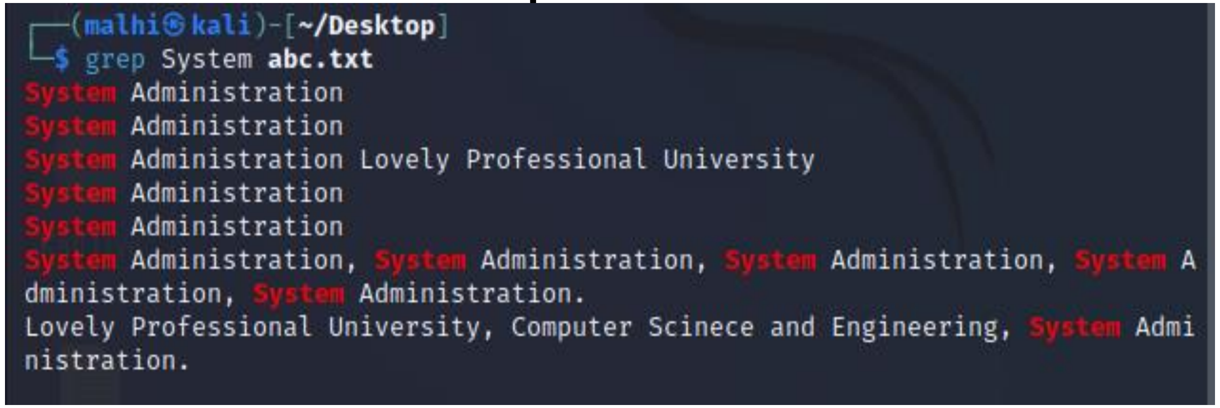
- Find where the specific file is situated



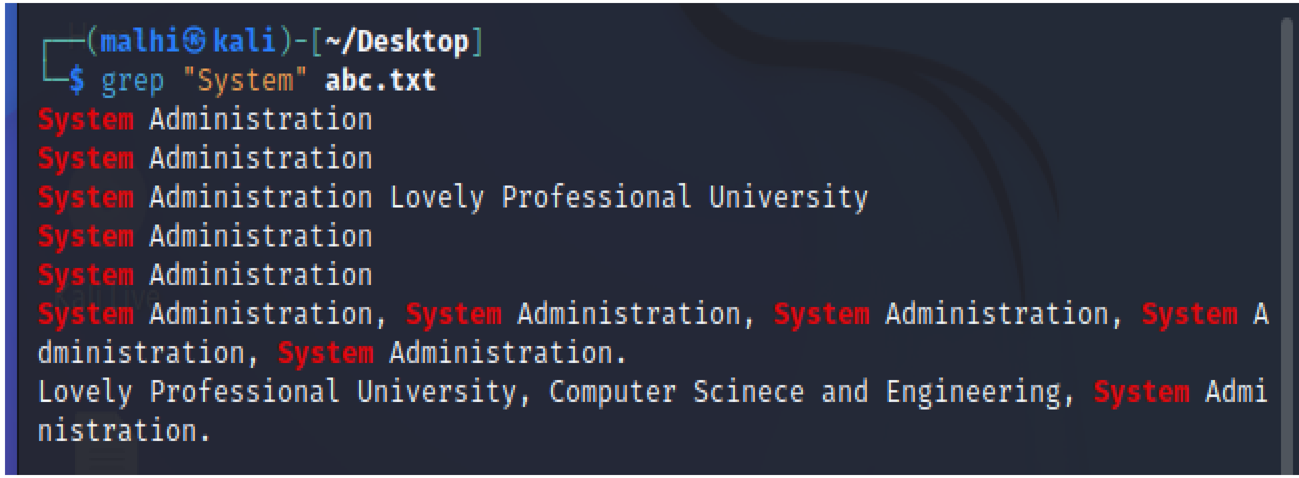
```
┌──(malhi㉿kali)-[~]
└─$ find -name abc.txt
./Desktop/abc.txt

┌──(malhi㉿kali)-[~]
└─$ 
```

grep

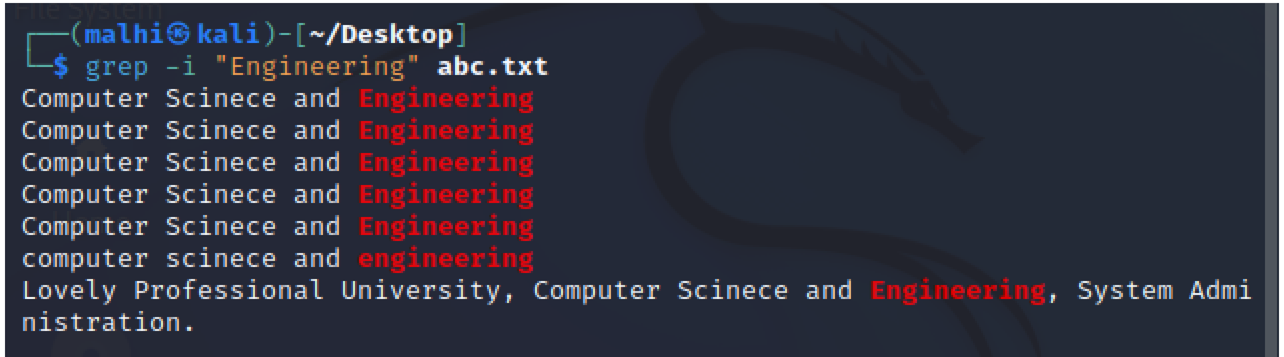
- **Grep** is an acronym that stands for **G**lobal **R**egular **E**xpression **P**rint.
- The grep command searches through the file, looking for matches to the pattern specified.
- Grep is case-sensitive.



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ grep System abc.txt
System Administration
System Administration
System Administration Lovely Professional University
System Administration
System Administration
System Administration, System Administration, System Administration, System A
dministration, System Administration.
Lovely Professional University, Computer Scinece and Engineering, System Admi
nistration.
```



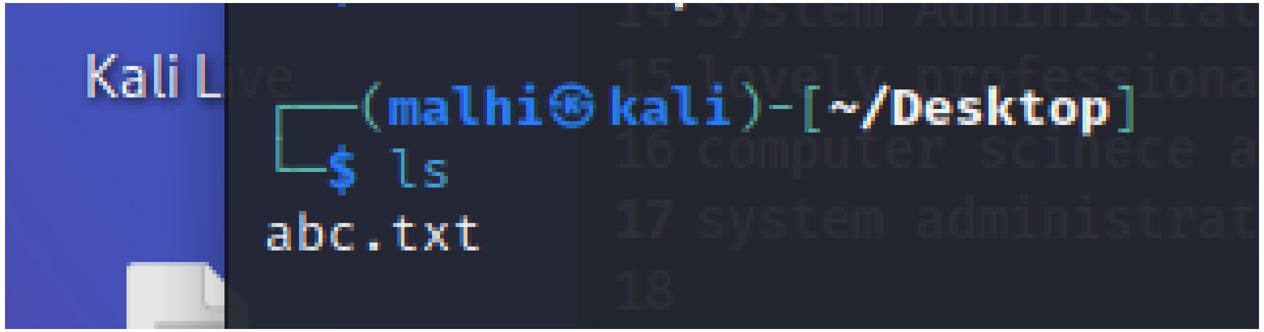
```
┌──(malhi㉿kali)-[~/Desktop]
└─$ grep "System" abc.txt
System Administration
System Administration
System Administration Lovely Professional University
System Administration
System Administration
System Administration, System Administration, System Administration, System A
dministration, System Administration.
Lovely Professional University, Computer Scinece and Engineering, System Admi
nistration.
```



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ grep -i "Engineering" abc.txt
Computer Scinece and Engineering
Computer Scinece and Engineering
Computer Scinece and Engineering
Computer Scinece and Engineering
Computer Scinece and Engineering
computer scinece and engineering
Lovely Professional University, Computer Scinece and Engineering, System Admi
nistration.
```

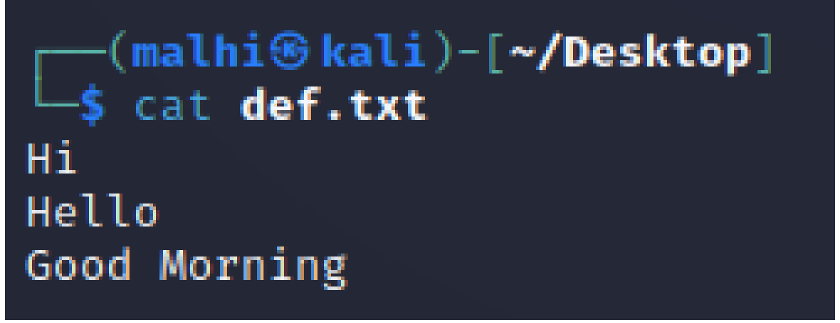
ls

- Displays files and folders present in the current directory



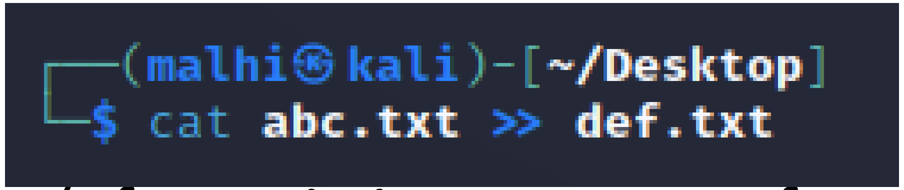
```
┌──(malhi㉿kali)-[~/Desktop]
└─$ ls
abc.txt
```

cat



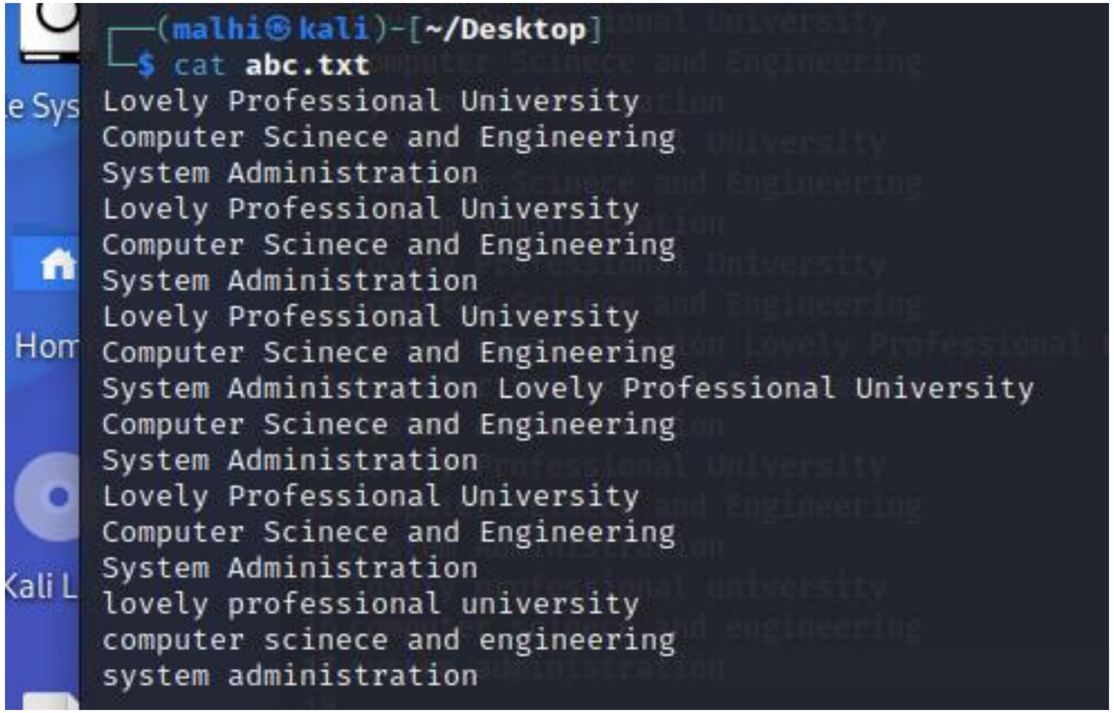
```
┌──(malhi㉿kali)-[~/Desktop]
└─$ cat def.txt
Hi
Hello
Good Morning
```

- Displays content of specific file

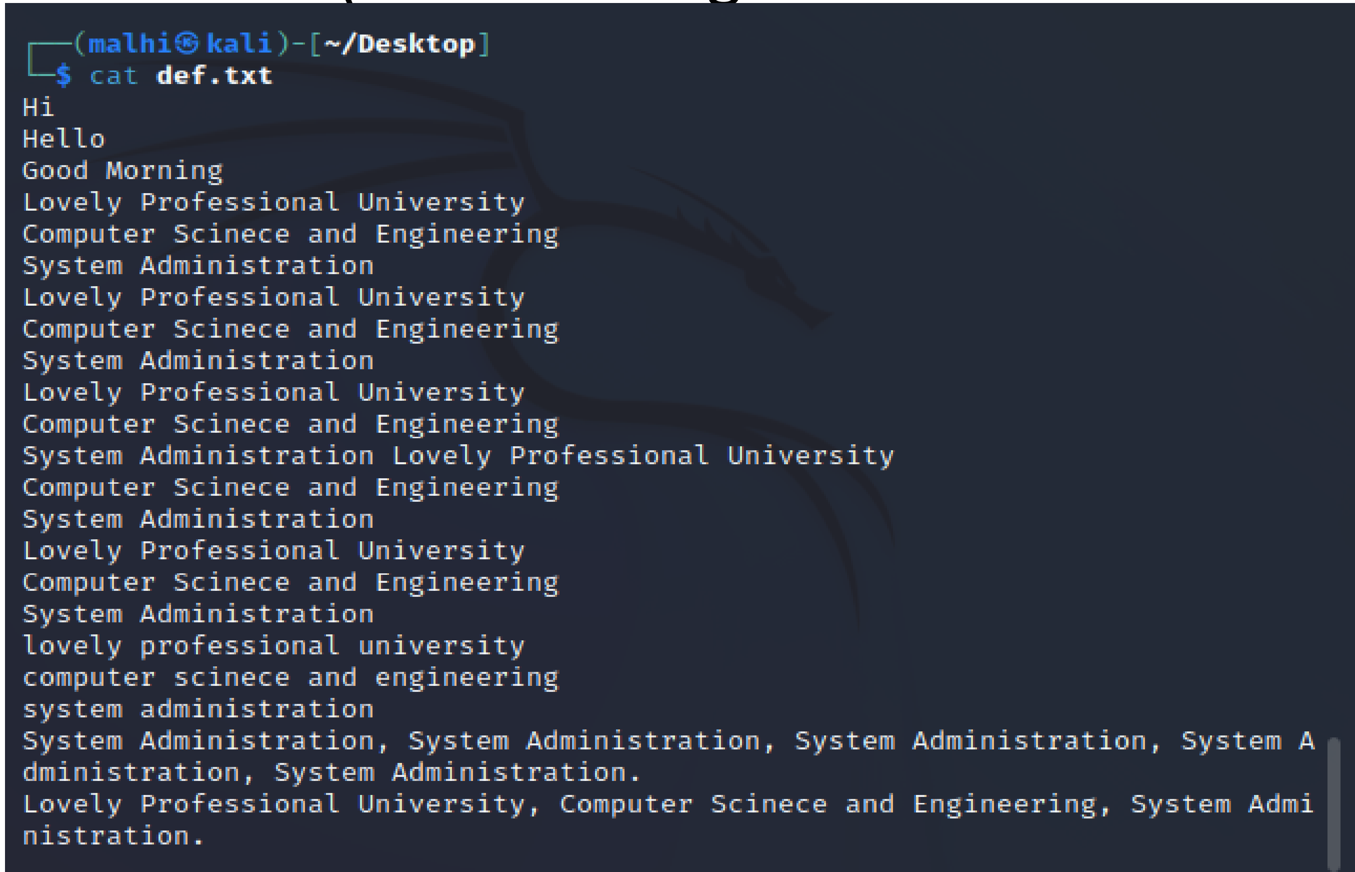


```
┌──(malhi㉿kali)-[~/Desktop]
└─$ cat abc.txt >> def.txt
```

- Copies content of one file into another file (after existing content of another file)



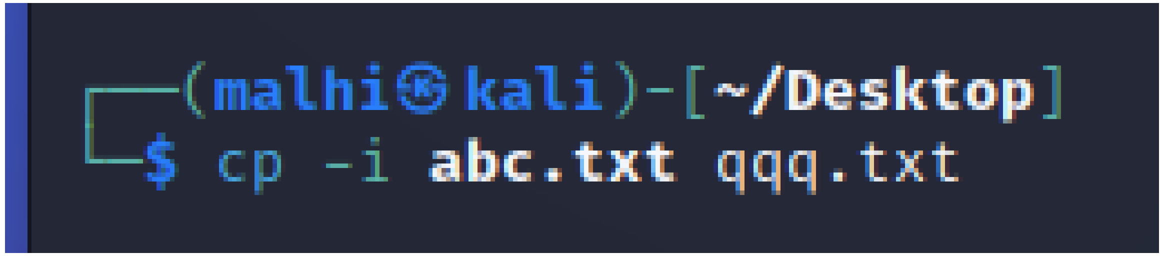
```
┌──(malhi㉿kali)-[~/Desktop]
└─$ cat abc.txt
Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration
lovely professional university
computer scinece and engineering
system administration
```



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ cat def.txt
Hi
Hello
Good Morning
Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration Lovely Professional University
Computer Scinece and Engineering
System Administration
Lovely Professional University
Computer Scinece and Engineering
System Administration
lovely professional university
computer scinece and engineering
system administration
System Administration, System Administration, System Administration, System A
dministration, System Administration.
Lovely Professional University, Computer Scinece and Engineering, System Admi
nistration.
```

cp

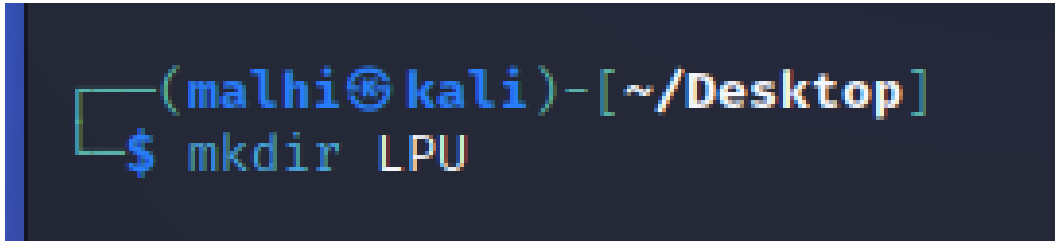
- Copies whole content of first file into another file (also creates another file if it is not available)



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ cp -i abc.txt qqq.txt
```

mkdir

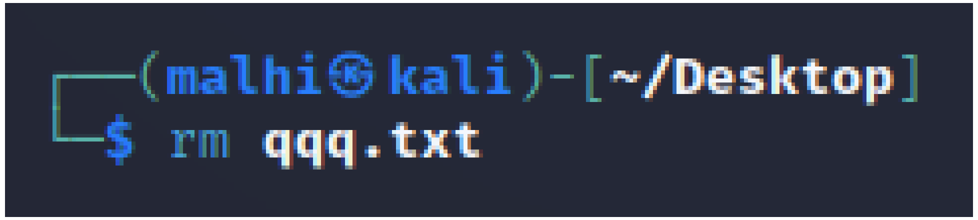
- It makes or creates the new directory



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ mkdir LPU
```

rm

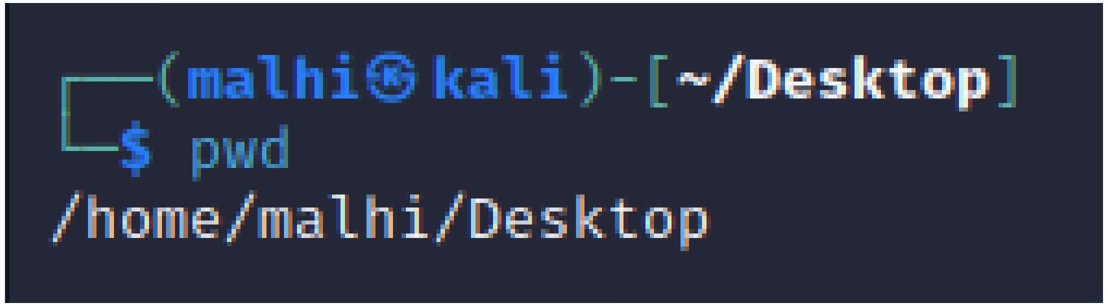
- It removes the specific file



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ rm qqq.txt
```

pwd (print working directory)

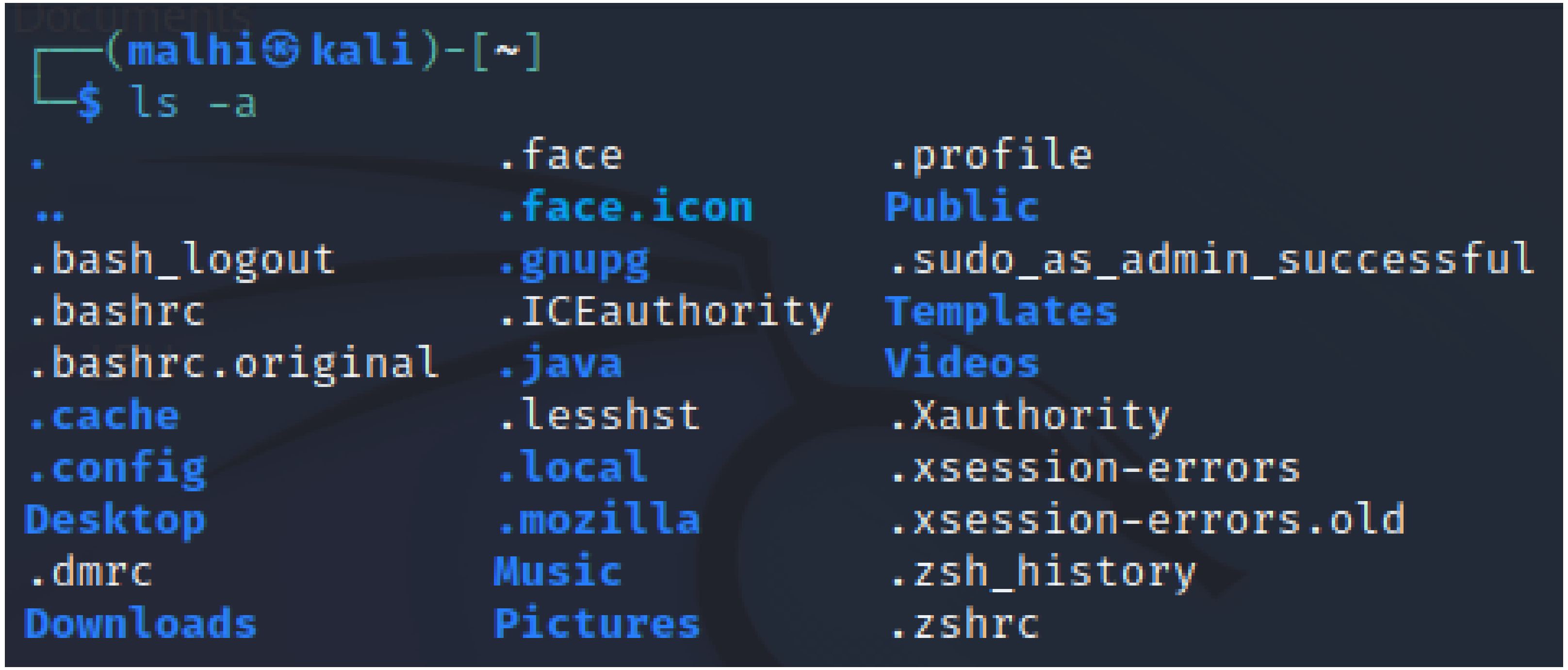
- It displays the path of working directory



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ pwd
/home/malhi/Desktop
```

ls -a

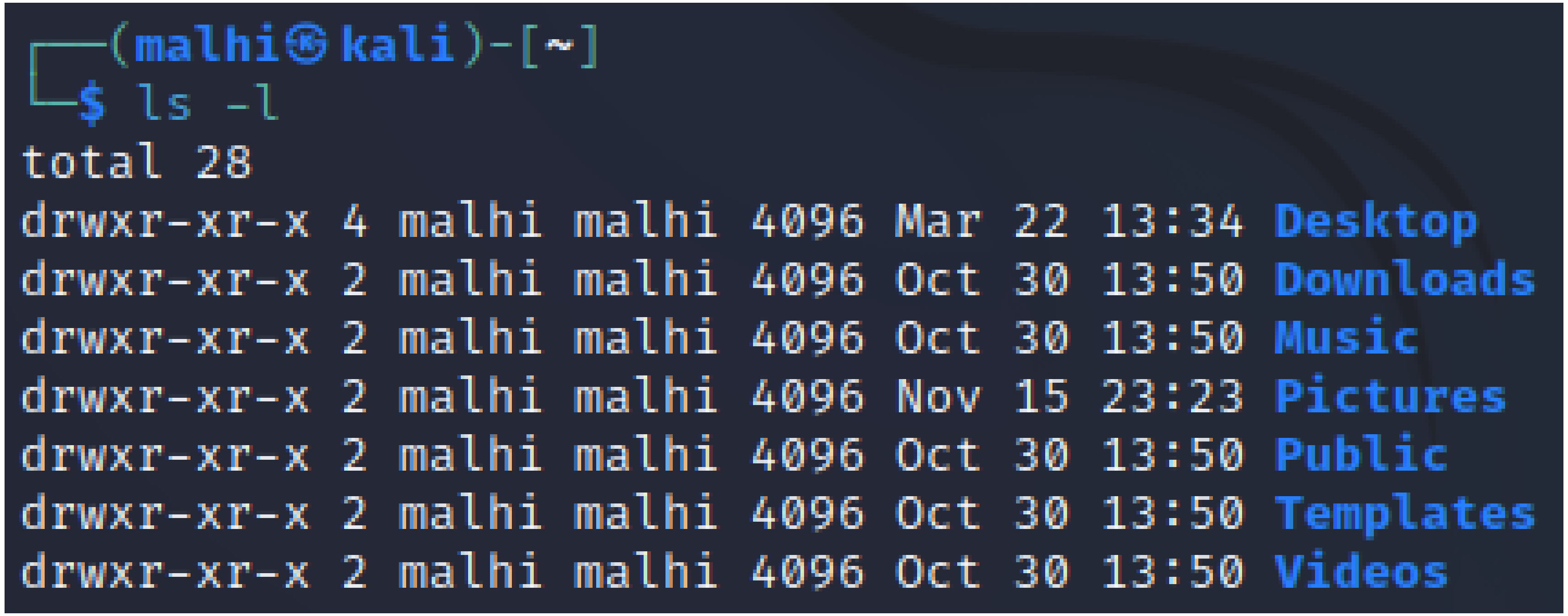
- It displays the hidden files and directories



```
┌──(malhi㉿kali)-[~]
└─$ ls -a
.                      .face             .profile
..                     .face.icon        Public
.bash_logout           .gnupg            .sudo_as_admin_successful
.bashrc                .ICEauthority     Templates
.bashrc.original       .java             Videos
.cache                 .lesshst          .Xauthority
.config                .local            .xsession-errors
Desktop                .mozilla          .xsession-errors.old
.dmrc                  Music             .zsh_history
Downloads              Pictures          .zshrc
```

ls -l

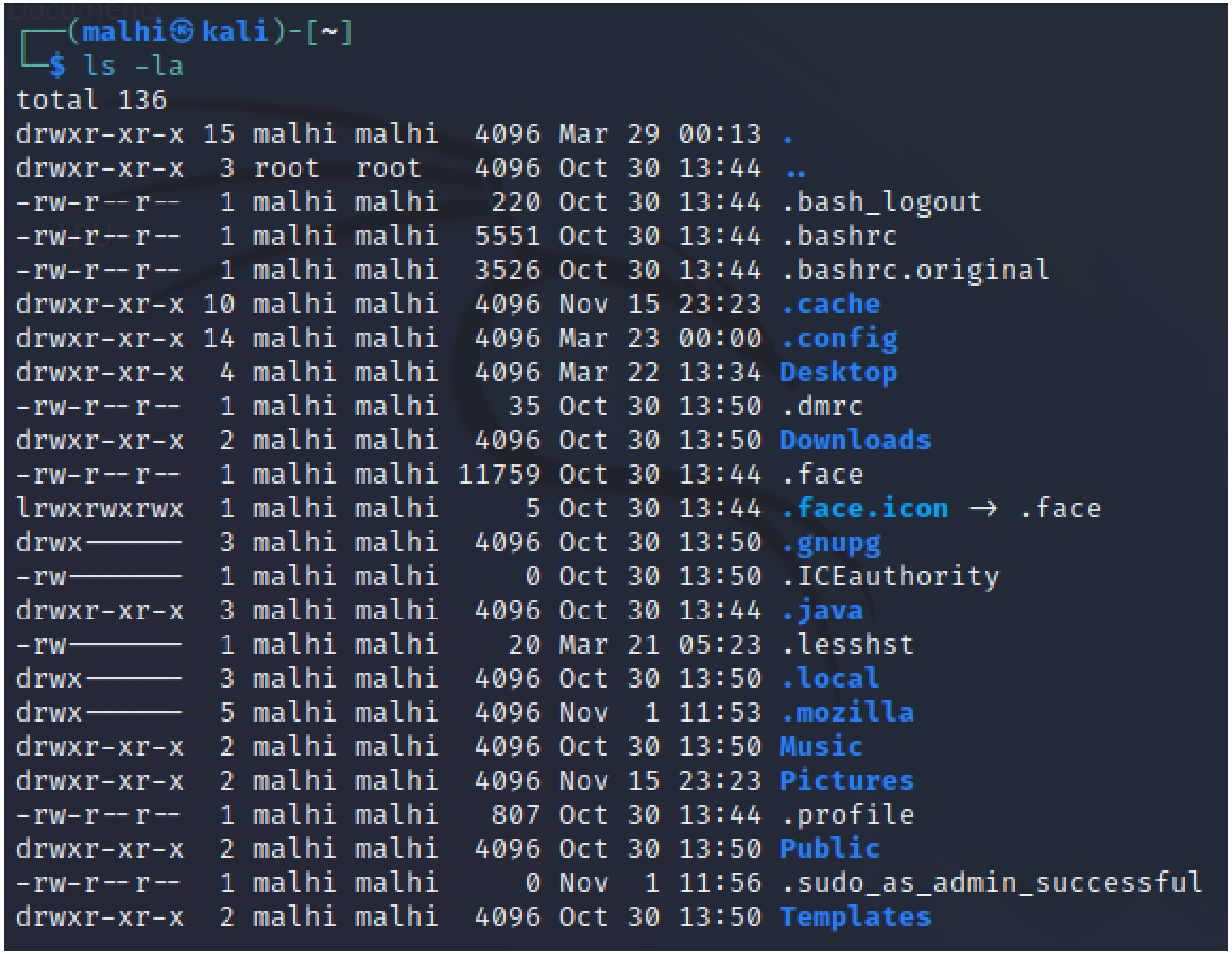
- It displays files and directories with permissions



```
┌──(malhi㉿kali)-[~]
└─$ ls -l
total 28
drwxr-xr-x 4 malhi malhi 4096 Mar 22 13:34 Desktop
drwxr-xr-x 2 malhi malhi 4096 Oct 30 13:50 Downloads
drwxr-xr-x 2 malhi malhi 4096 Oct 30 13:50 Music
drwxr-xr-x 2 malhi malhi 4096 Nov 15 23:23 Pictures
drwxr-xr-x 2 malhi malhi 4096 Oct 30 13:50 Public
drwxr-xr-x 2 malhi malhi 4096 Oct 30 13:50 Templates
drwxr-xr-x 2 malhi malhi 4096 Oct 30 13:50 Videos
```

ls -la

- It displays the hidden files and directories with permissions

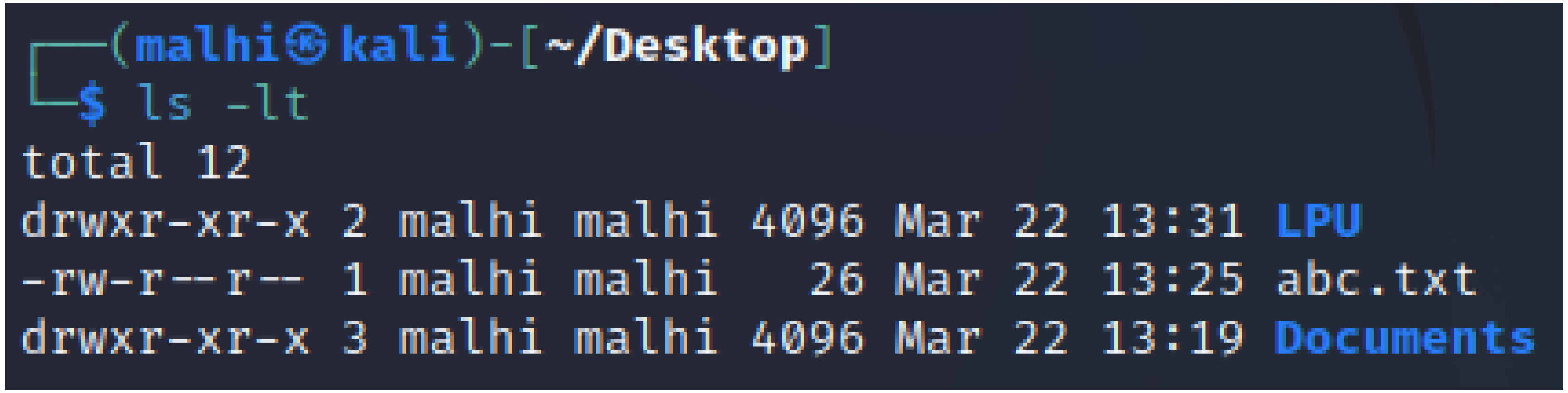


```
┌──(malhi㉿kali)-[~]
└─$ ls -la
total 136
drwxr-xr-x 15 malhi malhi  4096 Mar 29 00:13 .
drwxr-xr-x  3 root  root   4096 Oct 30 13:44 ..
-rw-r--r--  1 malhi malhi   220 Oct 30 13:44 .bash_logout
-rw-r--r--  1 malhi malhi  5551 Oct 30 13:44 .bashrc
-rw-r--r--  1 malhi malhi  3526 Oct 30 13:44 .bashrc.original
drwxr-xr-x 10 malhi malhi  4096 Nov 15 23:23 .cache
drwxr-xr-x 14 malhi malhi  4096 Mar 23 00:00 .config
drwxr-xr-x  4 malhi malhi  4096 Mar 22 13:34 Desktop
-rw-r--r--  1 malhi malhi    35 Oct 30 13:50 .dmrc
drwxr-xr-x  2 malhi malhi  4096 Oct 30 13:50 Downloads
-rw-r--r--  1 malhi malhi 11759 Oct 30 13:44 .face
lrwxrwxrwx  1 malhi malhi     5 Oct 30 13:44 .face.icon -> .face
drwx------  3 malhi malhi  4096 Oct 30 13:50 .gnupg
-rw-------  1 malhi malhi     0 Oct 30 13:50 .ICEauthority
drwxr-xr-x  3 malhi malhi  4096 Oct 30 13:44 .java
-rw-------  1 malhi malhi    20 Mar 21 05:23 .lesshst
drwx------  3 malhi malhi  4096 Oct 30 13:50 .local
drwx------  5 malhi malhi  4096 Nov  1 11:53 .mozilla
drwxr-xr-x  2 malhi malhi  4096 Oct 30 13:50 Music
drwxr-xr-x  2 malhi malhi  4096 Nov 15 23:23 Pictures
-rw-r--r--  1 malhi malhi   807 Oct 30 13:44 .profile
drwxr-xr-x  2 malhi malhi  4096 Oct 30 13:50 Public
-rw-r--r--  1 malhi malhi     0 Nov  1 11:56 .sudo_as_admin_successful
drwxr-xr-x  2 malhi malhi  4096 Oct 30 13:50 Templates
```

- The “ lrwxrwxrwx ” **shows that the file is the symbolic link to another file**, and it allows all the permissions, read, writes, and execution to all its users and groups.

ls -lt

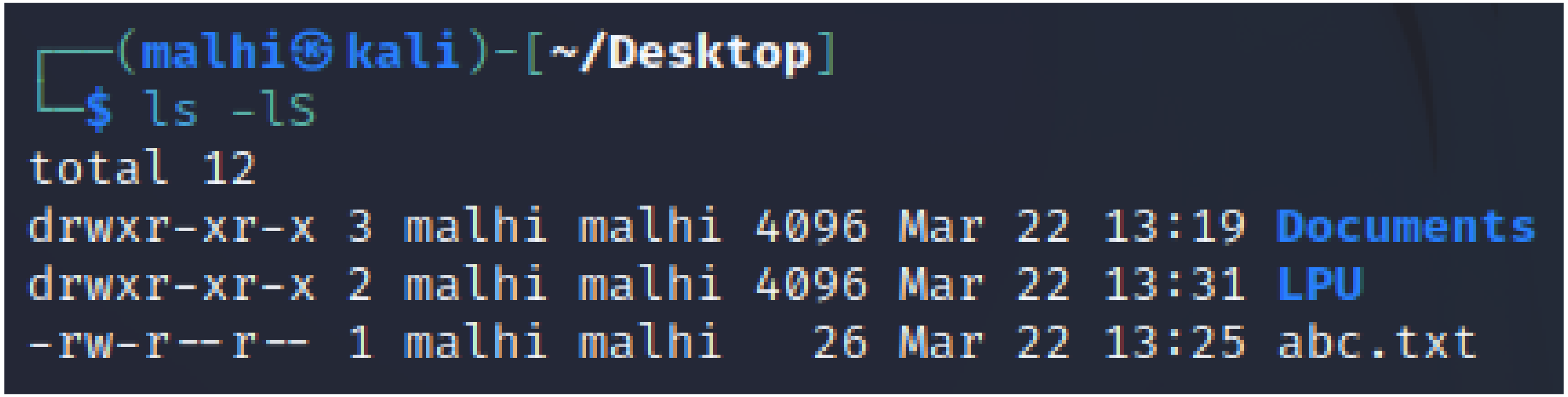
- It displays the files according to their modified date. The files under the specific directory will be listed according to their **modified date**. And **last modified** file will be at the top.



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ ls -lt
total 12
drwxr-xr-x 2 malhi malhi 4096 Mar 22 13:31 LPU
-rw-r--r-- 1 malhi malhi   26 Mar 22 13:25 abc.txt
drwxr-xr-x 3 malhi malhi 4096 Mar 22 13:19 Documents
```

ls -lS

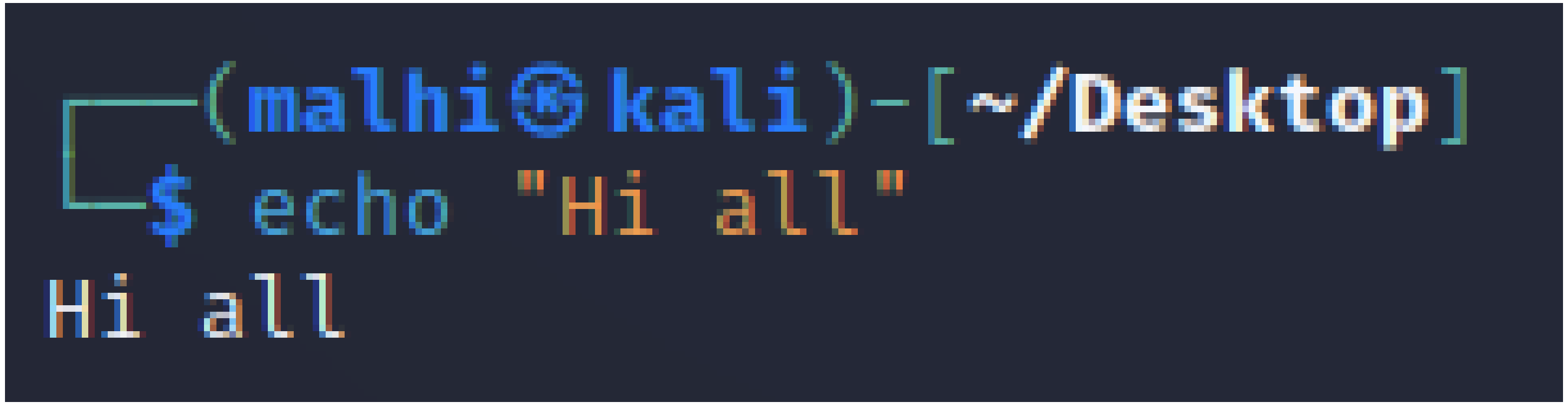
- It displays content of specific directory in a descending order that is largest file at the top.



```
┌──(malhi㉿kali)-[~/Desktop]
└─$ ls -lS
total 12
drwxr-xr-x 3 malhi malhi 4096 Mar 22 13:19 Documents
drwxr-xr-x 2 malhi malhi 4096 Mar 22 13:31 LPU
-rw-r--r-- 1 malhi malhi   26 Mar 22 13:25 abc.txt
```

echo

- It display the same message on the screen

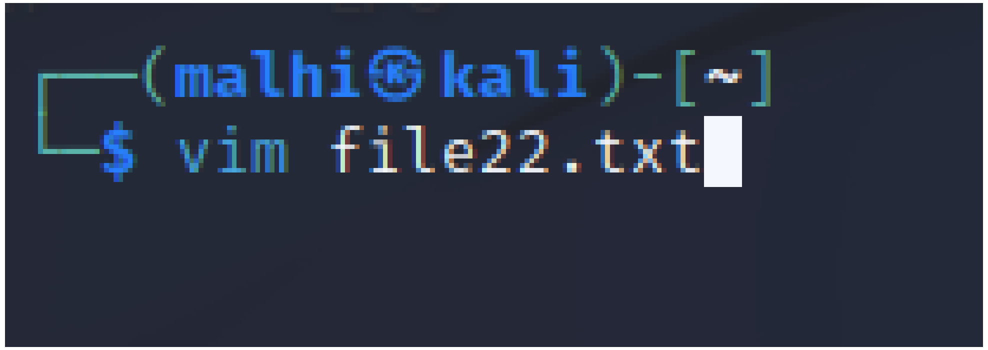


```
┌──(malhi㉿kali)-[~/Desktop]
└─$ echo "Hi all"
Hi all
```

File editing commands:

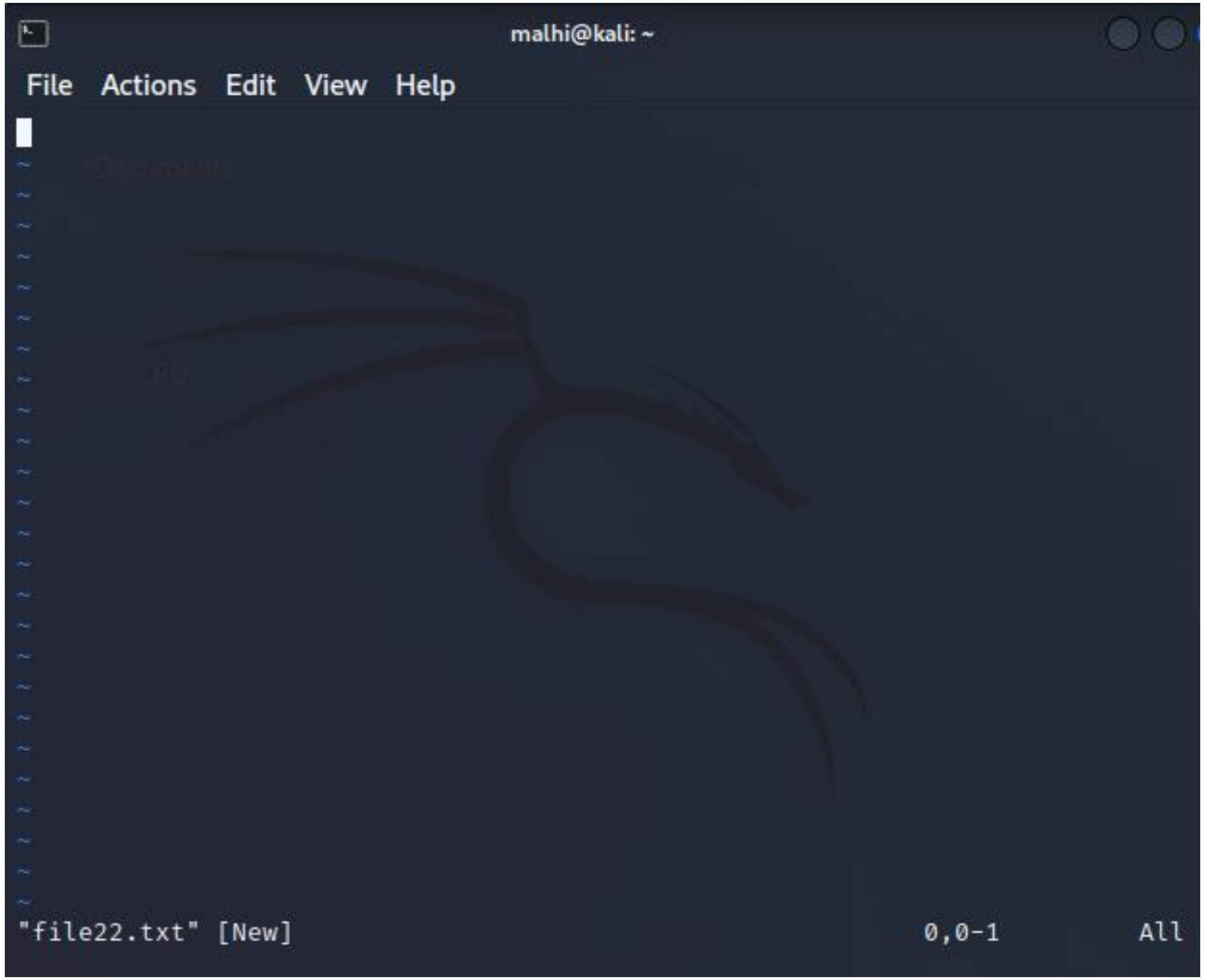
- To install vim on Debian based Linux like ubuntu run the command:
  - `sudo apt-get install vim`

vim

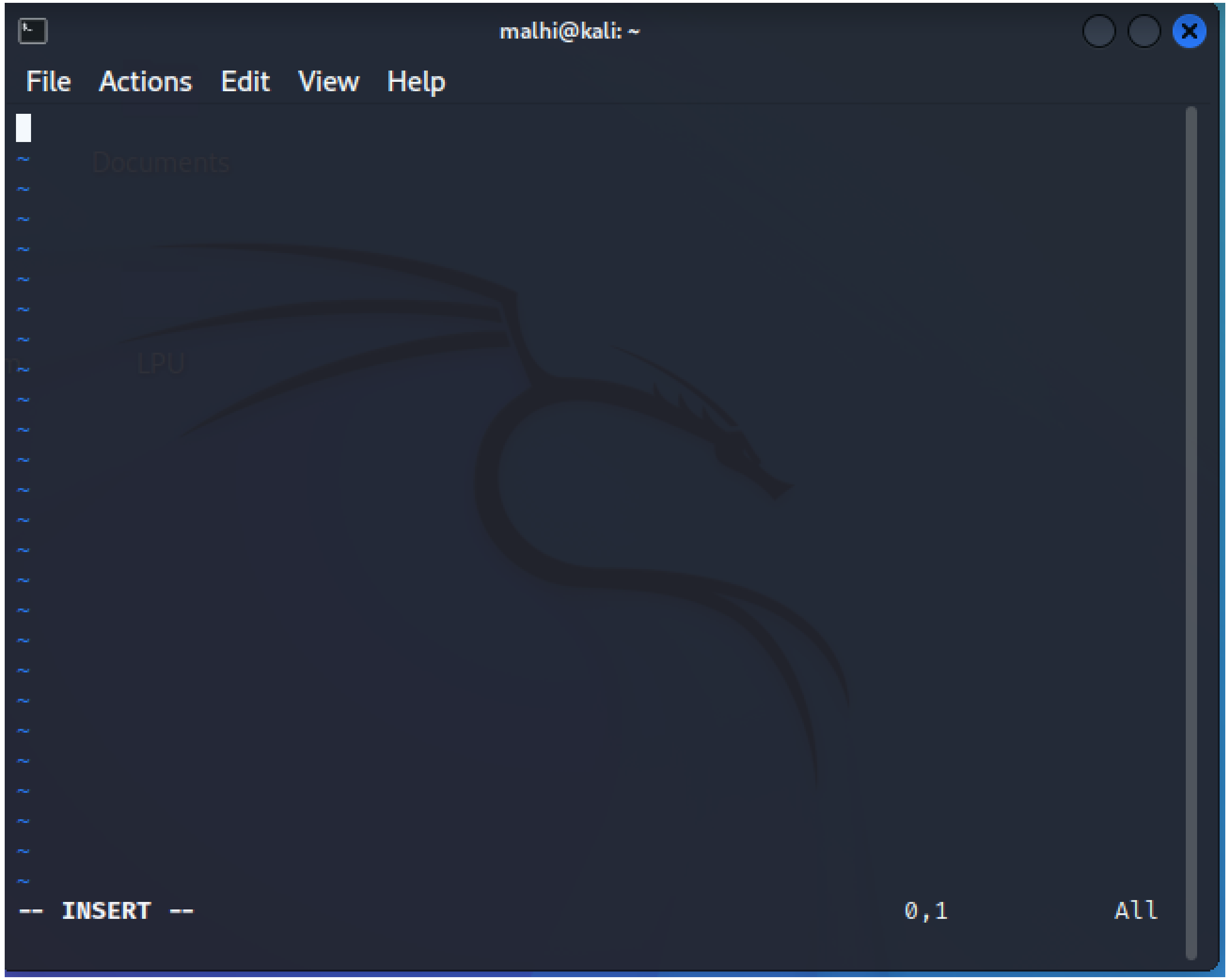


```
┌──(malhi㉿kali)-[~]
└─$ vim file22.txt
```

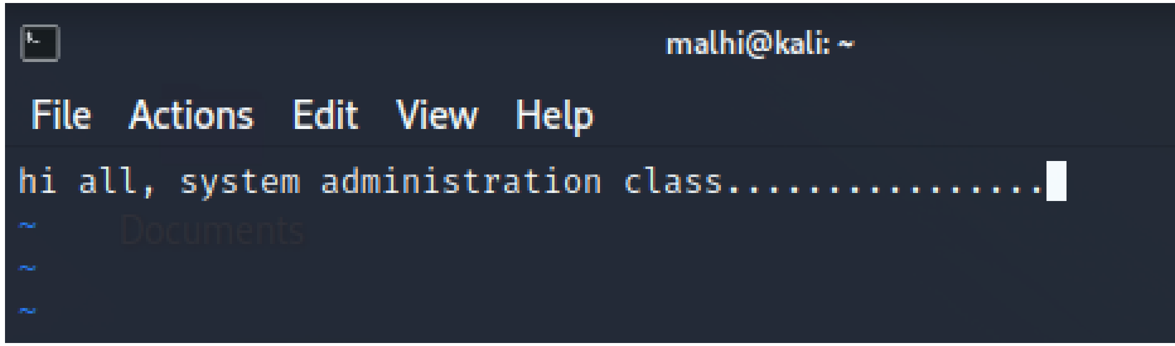
- Press i to enter into insert mode
- Then write your content
- After writing, exit from insert mode by pressing escape key
- To save write :wq



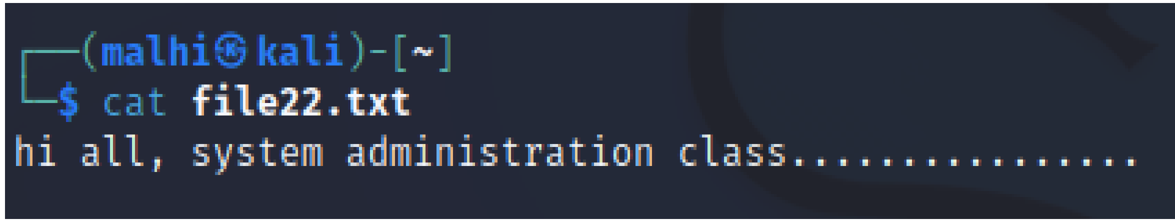
```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"file22.txt" [New]                                    0,0-1         All
```



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                    0,1          All
```



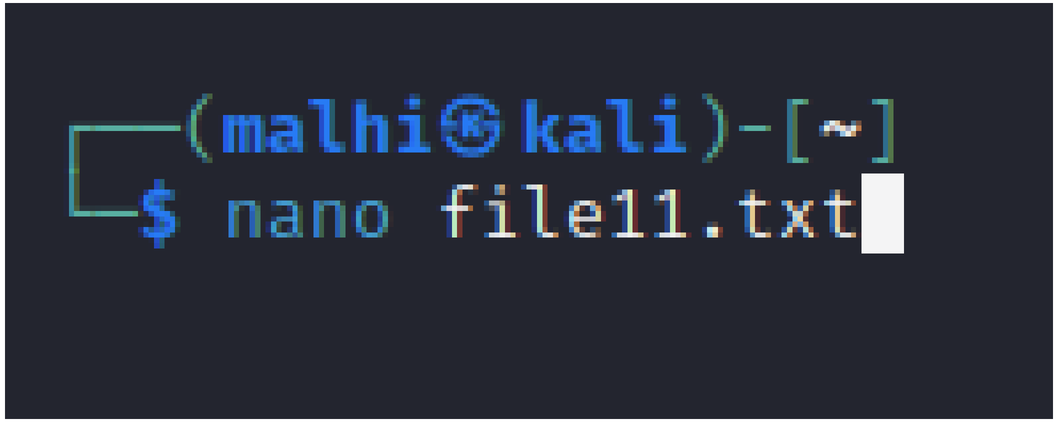
```
hi all, system administration class................
~
~
~
```



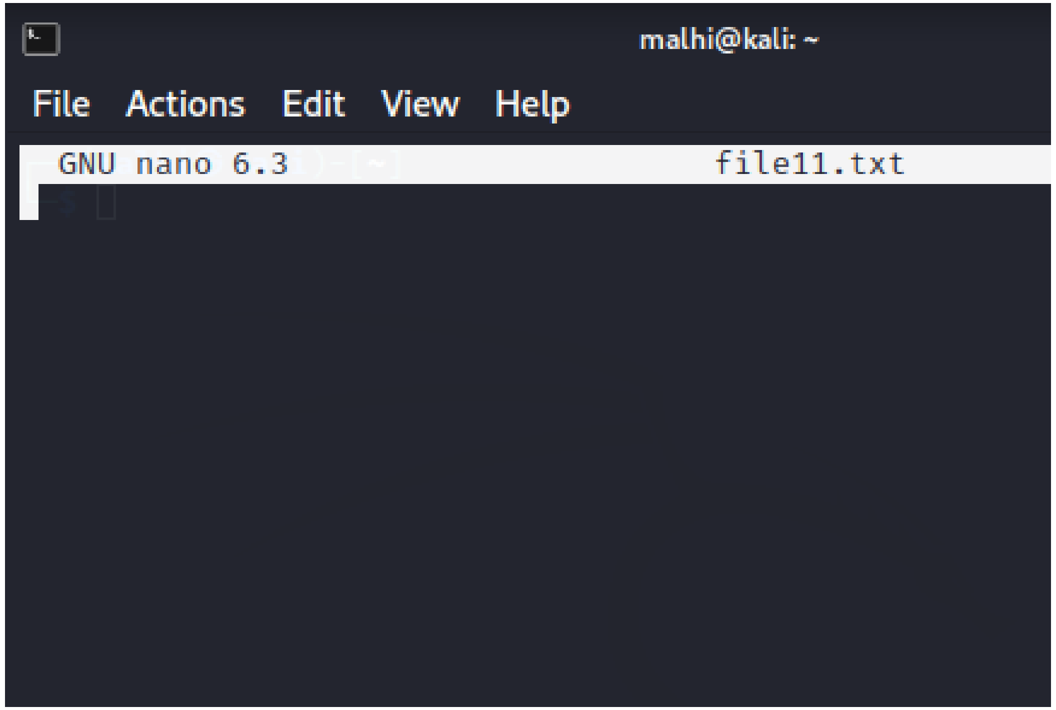
```
┌──(malhi㉿kali)-[~]
└─$ cat file22.txt
hi all, system administration class................
```

nano

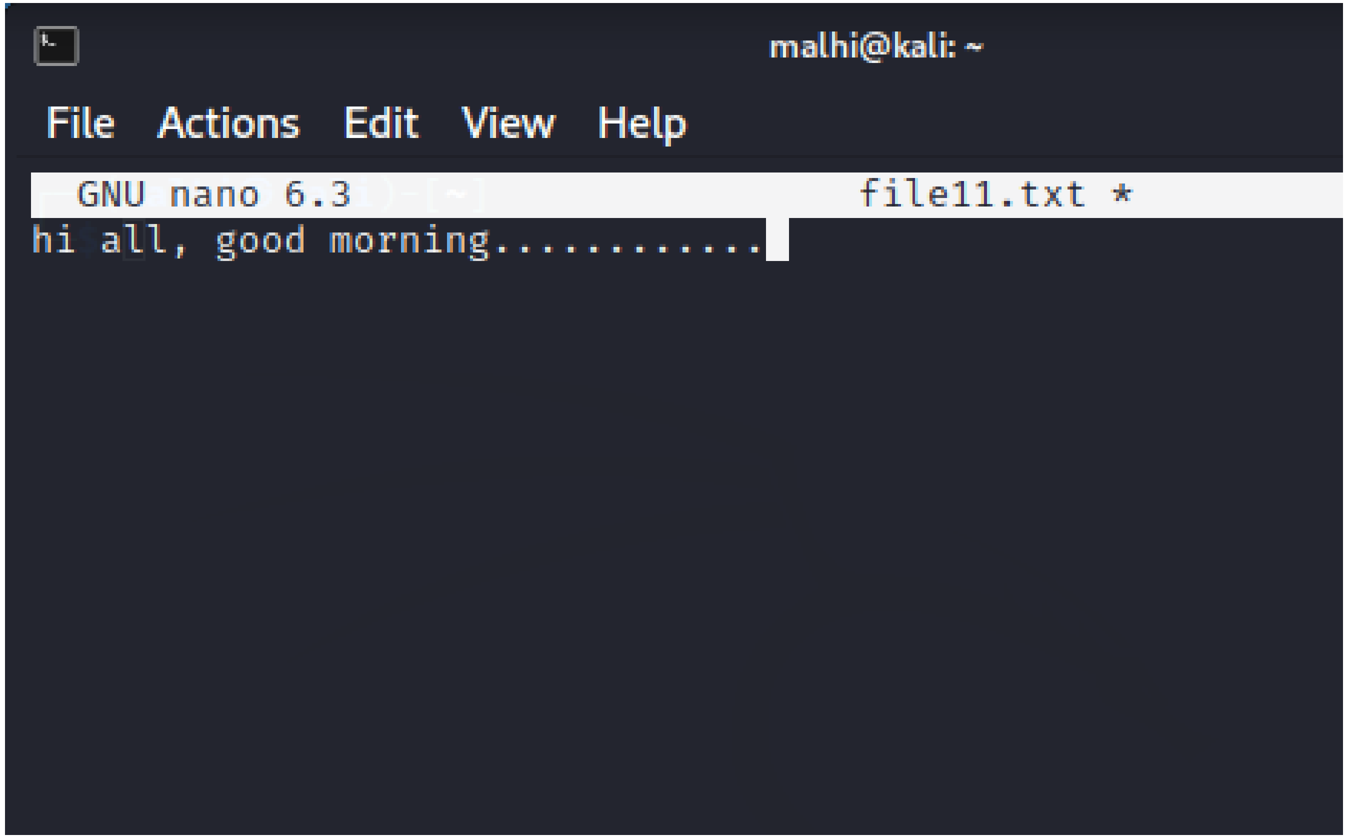
- nano



```
┌──(malhi㉿kali)-[~]
└─$ nano file11.txt
```

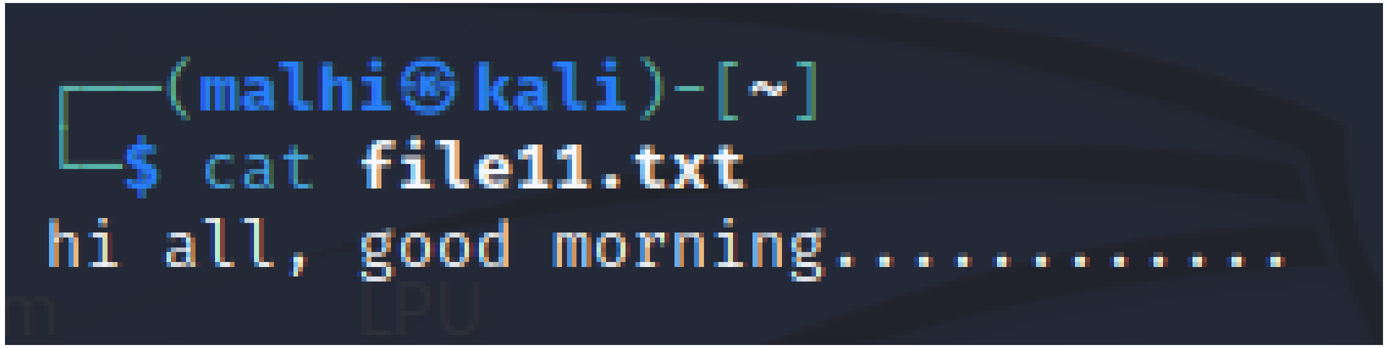


```
  GNU nano 6.3                              file11.txt
```



```
  GNU nano 6.3                          file11.txt *
hi all, good morning...........
```

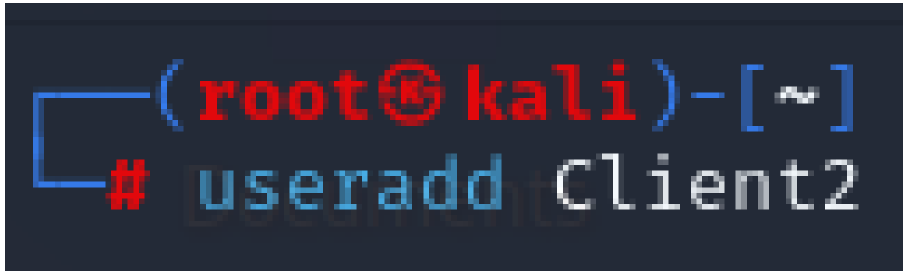
- Ctrl + O to save
- Ctrl + C to cancel the current work
- Ctrl + X to exit
- Then press Y



```
┌──(malhi㉿kali)-[~]
└─$ cat file11.txt
hi all, good morning............
```

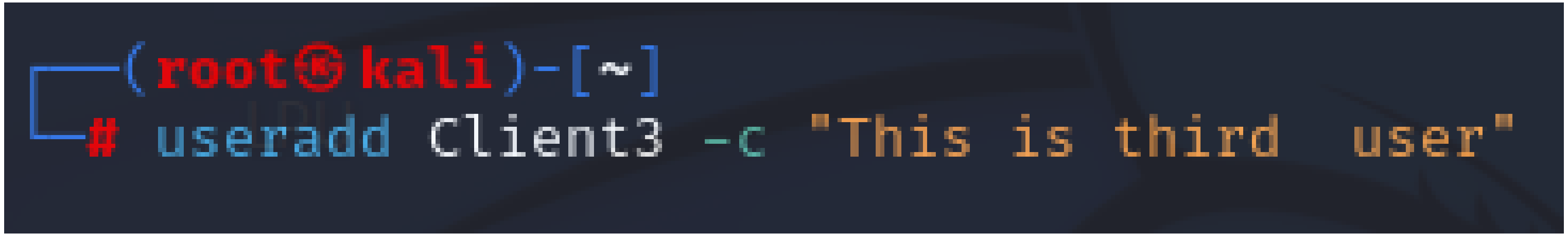
Users and groups

To create an user



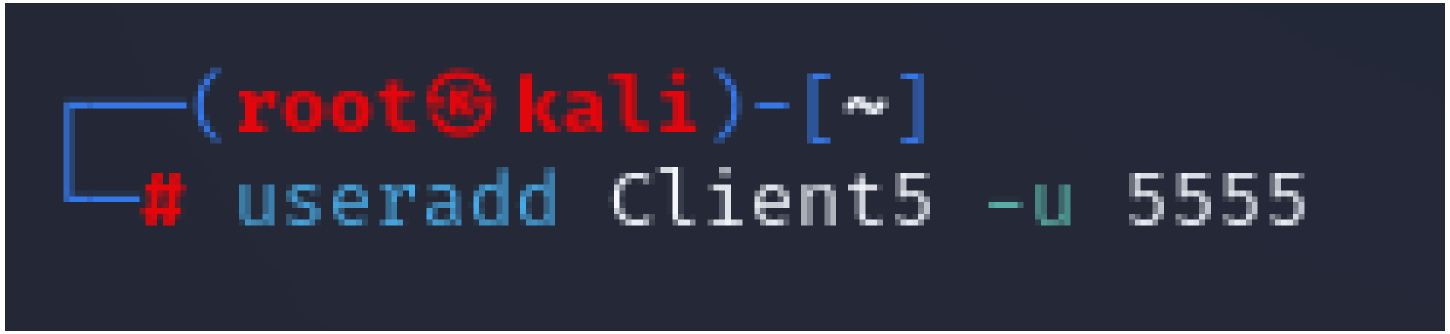
```
┌──(root㉿kali)-[~]
└─# useradd Client2
```

- User creation with comment



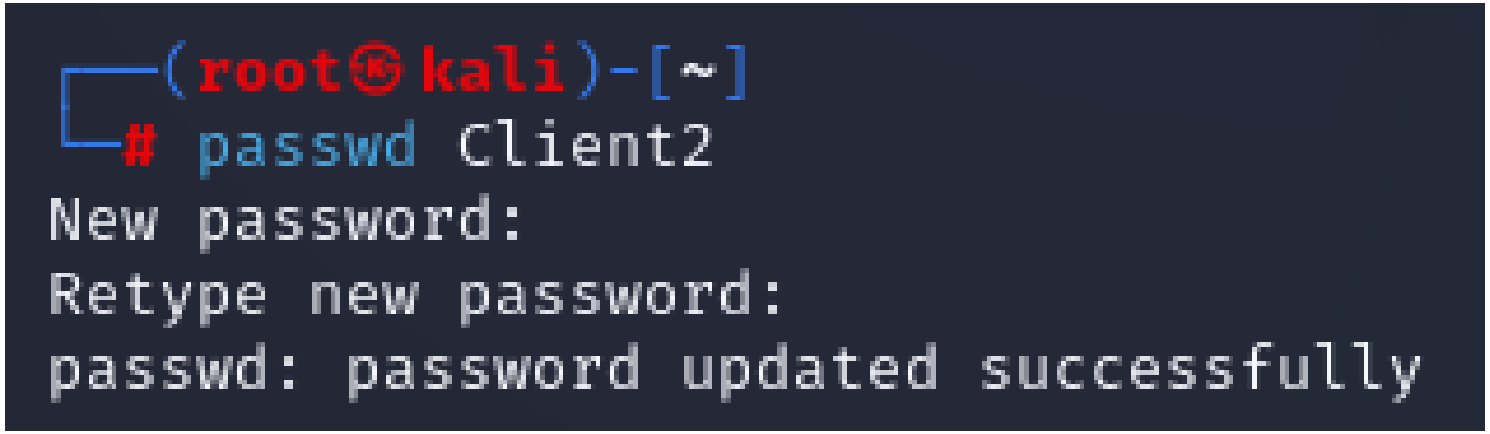
```
┌──(root㉿kali)-[~]
└─# useradd Client3 -c "This is third  user"
```

- User creation with user id



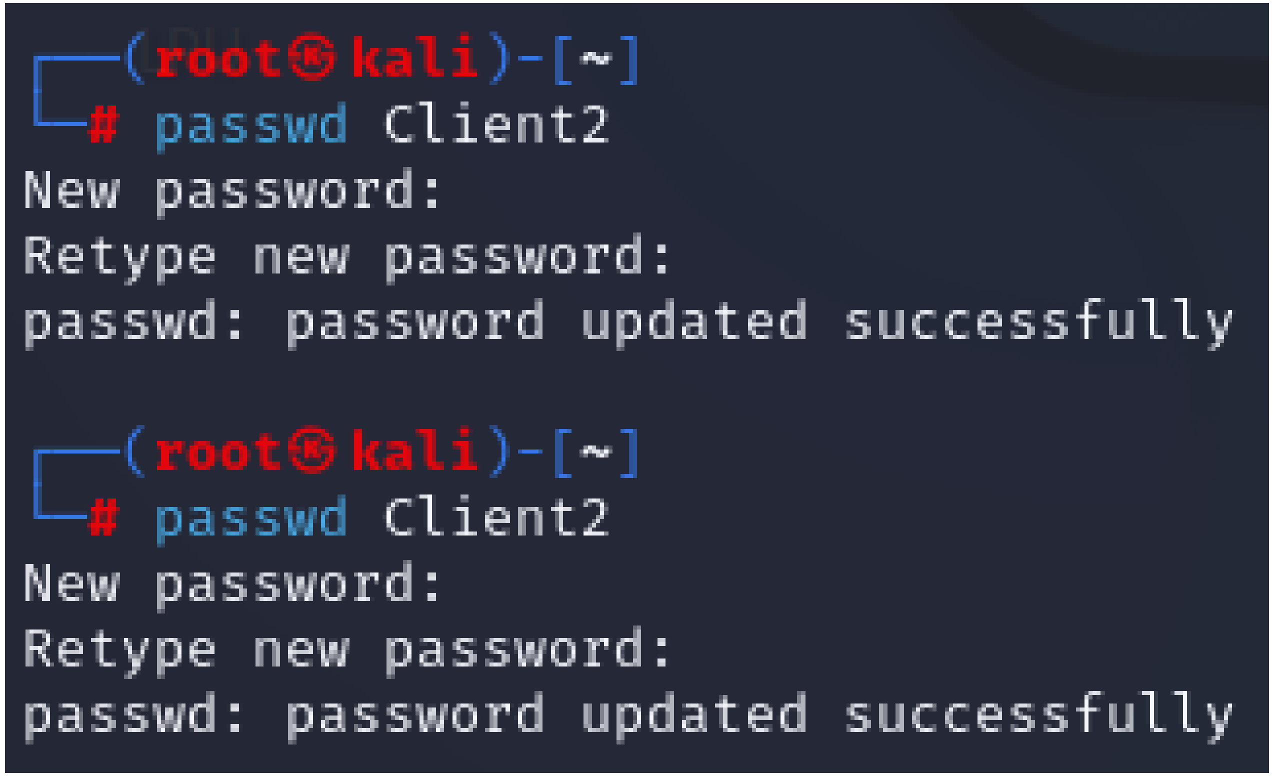
```
┌──(root㉿kali)-[~]
└─# useradd Client5 -u 5555
```

- To set the password of an user



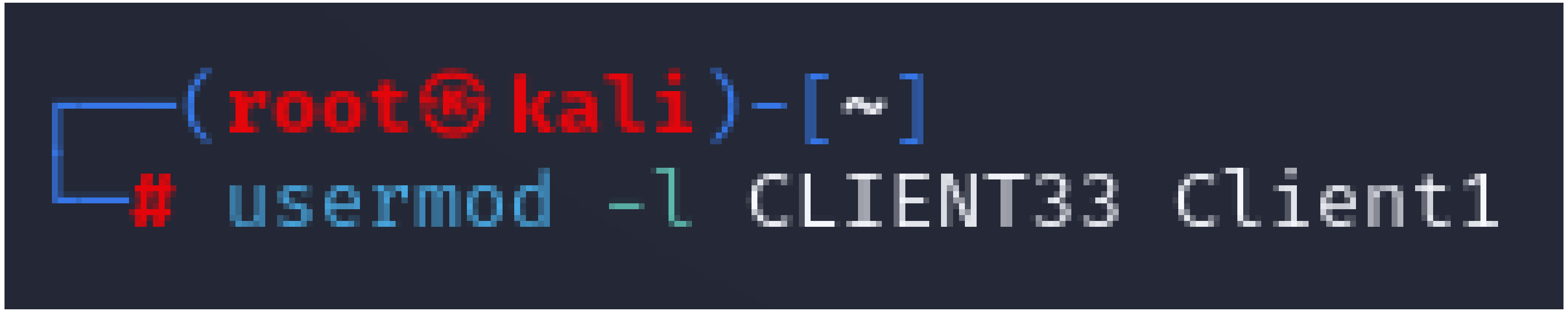
```
┌──(root㉿kali)-[~]
└─# passwd Client2
New password:
Retype new password:
passwd: password updated successfully
```

- To reset the password of an user

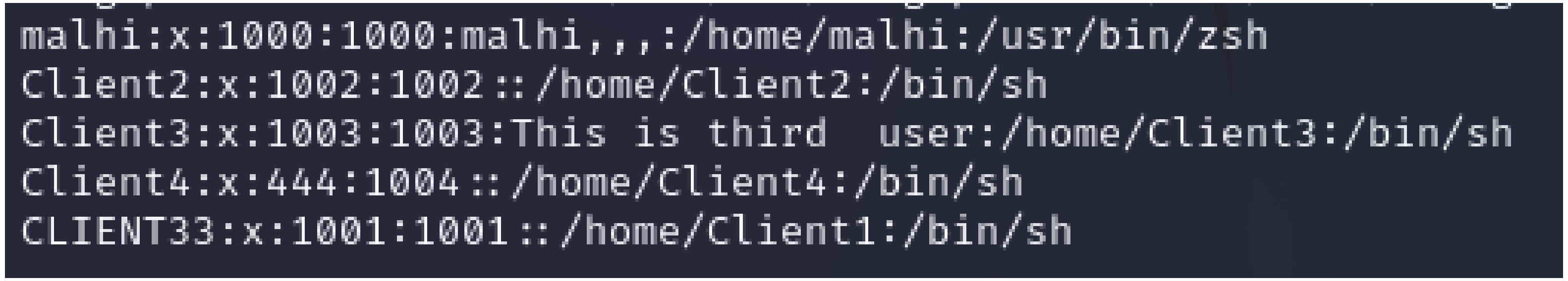


```
┌──(root㉿kali)-[~]
└─# passwd Client2
New password:
Retype new password:
passwd: password updated successfully

┌──(root㉿kali)-[~]
└─# passwd Client2
New password:
Retype new password:
passwd: password updated successfully
```

- To change the username

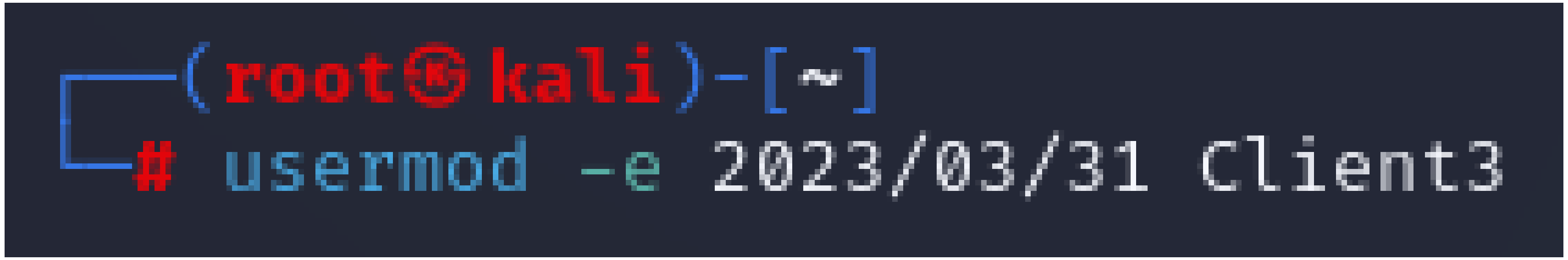


```
┌──(root㉿kali)-[~]
└─# usermod -l CLIENT33 Client1
```



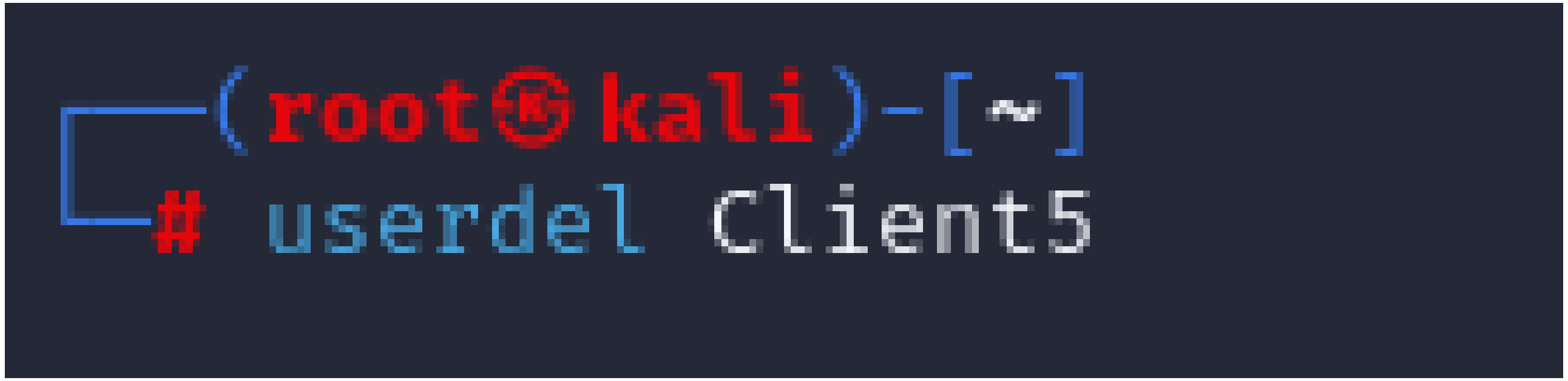
```
malhi:x:1000:1000:malhi,,,:/home/malhi:/usr/bin/zsh
Client2:x:1002:1002::/home/Client2:/bin/sh
Client3:x:1003:1003:This is third  user:/home/Client3:/bin/sh
Client4:x:444:1004::/home/Client4:/bin/sh
CLIENT33:x:1001:1001::/home/Client1:/bin/sh
```

- To set the user account expiration date



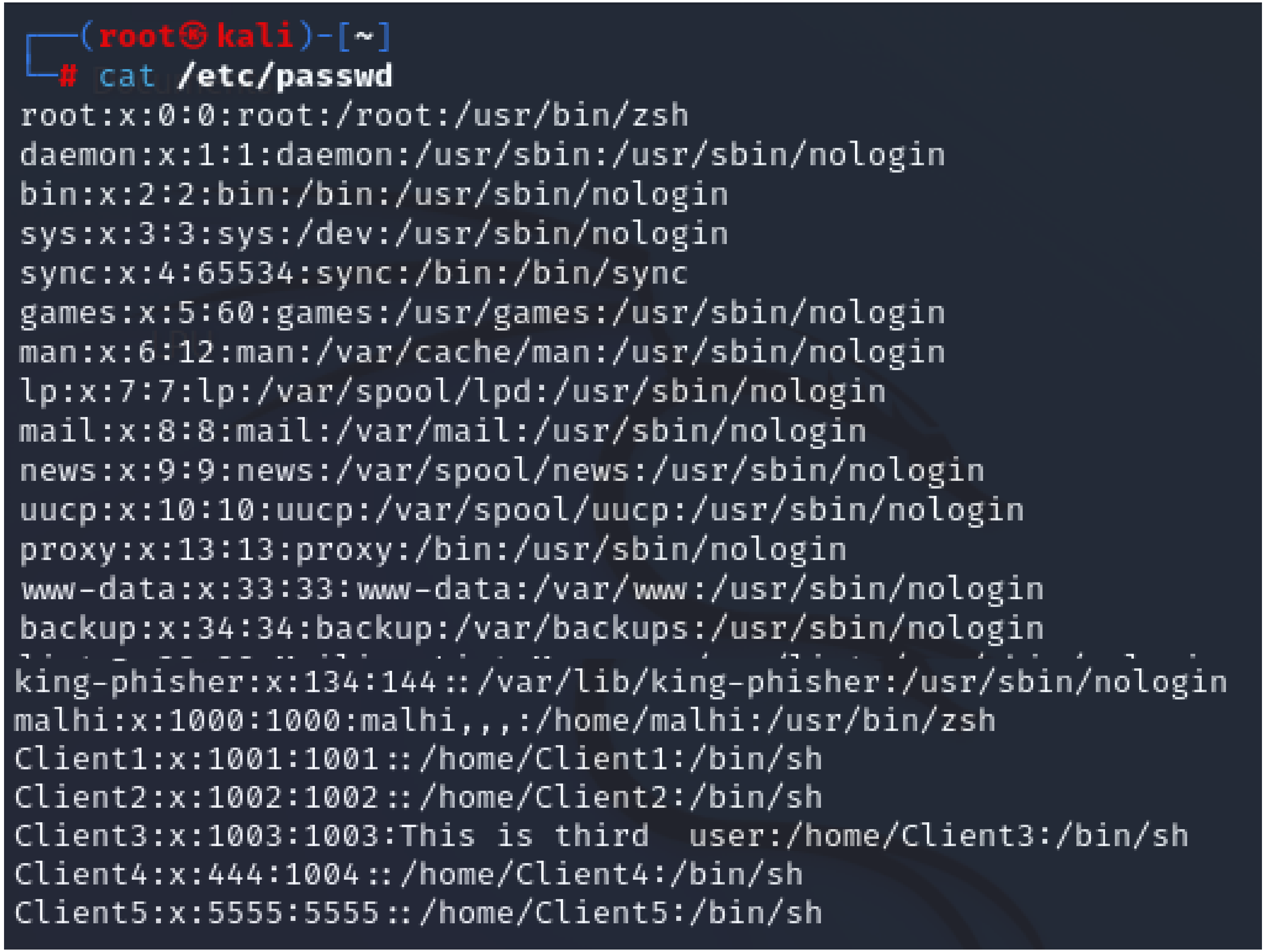
```
┌──(root㉿kali)-[~]
└─# usermod -e 2023/03/31 Client3
```

- To delete an user



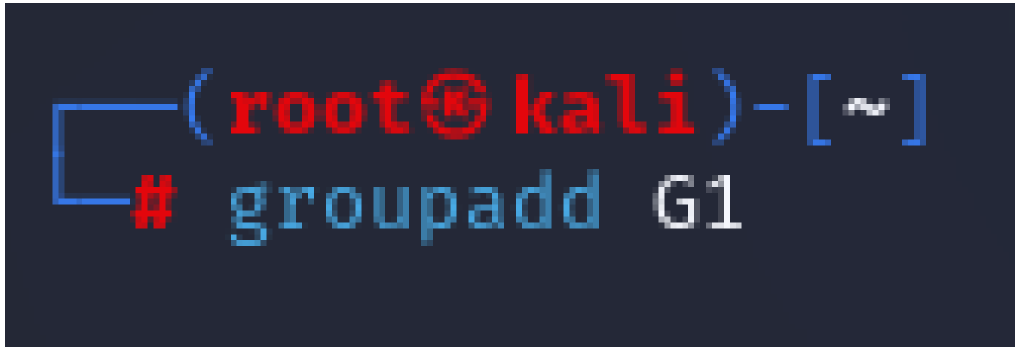
```
┌──(root㉿kali)-[~]
└─# userdel Client5
```

- To check the all users



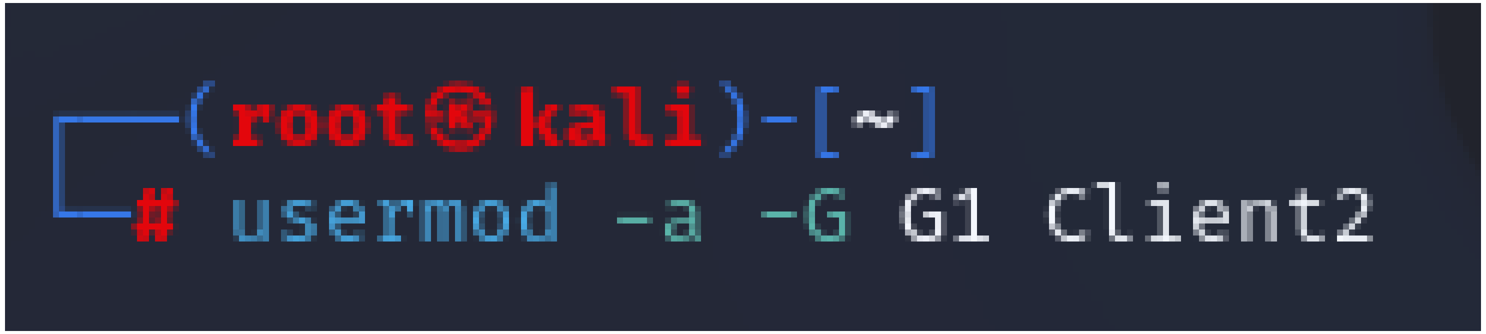
```
┌──(root㉿kali)-[~]
└─# cat /etc/passwd
root:x:0:0:root:/root:/usr/bin/zsh
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
king-phisher:x:134:144::/var/lib/king-phisher:/usr/sbin/nologin
malhi:x:1000:1000:malhi,,,:/home/malhi:/usr/bin/zsh
Client1:x:1001:1001::/home/Client1:/bin/sh
Client2:x:1002:1002::/home/Client2:/bin/sh
Client3:x:1003:1003:This is third  user:/home/Client3:/bin/sh
Client4:x:444:1004::/home/Client4:/bin/sh
Client5:x:5555:5555::/home/Client5:/bin/sh
```

- Create the group



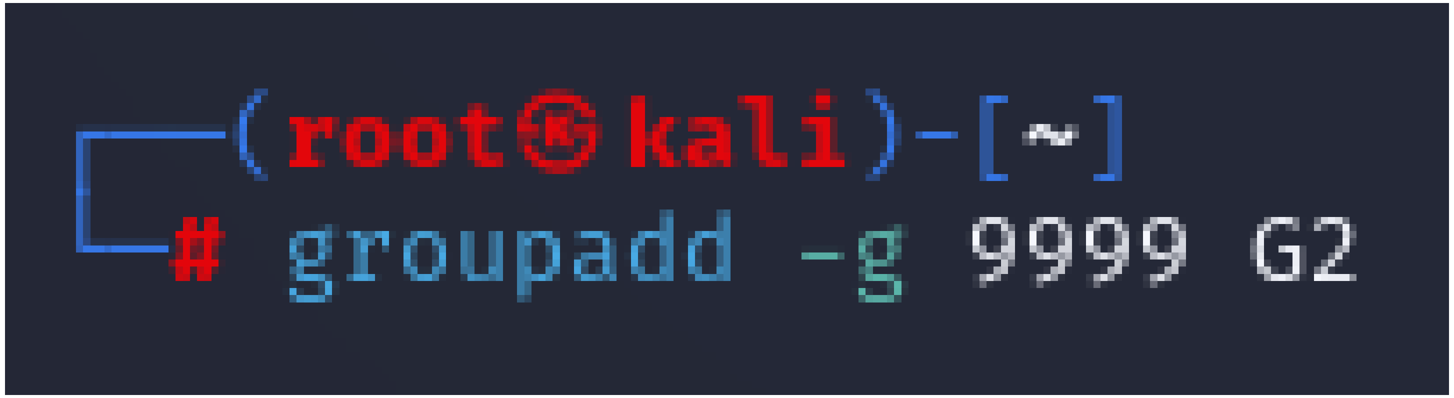
```
┌──(root㉿kali)-[~]
└─# groupadd G1
```

- To add user in the group



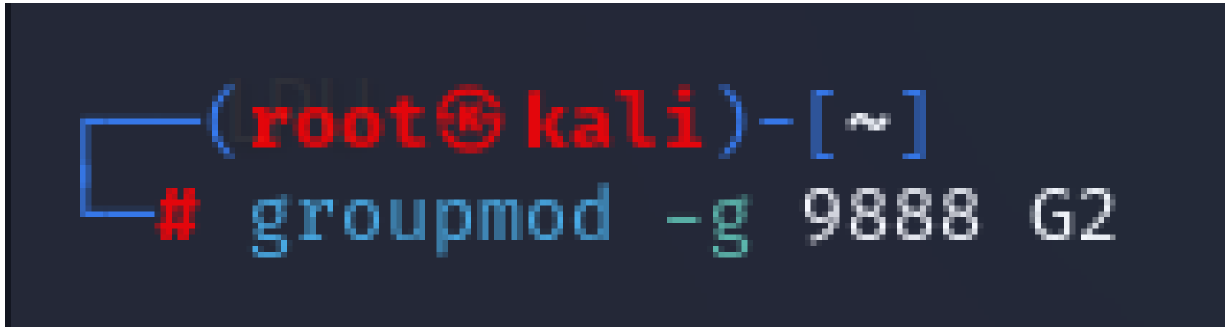
```
┌──(root㉿kali)-[~]
└─# usermod -a -G G1 Client2
```

- To create a group id

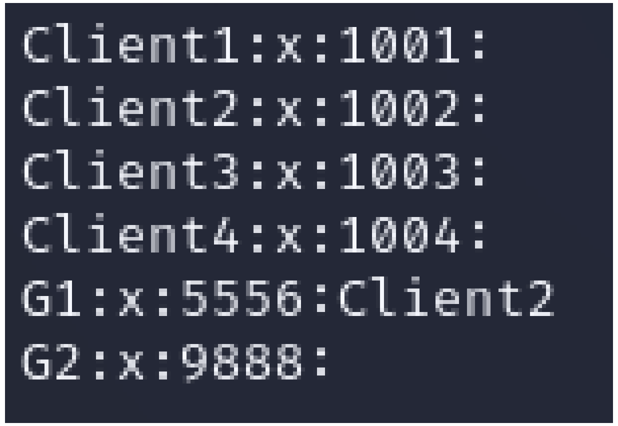


```
┌──(root㉿kali)-[~]
└─# groupadd -g 9999 G2
```

- To change the group id

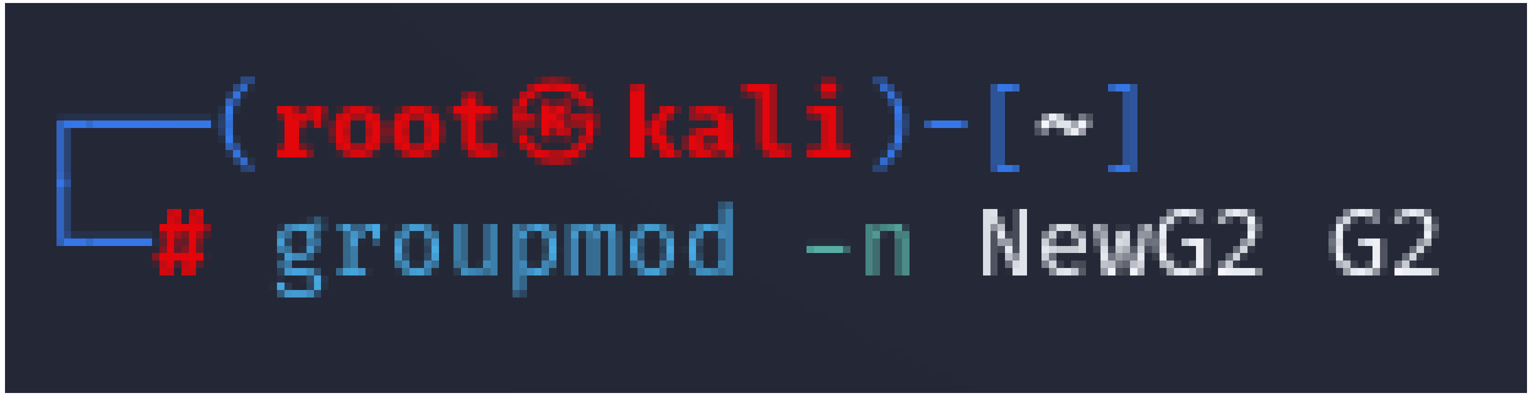


```
┌──(root㉿kali)-[~]
└─# groupmod -g 9888 G2
```

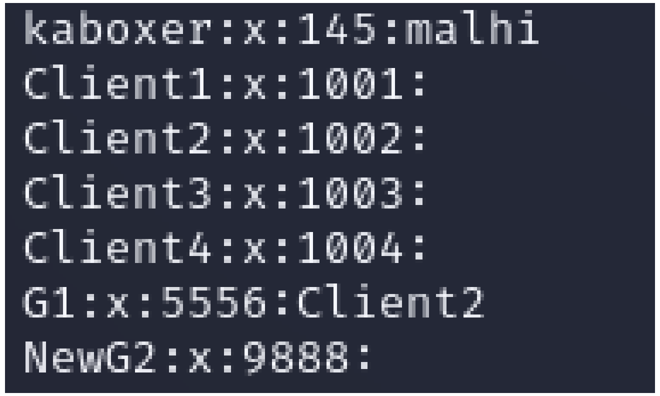


```
Client1:x:1001:
Client2:x:1002:
Client3:x:1003:
Client4:x:1004:
G1:x:5556:Client2
G2:x:9888:
```

- To change the group name (rename)

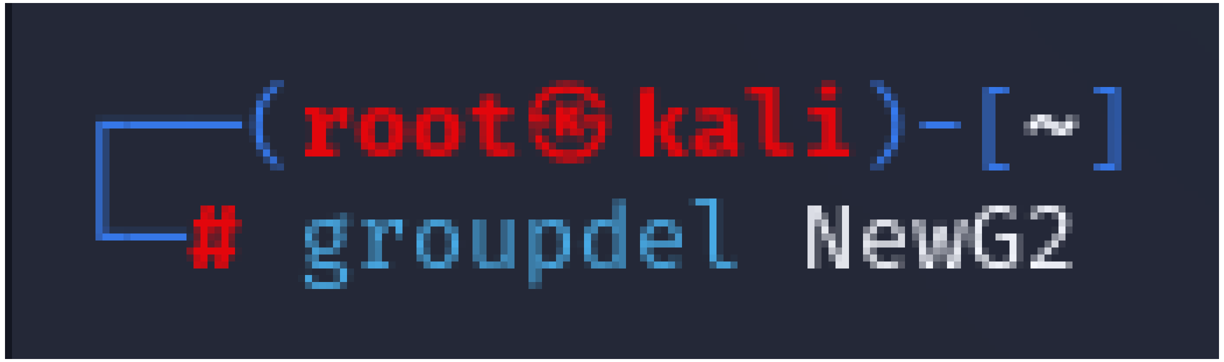


```
┌──(root㉿kali)-[~]
└─# groupmod -n NewG2 G2
```



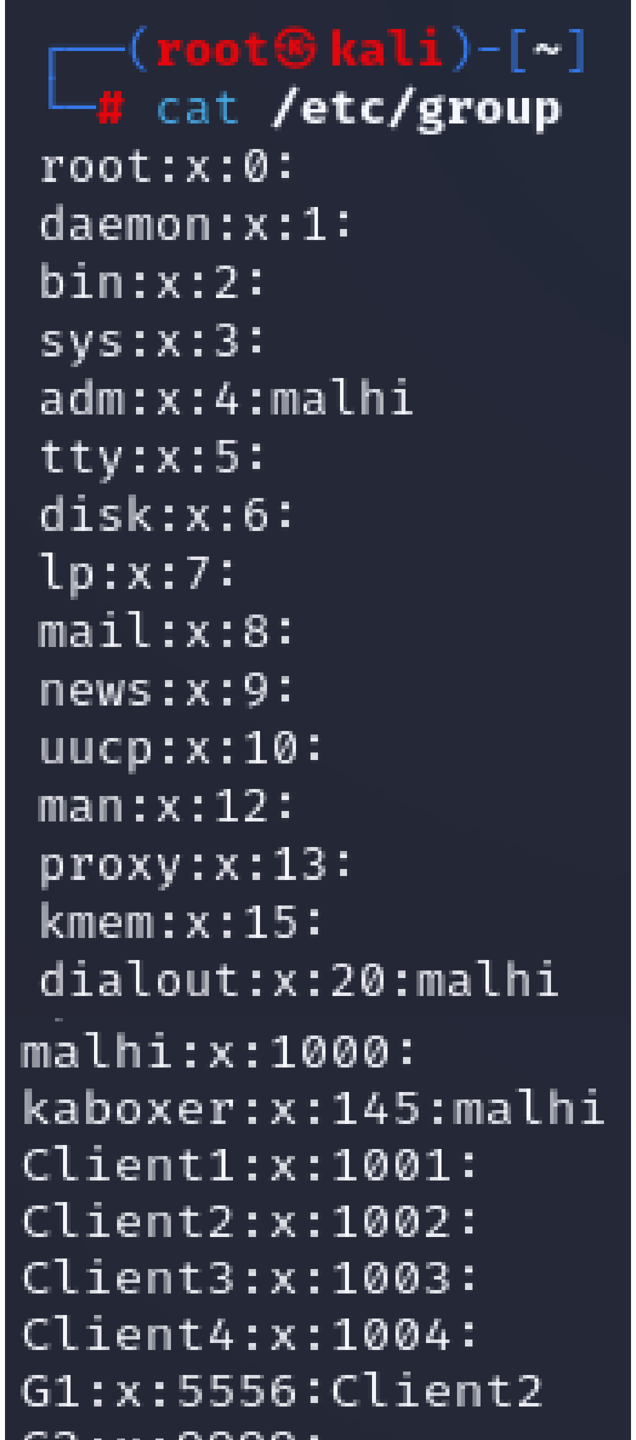
```
kaboxer:x:145:malhi
Client1:x:1001:
Client2:x:1002:
Client3:x:1003:
Client4:x:1004:
G1:x:5556:Client2
NewG2:x:9888:
```

- To delete any group. It does not delete user accounts that are members of the group.



```
┌──(root㉿kali)-[~]
└─# groupdel NewG2
```

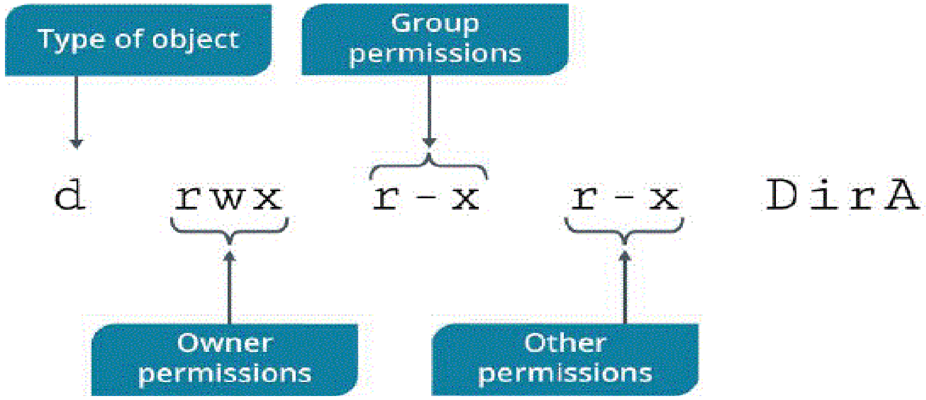
- To display all the groups along with users in the groups



```
┌──(root㉿kali)-[~]
└─# cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:malhi
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:malhi
malhi:x:1000:
kaboxer:x:145:malhi
Client1:x:1001:
Client2:x:1002:
Client3:x:1003:
Client4:x:1004:
G1:x:5556:Client2
```

File Permissions

- Every file has its own access rights for different groups of users.
- Access rights are – Read, Write and Execute.
- These rights are provided to different groups of users which are- Owner, Groups, Other Users.



- A file’s permission appears to be in format of three group of letters.
- Here three groups represent the permissions given to groups of the user.
- **Owner (**u**)**: The owner of the file or directory, also simply referred to as the user.
- **Group (**g**)**: The file or directory's group and all users belonging to that group.
- **Other (**o**)**: All other users (neither owner nor group member).

First part: Represents the access rights of the owner of the file.

Second part: Represents the access rights of group.

Third part: Represents the access rights of other users

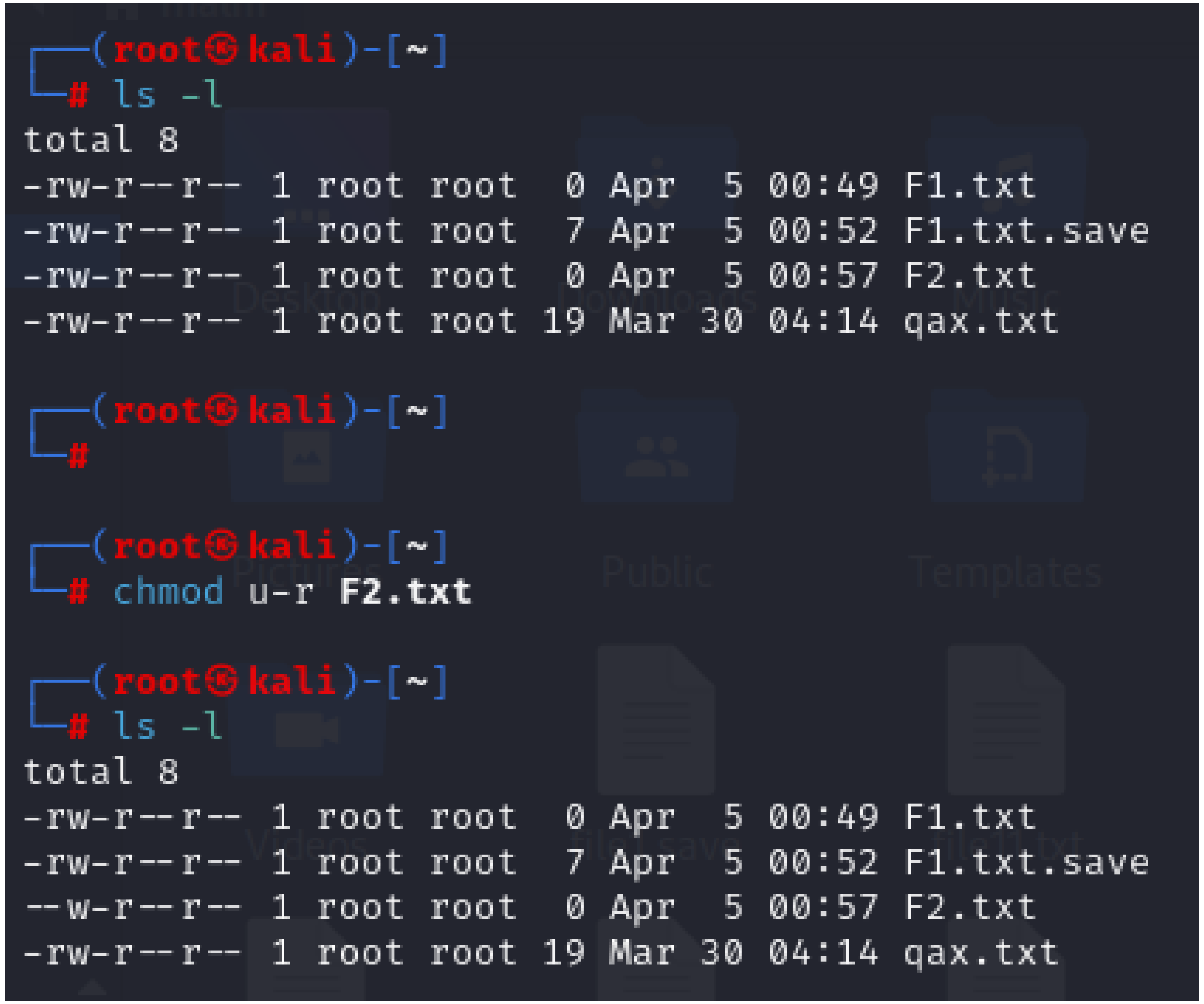
Here, meaning of symbols is as following:

- r- readable,
- w- writeable and
- x- executable

- chmod command is used to change the permissions of files and directories.

- 7- We are granting all rights- reading, writing and executing a file
- 6- We are granting two rights- reading and writing on file.
- 5- We are granting two rights- read and execute file.
- 4- We are granting one right- read a file.
- 3- We are granting two rights – write and execute a file.
- 2- We are granting one right- write on a file.
- 1- We are granting one right- execute a file.
- 0- We are granting no rights at all.

For Example: chmod 750 F2.txt



```
┌──(root㉿kali)-[~]
└─# ls -l
total 8
-rw-r--r-- 1 root root  0 Apr  5 00:49 F1.txt
-rw-r--r-- 1 root root  7 Apr  5 00:52 F1.txt.save
-rw-r--r-- 1 root root  0 Apr  5 00:57 F2.txt
-rw-r--r-- 1 root root 19 Mar 30 04:14 qax.txt

┌──(root㉿kali)-[~]
└─# 

┌──(root㉿kali)-[~]
└─# chmod u-r F2.txt

┌──(root㉿kali)-[~]
└─# ls -l
total 8
-rw-r--r-- 1 root root  0 Apr  5 00:49 F1.txt
-rw-r--r-- 1 root root  7 Apr  5 00:52 F1.txt.save
--w-r--r-- 1 root root  0 Apr  5 00:57 F2.txt
-rw-r--r-- 1 root root 19 Mar 30 04:14 qax.txt
```

- Suppose we want to allow write permissions to group:
    - chmod g+w F2.txt

  Suppose we want to deny execute permissions to owner:

  ```
  chmod u-x F2.txt
  ```

Create Partitions

- The first task in managing storage is to **partition** a storage device and **format** the partition with a file system.

STORAGE DEVICES

- A storage device is a physical component that can **record** data and hold it persistently.

- There are many **types** of storage devices that are supported by the Linux operating system.

- **Hard disk drive (HDD)—**electromechanical devices that use magnetic storage technology to store data, usually in large amounts.
- **Solid-state drive (SSD)—**storage devices that use non-mechanical solid-state technology to store data, usually in large amounts. They tend to support much quicker access times than HDDs.
- **USB thumb drive—**portable storage devices that use flash memory technology to store data, usually in small amounts compared to HDDs and SSDs. Their small size makes them easy to move around.
- **External storage drive—**portable storage drives that can use one of several technology types. They usually connect to a computer through a peripheral interface like USB, rather than being an internal component

BLOCK VS. CHARACTER DEVICES

- Linux refers to devices as either block or character devices.
  - Block devices are storage devices (like those listed previously) that can be read from and written to in blocks of data. (random access(buffering))
  - Character devices are devices like keyboards, mice, serial ports, etc., that can be read from and written to in streams of data. (sequential access(no buffering))

FILE SYSTEMS

- A file system is a data structure that is used by an operating system to **store, retrieve, organize, and manage** files and directories on storage devices.

- A file system maintains information such as the **date** of creation and modification of individual files, the **size** of files on the storage device, the **type** of files, and **permissions** associated with files.

- It also provides a **structured form** for data storage.

Linux supports many file system types. The most common are described in the following table.

- **FAT:** File Allocation Table (FAT) is an older file system that is compatible with many different operating systems, including all versions of Unix, Windows, and macOS.

- Improved versions include FAT32 and exFAT.

- **ext2 (second extended file system):** This used to be the native Linux file system of some older releases.

- It is still supported in current releases of Linux

- **ext3:** This is an improved version of ext2.
- In case of an abrupt system shutdown, ext3 is much faster in recovering data and better ensures data integrity.

- It can be easy to upgrade the file system from ext2 to ext3.

- **ext4:** This is one of two default file system for Linux distributions.
- It is backwards-compatible with the ext2 and ext3 file systems.
- Among ext4's improvements over ext3 are journaling, support of volumes of up to one exbibyte (EiB), and files up to 16 tebibyte (TiB) in size.
- This is the default file system for Ubuntu installations.
- (journaling: **keeps track of changes in a log before committing the changes to the main file system**.)

- **XFS(Extents File System):** This is a 64-bit, high-performance journaling file system that provides fast recovery and can handle large files efficiently.

NETWORK FILE SYSTEMS

- While the previous table lists general-purpose file systems, some file systems function as network protocols that enable the sharing of data over a network.

- Common **types** include the following:
- Server Message Block: The SMB protocol provides users shared access to files and other resources across a local area network (LAN). SMB clients make requests for resources to SMB servers, which respond and provide the appropriate level of access. This protocol is primarily used with Windows computers.

- **Common internet file system (CIFS)** is a specific implementation of SMB that is rarely in use.
- Microsoft designed it as a successor to SMB version 1, but SMB versions 2 and 3 superseded it.

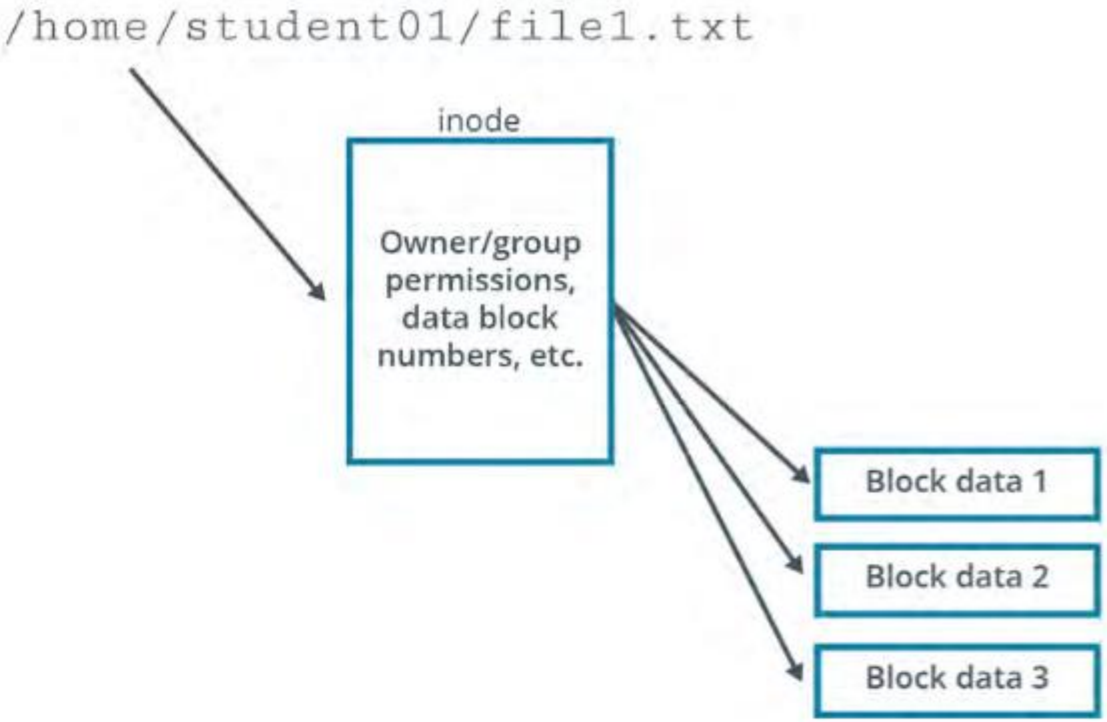
- **Network File System (NFS)** offers similar functionality to SMB, but the protocols are not compatible.

- NFS is preferred in situations where Linux clients access Linux servers.

- In environments that are a mix of Windows and Linux.

INODES

- An index node (inode) is an object that **stores** metadata about a file or directory on a file system.
- This metadata can **include** time-based values like when a file was created and last modified; permission and ownership information; the block locations of a file’s data on a storage device; and other miscellaneous information.


/home/student01/file1.txt
inode
Owner/group permissions, data block numbers, etc.
Block data 1
Block data 2
Block data 3

A specific file with its associated inode.

- Each inode on a file system is **identified** by a unique integer called an inode number.

- Whenever the system or an application tries to access a file, it searches for the **appropriate inode number** in a data structure called an inode table.

- The inode table **maps an inode number to its corresponding** file or directory name.

JOURNALING

- It is a method by which a file system records changes that have not yet been made to the file system itself in an object called a journal.

- This enables the file system to quickly recover after an unexpected interruption, such as a system crash, because the system can reference pending changes in the journal to resume where it had left off.

The journaling process involves the following phases:

- 1. The journal describes all the **changes** that must be made to the drive.
- 2. A **background process** makes each change as and when it is entered in the journal.
- 3. If the system **shuts down**, pending changes are performed when it is rebooted.
- 4. Incomplete entries in the journal are **discarded**.

VIRTUAL FILE SYSTEM

- A real file system refers to a discrete file system that the Linux kernel can normally work with directly.
- The problem is, Linux supports many different file system types that aren't necessarily compatible.
- The virtual file system (VFS) was created as a common software interface that sits between the kernel and real file systems.
- In effect, the VFS translates a real file system's details to the kernel so that the file system appears identical to any other file system.

- With VFS, you can mount multiple different types of file systems on the same Linux installation, and they will appear uniform to the user and to all other applications.
- Therefore, the user and these applications can work with the file system without
  actually knowing its underlying structure.
- This greatly increases interoperability between the system and running software.

- **Examples** of real file systems on a Linux system can include `xfs`, `ext4`, and several other types.

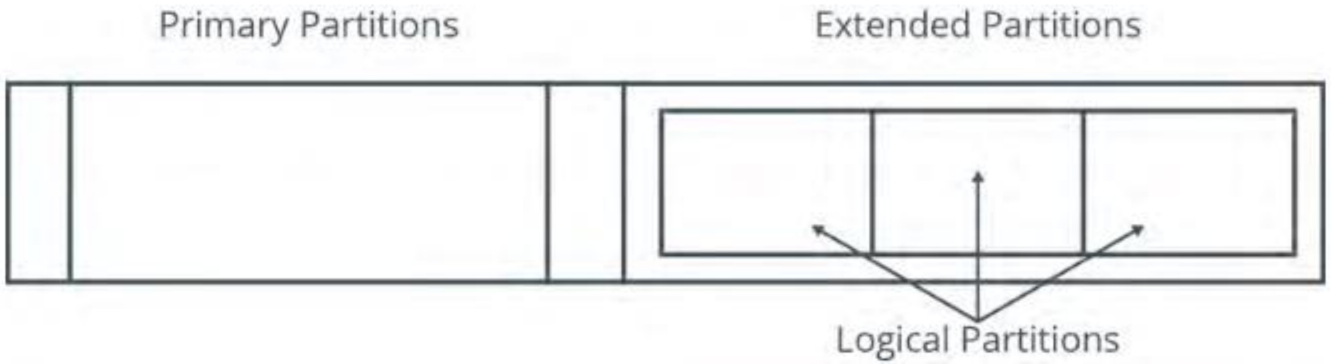
- **Examples** of virtual file systems can include `proc`, which contains system information during runtime; `devtmpfs`, which contains device nodes loaded by the kernel during system initialization; `debugfs`, which contains information useful in debugging the Linux kernel; and many more.

PARTITIONS

- A partition is a section of the storage drive that logically acts as a separate drive.
- Partitions enable you to convert a large drive to smaller manageable chunks, leading to better organization of information.
- A partition must be formatted and assigned a file system before data can be stored on it.
- Partitions are identified using a partition table, which is stored in one or more areas of the drive.
- The size of each partition can vary but cannot exceed the total free space of the storage drive.

TYPES OF PARTITIONS

- There are **three** types of partitions: primary, extended, and logical.



- **PRIMARY:** A partition that can contain one file system or logical drive and is sometimes referred to as a volume. The swap file system and the boot partition are normally created in a primary partition.

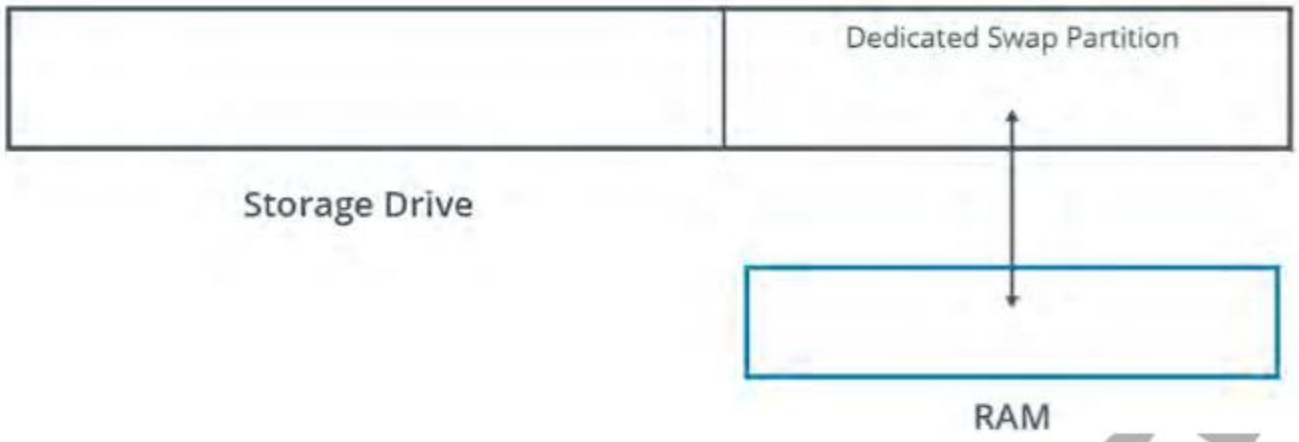
- **Extended:** An extended partition can contain several file systems, which are referred to as logical drives. There can be only one extended partition, which can be further subdivided. This partition type does not contain any data and has a separate partition table.

- **Logical:** A part of a physical drive that has been partitioned and allocated as an independent unit and functions as a separate drive. A logical partition is created within an extended partition, and is therefore a subset of an extended partition. There is no restriction on the number of logical partitions, but it is advisable to limit it to 12 logical partitions per drive.

SWAP SPACE

- It is a partition on the storage device that is used when the system runs out of physical memory.

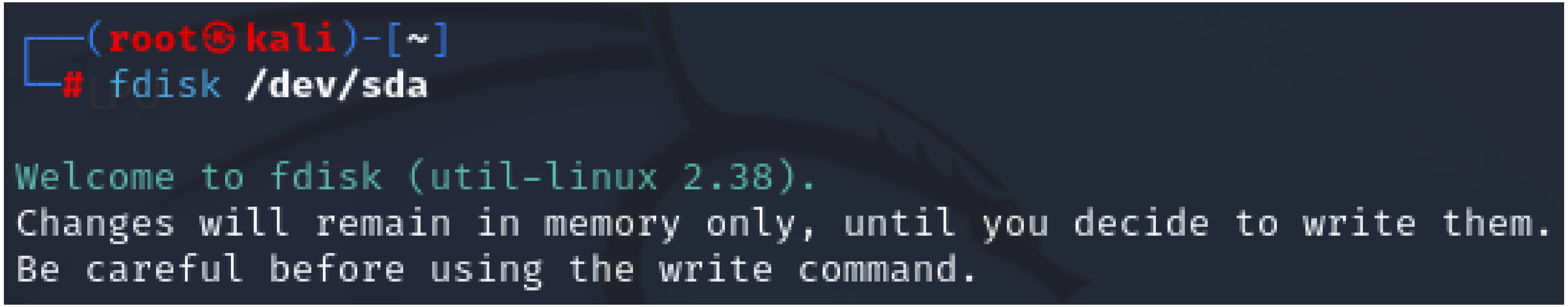
- Linux pushes some of the unused files from RAM to the swap space to free up memory. Usually, the swap space equals twice the RAM capacity.



Swap space on a storage drive.

THE fdisk UTILITY

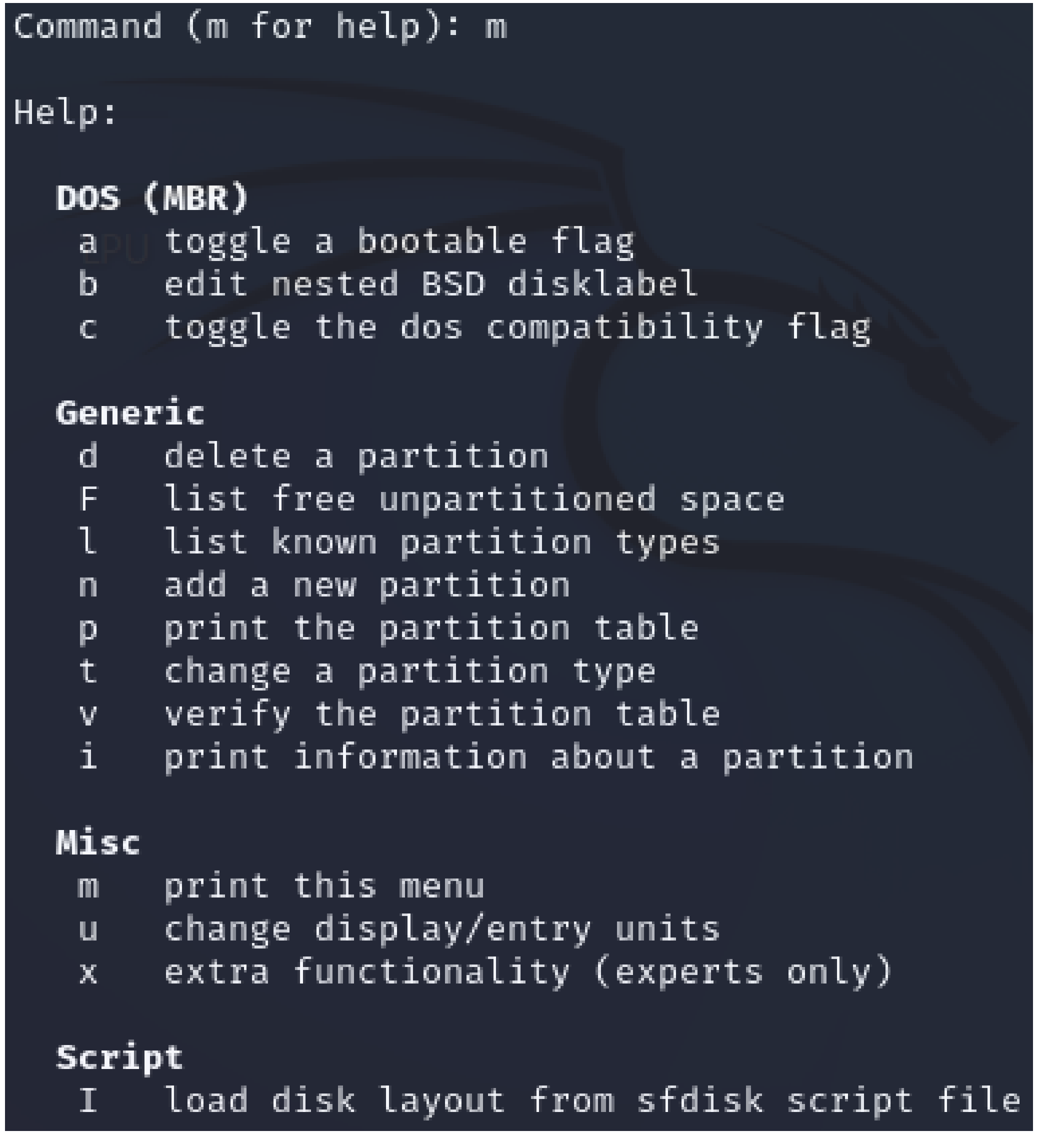
- The fdisk utility is a menu-driven program that is used to create, modify, or delete partitions on a storage drive.



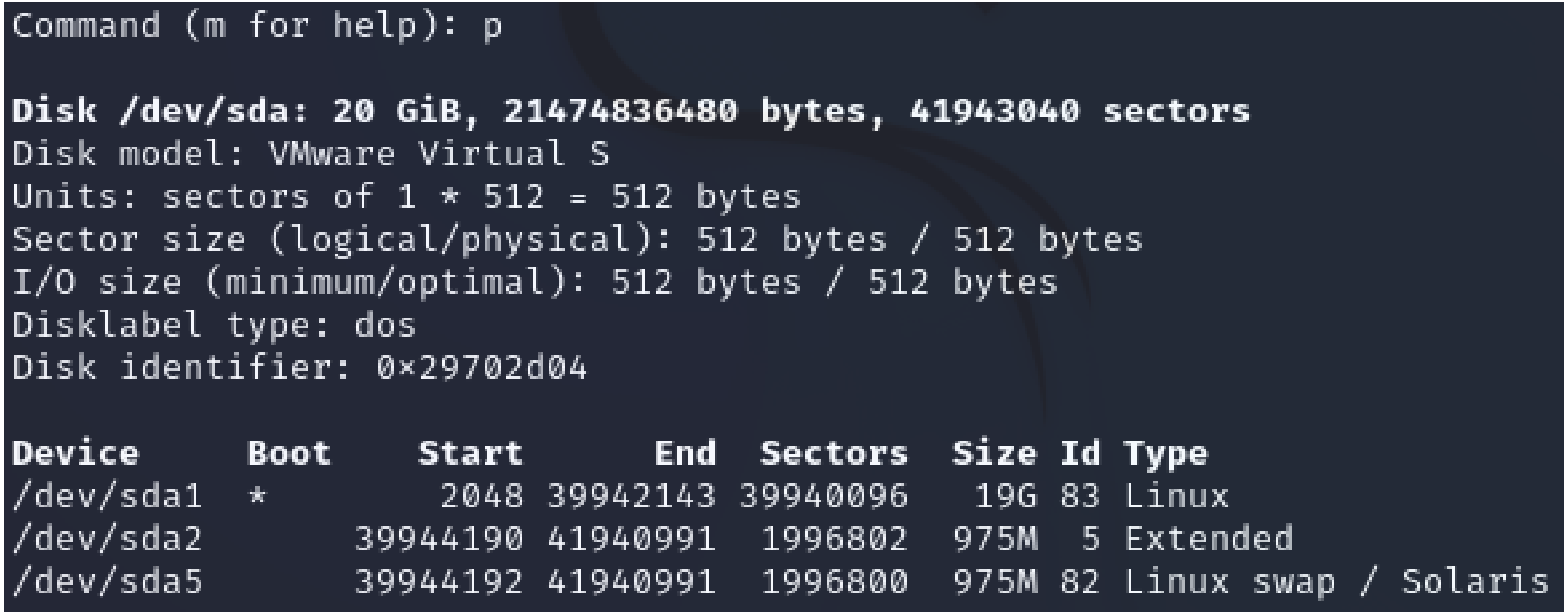
```
┌──(root㉿kali)-[~]
└─# fdisk /dev/sda

Welcome to fdisk (util-linux 2.38).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

- sda is a storage device

- m for help



```
Command (m for help): m

Help:

  DOS (MBR)
   a   toggle a bootable flag
   b   edit nested BSD disklabel
   c   toggle the dos compatibility flag

  Generic
   d   delete a partition
   F   list free unpartitioned space
   l   list known partition types
   n   add a new partition
   p   print the partition table
   t   change a partition type
   v   verify the partition table
   i   print information about a partition

  Misc
   m   print this menu
   u   change display/entry units
   x   extra functionality (experts only)

  Script
   I   load disk layout from sfdisk script file
```

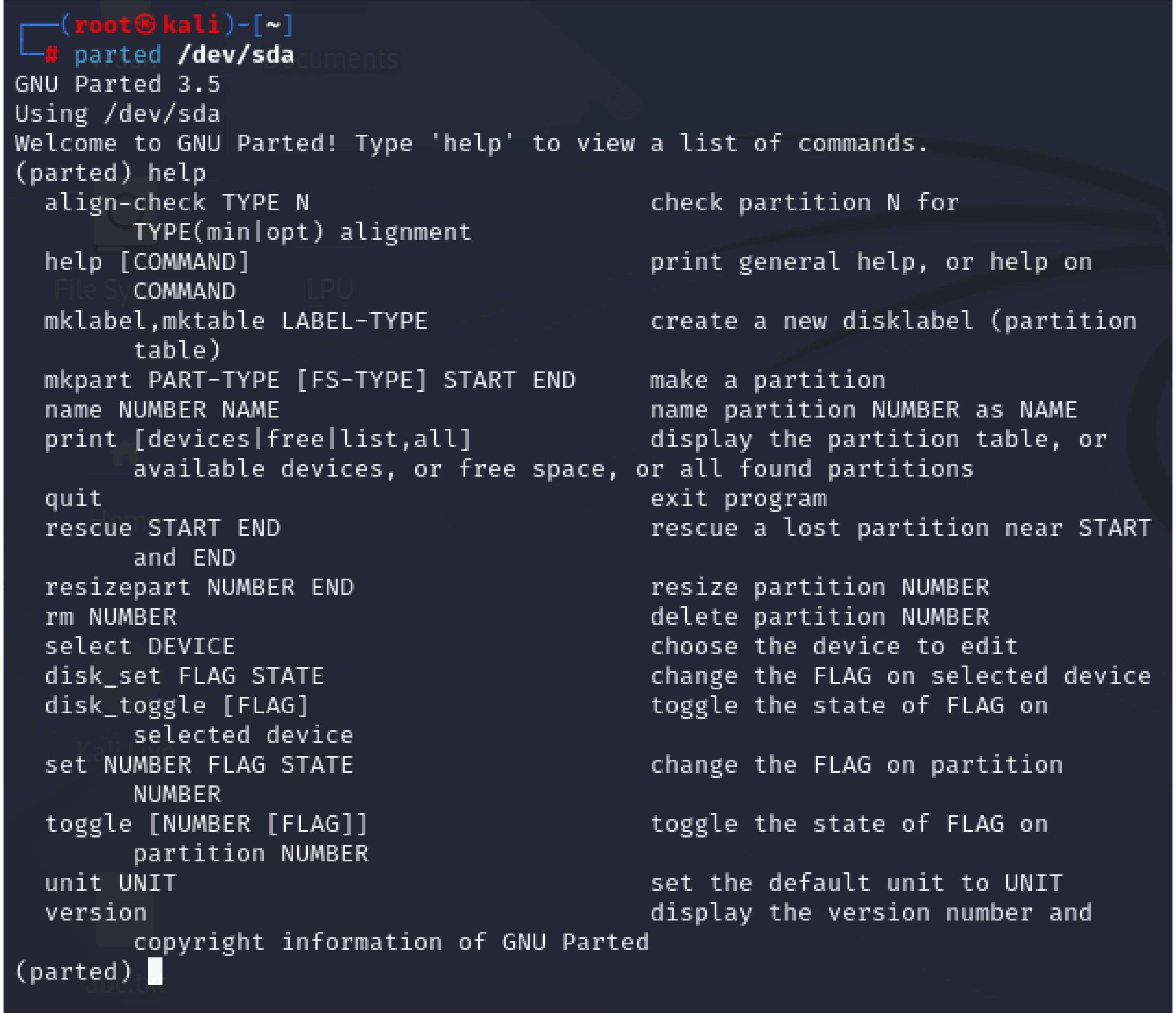


```
Command (m for help): p

Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Disk model: VMware Virtual S
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0×29702d04

Device     Boot    Start      End  Sectors  Size Id Type
/dev/sda1  *        2048 39942143 39940096   19G 83 Linux
/dev/sda2       39944190 41940991  1996802  975M  5 Extended
/dev/sda5       39944192 41940991  1996800  975M 82 Linux swap / Solaris
```

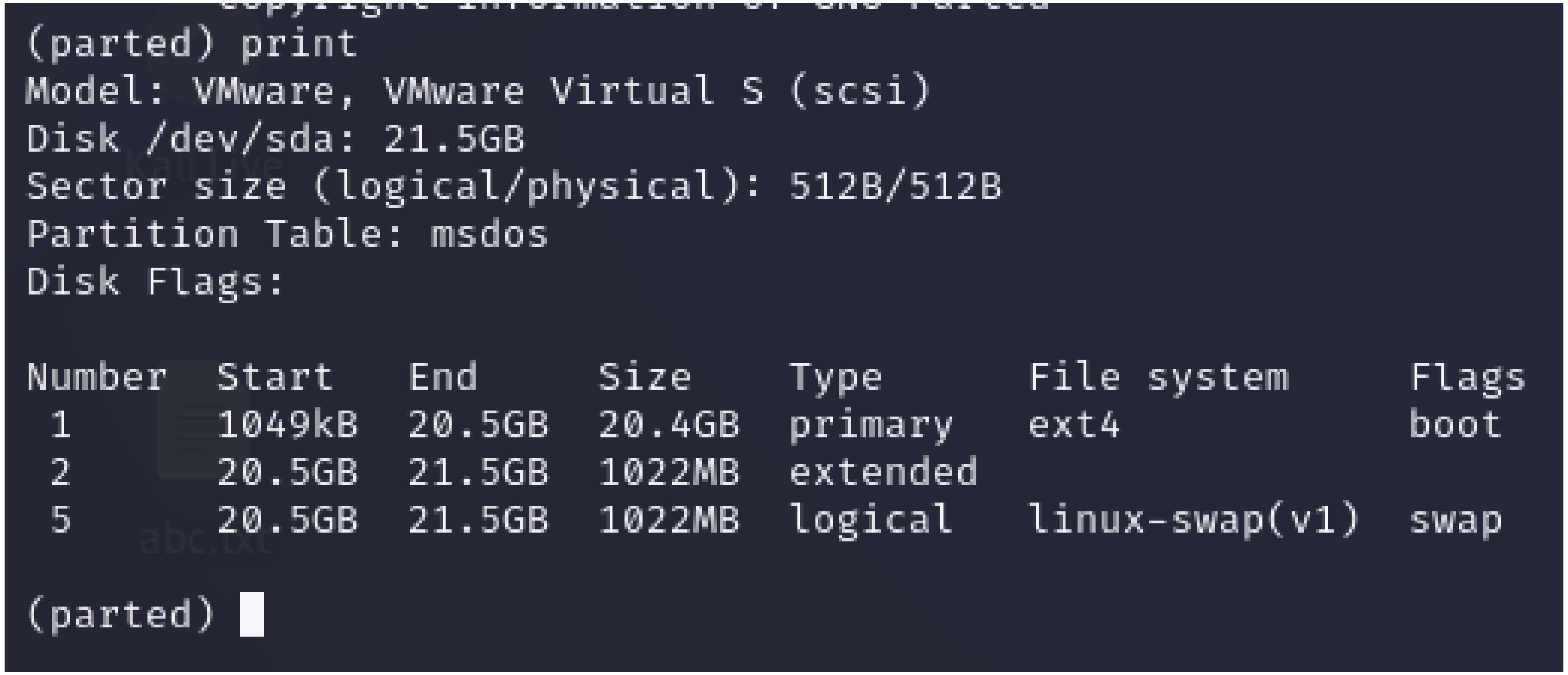
- p is print partitions

GNU PARTED

- The GNU Parted utility is also used to manage partitions.
- It is particularly useful when creating partitions on new storage drives.
- It can be used to create, destroy, and resize partitions.



```
┌──(root㉿kali)-[~]
└─# parted /dev/sda
GNU Parted 3.5
Using /dev/sda
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) help
  align-check TYPE N                        check partition N for
        TYPE(min|opt) alignment
  help [COMMAND]                            print general help, or help on
        COMMAND
  mklabel,mktable LABEL-TYPE                create a new disklabel (partition
        table)
  mkpart PART-TYPE [FS-TYPE] START END      make a partition
  name NUMBER NAME                          name partition NUMBER as NAME
  print [devices|free|list,all]             display the partition table, or
        available devices, or free space, or all found partitions
  quit                                      exit program
  rescue START END                          rescue a lost partition near START
        and END
  resizepart NUMBER END                     resize partition NUMBER
  rm NUMBER                                 delete partition NUMBER
  select DEVICE                             choose the device to edit
  disk_set FLAG STATE                       change the FLAG on selected device
  disk_toggle [FLAG]                        toggle the state of FLAG on
        selected device
  set NUMBER FLAG STATE                     change the FLAG on partition
        NUMBER
  toggle [NUMBER [FLAG]]                    toggle the state of FLAG on
        partition NUMBER
  unit UNIT                                 set the default unit to UNIT
  version                                   display the version number and
        copyright information of GNU Parted
(parted) 
```



```
        copyright information of GNU Parted
(parted) print
Model: VMware, VMware Virtual S (scsi)
Disk /dev/sda: 21.5GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type      File system     Flags
 1      1049kB  20.5GB  20.4GB  primary   ext4            boot
 2      20.5GB  21.5GB  1022MB  extended
 5      20.5GB  21.5GB  1022MB  logical   linux-swap(v1)  swap

(parted) 
```

THE partprobe COMMAND

- The `partprobe` command is used to update the kernel with changes in the partition table.

- The command first checks the partition table, and if there are any changes, it automatically updates the kernel with the changes.

- After creating a partition with fdisk, you cannot add a file system to that partition unless the kernel can read it from the partition table.
- You might receive an error like "Re-reading the partition table failed" during the fdisk operation.

- **Rebooting** the machine updates the table, or you can issue partprobe instead to update the table without a reboot.

THE mkfs COMMAND

- The `mkfs` command is used to build a Linux file system on a device, which is usually a drive partition.

THE fstab FILE

- The `fstab` file is a configuration file that stores information about storage devices and partitions and where and how the partitions should be mounted.

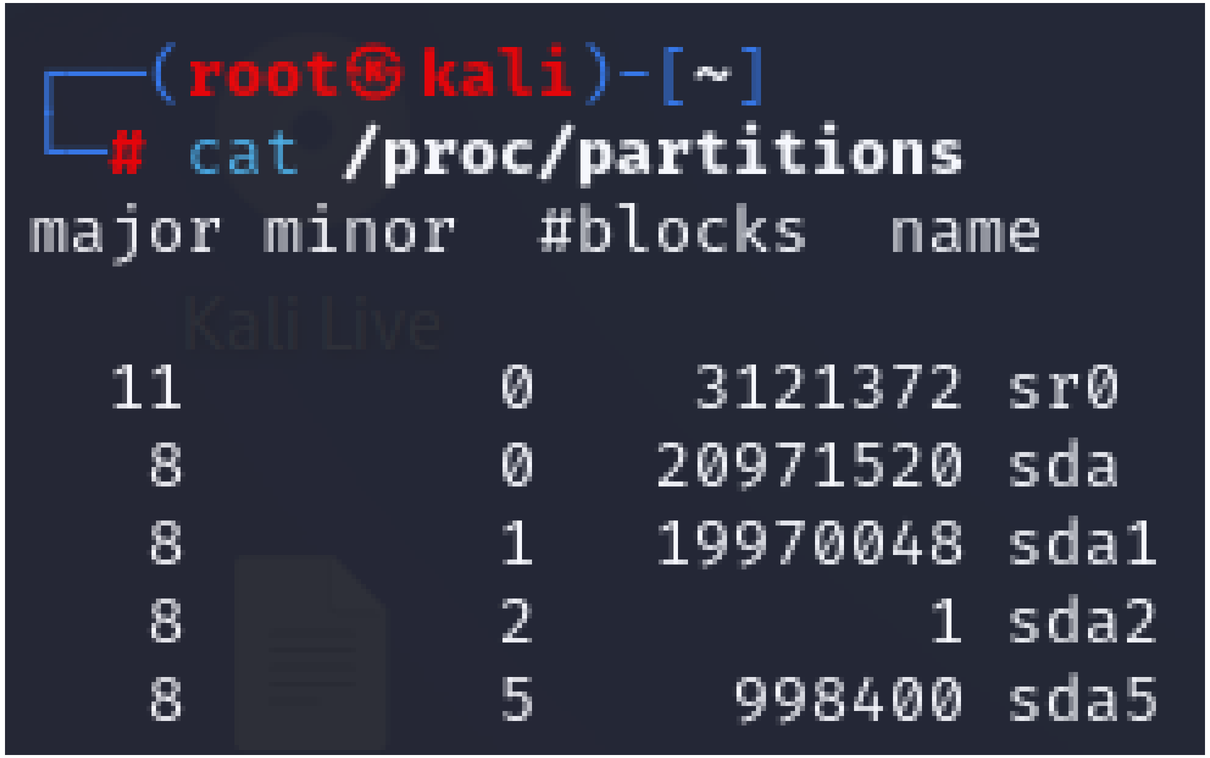
- The `fstab` file is located in the `/etc` directory. It is read by the system at boot time and can be edited only by a root user.

THE crypttab FILE

- The `/etc/crypttab` file performs a similar function to the `fstab` file, but its purpose is to store information about encrypted devices and partitions that must be unlocked and mounted on system boot.

THE /proc/partitions FILE

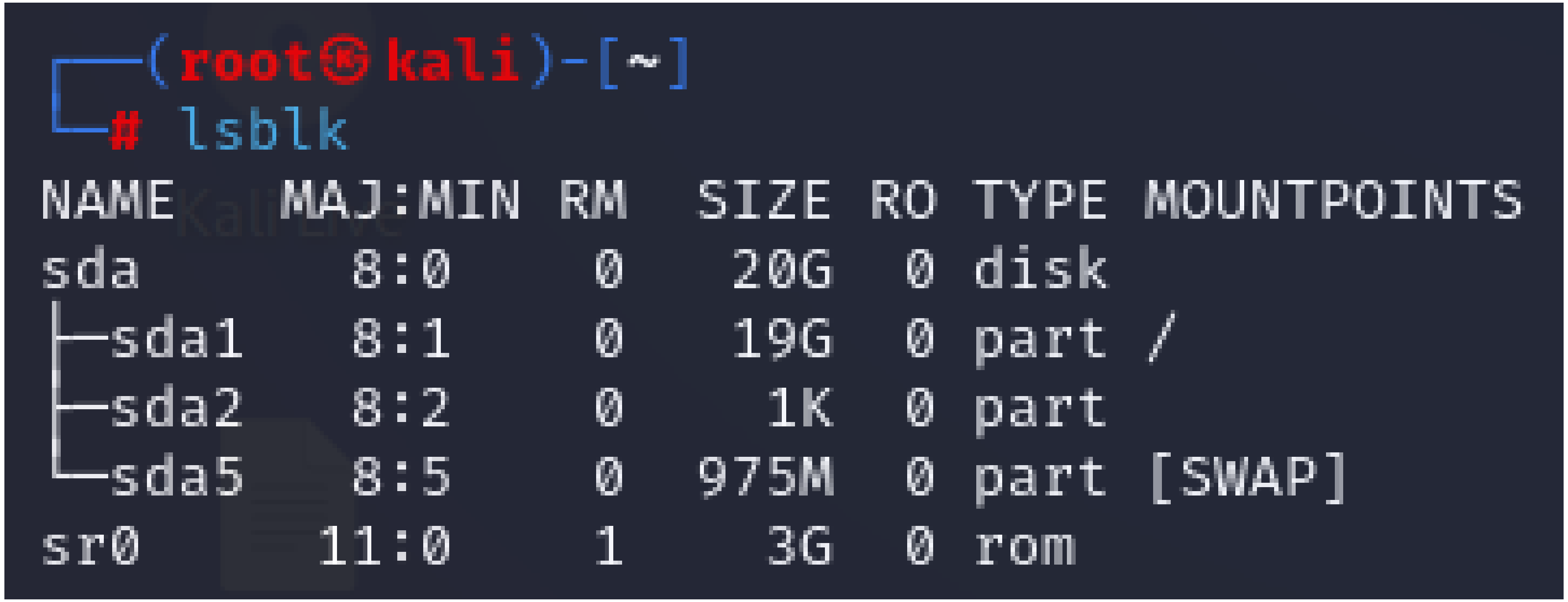
- The `/proc/partitions` file contains information about each partition that is currently attached to the system.



```
┌──(root㉿kali)-[~]
└─# cat /proc/partitions
major minor  #blocks  name

  11        0    3121372 sr0
   8        0   20971520 sda
   8        1   19970048 sda1
   8        2          1 sda2
   8        5     998400 sda5
```

THE lsblk COMMAND

- The `lsblk` command displays information about all block storage devices that are currently available on the system.



```
┌──(root㉿kali)-[~]
└─# lsblk
NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda      8:0    0    20G  0 disk
├─sda1   8:1    0    19G  0 part /
├─sda2   8:2    0     1K  0 part
└─sda5   8:5    0   975M  0 part [SWAP]
sr0     11:0    1     3G  0 rom
```

- ***ADDITIONAL READING IS STRONGLY RECOMMENDED***