BASICS LINUX COMMANDS

Linux is a family of open-source operating systems using a Kernel-based architecture. Most Linux distributions use a graphical user interface (GUI), making them beginner-friendly. However, most users prefer utilizing the command-line interface (CLI) because it's quicker and offers more control. A Linux command is a program or utility that runs on the CLI which is similar to the Command Prompt application in Windows.

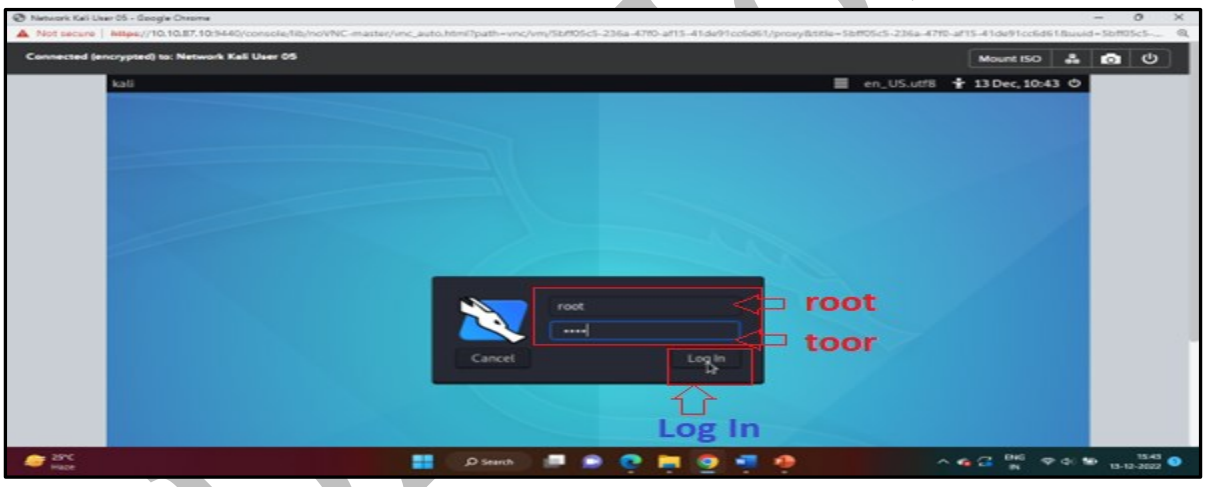
In Linux User Shell (CLI), the default prompt for a regular user is simply a dollar sign. i.e. $

For the root user(administrator) is a pound sign, also called a number sign or hashtag. i.e. #

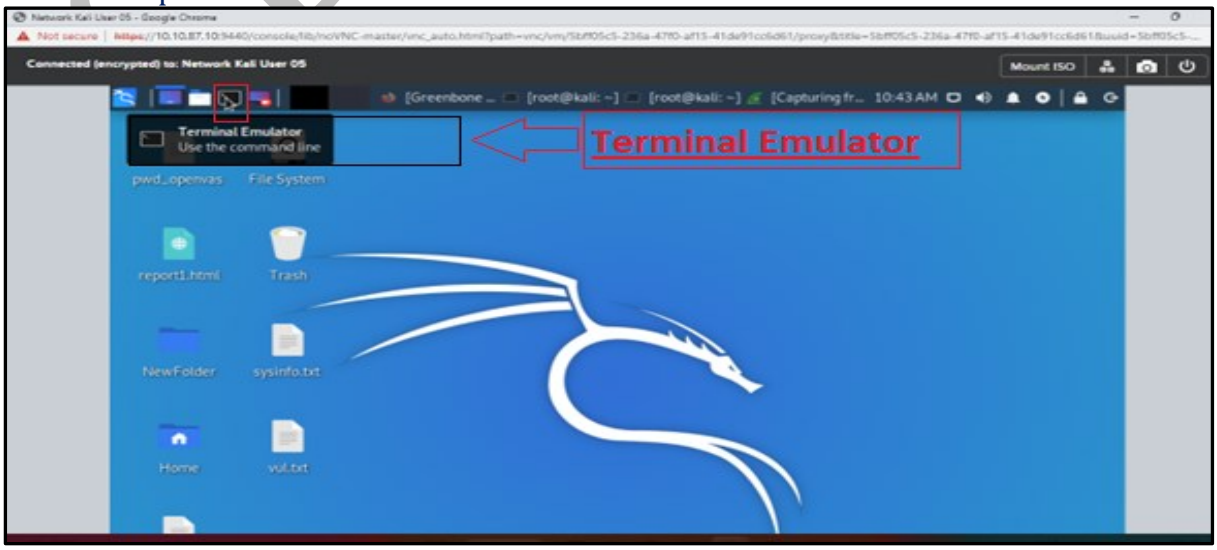
In this lab manual, you will learn about some basic Linux commands starting from the **man** command, used for getting the details of other commands to various other useful commands related to basic Linux utilities and processes.

For executing the commands demonstrated in this manual, follow the below-given steps.

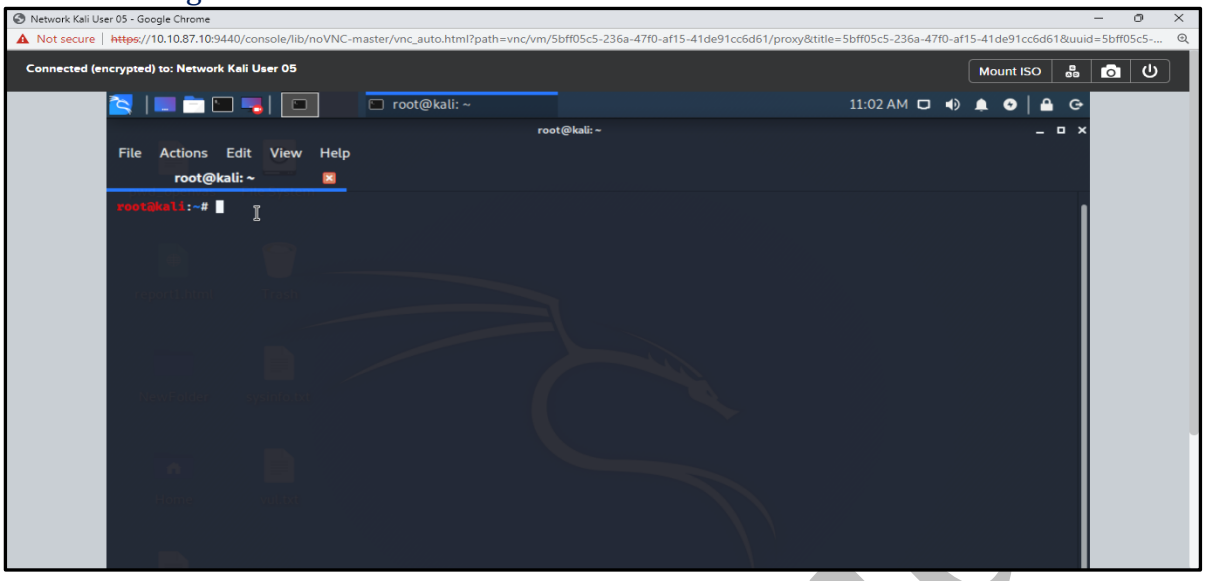
1. Connect to the kali Linux machine, created by you, using the RDP protocol.
2. When prompted for the username and password, enter **root** as username and **toor** as password. The root is the administrator user of the machine.



3. click on the black box icon (Terminal Emulator) in the top left corner of the Kali Linux Desktop.



Running the terminal while using the root account, allows you to run various commands with administrator rights.



Let's understand some basic Linux commands one by one.

1. MAN

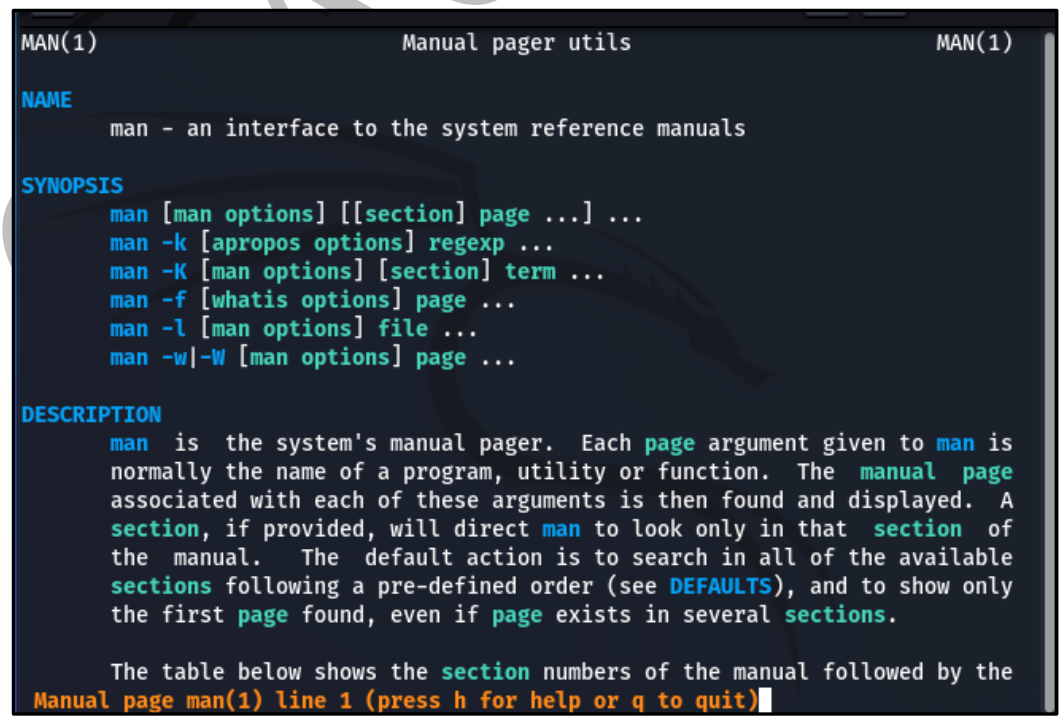
The man command is used to read the reference manual of any Linux command.

Here, **man** shows the output as manual of man command itself

Syntax : $man man

Or

$man, nano displays manual for nano



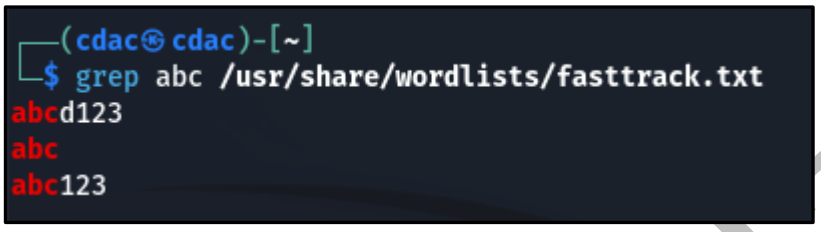
Try to use the man command for other Linux commands.

2. GREP

Syntax : $grep abc /usr/share/wordlists/fasttrack.txt

grep command is used to search within text files for finding lines in files that contain search text

In the following example, grep is used to search abc string in fasttrack.txt text file

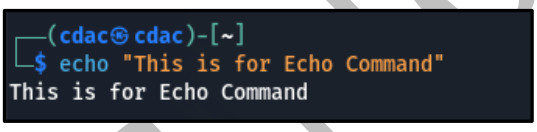


3. ECHO

echo command in Linux is used to display lines of text/string that are passed as an argument

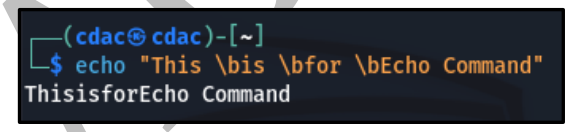
Syntax: $echo "This is for Echo Command"

This is a simple command to display a string passed under argument.



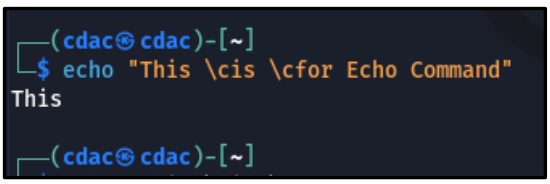
\b removes all the spaces in between the text while using echo, as shown below.

Command: $echo "This \bis \bfor \bEcho Command"



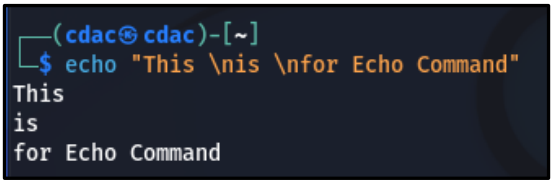
\c suppresses trailing new line with backspace

Command: $echo "This \cis \cfor Echo Command"



\n option creates a new line from where it is used.

Command: $echo "This \nis \nfor Echo Command"



\t option is used to create horizontal tab spaces.

Command: $echo "This \tis \tfor Echo Command"

```
┌──(cdac㉿cdac)-[~]
└─$ echo "This \tis \tfor Echo Command"
This    is      for Echo Command
```

\r is used to specify carriage return in output.

Command: $echo "This \ris \rfor Echo Command"

```
┌──(cdac㉿cdac)-[~]
└─$ echo "This \ris \rfor Echo Command"
for Echo Command
```

\v option is used to create vertical tab spaces.

Command: $echo "This \vis \vfor Echo Command"

```
┌──(cdac㉿cdac)-[~]
└─$ echo -e "This \vis \vfor Echo Command"
This
    is
       for Echo Command
```

Echo * command will print all files/folders, similar to the ls command

Command: $echo *

```
┌──(cdac㉿cdac)-[~]
└─$ echo *
Desktop Documents Downloads Music Pictures Public Templates Videos
```

4. Init

Using init command, you can restart and shut down the system. Using init zero, we can shut down the system.

Command: $init 0
To shutdown system

```
┌──(cdac㉿cdac)-[~]
└─$ init 0
```

Command: $init 6
To restart system.

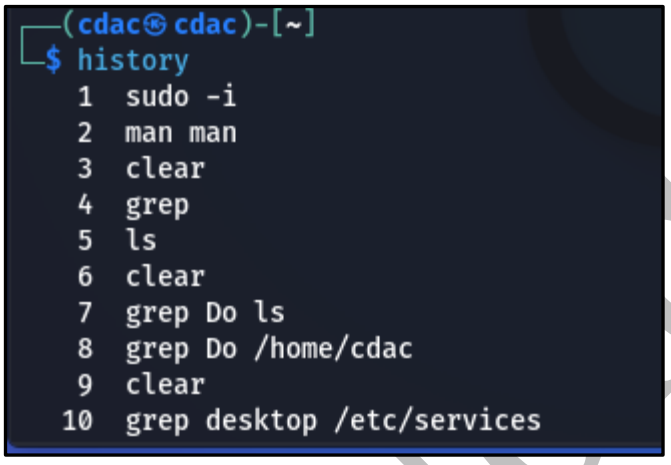
```
┌──(cdac㉿cdac)-[~]
└─$ init 6
```

5. History

After you type a command, the entire command line is saved in your shared history list. So using the history command, you can see all the commands that you executed earlier.

Enter the command without options or followed by a number to list that many of the most recent commands.

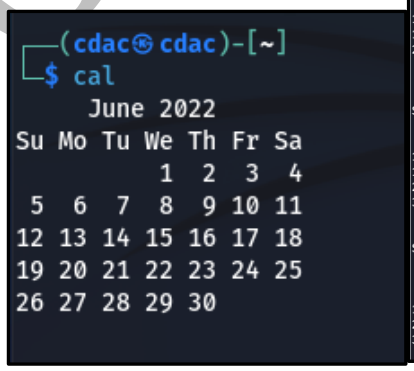
Command: $history

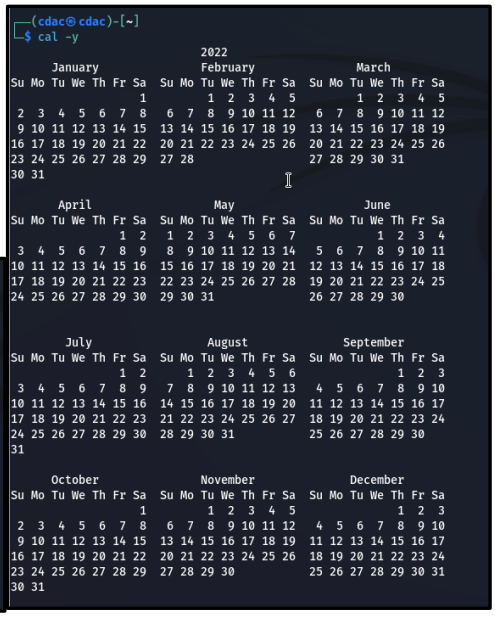


6. Cal

Cal command is used to list the current calendar

Command: $cal
Command: $cal -y
cal: Shows the current month calendar on the terminal
cal -y : Shows the calendar of the complete current year with the current date highlighted





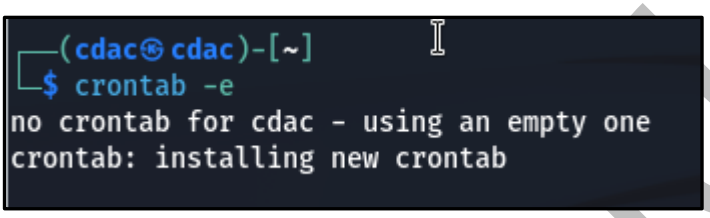
7. Cron

The **cron** is a software utility, that automates the scheduled task at a predetermined time. It is a **daemon process**, which runs as a background process and performs the specified operations at the predefined time when a certain event or condition is triggered without the intervention of a user.

For example, the cron command can be used to take a backup of files every day during working hours like 09:00 a.m. To 06:00. p.m.

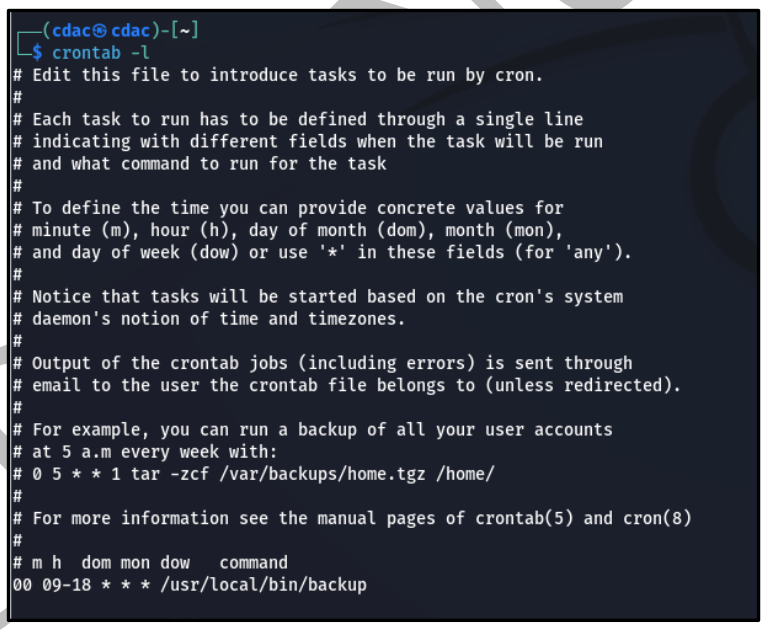
Command: $crontab –e

To add a schedule for a task, execute "crontab -e". e option is used for editing the cron file.



Command: $crontab -l

To see the scheduled task for the current user execute "crontab -l"



For example,

00 15 * * Thu /usr/local/bin/mycronjob.sh line in cronfile indicates running **mycronjob.sh** every Thursday at 3 p.m.

Crontab Command Fields and Ranges for Reference

Crontab Fields and Allowed Ranges (Linux Crontab Syntax)

Field	Description	Allowed Value
MIN	Minute field	0 to 59
HOUR	Hour field	0 to 23
DOM	Day of Month	1-31
MON	Month field	1-12
DOW	Day Of Week	0-6
CMD	Command	Any command to be executed.

8. DF

To see the amount of space available on all of the mounted filesystems on your Linux computer, type df with no options:

Command: $df

This example output shows the space available on the hard disk partition mounted on the / (root) directory (/dev/sda1). Disk space is shown in 1KB blocks

```
┌──(root㉿cdac)-[~]
└─# df
Filesystem     1K-blocks     Used Available Use% Mounted on
udev             1934252        0   1934252   0% /dev
tmpfs             395644     1112    394532   1% /run
/dev/sda1       81000912 10853348  65986952  15% /
tmpfs            1978212        0   1978212   0% /dev/shm
tmpfs               5120        0      5120   0% /run/lock
tmpfs             395640       76    395564   1% /run/user/1000
```

9. Uname

The uname command shows the type of system you are running (Linux). When you add -a, you also can see the hostname, kernel release, and kernel version.

Command: $uname -a

```
┌──(cdac㉿cdac)-[~]
└─$ uname -a
Linux cdac 5.16.0-kali7-amd64 #1 SMP PREEMPT Debian 5.16.18-1kali1 (2022-04-01) x86_64 GNU/Linux
```

10. Hostname

By simply entering the hostname command, it will show you the hostname that is currently allocated to the current user.

Syntax: $hostname

To show or set the system's hostname

11. Whatis

whatis command in Linux is used to get a one-line manual page description.

Syntax: $whatis -h

12. Who

Syntax: $who -uH

You can list the current login session by using the who command. In the following example, the -u option says to add information about idle time and the process ID, and -H asks that a header be printed:

The output of the below example shows that the user "cdac" is logged in on tty1 (which is the first virtual console on the monitor connected to the computer) and his login session began at 20:57 on January 13. The IDLE time shows how long the shell has been open without any command being typed (the dot indicates that it is currently active). PID shows the process ID

of the user's login shell. COMMENT would show the name of the remote computer from which the user had logged in.

```
┌──(cdac㉿cdac)-[~]
└─$ who -uH
NAME     LINE         TIME             IDLE          PID COMMENT
cdac     :1           2022-06-21 17:15   ?          1014 (:1)
```

13. Top

Syntax: $top

With top, the default is to display processes based on how much CPU time they are currently consuming.

```
top - 20:35:01 up  3:21,  1 user,  load average: 0.35, 0.11, 0.03
Tasks: 172 total,   1 running, 171 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.4 us,  0.2 sy,  0.0 ni, 98.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3863.7 total,   1882.1 free,   1138.6 used,    843.0 buff/cache
MiB Swap:    975.0 total,    975.0 free,      0.0 used.   2470.1 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1199 cdac      20   0 5086928 400532 147232 S   5.0  10.1   3:12.61 gnome-shell
   1018 cdac      20   0  909520 125740  72232 S   1.0   3.2   0:29.89 Xorg
   1584 cdac      20   0  563144  58356  43484 S   1.0   1.5   0:10.63 gnome-terminal-
   4342 cdac      20   0   10508   4128   3340 R   0.3   0.1   0:00.01 top
      1 root      20   0  166112  11756   8804 S   0.0   0.3   0:01.69 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.01 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
```

14. Banner

Syntax: $banner CDAC

banner command in Linux is used to print the ASCII character string in a large letter to standard output.

User may have to install it by “apt install systemvbanner”

```
┌──(cdac㉿cdac)-[~]
└─$ banner CDAC
 #####   ######      #      #####
#     # #     #    # #    #     #
#       #     #   #   #   #
#       #     #  #     #  #
#       #     #  #######  #
#     # #     #  #     #  #     #
 #####  ######   #     #   #####
```

15. Whoami

Syntax: $whoami

whoami command is used both in *Unix Operating System* and as well as in *Windows Operating System*. This command outputs the current user.

```
┌──(cdac㉿cdac)-[~]
└─$ whoami
cdac
```

```
┌──(cdac㉿cdac)-[~]
└─$ whoami --help
Usage: whoami [OPTION]...
Print the user name associated with the current effective user ID.
Same as id -un.

      --help     display this help and exit
      --version  output version information and exit

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/whoami>
or available locally via: info '(coreutils) whoami invocation'
```

16. PS

Syntax: $ps u

In this example, the u option (equivalent to -u) asks that usernames be shown, as well as other information such as the time the process started and memory and CPU usage for processes associated with the current user.

```
┌──(cdac㉿cdac)-[~]
└─$ ps u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
cdac        1014  0.0  0.2 163928  9312 tty2     Ssl+ 17:15   0:00 /usr/libexec/gdm-x-session --run-script /usr/bin/gnome-session
cdac        1018  0.2  3.1 909520 125748 tty2    Sl+  17:15   0:34 /usr/lib/xorg/Xorg vt2 -displayfd 3 -auth /run/user/1000/gdm/Xauthority -nolisten tcp -back
cdac        1051  0.0  0.4 300056 16444 tty2     Sl+  17:15   0:00 /usr/libexec/gnome-session-binary
cdac        1609  0.0  0.2  13904  7968 pts/0    Ss   17:16   0:12 zsh
cdac        3745  0.0  0.1  13616  7264 pts/1    Ss+  20:01   0:00 zsh
cdac        4919  0.0  0.0   9992  3492 pts/0    R+   21:10   0:00 ps u

┌──(cdac㉿cdac)-[~]
└─$ ps
    PID TTY          TIME CMD
   1609 pts/0    00:00:12 zsh
   4931 pts/0    00:00:00 ps
```

17. Kill

Syntax: $kill -l 1199

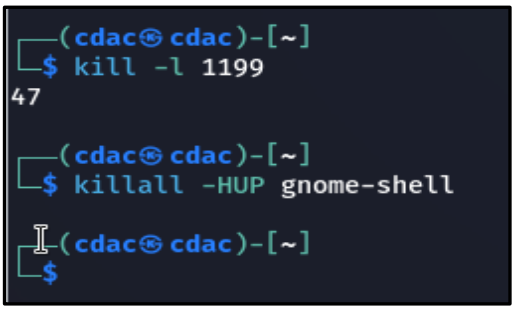
The kill command can send a kill signal to any process to end it

To list process id, execute: **top**

```
top - 21:19:29 up  4:05,  1 user,  load average: 0.33, 0.11, 0.04
Tasks: 169 total,   1 running, 168 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.8 us,  0.1 sy,  0.0 ni, 99.1 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :   3863.7 total,   2002.4 free,   1025.4 used,    836.0 buff/cache
MiB Swap:    975.0 total,    975.0 free,      0.0 used.   2597.7 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
   1199 cdac      20   0 5096332 413356 137376 S   2.7  10.4   4:01.35 gnome-shell
   1018 cdac      20   0  879660  98152  57480 S   0.3   2.5   0:37.97 Xorg
```

To end the process with PID 1199, execute: **kill -l 1199**



18. Tail

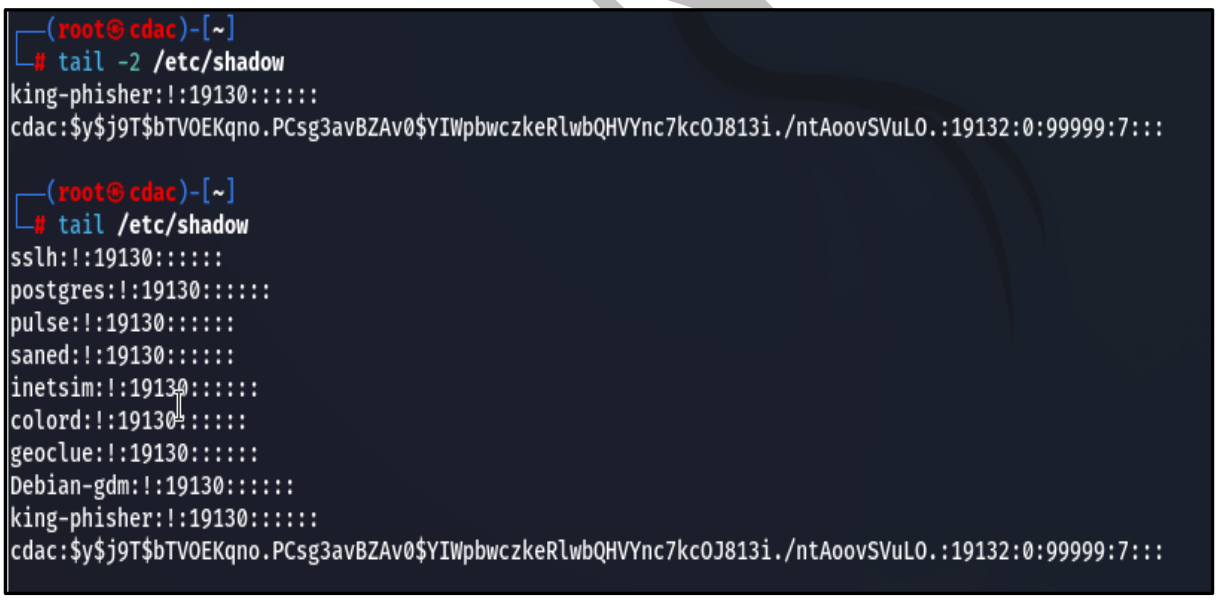
Syntax: #tail /etc/shadow

The tail command, as the name implies, prints the last N lines of the given input file. By default, it prints the last 10 lines of the specified files.

Syntax: #tail -2 /etc/shadow

Tail -2 shows the last 2 lines of the shadow file which stores passwords in encrypted form.

Note: shadow is a system file, so the command must be executed in a root shell.



19. HEAD

Syntax: $head /proc/cpuinfo

The head command, as the name implies, prints the top N lines of the given input file. By default, it prints the first 10 lines of the specified files.

```
┌──(cdac㉿cdac)-[~]
└─$ head /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 15
model           : 6
model name      : Intel(R) Xeon(R) Silver 4208 CPU @ 2.10GHz
stepping        : 1
microcode       : 0x1
cpu MHz         : 2095.078
cache size      : 16384 KB
physical id     : 0
```

20. ifconfig

This command will show you the IP address of your system. To use the ifconfig command, the net-tools package must be installed in Linux.

Syntax: $ifconfig

inet addr is followed by IP address in the ifconfig command's output

```
┌──(cdac㉿cdac)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.31.101.173  netmask 255.255.252.0  broadcast 172.31.103.255
        inet6 fe80::526b:8dff:fee4:f27b  prefixlen 64  scopeid 0x20<link>
        ether 50:6b:8d:e4:f2:7b  txqueuelen 1000  (Ethernet)
        RX packets 162552  bytes 17261199 (16.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 535  bytes 115200 (112.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 32  bytes 1920 (1.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 1920 (1.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

21. To activate/deactivate the given interface

ifconfig command can also be used to activate and deactivate the given interface.

Syntax: $ifconfig eth0 up

Syntax: $ifconfig eth0 down

In case, there is an error “ifconfig: command not found”, Then execute the following command to install ifconfig.

For Debian, Ubuntu, and related Linux distributions

sudo apt-get install net-tools

For CentOS or RPM (RedHat Package Manager) based Linux: **yum install net-tools**

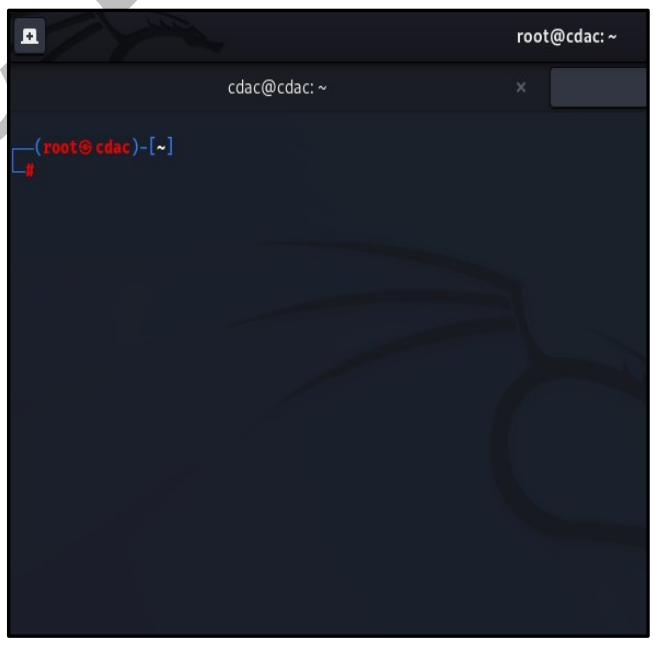
22. clear

Syntax: $clear

clear is a standard Unix computer operating system command that is used to clear the terminal screen

Before

After



23. Locate

locate command in Linux is used to find the files by name

Syntax: $locate chage

In the above command, locate not only found the chage command, but it also found the chage command and a variety of man pages associated with chage for different languages. The

locate command looks all over your filesystem, not just in directories that contain commands.

```
┌──(root㉿cdac)-[~]
└─# locate chage
/usr/bin/chage
/usr/share/bash-completion/completions/chage
/usr/share/man/de/man1/chage.1.gz
/usr/share/man/fr/man1/chage.1.gz
/usr/share/man/it/man1/chage.1.gz
/usr/share/man/ja/man1/chage.1.gz
/usr/share/man/man1/chage.1.gz
/usr/share/man/pl/man1/chage.1.gz
/usr/share/man/ru/man1/chage.1.gz
/usr/share/man/sv/man1/chage.1.gz
/usr/share/man/tr/man1/chage.1.gz
/usr/share/man/zh_CN/man1/chage.1.gz

┌──(root㉿cdac)-[~]
└─#
```

Syntax: $locate “*.html” -n 20

In the above command. Locate finds all the files with html extension showing only 20 lines in output.

```
┌──(root㉿cdac)-[~]
└─# locate "*.html" -n 20
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/404.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/500.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/actions.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/app_index.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/app_list.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/base.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/base_site.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/change_form.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/change_form_object_tools.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/change_list.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/change_list_object_tools.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/change_list_results.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/date_hierarchy.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/delete_confirmation.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/delete_selected_confirmation.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/filter.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/index.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/invalid_setup.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/login.html
/usr/lib/python3/dist-packages/django/contrib/admin/templates/admin/nav_sidebar.html

┌──(root㉿cdac)-[~]
└─#
```

24. Netstat

You can use the netstat command to view the list of programs (including Apache) with TCP ports in the LISTEN state

Syntax: $netstat -ntlp

The output from netstat (which was shortened to fit here) indicates that an instance of the python process with a process ID of 6278 is listening (as indicated by the LISTEN state) for

connections to any local IP address (indicated by:::80) on port 80 (the standard HTTP port). If a different program is listening to port 80, it is shown there.

```
┌──(cdac㉿cdac)-[~]
└─$ netstat -ntlp
(Not all processes could be identified, non-owned process info
 will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      6278/python3
┌──(cdac㉿cdac)-[~]
└─$
```

25. ZIP

Zip command is used both in *Unix Operating System* and as well as in *Windows Operating System*.

Syntax: $zip file.zip file1.txt file2.txt file3.txt file4.txt file5.txt

It will compress all the files provided in the argument in a zip format file.

```
┌──(cdac㉿cdac)-[~]
└─$ zip fie.zip file1.txt file2.txt file3.txt file4.txt file5.txt
  adding: file1.txt (stored 0%)
  adding: file2.txt (stored 0%)
  adding: file3.txt (stored 0%)
  adding: file4.txt (stored 0%)
  adding: file5.txt (stored 0%)

┌──(cdac㉿cdac)-[~]
└─$
```

26. Unzip

The default behavior of unzip command (with no options) is to extract into the current directory (and sub-directories below it) all files from the specified ZIP archive.

Syntax: $unzip file.zip

```
┌──(cdac㉿cdac)-[~/Desktop]
└─$ unzip ../fie.zip
Archive:  ../fie.zip
 extracting: file1.txt
 extracting: file2.txt
 extracting: file3.txt
 extracting: file4.txt
 extracting: file5.txt
┌──(cdac㉿cdac)-[~/Desktop]
└─$
```

27. Tar

The Linux 'tar' stands for tape archive, which is used to create an Archive and extract the Archive files. Here, first, we have an archive file named file.tar.gz where tar.gz is the archive format created by tar.

Syntax: $tar xzvf file.tar.gz

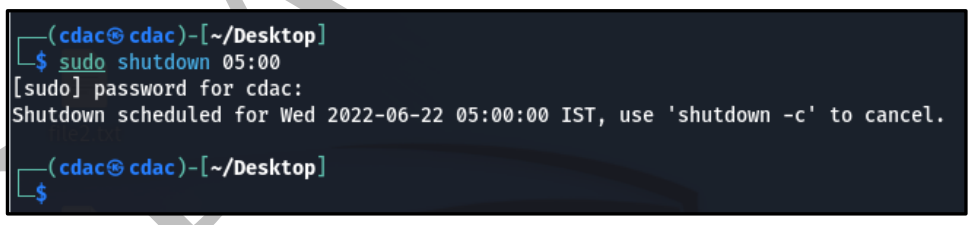
Then, file.tar.gz is decompressed with tar xvzf command.

28. Shutdown

You can shut down the machine immediately, or schedule a shutdown using a 24-hour format using the shutdown command. It brings the system down in a secure way. When the shutdown is initiated, all logged-in users and processes are notified that the system is going down, and no further logins are allowed.

Command: $sudo shutdown 05:00
Only the root user can execute the shutdown command.

"Sudo shutdown 05:00" will shut down the system at a specified time.



29. Reboot

The Reboot command is used to restart or reboot the system. In a Linux system administration, there comes a need to restart the server after the completion of some network and other major updates. reboot command may be used for this purpose.

Syntax: $reboot