

Fundamental of Linux Operating System

Facilitator's Profile



Aminu Idris

BSc. Ed, CCNA, CompTIA Security+, CEH, OSCP, CISSP, CISM, MPCSEAN

Head of Digital Transformation & Cybersecurity

Nasarawa State Information Technology and

Digital Economy Agency

Founder ICDFA

aminu.idris@nasidea.na.gov.ng

Overview

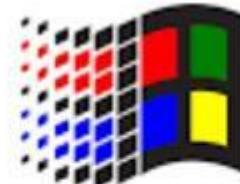
- Introduction to OS and file Systems
- Linux for digital forensics - Good and Bad
- Virtual file system
- File structure
- Path and path variable
- Linux commands
 - Create folders and files, File copy & deletion, Search for information, Networks, Create a batch file, Update/install software



Android



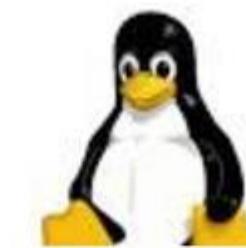
Windows 10



Microsoft
Windows



Ubuntu



Linux

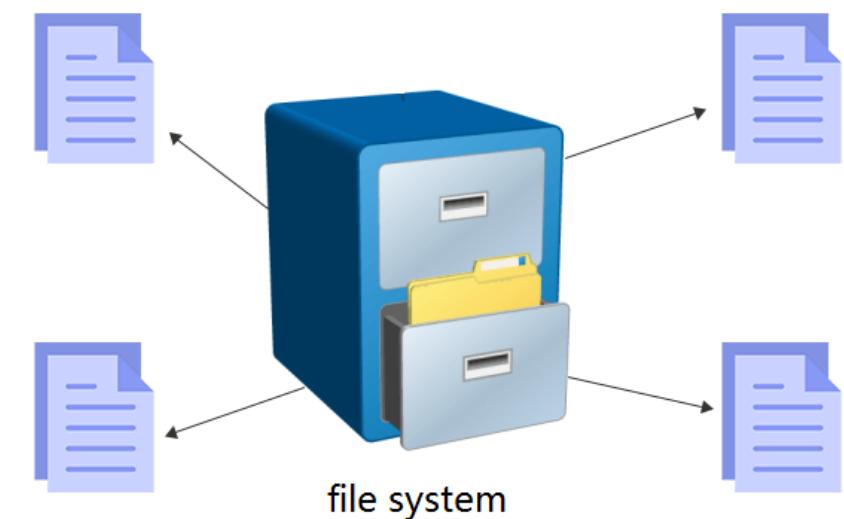


macOS



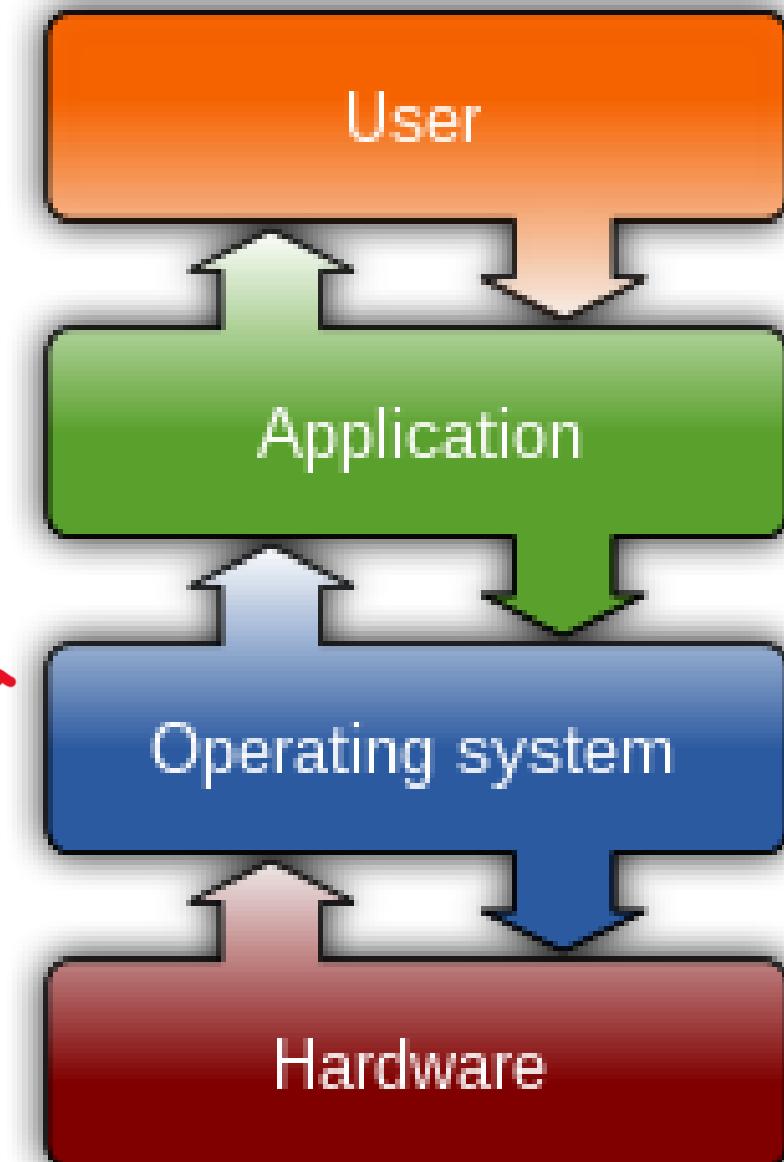
Chrome OS

OS and File Systems



Common features of OS

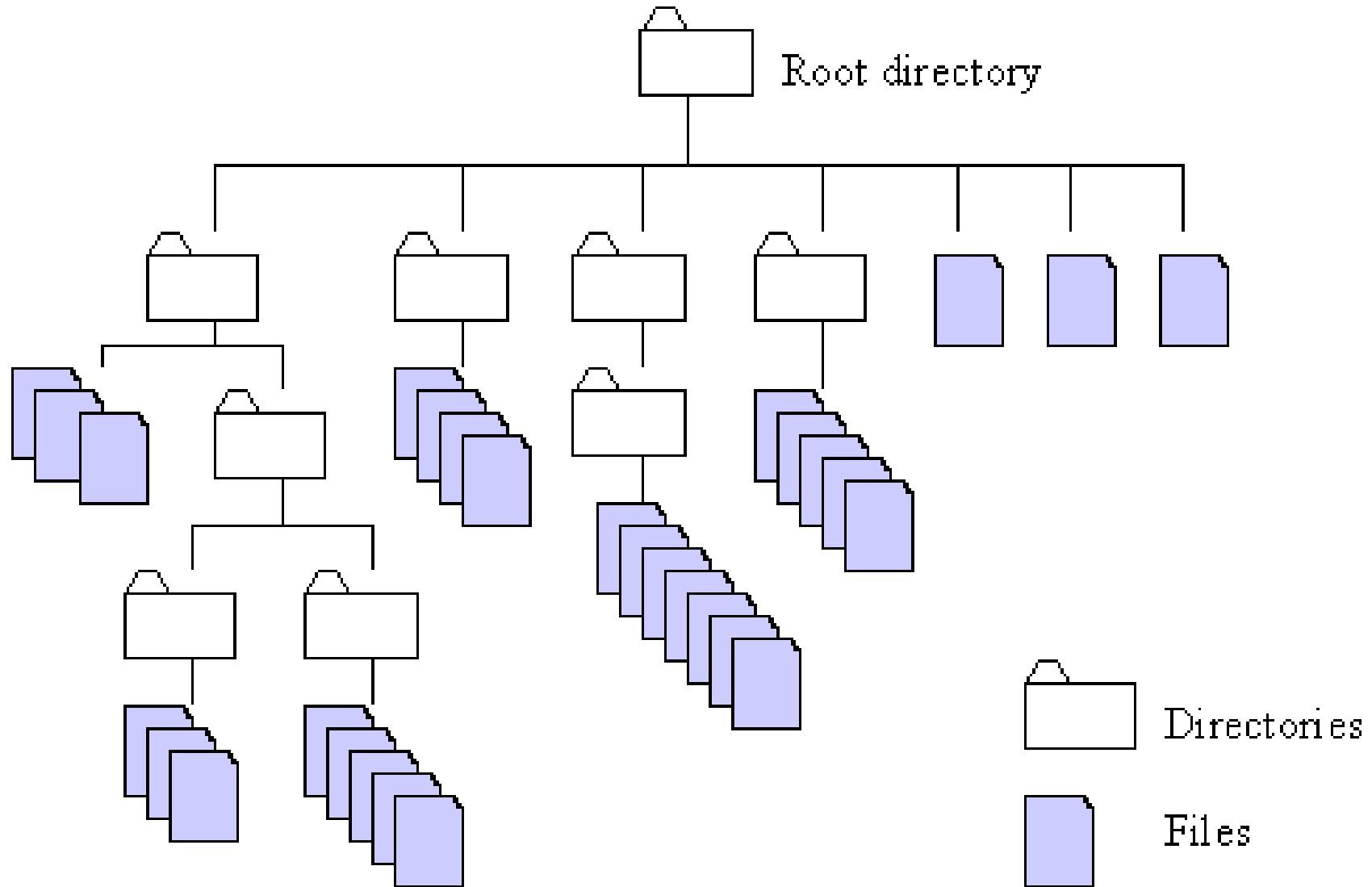
- Process management
- Memory management
- File system
- Device drivers
- Networking
- Security
- I/O



Files and File Systems

- A file
 - is a collection of correlated information
 - information is recorded on secondary or non-volatile storage like magnetic disks, optical disks, and tapes.
- A file system
 - defines how files are named, stored, and retrieved from a storage device.





File systems used by operating systems

- By Unix and Unix-like operating systems

- Linux: XFS, JFS, and btrfs.

- Solaris

- macOS:

- Hierarchical File System (HFS) + : No support for dates beyond February 6, 2040

- By Microsoft Windows

- FAT: File Allocation Table

- NTFS: New Technology File System



The logo for APFS (Apple File System), consisting of the acronym "APFS" in a bold, black, sans-serif font above the text "Apple File System".



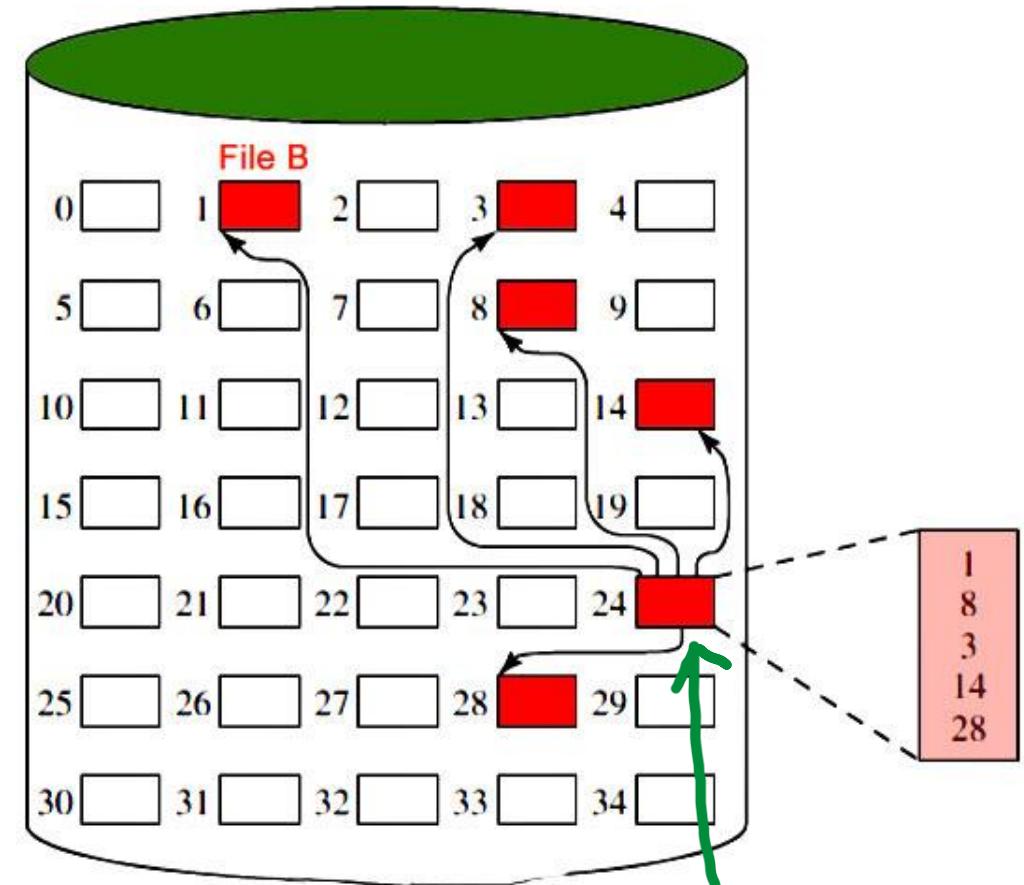
How to manage storage boxes?

Assumption: One person rents one or multiple boxes.

Name	Boxes ID
Frank	1,2,3
...	...



Storage Unit	File Storage Unit
Box	Sector
Person rents boxes	File uses sectors

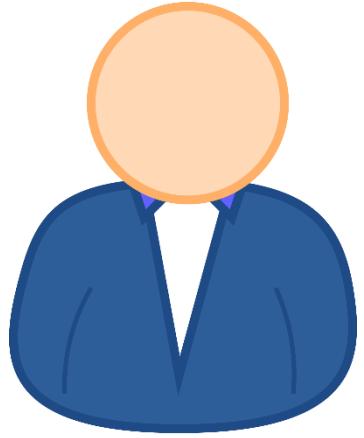


File allocation table

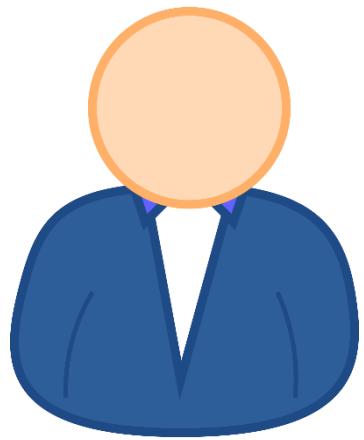
File name	Index block
•••	•••
File B	24
•••	•••

The simplest file system: FAT
(think about an index card)

Virtual file system (VFS)



Without an interpreter

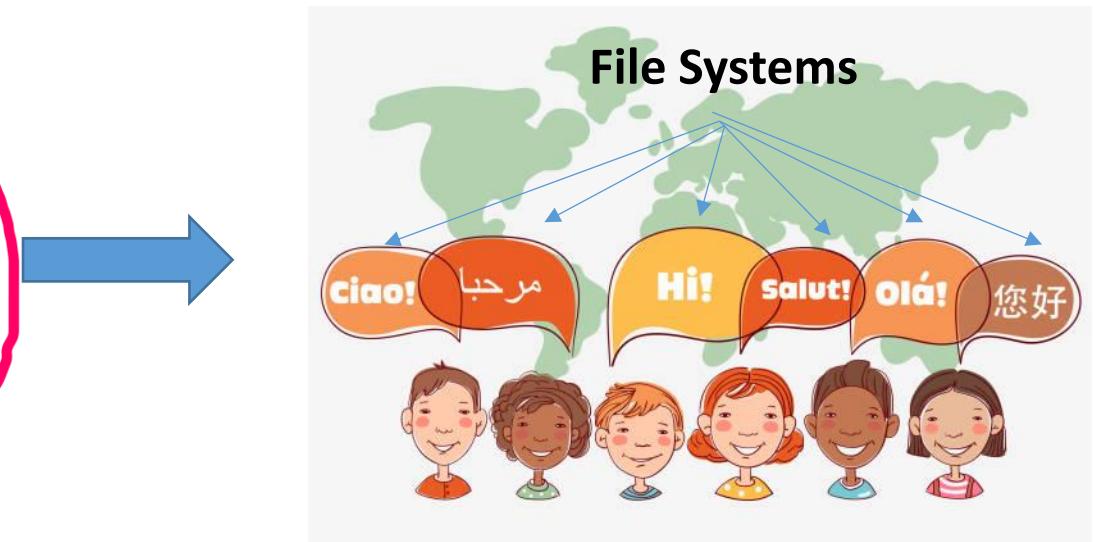


With an interpreter



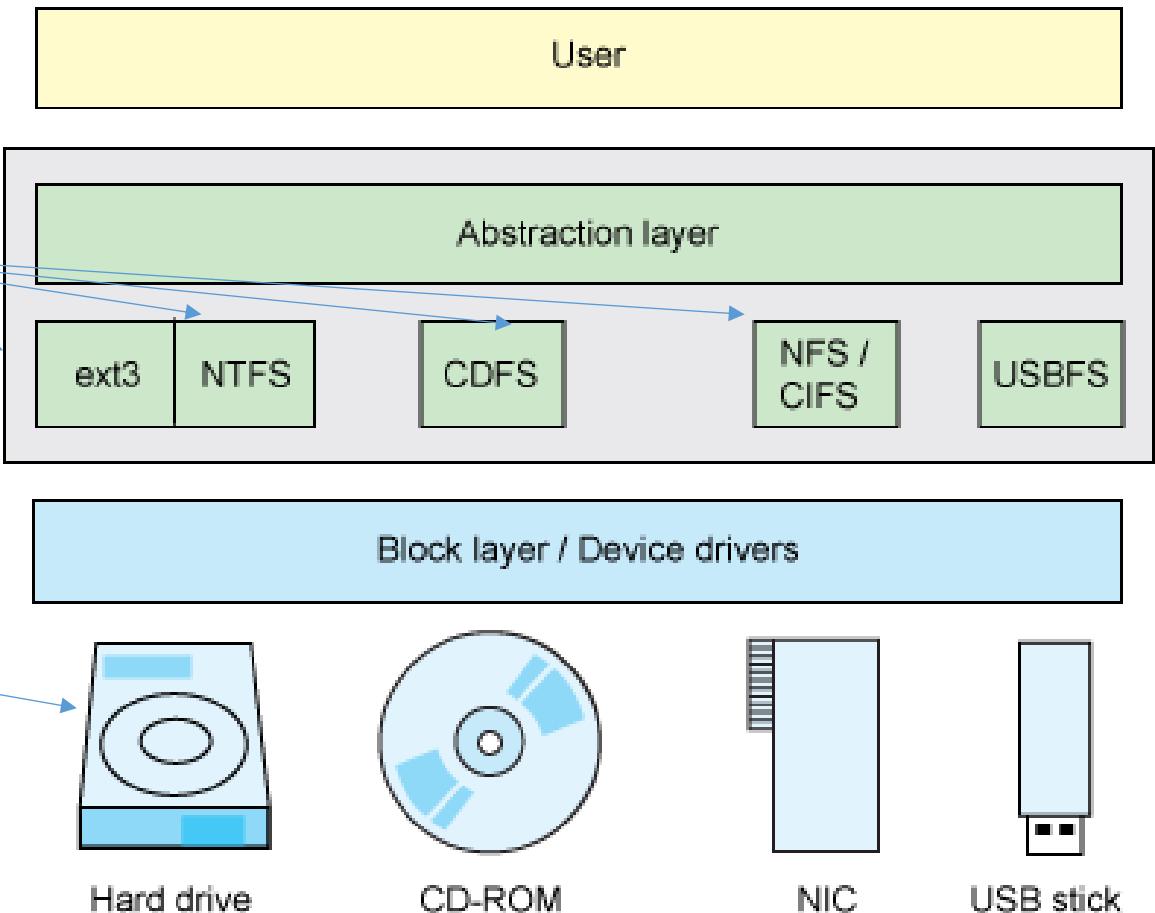
Google
Translate

Virtual file system



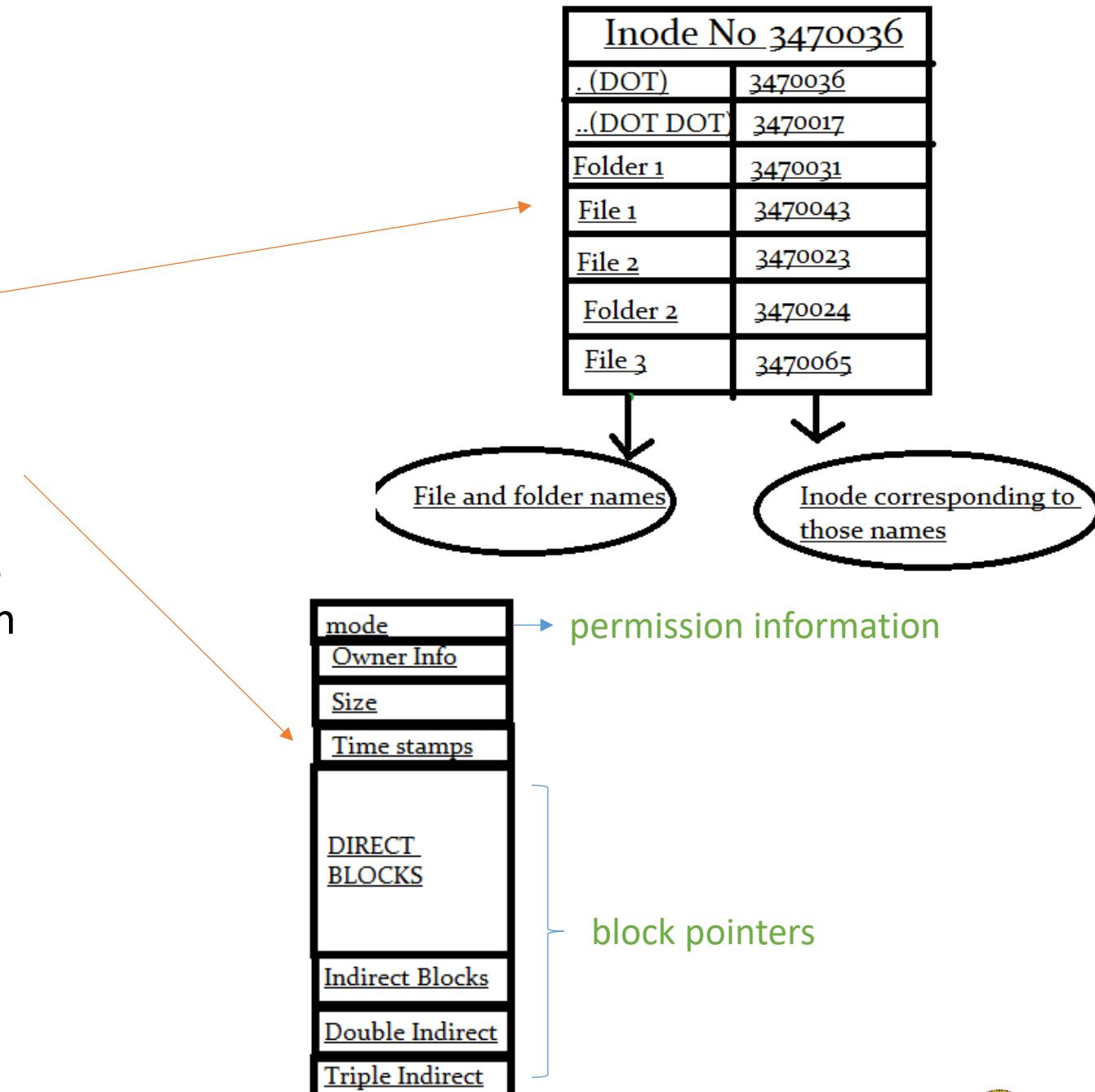
Linux is a VFS

- VFS allows client applications to access different types of **concrete** file systems in a uniform way
 - provides an abstract layer for upper-layer applications
 - The same techniques can be utilized to investigate different types of devices
- Each and everything in Linux is a **file** (**Everything appears somewhere in the filesystem**)
 - file, directory, hard disks, CD/DVD, NIC, USB
 - devices can be represented as file-like objects under `/dev/` filesystem.
- OS recognizes files by
 - inode (index node)



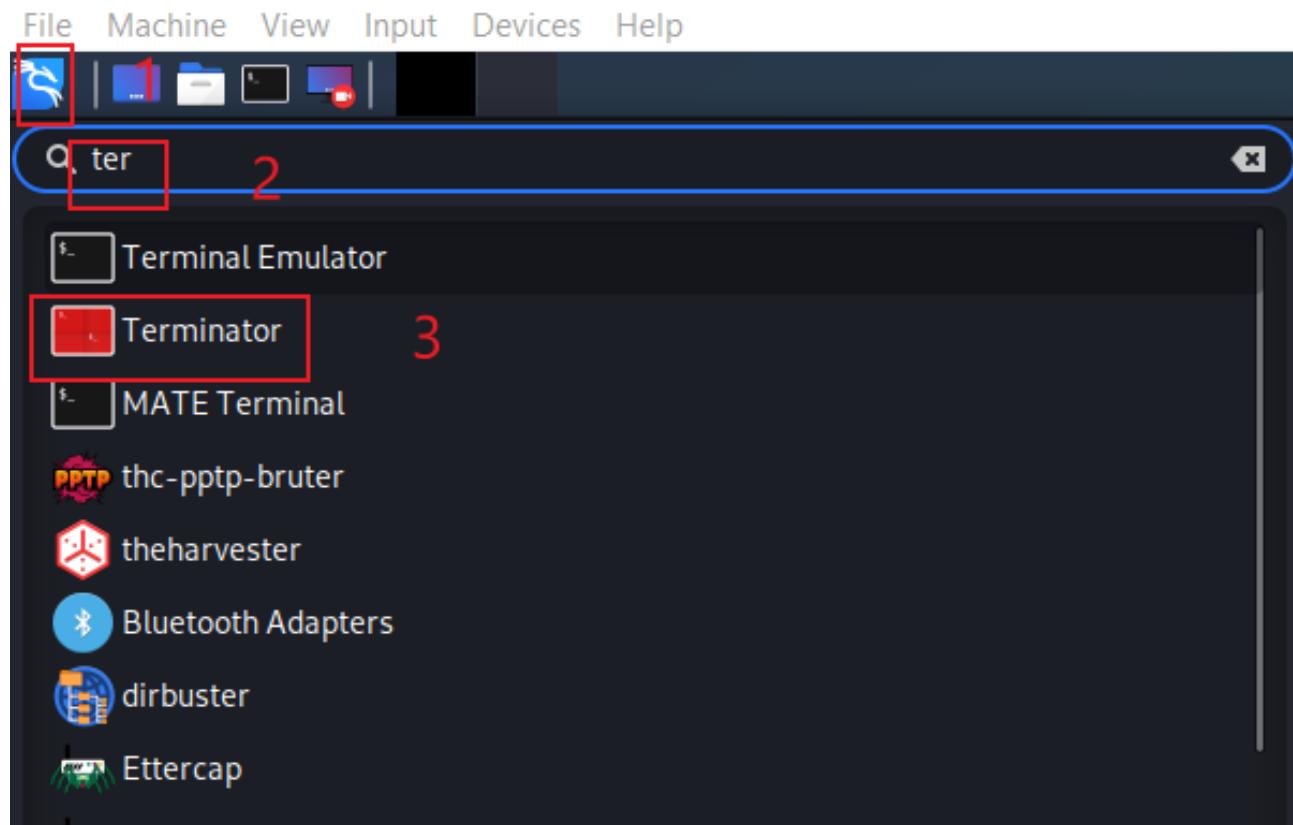
What is inode?

- inode is an ID of a file/folder
 - unique number
- inode is a data structure of a file
 - You store your information in a file, and the operating system stores the information/metadata about a file in an inode
- inode points to the content of a file
- How to access to a file?
 - inode ID
 - file name=>inode ID



Open a Linux Terminator

Kali-Linux-2020.4-vbox-amd64 (Snapshot 1) [Running] - Oracle VM VirtualBox



same as cmd



File inode

```
(kali㉿kali)-[~]
$ ls
Desktop    Downloads    Marvic_air   Pictures   Public      UB-730-Tools
Documents  hacking_case Music       pixel3     Templates   Videos

(kali㉿kali)-[~]
$ ls -ia
401164  .
393217  ..
401192  .bash_history
401174  .bash_logout
401170  .bashrc
401176  .bashrc.original
531550  .cache
401180  .config
531551  Desktop
401178  .dmrc
660979  Documents
660975  Downloads
401173  .face
401175  .face.icon
401177  .gnupg
401370  hacking_case
401198  .ICEauthority
401472  .java
660983  .local
401498  Marvic_air
401206  .mozilla
660980  Music
401172  .zshrc
660981  Pictures
796226  pixel3
401171  .profile
660978  Public
401468  .razorsql
401455  .rs
401456  .sqlite_history
660977  Templates
661438  UB-730-Tools
401193  .vboxclient-clipboard.pid
401200  .vboxclient-display-svga-x11.pid
401199  .vboxclient-draganddrop.pid
401195  .vboxclient-seamless.pid
660982  Videos
402204  .viminfo
401203  .wget-hsts
401187  .Xauthority
401188  .xsession-errors
401191  .xsession-errors.old
401369  .zsh_aliases
402209  .zsh_history
```

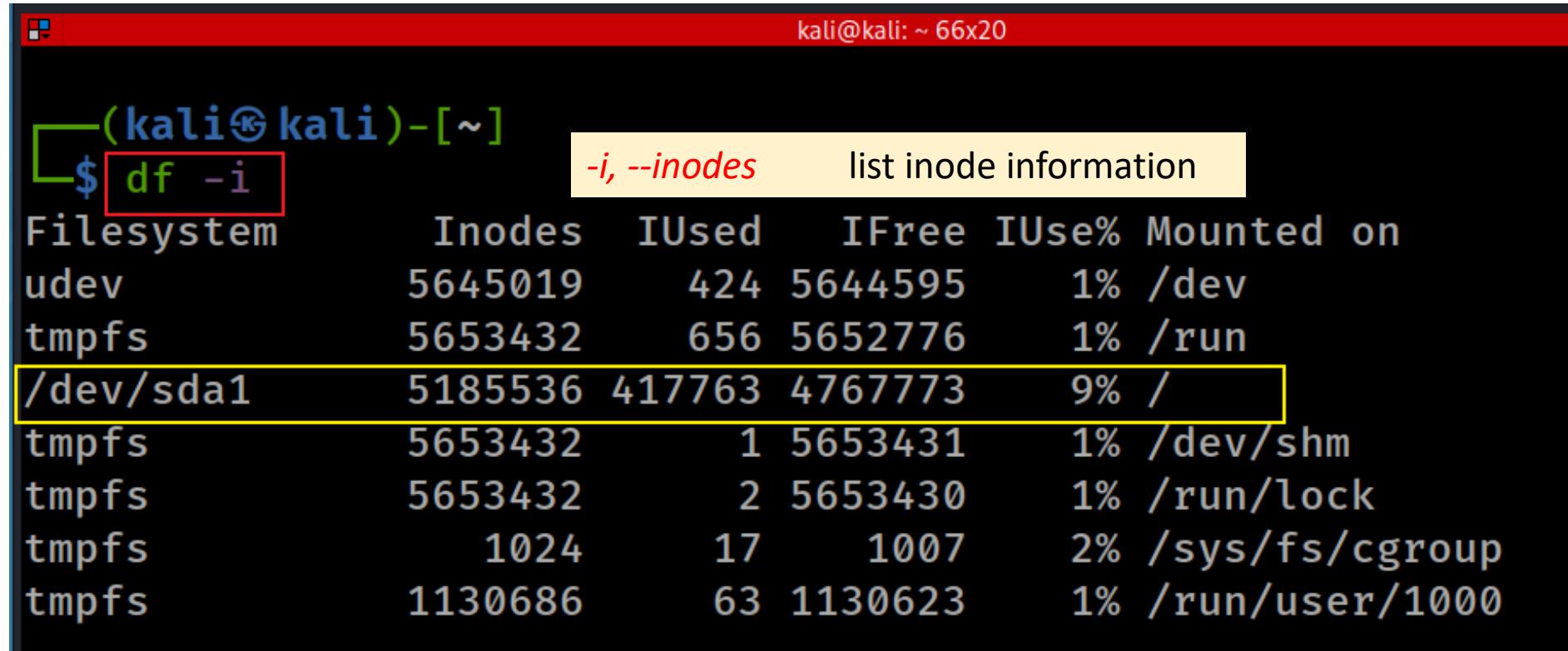
- **-i, --inode** print the index number of each file
- **-a, --all** do not ignore entries starting with . (hidden file)

```
(kali㉿kali)-[~]
$ ls --help
Usage: ls [OPTION]... [FILE]...
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.
-a, --all                                do not ignore entries starting with .
-A, --almost-all                           do not list implied . and ..
--author                                 with -l, print the author of each file
-b, --escape                               print C-style escapes for nongraphic characters
--block-size=SIZE                          with -l, scale sizes by SIZE when printing them;
                                         e.g., '--block-size=M'; see SIZE format below
-B, --ignore-backups                      do not list implied entries ending with ~
-c                                         with -lt: sort by, and show, ctime (time of last
                                         modification of file status information);
                                         with -l: show ctime and sort by name;
                                         otherwise: sort by ctime, newest first
-C                                         list entries by columns
--color[=WHEN]                            colorize the output; WHEN can be 'always' (default
                                         if omitted), 'auto', or 'never'; more info below
-d, --directory                           list directories themselves, not their contents
-D, --dired                               generate output designed for Emacs' dired mode
-f                                         do not sort, enable -aU, disable -ls --color
-F, --classify                            append indicator (one of */=>@|) to entries
                                         likewise, except do not append '*'
                                         across -x, commas -m, horizontal -x, long -l,
                                         single-column -1, verbose -l, vertical -C
                                         like -l --time-style=full-iso
-g                                         like -l, but do not list owner
--group-directories-first                 group directories before files
```

Disk free information (*df*) inode

wmic logicaldisk get size, freespace, caption



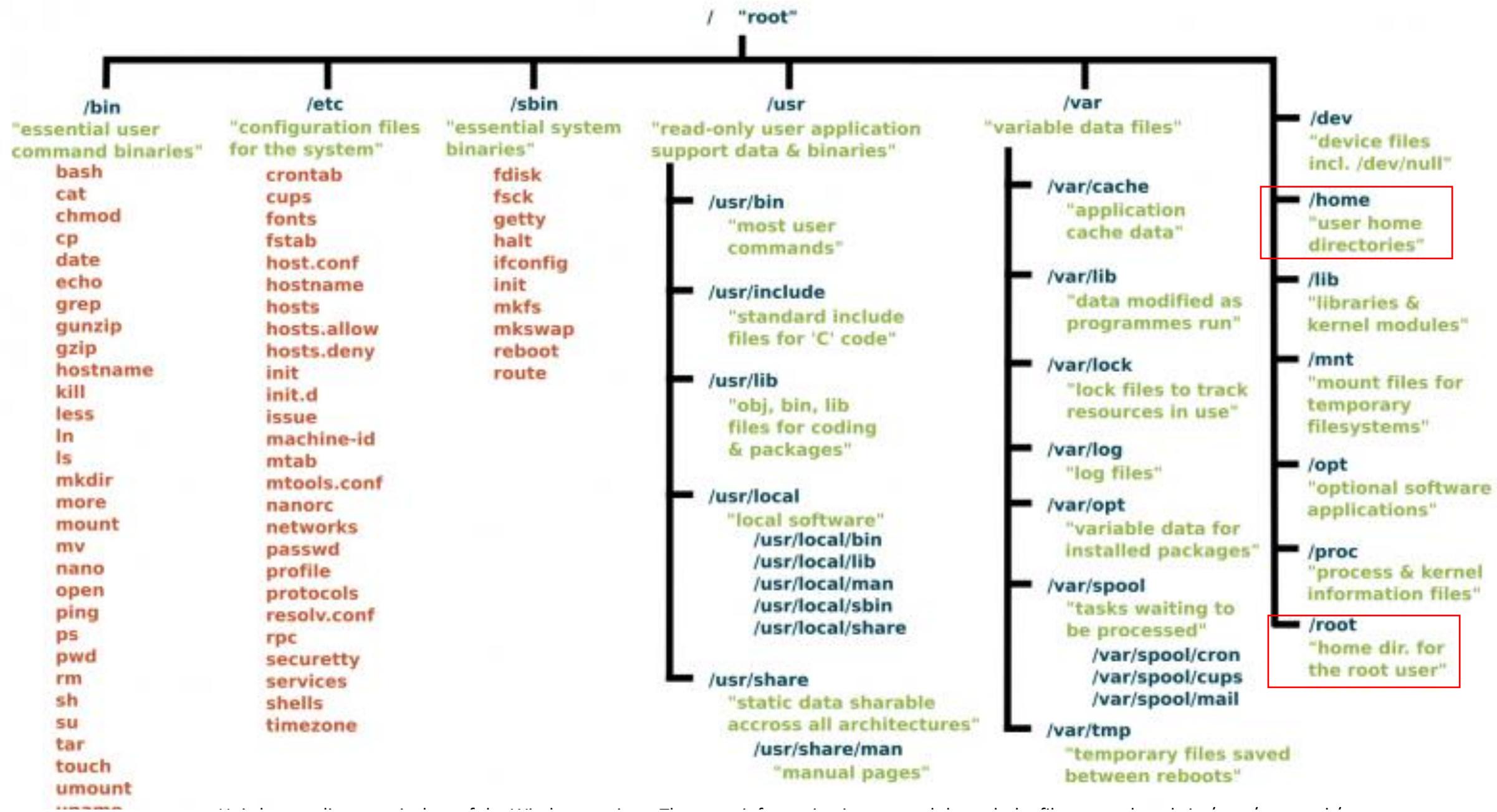
The screenshot shows a terminal window with a red header bar containing the text "kali@kali: ~ 66x20". The terminal prompt is "(kali㉿kali)-[~]". The command entered is "\$ df -i". A yellow box highlights the command. To the right of the command, the help text "-i, --inodes list inode information" is displayed. The output of the command is a table showing disk usage statistics for various filesystems. The table has columns: Filesystem, Inodes, IUsed, IFree, IUse%, and Mounted on. The row for the root filesystem, "/dev/sda1", is highlighted with a yellow box. The data for the root filesystem is: Inodes 5185536, IUsed 417763, IFree 4767773, IUse% 9%, Mounted on /. Other filesystems listed include udev, tmpfs, /dev/shm, /run/lock, /sys/fs/cgroup, and /run/user/1000.

Filesystem	Inodes	IUsed	IFree	IUse%	Mounted on
udev	5645019	424	5644595	1%	/dev
tmpfs	5653432	656	5652776	1%	/run
/dev/sda1	5185536	417763	4767773	9%	/
tmpfs	5653432	1	5653431	1%	/dev/shm
tmpfs	5653432	2	5653430	1%	/run/lock
tmpfs	1024	17	1007	2%	/sys/fs/cgroup
tmpfs	1130686	63	1130623	1%	/run/user/1000

Show human readable format

```
(kali㉿kali)-[~]
$ df -h
Filesystem      Size  Used  Avail Use% Mounted on
udev            22G   0     22G   0% /dev
tmpfs           4.4G  964K  4.4G  1% /run
/dev/sda1        78G  52G   23G  70% /
tmpfs           22G   0     22G   0% /dev/shm
tmpfs           5.0M  0     5.0M  0% /run/lock
tmpfs           4.0M  0     4.0M  0% /sys/fs/cgroup
tmpfs           4.4G  52K   4.4G  1% /run/user/1000
```

File Structure



Trash



File System



Home



RazorSQL



Seagate3T



157 GB
Volume

File Edit View Go Help



File System - File Manager

- □ ×

DEVICES

File System

157 GB Volume

Seagate3T

PLACES

kali

Desktop

Trash

Documents

Music

Pictures

Videos

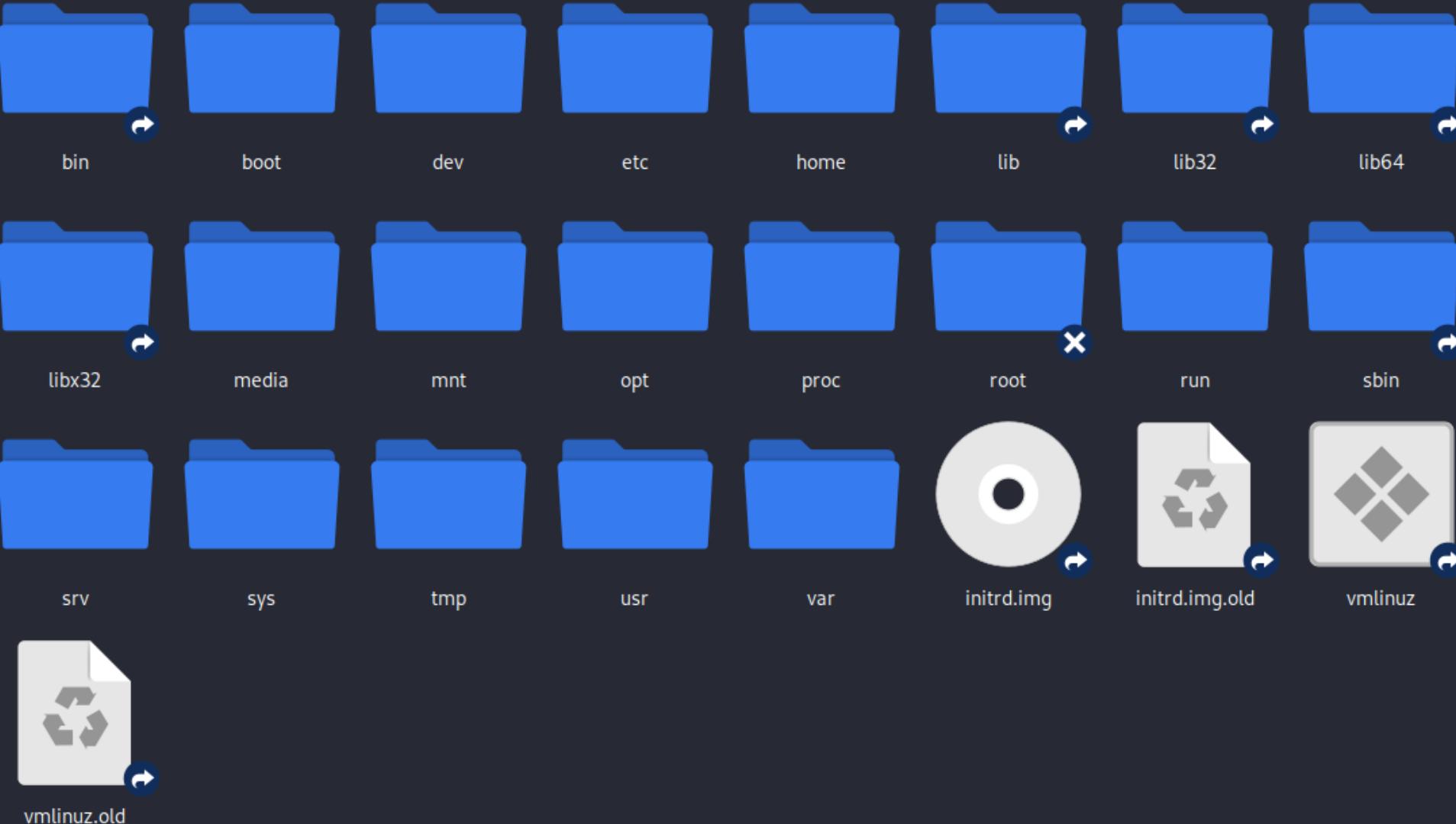
Downloads

NETWORK

Browse Network

File System - File Manager

C



Show Linux file system from root /

Check where you are (impact all the commands you entered)  cd, ls
Kali㉿kali: ~ 92x38

```
(kali㉿kali)-[~]
$ pwd      show current working directory
/home/kali

(kali㉿kali)-[~]
$ ls /      "/" mean root
bin  etc          initrd.img.old  lib64      media    proc    sbin    tmp     vmlinuz
boot home         lib           libx32     mnt      root    srv     usr     vmlinuz.old
dev   initrd.img  lib32        lost+found  opt      run     sys     var
```

Clear screen

```
(kali㉿kali)-[~]
$ clear
```

Show details of a file/directory

```
└──(kali㉿kali)-[~]
$ ls -l /
total 68
lrwxrwxrwx  1 root root   7 Nov 17  2020 bin  -> usr/bin
drwxr-xr-x  3 root root 4096 Nov 17  2020 boot
drwxr-xr-x 17 root root 3360 Jun 27 15:41 dev
drwxr-xr-x 157 root root 12288 Jun 27 15:29 etc
drwxr-xr-x  3 root root 4096 Nov 17  2020 home
lrwxrwxrwx  1 root root   33 Nov 17  2020 initrd.img -> boot/initrd.img-5.9.0-kali1-amd64
lrwxrwxrwx  1 root root   33 Nov 17  2020 initrd.img.old -> boot/initrd.img-5.9.0-kali1-amd64
lrwxrwxrwx  1 root root   7 Nov 17  2020 lib  -> usr/lib
lrwxrwxrwx  1 root root   9 Nov 17  2020 lib32 -> usr/lib32
lrwxrwxrwx  1 root root   9 Nov 17  2020 lib64 -> usr/lib64
lrwxrwxrwx  1 root root  10 Nov 17  2020 libx32 -> usr/libx32
drwx-----  2 root root 16384 Nov 17  2020 lost+found
.
```

Show Linux file system from root with *tree*

```
(kali㉿kali)-[~]
$ tree -L 1 /
only show Level 1
/
bin    -> usr/bin
boot
dev
etc
home
initrd.img    -> boot/initrd.img-5.9.0-kali1-amd64
initrd.img.old -> boot/initrd.img-5.9.0-kali1-amd64
lib    -> usr/lib
lib32  -> usr/lib32
lib64  -> usr/lib64
libx32 -> usr/libx32
lost+found
media
mnt
opt
proc
```

Remember use following command for help
`tree --help`

Show current working directory (*pwd*) with *tree*

```
(kali㉿kali)-[~]
$ pwd
/home/kali ←

(kali㉿kali)-[~]
$ tree -L 1
.
├── Desktop
├── Documents
├── Downloads
├── hacking_case
├── Marvic_air
├── Music
├── Pictures
├── pixel3
├── Public
├── Templates
└── UB-730-Tools
└── Videos

12 directories, 0 files
```

```
(kali㉿kali)-[~]
$ tree -L 2 | more ←
.
├── Desktop
│   └── razorsql.desktop
├── Documents
├── Downloads
│   ├── JLECmd.zip
│   ├── razorsql
│   ├── razorsql9_3_3_linux_x64.zip
│   ├── RegRipper30-apt-git-Install.sh
│   └── wine-mono-5.0.0-x86.msi
└── hacking_case
    ├── ip.txt
    ├── mac.txt
    ├── mirc.ini
    ├── NTUSER_Evil.DAT
    └── RegRipper2.8
        └── SAM
```

Linux commands are files!

Display `ls` command under `/bin`

```
(kali㉿kali)-[~]
$ ls -l /bin/ls
-rwxr-xr-x 1 root root 147176 Sep 24 2020 /bin/ls ←
```

```
(kali㉿kali)-[~]
$ █
```

Why can `ls` be executed in any folder? => Need to understand path (see next slide)

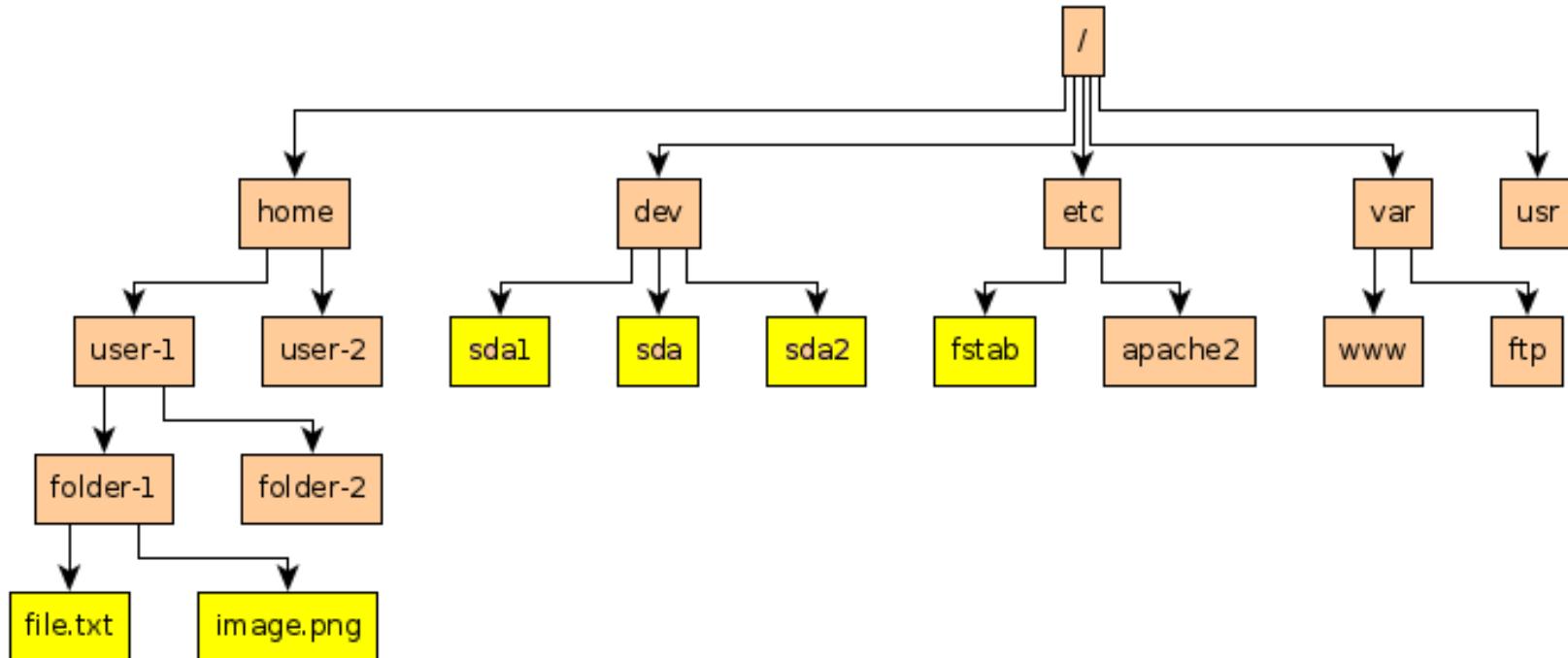
Path and Path variable

What is path in a file system?



- A path is a **hierarchical representation of the location** (address) of a file or directory within the file system's directory structure.
- A path describes the route or sequence of directories (folders) you must navigate through to reach a specific file or directory.
- Paths are used to uniquely identify and access files and directories on a computer or storage device.

Example of paths



- Absolute path to *file.txt*: */home/user-1/folder-1/file.txt*
- Relative path to *file.txt* (Under *folder-2*): *../folder-1/file.txt*

Switch to the parent path using relative path



cd ..

```
kali㉿kali:[~]                               kali@kali: /home 82x25
└─$ pwd
/home/kali ←

└─$ cd ..
.. means parent folder

└─$ pwd
/home ←
```

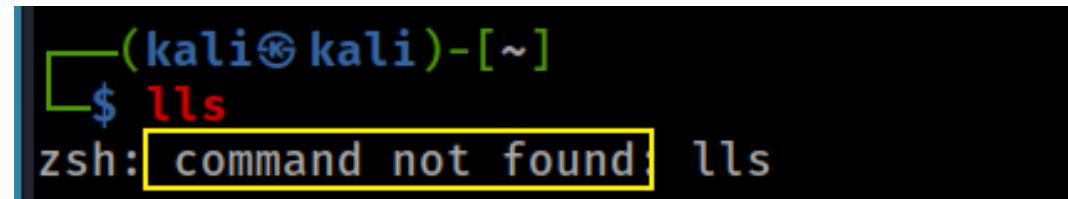
Path variable



```
kali㉿kali: ~ 82x25
└─$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin /usr/bin:/sbin:/bin:/usr/local/games:/usr
/games
```

The terminal window shows the command `echo $PATH` being run. The output displays the system's path environment variable, which includes several directories: `/usr/local/sbin`, `/usr/local/bin`, `/usr/sbin`, `/usr/bin`, `/sbin`, `/bin`, `/usr/local/games`, and `/usr/games`. The directory `/usr/bin` is highlighted with a yellow box.

- How does OS execute a command, e.g., `ls`?
 - search the command in the current directory
 - if can't find it, search for the command in each path defined in the path variable
 - if can't find it, OS throws the *command not found* error message



```
└─$ ll
zsh: command not found: ll
```

The terminal window shows the command `ll` being run. The output indicates that the command was not found, with the error message `zsh: command not found: ll`. The word `ll` is highlighted with a yellow box.



path

Adding a new path

```
(kali㉿kali)-[~]
$ export PATH=/some/new/path:$PATH

(kali㉿kali)-[~]      new added path
$ echo $PATH
/some/new/path:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/games:/usr/games
```



```
setx path "%path%;C:\Program Files\Java\jdk1.8.0_202\bin" -m
```

Persisting the new path

```
(kali㉿kali)-[~]
$ ls -a
.
..
.bash_history
.bash_logout
.bashrc
.bashrc.original
.cache
.config
Desktop
.dmrc
Documents
Downloads
.face
.face.icon
.gnupg
.hacking_case
.ICEauthority
.java
.local
.Marvic_air
.mozilla
.Music
.Pictures
.pixel3
.profile
.Public
.razorsql
.rs
.sqlite_history
.Templates
.UB-730-Tools
.vboxclient-clipboard.pid
.vboxclient-display-svga-x11.pid
.vboxclient-draganddrop.pid
.vboxclient-seamless.pid
.Videos
.viminfo
.wget-hsts
.Xauthority
.xsession-errors
.xsession-errors.old
.zsh_aliases
.zsh_history
.zshrc
```

← add to this file

we discuss the details later

Create folders and files

Create folders

```
root@kali:~# mkdir myfolder
root@kali:~# cd myfolder/
root@kali:~/myfolder# mkdir mysubfolder
root@kali:~/myfolder# ls
mysubfolder
root@kali:~/myfolder#
```



Delete folders



rmdir

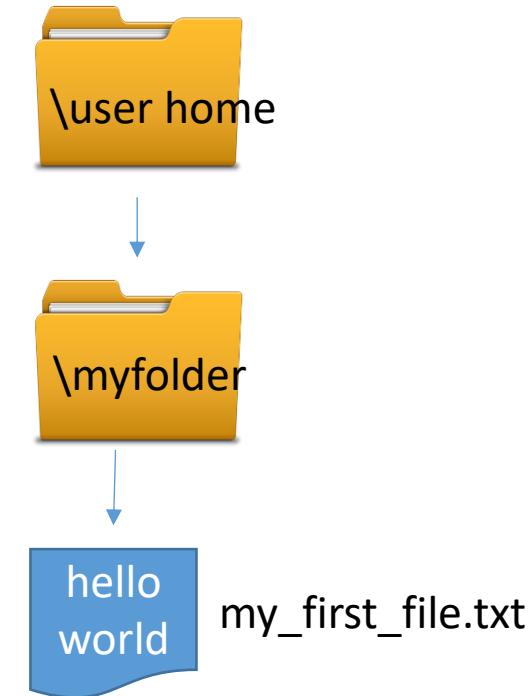
```
root@kali:~/myfolder# ls
mysubfolder
root@kali:~/myfolder# rmdir mysubfolder/
root@kali:~/myfolder# ls
root@kali:~/myfolder#
```

Create a new text file

```
root@kali:~/myfolder# echo "hello world" > my first file.txt  
root@kali:~/myfolder# ls  
my_first_file.txt  
root@kali:~/myfolder#
```

Show the content of the file

```
root@kali:~/myfolder# cat my first file.txt  
hello world  
root@kali:~/myfolder#
```



Show the content of the file

```
root@kali:~/myfolder# leafpad my_first_file.txt
```

or nano command

The image shows a terminal window at the top with the command `leafpad my_first_file.txt` entered. A red box highlights this command. To the right, a file editor window titled "my_first_file.txt" is open, displaying the text "hello world". A green box highlights the word "hello". The terminal window has a black background with white text, and the file editor window has a light gray background.

File Copy & Deletion

Copy (**cp**) a file

```
root@kali:~/myfolder# cp my_first_file.txt my_first_file_dup.txt
root@kali:~/myfolder# ls
my_first_file_dup.txt  my_first_file.txt
root@kali:~/myfolder#
```

Copy (**cp**) a file to a different location

```
root@kali:~/myfolder# cp my_first_file.txt /var/www/html  
root@kali:~/myfolder# ls /var/www/html/my_first_file.txt  
/var/www/html/my_first_file.txt  
root@kali:~/myfolder#
```

How to copy a folder? **-r** recursive

- Basic syntax: **cp -r source_folder destination_folder**
- Example: **cp -r ~/myfolder ~/myfolder_copy**

Remove a file: rm

```
root@kali:~/myfolder# ls  
my_first_file_dup.txt my_first_file.txt  
root@kali:~/myfolder# rm my_first_file_dup.txt  
root@kali:~/myfolder# ls  
my_first_file.txt  
root@kali:~/myfolder# 
```

Rename a file (mv)

 move/ copy and ren

```
root@kali:~/myfolder# ls  
my_first_file.txt  
root@kali:~/myfolder# mv my_first_file.txt renamed_file.txt  
root@kali:~/myfolder# ls  
renamed_file.txt  
root@kali:~/myfolder#
```

Search for information

Search for a string in a text file

grep search

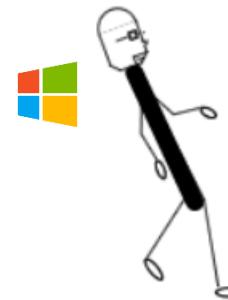
```
(kali㉿kali)-[~/myfolder]
$ ls
my_first_file.txt ←

(kali㉿kali)-[~/myfolder]
$ cat my_first_file.txt
hello world ←

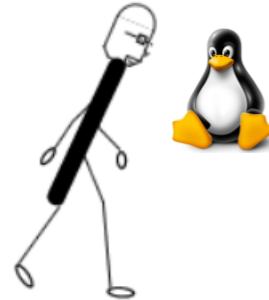
(kali㉿kali)-[~/myfolder]
$ grep "hello" my_first_file.txt
hello world ←
```

Show line number while displaying the output

```
(kali㉿kali)-[~/myfolder]
$ grep -n "hello" my_first_file.txt
1:hello world ←
```



backslash



forward slash



type/findstr /c:"error" log.txt

Search for file names that contains the given string/pattern

grep help

```
(kali㉿kali)-[~/myfolder]
$ grep --help
Usage: grep [OPTION]... PATTERNS [FILE]...
Search for PATTERNS in each FILE.
Example: grep -i 'hello world' menu.h main.c
PATTERNS can contain multiple patterns separated by newlines.

          . . . . .
-r, --recursive      like --directories=recurse
-R, --dereference-recursive  likewise, but follow all symlinks
--include=GLOB       search only files that match GLOB (a file pattern)
--exclude=GLOB       skip files that match GLOB
--exclude-from=FILE  skip files that match any file pattern from FILE
--exclude-dir=GLOB   skip directories that match GLOB
-L, --files-without-match  print only names of FILES with no selected lines
-l, --files-with-matches  print only names of FILES with selected lines
-c, --count           print only a count of selected lines per FILE
```

Search for file contains “hello”

```
(kali㉿kali)-[~/myfolder]
$ grep -l "hello" *
my_first_file.txt
```

Networking

Test network connection using domain name

```
root@kali:~# ping google.com
PING google.com (216.58.219.206) 56(84) bytes of data.
64 bytes from lga25s40-in-f14.1e100.net (216.58.219.206): icmp_seq=1 ttl=56 time=10.9 ms
64 bytes from lga25s40-in-f14.1e100.net (216.58.219.206): icmp_seq=2 ttl=56 time=11.4 ms
64 bytes from lga25s40-in-f14.1e100.net (216.58.219.206): icmp_seq=3 ttl=56 time=12.9 ms
64 bytes from lga25s40-in-f14.1e100.net (216.58.219.206): icmp_seq=4 ttl=56 time=10.10 ms
64 bytes from lga25s40-in-f14.1e100.net (216.58.219.206): icmp_seq=5 ttl=56 time=12.2 ms
```

Test network connection using IP

```
root@kali:~# ping 216.58.219.206
PING 216.58.219.206 (216.58.219.206) 56(84) bytes of data.
64 bytes from 216.58.219.206: icmp_seq=1 ttl=56 time=11.1 ms
64 bytes from 216.58.219.206: icmp_seq=2 ttl=56 time=12.3 ms
64 bytes from 216.58.219.206: icmp_seq=3 ttl=56 time=12.4 ms
64 bytes from 216.58.219.206: icmp_seq=4 ttl=56 time=15.3 ms
^C
```

List all open TCP ports

```
-l, --listening      display listening server sockets  
-n, --numeric        port number (don't resolve names)  
-t, --tcp
```

```
student@kalit01: ~ 80x24  
└─(student㉿kalit01)─[~]  
$ netstat -lnt  
Active Internet connections (only servers)  
Proto Recv-Q Send-Q Local Address          Foreign Address        State  
tcp     0      0 127.0.0.1:5432            0.0.0.0:*              LISTEN  
tcp     0      0 0.0.0.0:80               0.0.0.0:*              LISTEN  
tcp     0      0 0.0.0.0:22               0.0.0.0:*              LISTEN  
tcp6    0      0 :::3389                ::::*                LISTEN  
tcp6    0      0 :::22      ssh (22) port is listening  ::::*                LISTEN  
tcp6    0      0 :::1:3350              ::::*                LISTEN  
tcp6    0      0 :::1:5432              ::::*                LISTEN
```

Open port 21 and verify the port is listening

```
nmap localhost -p 21
```

```
[student@kalit01]~]
```

```
$ netstat -lnt
```

Active Internet connections (only servers)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	127.0.0.1:5432	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:80	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:21	2. listening	LISTEN
tcp6	0	0	:::3389	:::*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN
tcp6	0	0	::1:3350	:::*	LISTEN
tcp6	0	0	::1:5432	:::*	LISTEN

```
[student@kalit01]~]
```

```
$
```

student@kalit01: ~ 80x6

```
[student@kalit01]~]
```

```
$ nc -lp 21
```

1. listen to port 21

Download a file

```
root@kali: ~/myfolder
File Edit View Search Terminal Help
root@kali:~# cd myfolder/
root@kali:~/myfolder# wget https://pbs.twimg.com/media/DulILzQXcAAkFMV.jpg
--2018-12-16 22:49:03-- https://pbs.twimg.com/media/DulILzQXcAAkFMV.jpg
Resolving pbs.twimg.com (pbs.twimg.com)... 72.21.91.70, 2606:2800:220:1410:4
89:141e:20bb:12f6
Connecting to pbs.twimg.com (pbs.twimg.com)|72.21.91.70|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 109010 (106K) [image/jpeg]
Saving to: 'DulILzQXcAAkFMV.jpg'

DulILzQXcAAkFMV.jpg      100%[=====] 106.46K  --.-KB/s
in 0.1s

2018-12-16 22:49:03 (894 KB/s) - 'DulILzQXcAAkFMV.jpg' saved [109010/109010]

root@kali:~/myfolder# ls
DulILzQXcAAkFMV.jpg  renamed_file.txt
root@kali:~/myfolder# display DulILzQXcAAkFMV.jpg
```



<https://pbs.twimg.com/media/DulILzQXcAAkFMV.jpg>

Create a script file



.bat

Create a simple script file

Create a script using leafpad

```
(kali㉿kali)-[~/myfolder]$ leafpad myFirstScript.sh
#!/bin/sh
File Edit Search Options Help
echo "Hello World" > myFile.txt
ls myFile.txt -l
```

- **shebang:** indicate the interpreter that should be used to execute the script or program that follows

Check permission of files

```
(kali㉿kali)-[~/myfolder]
$ ls -l
total 8
-rw-r--r-- 1 kali kali 12 Jun 27 22:19 my_first_file.txt
-rw-r--r-- 1 kali kali 48 Jun 27 23:01 myFirstScript.sh
```

Linux Permissions

	u	g	o
	7	5	4
access	r w x	r w x	r w x
binary	4 2 1	4 2 1	4 2 1
enabled	1 1 1	1 0 1	1 0 0
result	4 2 1	4 0 1	4 0 0
total	7	5	4

```
root@kali: ~/myfolder
root@kali: ~/myfolder 80x24
root@kali:~/myfolder# chmod 777 myFirstScript.sh
root@kali:~/myfolder# ls myFirstScript.sh -l
-rwxrwxrwx 1 root root 49 Mar 20 11:13 myFirstScript.sh
root@kali:~/myfolder#
```

```
root@kali: ~/myfolder
root@kali: ~/myfolder 80x24
root@kali:~/myfolder# ./myFirstScript.sh
-rw-r--r-- 1 root root 12 Mar 20 11:14 myFile.txt
root@kali:~/myfolder#
```

Execute a script file in a different folder

```
(kali㉿kali)-[~/myfolder]
$ cd ..
(kali㉿kali)-[~]
$ pwd
/home/kali
MUST include path
(kali㉿kali)-[~]
$ ./myfolder/myFirstScript.sh
-rw-r--r-- 1 kali kali 12 Jun 28 11:06 myFile.txt
```

Add a path to the path variable

```
(kali㉿kali)-[~]
$ pwd
/home/kali

(kali㉿kali)-[~]
$ ls ./myfolder -l
total 12
-rw-r--r-- 1 kali kali 12 Jun 27 23:06 myFile.txt
-rw-r--r-- 1 kali kali 12 Jun 27 22:19 my_first_file.txt
-rwxrwxrwx 1 kali kali 48 Jun 27 23:01 myFirstScript.sh

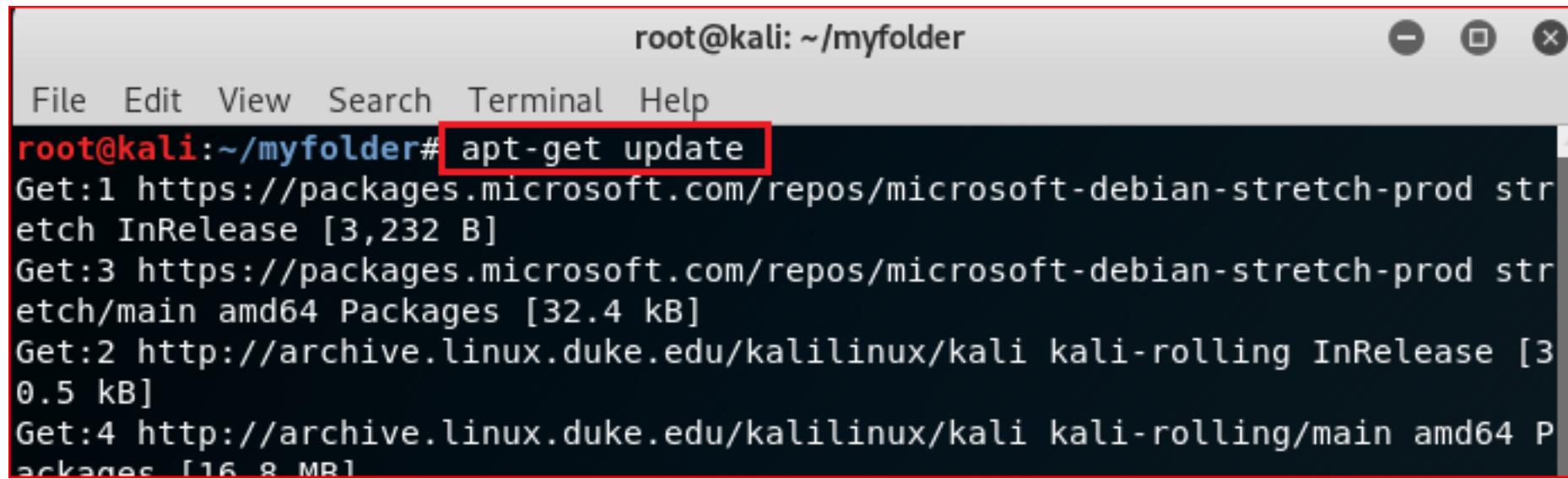
(kali㉿kali)-[~]
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games

(kali㉿kali)-[~]
$ export PATH=/home/kali/myfolder:$PATH
(kali㉿kali)-[~]
$ echo $PATH
/home/kali/myfolder:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
(kali㉿kali)-[~]
$ myFirstScript.sh
-rw-r--r-- 1 kali kali 12 Jun 28 11:18 myFile.txt
```

executed without specifying path

Update/Install software

Update software

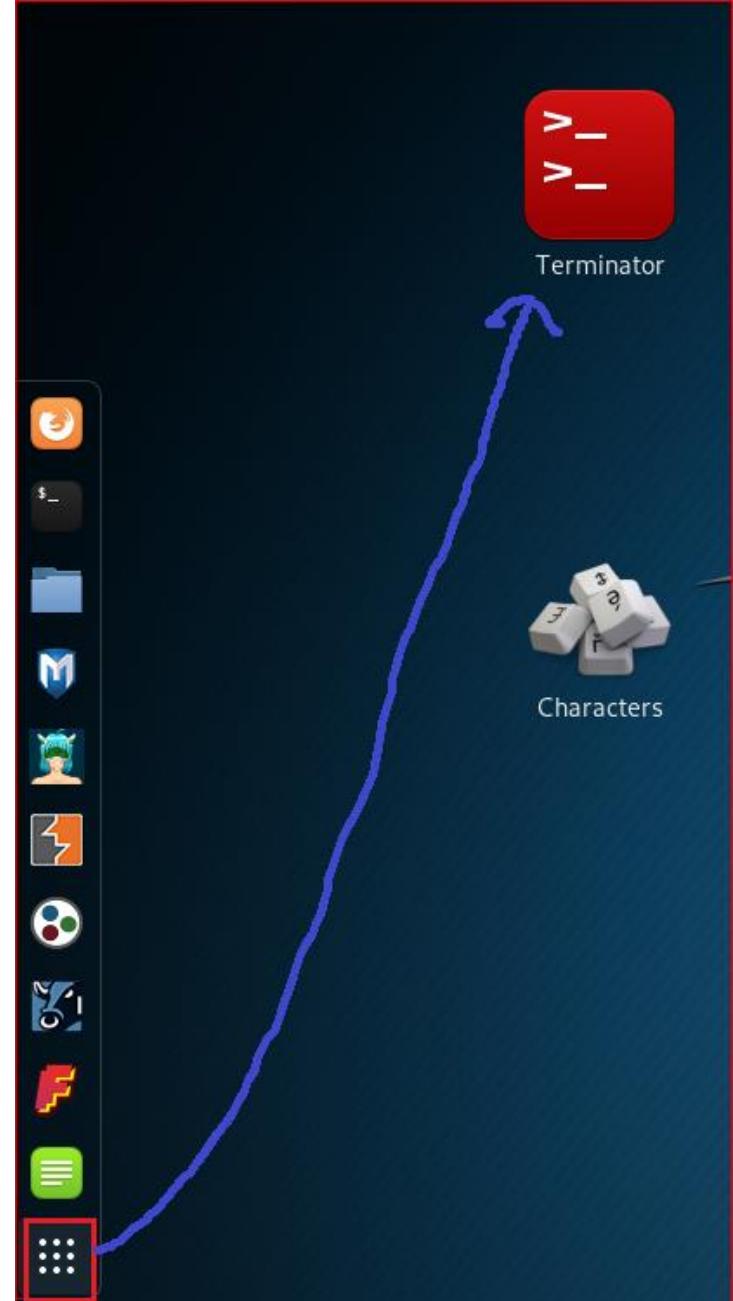


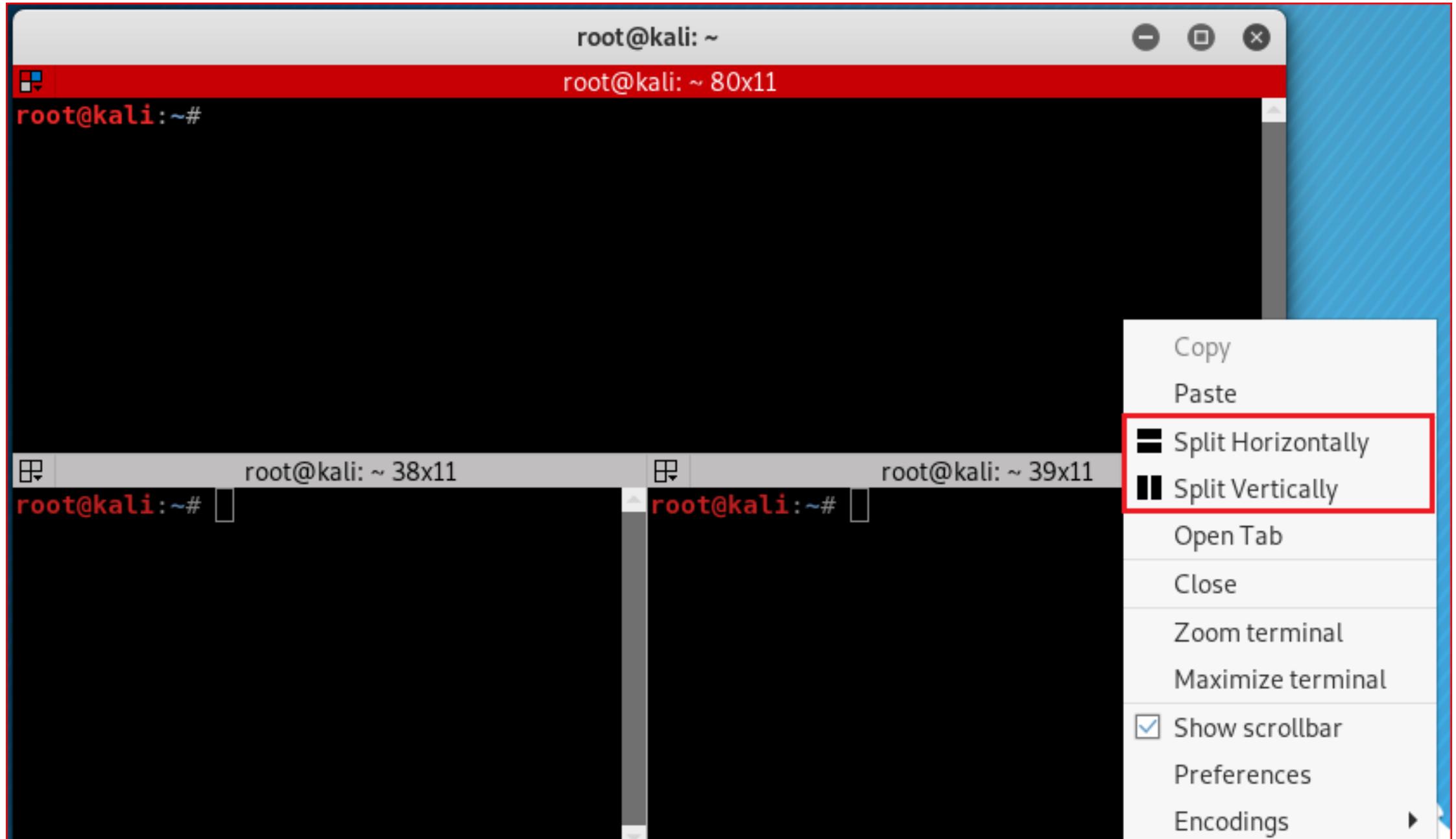
A screenshot of a terminal window titled "root@kali: ~/myfolder". The window has a red border. The title bar shows the user "root@kali" and the directory "~/myfolder". The menu bar includes "File", "Edit", "View", "Search", "Terminal", and "Help". Below the menu bar, the command "root@kali:~/myfolder# apt-get update" is highlighted with a red box. The terminal output shows the process of updating packages from two sources: Microsoft's repository and Kali Linux's repository. The output includes file names, sizes, and download counts.

```
root@kali:~/myfolder# apt-get update
Get:1 https://packages.microsoft.com/repos/microsoft-debian-stretch-prod stretch InRelease [3,232 B]
Get:3 https://packages.microsoft.com/repos/microsoft-debian-stretch-prod stretch/main amd64 Packages [32.4 kB]
Get:2 http://archive.linux.duke.edu/kalilinux/kali kali-rolling InRelease [30.5 kB]
Get:4 http://archive.linux.duke.edu/kalilinux/kali kali-rolling/main amd64 Packages [16.8 MB]
```

Install terminator

```
root@kali:~/myfolder# apt-get install terminator
Reading package lists... Done
Building dependency tree
Reading state information... Done
terminator is already the newest version (1.91-1).
```





Zip/unzip files

```
root@kali:~/myfolder# ls
DulILzQXcAAkFMV.jpg  renamed_file.txt
root@kali:~/myfolder# zip zipped_image.zip DulILzQXcAAkFMV.jpg
  adding: DulILzQXcAAkFMV.jpg (deflated 1%)
root@kali:~/myfolder# ls
DulILzQXcAAkFMV.jpg  renamed_file.txt  zipped_image.zip
root@kali:~/myfolder# rm DulILzQXcAAkFMV.jpg
root@kali:~/myfolder# ls
renamed_file.txt  zipped_image.zip
root@kali:~/myfolder# 
```

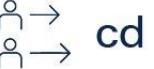
```
root@kali: ~/myfolder 80x11
root@kali:~/myfolder# unzip zipped_image.zip
Archive: zipped_image.zip
  inflating: DulILzQXcAAkFMV.jpg
root@kali:~/myfolder# ls
DulILzQXcAAkFMV.jpg  renamed_file.txt  zipped_image.zip
root@kali:~/myfolder# 
```

Linux Basic Commands Every User Should Know



pwd

To find out the path of the current working directory



cd

To navigate through the Linux files and directories



ls

ls used to view the contents of a directory



cat

ls used to create a new file



cp

To copy files from the current directory to a different directory



mv

The command is to move files



mkdir

Use mkdir command to make a new directory



rm

The rm command is used to delete directories and the contents within them



locate

You can use this command to locate a file, just like the search command in Windows



sudo

This command enables you to perform tasks that require administrative or root permissions



head

The head command is used to view the first lines of any text



Some Basic Linux Commands



by @SecurityGuill

FILE COMMANDS

- **ls** = directory listing
- **ls -al** = formatted listing with hidden files
- **cd dir** = change directory to dir
- **pwd** = show current directory
- **mkdir dir** = create directory dir
- **rm file** = delete file
- **rm -r dir** = delete directory dir
- **rm -f file** = force remove file
- **rm -rf dir** = force remove directory
- **cp file1 file2** = copy file1 to file2
- **mv file1 file2** = rename file1 to file2
- **ln -s file link** = create symbolic link 'link' to file
- **touch file** = create or update file
- **cat > file** = place standard input into file
- **more file** = output the contents of the file
- **less file** = output the contents of the file
- **head file** = output first 10 lines of file
- **tail file** = output last 10 lines of file
- **tail -f file** = output contents of file as it grows



NETWORK



- **ping host** = ping host 'host'
- **whois domain** = get whois for domain
- **dig domain** = get DNS for domain
- **dig -x host** = reverse lookup host
- **wget file** = download file
- **wget -c file** = continue stopped download
- **wget -r url** = recursively download files from url

SYSTEM INFO

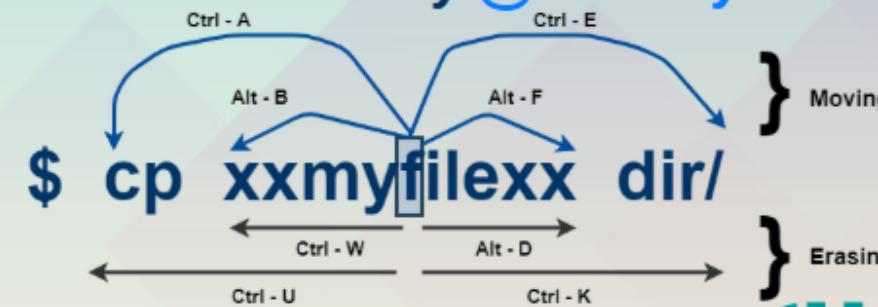


- **date** = show current date/time
- **cal** = show this month's calendar
- **uptime** = show uptime
- **w** = display who is online
- **whoami** = who are you logged in as
- **uname -a** = show kernel config
- **cat /proc/cpuinfo** = cpu info
- **cat /proc/meminfo** = memory info
- **man command** = show manual for command
- **df** = show disk usage
- **du** = show directory space usage
- **du -sh** = human readable size in GB
- **free** = show memory & swap usage
- **whereis app** = show possible locations of app
- **which app** = show which app will be run by default

SSH



- **ssh user@host** = connect to host
- **ssh -p port user@host** = connect using port p
- **ssh -D port user@host** = connect & use bind port



SEARCHING

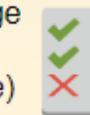
- **grep pattern files** = search for pattern in files
- **grep -r pattern dir** = search recursively for pattern in dir
- **commad | grep pattern** = search for pattern in the output of command
- **locate file** = find all instances of file

PROCESS MANAGEMENT



- **ps** = display currently active processes
- **ps aux** = ps with a lot of detail
- **kill pid** = kill process with pid 'pid'
- **killall proc** = kill all processes named proc
- **bg** = lists stopped/background jobs
- **fg** = bring most recent job to foreground
- **fg n** = brings job n to foreground

FILE PERMISSIONS



- **chmod octal file** = change permission of file
(4:read / 2:write / 1:execute)
Order: owner/group/world
Eg: chmod 755 file = read write for owner, read execute for group/world

COMPRESSION



- **tar cf file.tar files** = tar files into file.tar
- **tar xf file.tar** = untar into current directory
- **tar tf file.tar** = show contents of archive



Follow [@SecurityGuill](#) on [Twitter](#) for more about [Infosec](#) / [Cybersecurity](#)