Session 5

- recap

agenda

6:40 putty super putty installation

19:32 winscp installation

21:34 installation and start ,enable nginx in aws linux

nginx configuration and hosting a static website

copy a static website and copy it to server nginx location using winscp
if you are facing permission issue chmod the dir of html for others and object of others
Restart the nginx service once the files are copied

location of files is `/usr/share/nginx/html/`

29:26 Attaching existing domain to existing ip

you can buy from hostinger

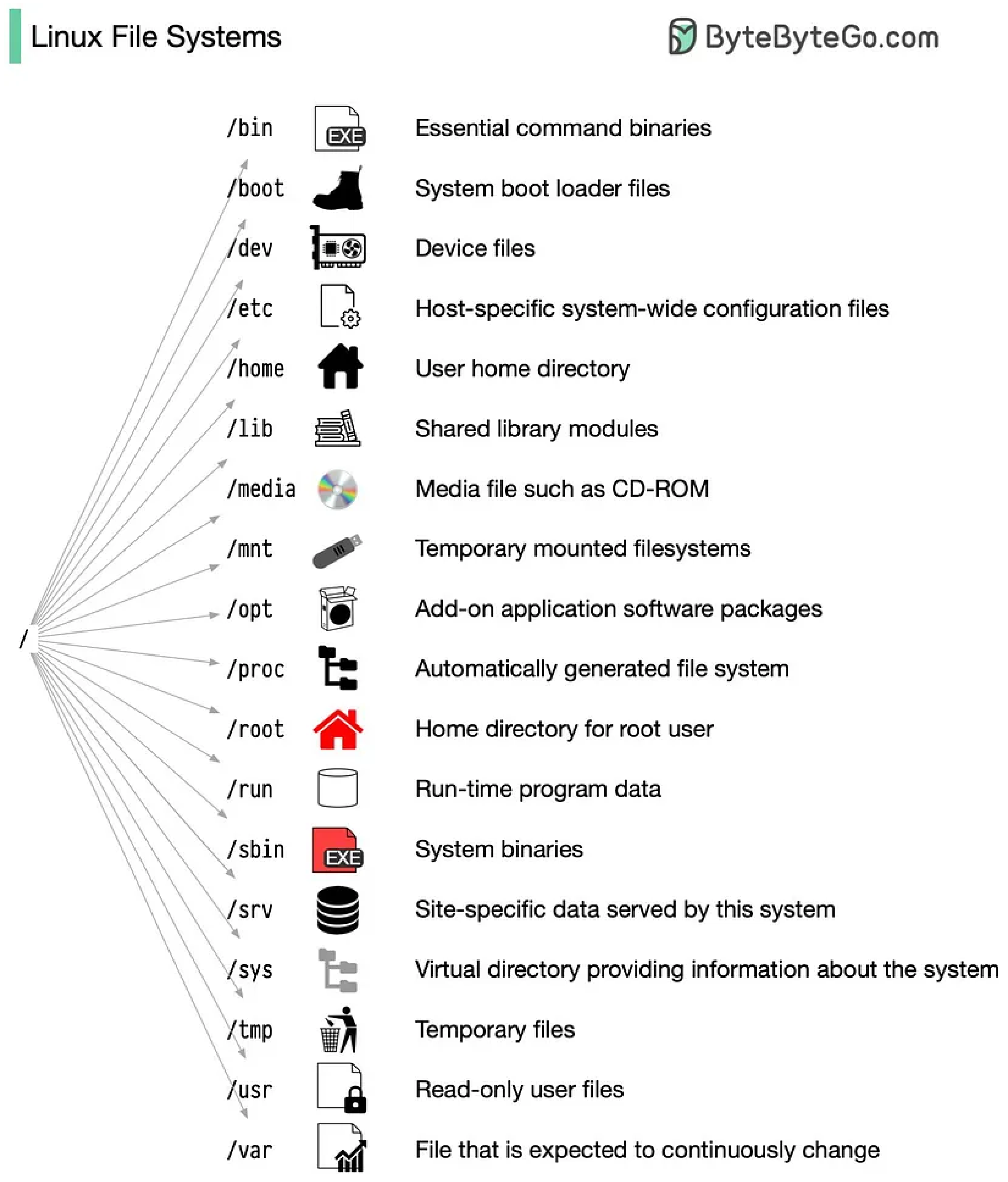
30:17 loading website changes

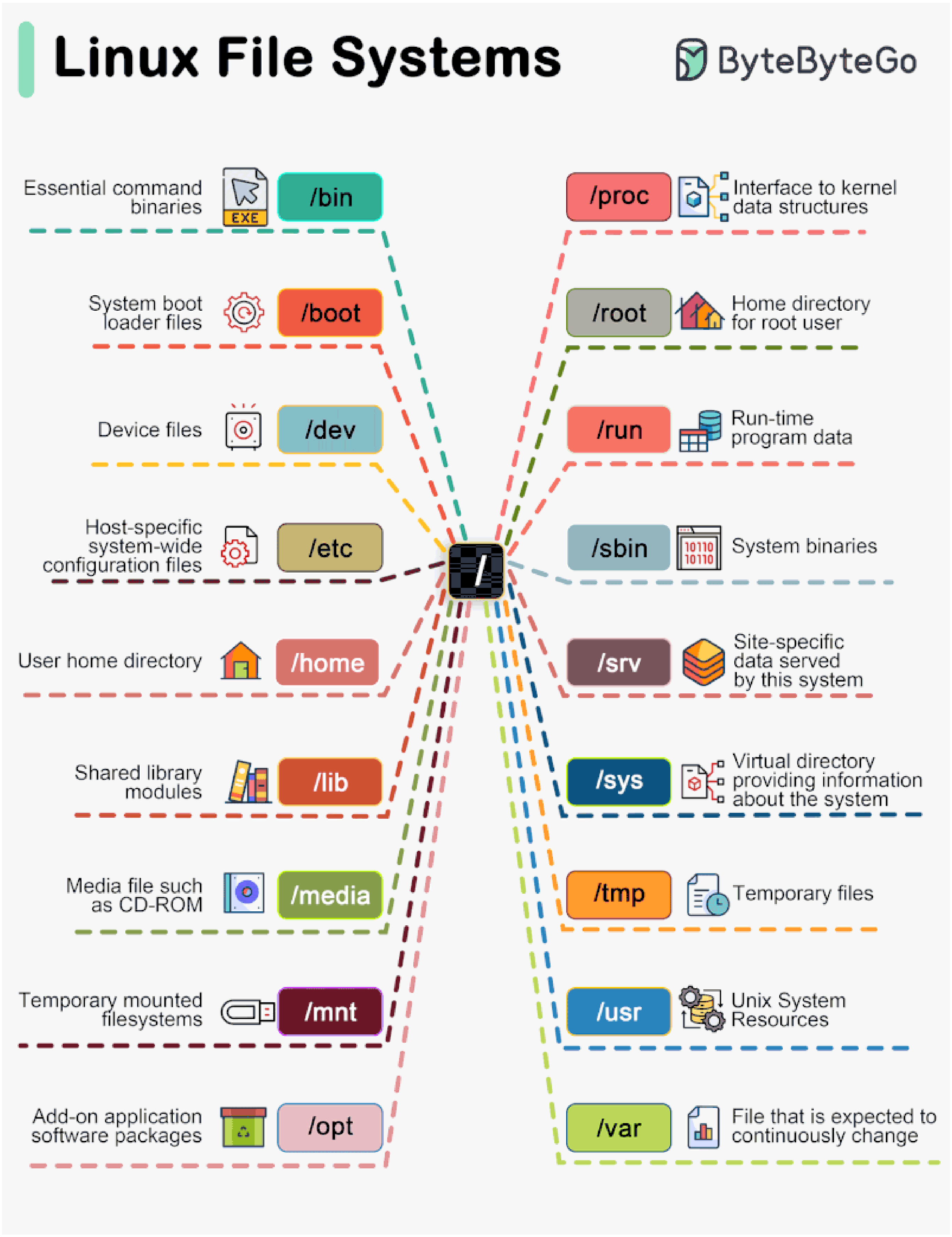
31:30 servers Front end and backend

front end servers - hosts what users see backend servers - 8080 , apache , nginx, java , .net ,python

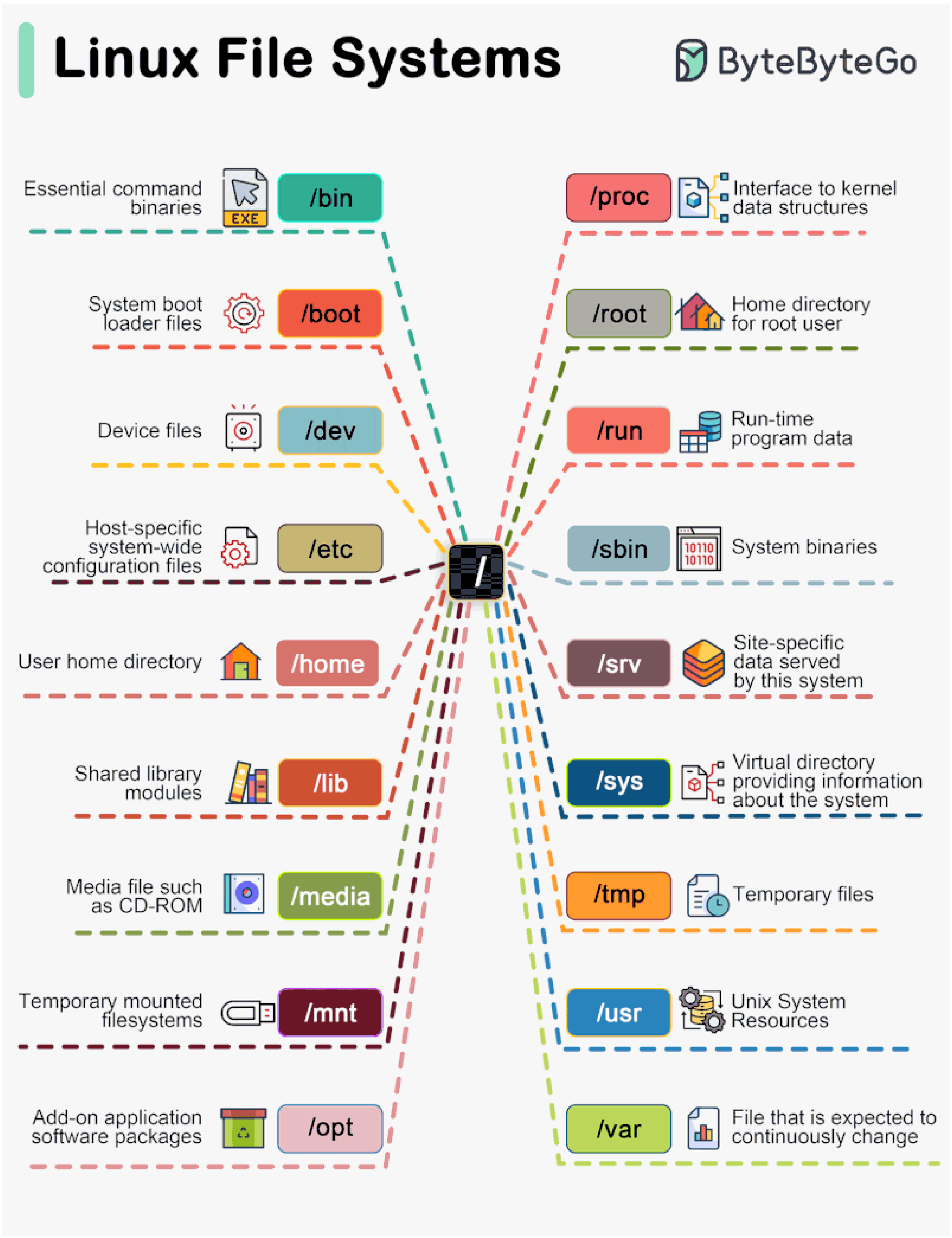
frontend engineers write html to make api calls

33:10 Linux file system directory structure


Linux File Systems
ByteByteGo.com
/
/bin
Essential command binaries
/boot
System boot loader files
/dev
Device files
/etc
Host-specific system-wide configuration files
/home
User home directory
/lib
Shared library modules
/media
Media file such as CD-ROM
/mnt
Temporary mounted filesystems
/opt
Add-on application software packages
/proc
Automatically generated file system
/root
Home directory for root user
/run
Run-time program data
/sbin
System binaries
/srv
Site-specific data served by this system
/sys
Virtual directory providing information about the system
/tmp
Temporary files
/usr
Read-only user files
/var
File that is expected to continuously change


Linux File Systems
ByteByteGo
/
Essential command binaries
/bin
System boot loader files
/boot
Device files
/dev
Host-specific system-wide configuration files
/etc
User home directory
/home
Shared library modules
/lib
Media file such as CD-ROM
/media
Temporary mounted filesystems
/mnt
Add-on application software packages
/opt
/proc
Interface to kernel data structures
/root
Home directory for root user
/run
Run-time program data
/sbin
System binaries
/srv
Site-specific data served by this system
/sys
Virtual directory providing information about the system
/tmp
Temporary files
/usr
Unix System Resources
/var
File that is expected to continuously change

[Linux file system explained]


Linux File Systems
ByteByteGo
/
Essential command binaries
EXE
/bin
System boot loader files
/boot
Device files
/dev
Host-specific system-wide configuration files
/etc
User home directory
/home
Shared library modules
/lib
Media file such as CD-ROM
/media
Temporary mounted filesystems
/mnt
Add-on application software packages
/opt
/proc
Interface to kernel data structures
/root
Home directory for root user
/run
Run-time program data
/sbin
System binaries
/srv
Site-specific data served by this system
/sys
Virtual directory providing information about the system
/tmp
Temporary files
/usr
Unix System Resources
/var
File that is expected to continuously change

46:01 reality of devops projects

Learn properly , there are a lot of projects in migration , if you are skilled enough to industry standard you will get a job, i*f* there is error, read, research, understand, resolve

49:45 setting putty session to no timeout

`/` --> root folder for liux server `boot` --> when server is getting started, server refers this directory and config from grub.cfg
`dev` --> devices --> external devices are mounted here
`etc` --> extra configuration
`home` --> users home directory are created here
`lib/lib64` --> system needs libraries
`/media` --> cd/dvd drive
`/mnt` --> extra disks mount here
/opt --> optional , manual installation of services or applications will be downloaded here
/proc --> running process will store here with their PID
`/proc/cpuinfo` - cpu info
`/proc/meminfo` - process info
/root --> root user home folder
/run --> when server starts, server uses this directory as storage
/bin --> cat,vim, touch, cd
/sbin --> system commands like root access commands
/tmp --> temp files, not important
/usr --> all users,
/var --> variables --> log files,

1:01:34 Symlink vs Hardlink

V. important

Domain (google.com) is not understand by computer, system can only understand numbers
when we type `ping google.com` it fetches the ip address

storage

storage contains memory locations when a file is created it points to memory location these memory location are called inode inode is representation for file and directory inside memory

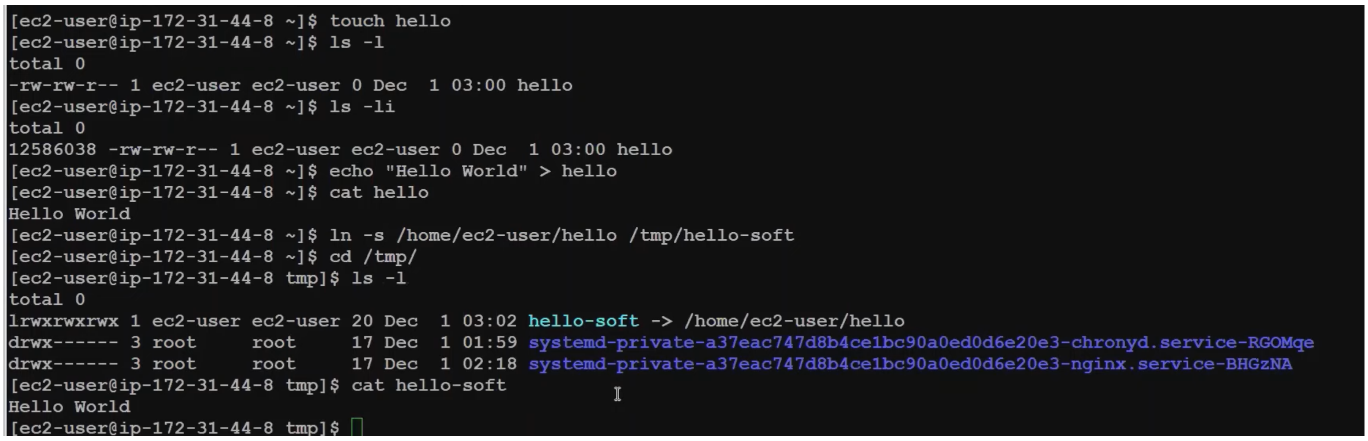
checking inode

touch hello

`ls -li` contains memory location when we open file it opens the memory location symlink - similar to shortcut in windows, it doesn't contain any data but it points to data
for ex -

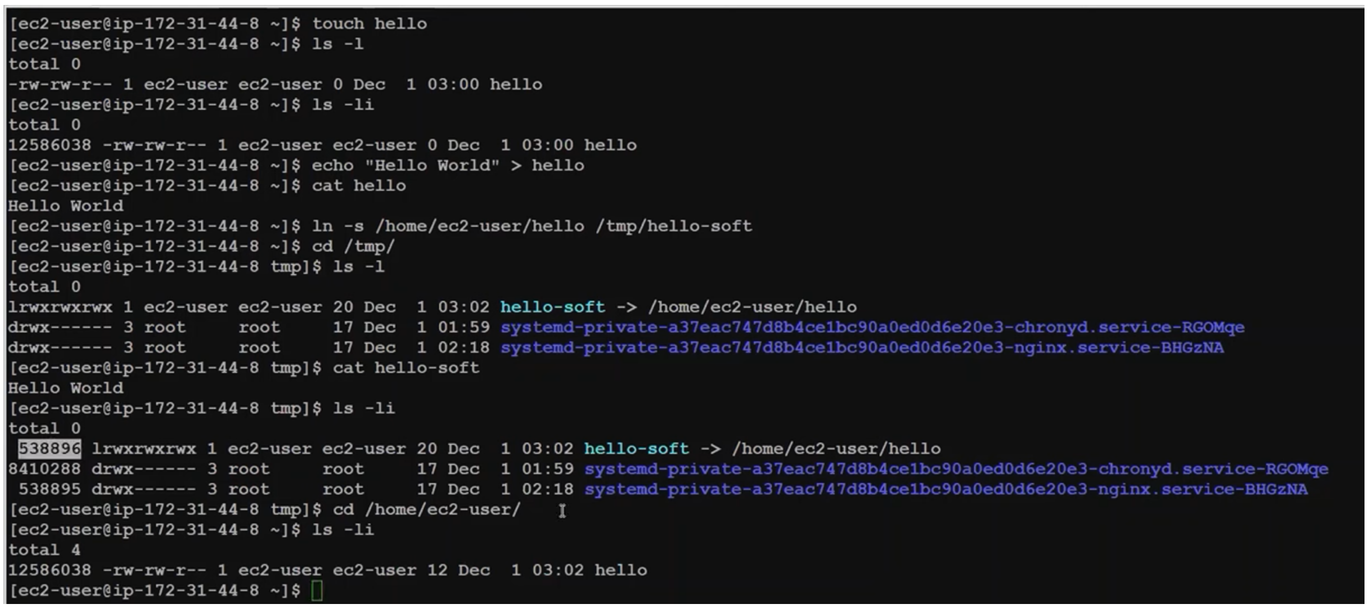
create a file hello

`ln -s /home/ec2-user/hello /tmp/hello-soft` `ln -s sourceFile destinationForLink`



```
[ec2-user@ip-172-31-44-8 ~]$ touch hello
[ec2-user@ip-172-31-44-8 ~]$ ls -l
total 0
-rw-rw-r-- 1 ec2-user ec2-user 0 Dec  1 03:00 hello
[ec2-user@ip-172-31-44-8 ~]$ ls -li
total 0
12586038 -rw-rw-r-- 1 ec2-user ec2-user 0 Dec  1 03:00 hello
[ec2-user@ip-172-31-44-8 ~]$ echo "Hello World" > hello
[ec2-user@ip-172-31-44-8 ~]$ cat hello
Hello World
[ec2-user@ip-172-31-44-8 ~]$ ln -s /home/ec2-user/hello /tmp/hello-soft
[ec2-user@ip-172-31-44-8 ~]$ cd /tmp/
[ec2-user@ip-172-31-44-8 tmp]$ ls -l
total 0
lrwxrwxrwx 1 ec2-user ec2-user 20 Dec  1 03:02 hello-soft -> /home/ec2-user/hello
drwx------ 3 root     root     17 Dec  1 01:59 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-chronyd.service-RGOMqe
drwx------ 3 root     root     17 Dec  1 02:18 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-nginx.service-BHGzNA
[ec2-user@ip-172-31-44-8 tmp]$ cat hello-soft
Hello World
[ec2-user@ip-172-31-44-8 tmp]$
```

symlink points to actual file Symlink will have its own inode will not be same as file inode



```
[ec2-user@ip-172-31-44-8 ~]$ touch hello
[ec2-user@ip-172-31-44-8 ~]$ ls -l
total 0
-rw-rw-r-- 1 ec2-user ec2-user 0 Dec  1 03:00 hello
[ec2-user@ip-172-31-44-8 ~]$ ls -li
total 0
12586038 -rw-rw-r-- 1 ec2-user ec2-user 0 Dec  1 03:00 hello
[ec2-user@ip-172-31-44-8 ~]$ echo "Hello World" > hello
[ec2-user@ip-172-31-44-8 ~]$ cat hello
Hello World
[ec2-user@ip-172-31-44-8 ~]$ ln -s /home/ec2-user/hello /tmp/hello-soft
[ec2-user@ip-172-31-44-8 ~]$ cd /tmp/
[ec2-user@ip-172-31-44-8 tmp]$ ls -l
total 0
lrwxrwxrwx 1 ec2-user ec2-user 20 Dec  1 03:02 hello-soft -> /home/ec2-user/hello
drwx------ 3 root     root     17 Dec  1 01:59 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-chronyd.service-RGOMqe
drwx------ 3 root     root     17 Dec  1 02:18 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-nginx.service-BHGzNA
[ec2-user@ip-172-31-44-8 tmp]$ cat hello-soft
Hello World
[ec2-user@ip-172-31-44-8 tmp]$ ls -li
total 0
 538896 lrwxrwxrwx 1 ec2-user ec2-user 20 Dec  1 03:02 hello-soft -> /home/ec2-user/hello
8410288 drwx------ 3 root     root     17 Dec  1 01:59 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-chronyd.service-RGOMqe
 538895 drwx------ 3 root     root     17 Dec  1 02:18 systemd-private-a37eac747d8b4ce1bc90a0ed0d6e20e3-nginx.service-BHGzNA
[ec2-user@ip-172-31-44-8 tmp]$ cd /home/ec2-user/
[ec2-user@ip-172-31-44-8 ~]$ ls -li
total 4
12586038 -rw-rw-r-- 1 ec2-user ec2-user 12 Dec  1 03:02 hello
[ec2-user@ip-172-31-44-8 ~]$
```

Hardlink

Hardlink points to inode of a file but symlink will point to file

creating hardlink `ln /home/ec2-user/hello hello-hard`

simlink will be red in color if we delete file hardlink works even if source is deleted

inode will be only deleted only when all hardlinks are deleted to that memory location

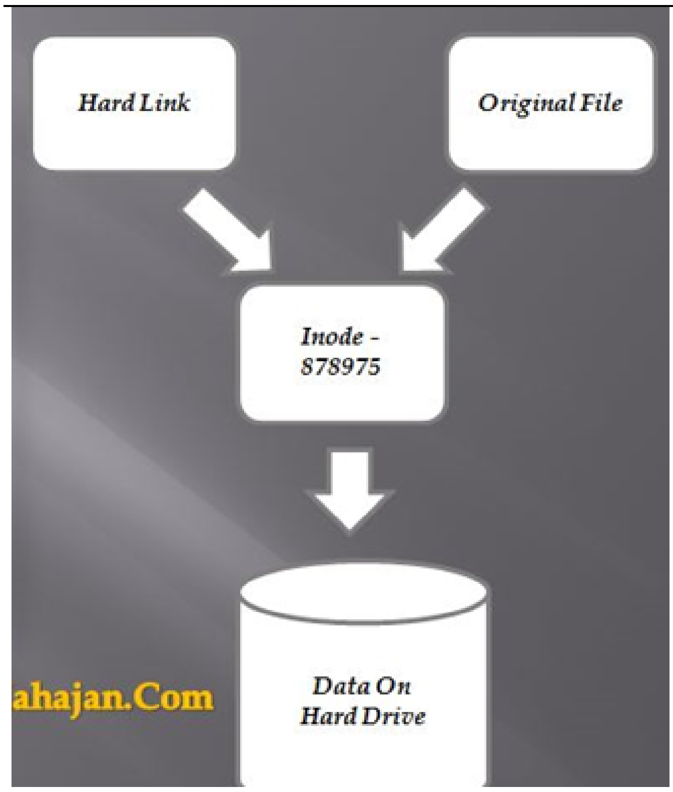
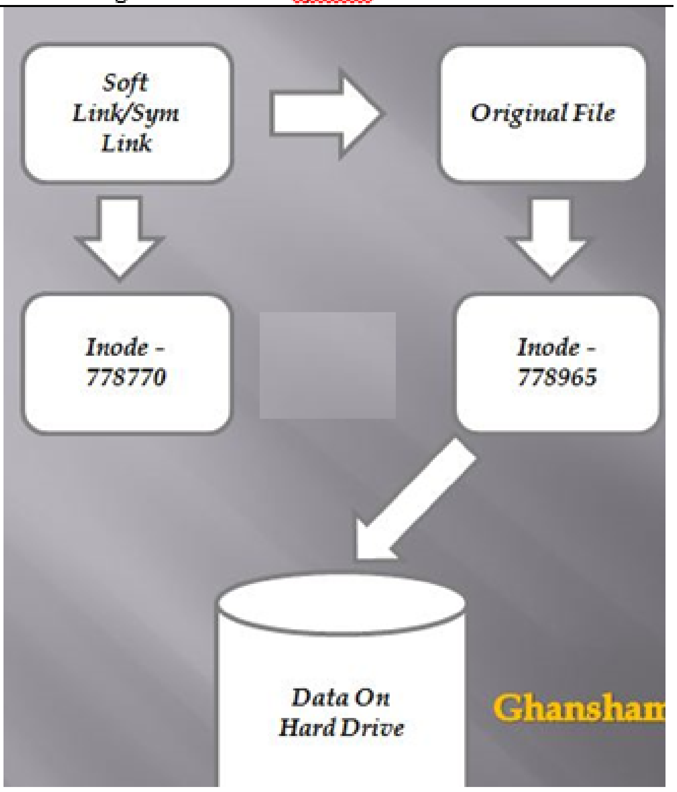
A file in the file system is basically a link to an inode. A hard link, then, just creates another file with a link to the same underlying inode.

When you delete a file, it removes one link to the underlying inode. The inode is only deleted (or deletable/over-writable) when all links to the inode have been deleted.

A symbolic link is a link to another name in the file system.

Once a hard link has been made the link is to the inode. Deleting, renaming, or moving the original file will not affect the hard link as it links to the underlying inode. Any changes to the data on the inode is reflected in all files that refer to that inode.

Note: Hard links are only valid within the same File System. Symbolic links can span file systems as they are simply the name of another file.

Hard Link	Soft Link
If the original file is deleted, nothing will happen to link file. It will still present with the active status.	If the original file is deleted, link file remains there as inactive status.
Both the file (original and link file) have same inode number	Both the file have different inode number.
Can't be used to create a hard link for a directory and different file system. [root@localhost ~]# ln test testhardlink ln: 'test': hard link not allowed for directory	Can be used to create a soft link for a directory and different file system. [root@localhost ~]# ln -s test testsoftlink [root@localhost ~]#
Hard link will have actual file content.	Soft link has the path for original file and not the file content. So basically it is a shortcut of any file or directory.
#ln {source} {destination}	#ln –s {Source} {destination} -s = This flag tells to create a symlink
 Hard Link Original File Inode - 878975 Data On Hard Drive ahajan.Com	 Soft Link/Sym Link Original File Inode - 778770 Inode - 778965 Data On Hard Drive Ghansham

symlink vs hardlink

about 'do not understand the use of a *hard link', it can be used in build systems which do lot of copying of binaries. Creating hard link instead of actual copy speeds things up. MSBuild 4.0 supports this.**

The different semantics between hard and soft links make them suitable for different things.

Hard links:

```
indistinguishable from other directory entries, because every directory entry is
hard link
"original" can be moved or deleted without breaking other hard links to the same
inode
only possible within the same filesystem
permissions must be the same as those on the "original" (permissions are stored in
the inode, not the directory entry)
can only be made to files, not directories
```

Symbolic links (soft links)

```
simply records that point to another file path. (ls -l will show what path a
symlink points to)
will break if original is moved or deleted. (In some cases it is actually
desirable for a link to point to whatever file currently occupies a particular
location)
can point to a file in a different filesystem
can point to a directory
on some file system formats, it is possible for the symlink to have different
permissions than the file it points to (this is uncommon)
```

you can create a group of commands for easy management, command aliases

about 'do not understand the use of a hard link', it can be used in build systems which do lot of copying of binaries. Creating hard link instead of actual copy speeds things up. MSBuild 4.0 supports this.

linux password is stored in `/etc/shadow` file.