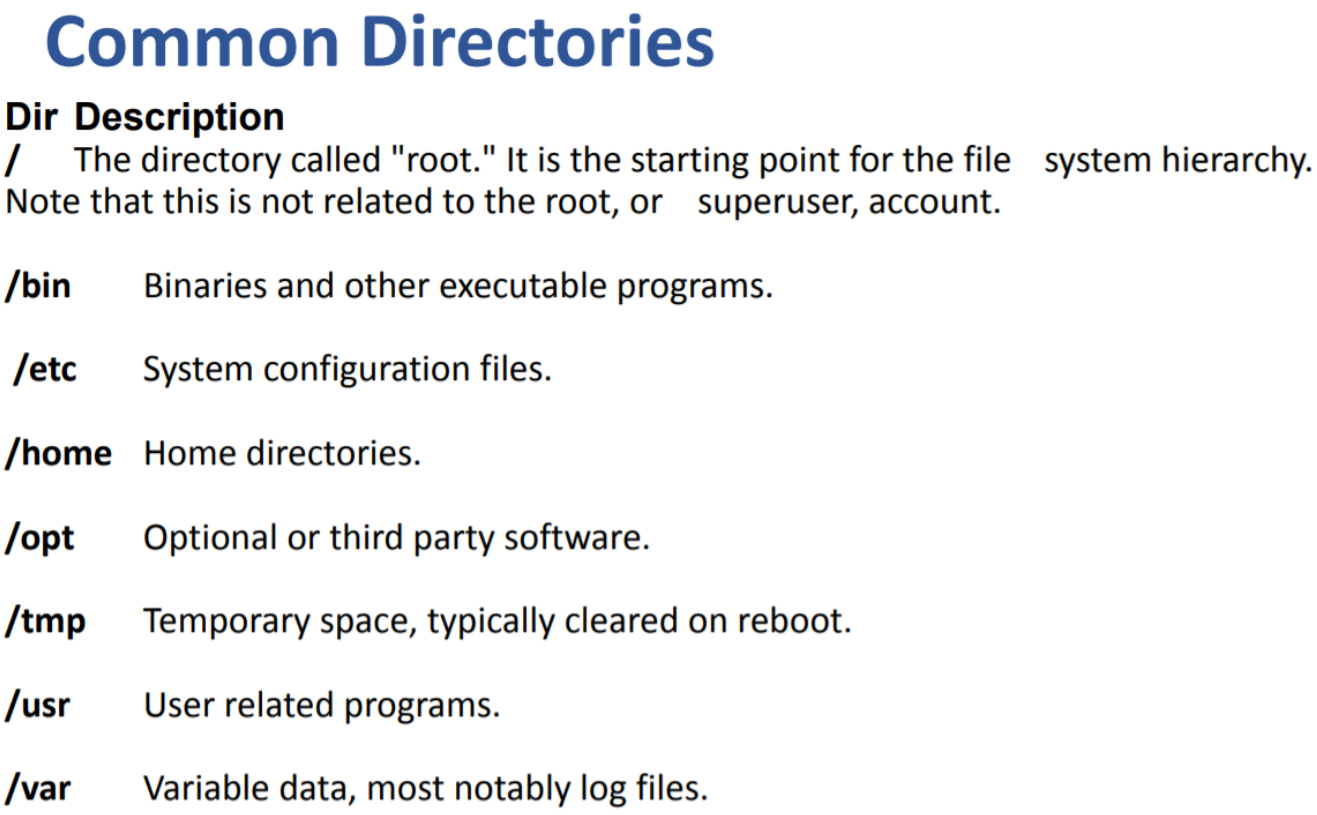
Table of contents

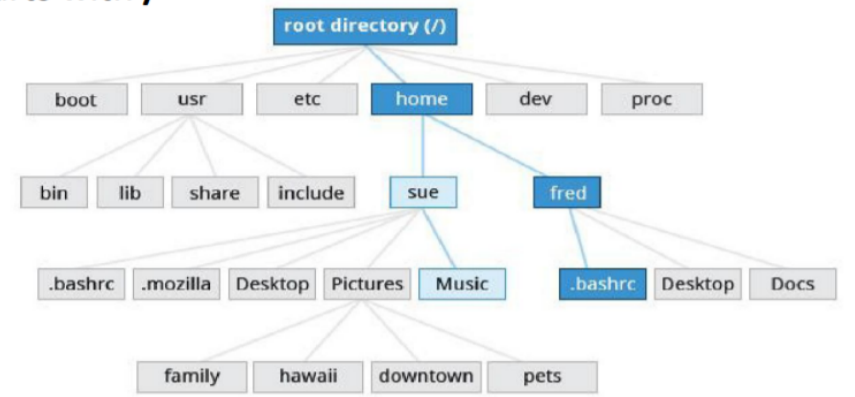
Common Directories



Absolute Path and Relative Path

Absolute Path and Relative Path

- **Absolute pathname**: begins with the root directory , it always starts with **/**
- **Relative pathname**: starts from the present working directory , it never starts with **/**



1.In the above example, we use the relative path method to list the files under Music from your current working directory (sue)
$ ls ../sue/Music

2.In the above example, we use the Absolute pathname method to edit the .bashrc file:
$ gedit /home/fred/.bashrc

Basic Linux Commands

- `ls` - Lists directory contents.
- `cd` - Changes the current directory.
- `pwd` - Displays the present working directory.
- `cat` - Concatenates and displays files.
- `echo` - Displays arguments to the screen.
- `man` - Displays the online manual.
- `exit`-Exits the shell or your current session.
- `clear` - Clears the screen
- `q` - quit
- Look at the man pages for `ls`
    - `man ls`

Files and Directory Operations

- All input lines entered at the shell prompt have three basic elements:

```
Command
Options
```

```
Arguments
```

- The command is the name of the program you are executing.
- It may be followed by one or more options that modify what the command may do.
- Arguments specify on what the command will operate on.

Listing Files

File Operations

- Directory Separator is `/`

```
which ls
whereis ls
locate ls
head - print first 10 lines
tail - print last 10 lines
touch - create an empty file
file <name> - will give type of file
mkdir – create a directory
rmdir – remove directory*
mv – move or rename a file
rm - remove a file (Use cautiously)
rm -f : forcefully remove (Use cautiously)
rm -i : interactively remove a file (Use cautiously)
rm -rf : recursively remove entire tree structure (Use cautiously)
```

File Permissions

Files have three kinds of permissions: read (r), write (w), execute (x). rwx. • These permissions affect three groups of owners: user/owner (u), group (g), and others (o).

`rwx: rwx: rwx u: g: o`

- Numbers for permissions
  - 4 if read permission is desired.
  - 2 if write permission is desired.
  - 1 if execute permission is desired

Directory Shortcuts

Symbol Command	Description
.	This directory
..	The parent directory
`cd -`	Change to the previous directory

Linux User operations

- Agenda for today would be understanding and performing some of the basic linux operations that will help our understanding

User creation

- Every operating system has a default user that comes with it. For ex. Amazon linux 1 and 2 has `ec2-user` as the default user. Ubuntu operating system has `ubuntu` and so on
- We can create other users as well based on the requirement .
- These users are generally created for project teams which use the instance for various developement purposes

```
sudo adduser test1
```

- Above command will create a user named test1
- Now that we have created a user , in order to use it , that user need to have a password

```
sudo passwd test1
```

- Once you use above command , it will prompt for entering a password .
- To test the above , login to the machine as ec2-user and use

```
su test1
```

- su stands for switch user . Above command will ask for a password , once it is entered , you will be able to log in as test1 user
- Super users do not require a password while using su command. So from root you can switch to any user that you wish without having to worry about entering the password

Enabling password based authentication for EC2

- Since the pem key allows access to ec2-user which has sudo access , ideally we shoudnt be sharing the key with the project users
- We should be creating seperate users for each user
- By Default , ec2 instances only have key based authentication . We need to enable password based authentication in order for users to access the instance with just username and password

```
cd /etc/ssh/
sudo vi sshd_config
```

- Once the file is opened , change the PasswordAuthentication as yes

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile .ssh/authorized_keys

#AuthorizedPrincipalsFile none


# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
PasswordAuthentication yes

# Change to no to disable s/key passwords
#ChallengeResponseAuthentication yes
ChallengeResponseAuthentication no

# Kerberos options
#KerberosAuthentication no
-- INSERT --
```

- Once the changes are made save and exit

```
sudo service sshd restart
```

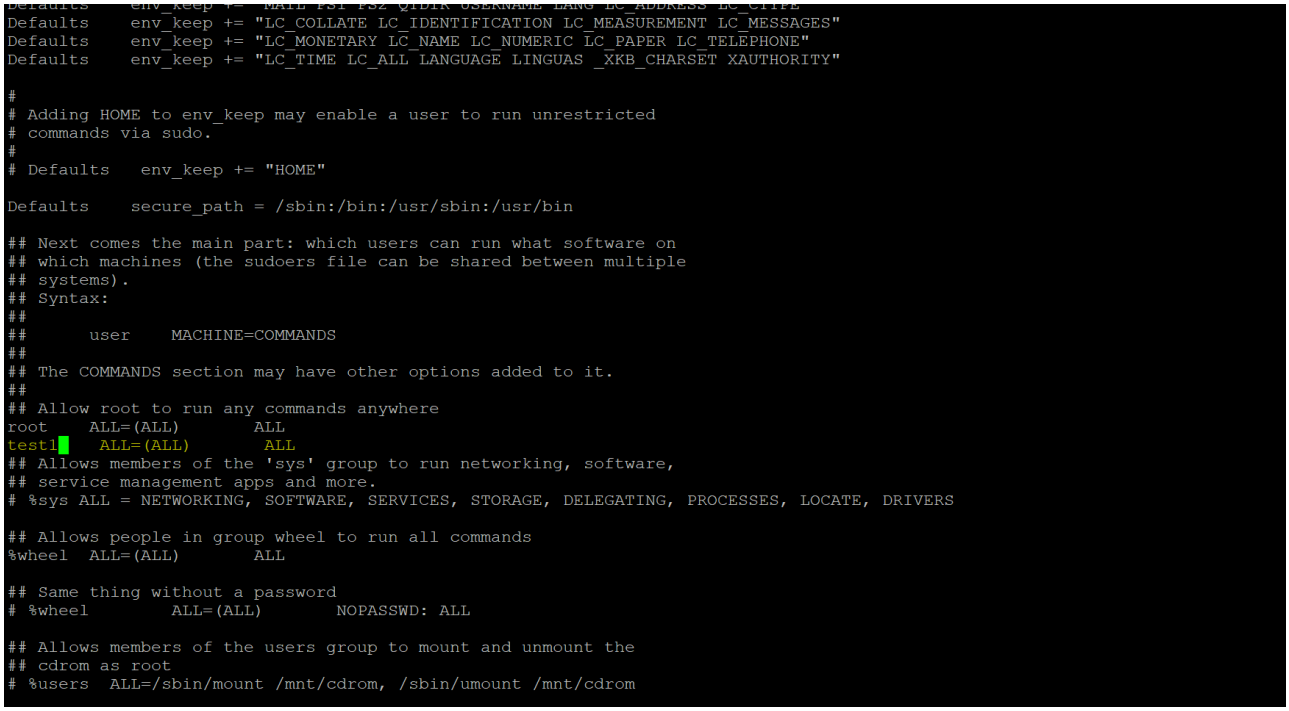
- Above command will restart the ssh daemon which releads the configuration changes if any
- You can now try logging in using just the username (for ex- test1) and password

Granting sudo privileges to user

- Whenever any new user is created , it only has permissions to operate on its own home folder i.e. /home/test1
- If there is a scenario where the user needs permissions to install or make some changes in other folder , user will need sudo privileges
- In order to grant the permissions

```
sudo visudo
```

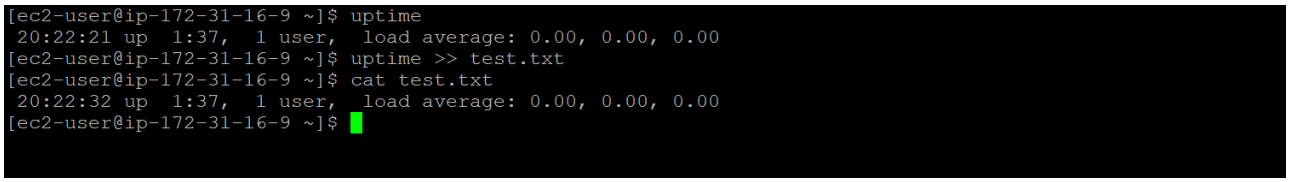
- This will open a file , make the changes as below



- Save the file and exit . Post which user will be able to run sudo commands

Cronjob

- Cronjobs are a great way of scheduling linux level tasks or scripts
- We have briefly seen cloudwatch rules which used cron expression
- For ex we want to run "uptime " command which shows duration since the last reboot
- uptime >> test.txt will stora the output in the test.txt file



- If we wish to schedule above command to run at a certain schedule , we can use cronjob

```
crontab -e
# this opens up the cronjob editor which is exactly like vi editor
# For testing purposes we will make entry "* * * * * uptime >> test.txt"
# Save and exit

crontab -l

# use above to list the cronjobs. Remember cronjobs are user specific , so while
scheduling dont use sudo or else the jobs will be scheduled from root user
```

```
* * * * * uptime >> test.txt
~
~
~
~
~
```

```
[ec2-user@ip-172-31-16-9 ~]$ cat test.txt
 20:22:32 up  1:37,  1 user,  load average: 0.00, 0.00, 0.00
 20:27:01 up  1:42,  1 user,  load average: 0.00, 0.00, 0.00
 20:28:01 up  1:43,  1 user,  load average: 0.00, 0.00, 0.00
[ec2-user@ip-172-31-16-9 ~]$
```

- This can even be used to schedule shell/python scripts to be executed at a certain schedule