Linux commands:

**Impartant:**

**If any one asked me to list most powerfull commands in linux I will tell below commands**

**\*ssh → Connecting to remote server**

**\*scp → copying files from remote or to remote server.**

**\*grep**

**\*cp**

**\*find**

**1) EUID** →is a env in linux, which store the logged in user id.

In linux user id for roo user is always 0.

To check the logged-in user is root or normal use:

echo $EUID → give zero as output if user is root user otherwise normal user id

**2) uname --------> give info about your system**

$ uname -a ---- give all info of your system

$ uname --kernel-release ----- kernel release details

$ uname --kernel-version ------- kernel version

$ uname --kernel-name -------- name

**3) uptime**

$ uptime -p ------- give details about the from when this system is up

**4) $w ------- give details about the logged in users.**

**5) rm command ------> Remove commands**

**rm -r** /home/nraghu/naga → this will delete the naga folder (if it’s not worked use -f)

or **rm -fr** /*home/*nraghu/naga → this will forcefully delete the naga folder

**rm -r** /*home*/nraghu/naga/raghu.txt → this will delete the raghu.txt

cat /*etc*/shells → give what and all shell will support in your system

which bash → give where the executable file is available in system

**6) echo command:**

**\* echo "Raghu M N" >> Nagendra.txt → >> will append the content(Raghu M N) into Nagendra.txt**

**\* echo "Ravi M N" > Nagendra.txt → > will overwrite the content of Nagendra.txt**

**7) ls command:**

**ls** → list the file and directories except hidden files

**ls -l**  → list files and directories with owner and permission details

**ls -a** -->list the file and directories with hidden files

**ls -al or ll** → list all files and directories with hidden files and their permissions

-rwxr-xr-x 1 nraghu nraghu 53480440 Dec 19 10:59 beam.jit

**ls -lh** → give list of files with time

-rwxr-xr-x 1 nraghu nraghu 52M Dec 19 10:59 beam.jit

**8) Copy and move command:**

**$** **cp file2.txt file3.txt** --> copying the content of file2 into file3

**$ cp -r folder1 folder1\_bk** --> copy the entire folder1 folder into folder1\_bk folder.

Inside folder1\_bk have the folder1

**$ cp -r folder1/\* folder1\_bk** --> copy the only content of folder1 into folder1\_bk folder

**$ mv file3.txt file4.txt** --> moving the file3.txt into file3.txt

**$ mv folder3 folder4 --> moving the folder3 into folder4**

**9) cat command:**

**$ cat > larger-text.txt** --> if don’t pass any input into cat, will connect the stdin of cat process and redirecting stdout into larger-text.txt file. Ctrl+c to exit once the content pasted.

**5) Head command:**is used to read the starting of file,by default first 10 lines of file we can read

**$ head file\_name**

**$ head -n 23 file\_name -->** show the first 23 lines of file

options in head command: --> need to explore about the options

-f option to pass file and keep it open the file and give live changes to top of file if any changes happen to that file. Help in checking the live logs of log file

**6) Tail command:** is used read last part of the file. By default we can read last 10 lines of file

**$ tail file\_name** --> show the last 10 lines of file

**$ tail -n 13 file\_name** --> ready last 13 lines of file

**$ tail -f file\_name** --> -f option to pass file and keep it open the file and give live changes to top of file if any changes happen to that file. Help in checking the live logs of log file

options in tail command: need to explore about the options

**7) more command:**

is used to open the file and keep open to read the content

**$ more file\_name -->** show content and keep it open

**$ more -6 larger-text.txt** --> first show the 6 lines of file and keep it open

**$ more +10 larger-text.txt** --> start showing the from line 10 and keep it open

**8) grep command: --> need to explore on grep and reg expression**

grep command is used to filter the pattern in file or files in directory(need to use -r option).it is very import to devops engineer to automate process.

Ex: **$ grep raghu my\_details.txt** --> show the line which contain the word raghu in my\_details.txt file

**$ ls /bin | grep hostname** -->

**$ ls /bin | grep ^l** --> list files starts with l letter

**$ ls /bin | grep ch$** --> list files ends with ch word

GNU grep home page: <http://www.gnu.org/software/grep/>

General help using GNU software: <https://www.gnu.org/gethelp/>

**8)softlink and hardlink :**

**Soft link** is just a pointer to actual file or directory. Link name have it’s own permission and size(the size always same only).create the multiple softlinks into same file or folder. I node will be different for link and actual file or folder

**$ ls -lia /usr/bin | grep "\->"** --> the leftside shows the link name and rightside show the actual file or folder

**$ ls -lia**



**Hard link** is creation of file for actual file. The i node(index node) will be same for both files.create the multiple hardlinks into same file but all inode are same.

Linux follow the i node structure to store the content,every file have inode

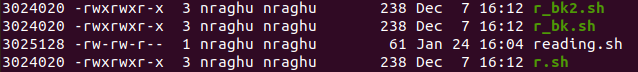
If you change anything in one file,which will be happended in another file also.

If you delete the one file,the content will present in another file.

**$ ln r.sh r\_bk.sh -->** creation of hard link

**$ ln r.sh r\_bk2.sh -->** creation of another hardlink

$ ls -il

****

**We can’t create the hard link to the directory**

**Diffrence between the softlink and hardlink:**

**softlink:**

\*softlink will be created into file or folder

\*i node will be diffent for softlink and souce file

\*once the source file deleted,the link will be invalid,i will lose the data

**hardlink:**

\*hardlink will be created for only file.

\*i node will be same for hard link and source file

\*even if we delete the source file,hardlink contain the data,we will not lose the data

**9) find command:**

find command is used to find the file and directory.

The structure of find command is

find options argument expression

Note: path is argument for find command

[**https://linux.die.net/man/1/find**](https://linux.die.net/man/1/find)

**In general,command have following:**

1)arugument is mandatory for any command

2)long version of option is represented by --word

3)short version of option is represented by -single\_letter

4)expressions is represented by -word(expression is made up of options),all command don’t have expression.

Ex:

$ **find . -name "\*.txt"**



$ **find /home/nraghu/knowledge/Docker/docker\_udemy -name "\*.txt"**

****

find command give path based on type of path passed in command.if you pass relative path,give result in relative path and same for absolute path.

$ **find . -type f** --> find all files in current directory

$ **find . -type d**  --> find all directories under current directory

$ **find . -type l ls**  --> find all links in current directory and give in table format

$ **find . -type f -empty** --> search for all emplty files in currenr directory

$ **find . -type f -size +10M -ls** --> give files which size is greater than 10M

$ **find . -type f -perm 755 -ls** --> give files which all 755 permission

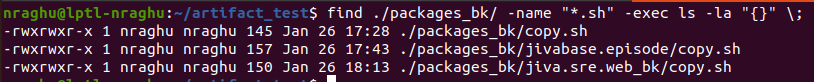
**Executing the additional command inside the find command:**

In find command,have exec option to execute the command into result of find command.

$ **find ./packages\_bk/ -name "\*.sh" -exec ls -la {} \;**

**or**

**$ find ./packages\_bk/ -name "\*.sh" -exec ls -la "{}" \;**

****

The above command operation:

the find command find the files which have sh and exec option will execute the additional ls -la command into output of find command,{} or “{}” represents iteration into all output of find command and ; is argument into exec option,escaping using \

**10) xargs command:**

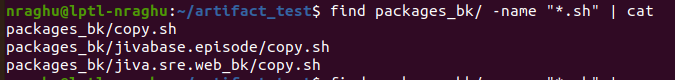
xargs i used to pass the stdout of command into argument into another command

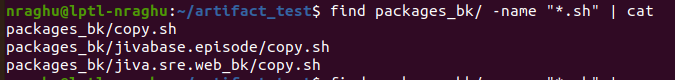
ex:

**$ ls | cat**  --> passing the stdout of ls into stdin of cat command,cat without argument will just redirect stdin into stdout

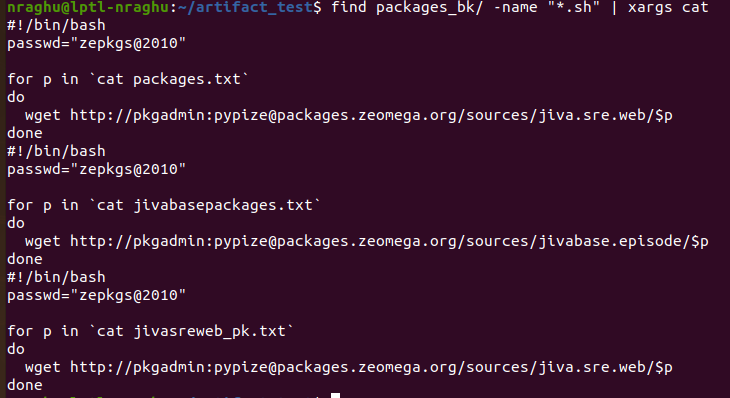
**$ ls | xarg cat -->** passing the stdout of ls command as argument into cat command using xarg command

$ **find packages\_bk/ -name "\*.sh" | cat**

$ **find packages\_bk/ -name "\*.sh" | cat**



$ **find packages\_bk/ -name "\*.sh" | xargs cat**

****

**11) tar and gzip command:**

**Compressing and sorting in linux:**

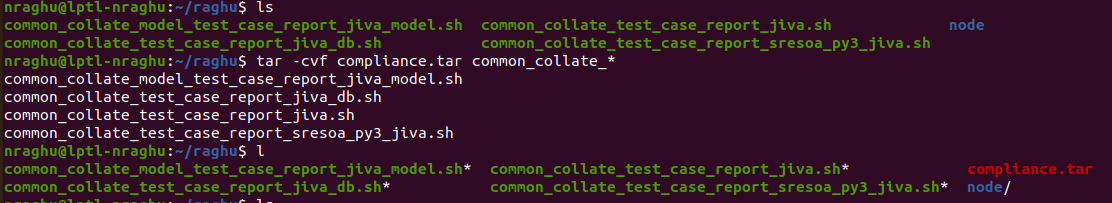
**tar** : is used create the archive file from multiple files(archive file is set of files), used to compress the archive file and restore the indiviuals files from archive. In tar we have option to compress once the archive.tar is created,that archive.tar.gz

$ **tar -cvf raghu\_bk.tar file1.txt file2.txt -->** create the tar archive file,here we are not compressing the tar file

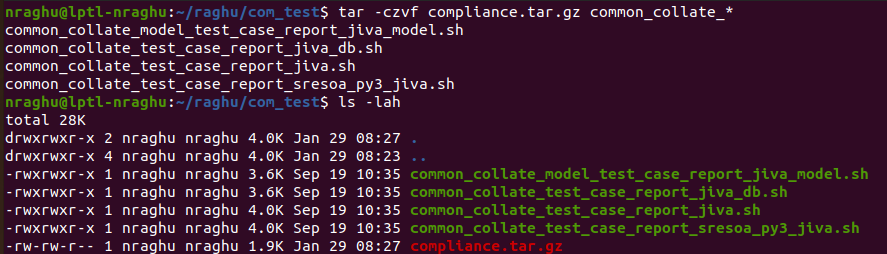
c is used to mention as create tar file,not a compressed file,use z option to compress

f option is used to mentinon which file to create or extract

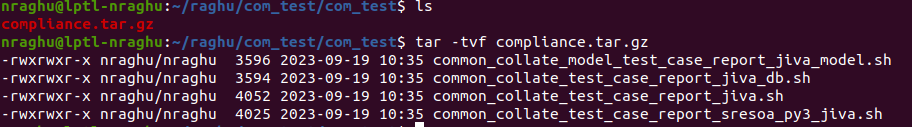
v option is used to mention,it is verbose mode



$ **tar -czvf raghu\_bk.tar.gz file1.txt file2.txt -->** create the tar and compress it using the z option

****

$ **tar -tvf compliance.tar.gz** --> list files inside of archive

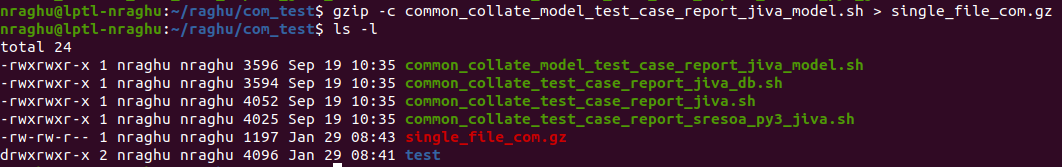


**gzip**: is used to compreses and uncompress single file.

Go with tar command with z option to compress files and directories.

Gzip is worse command to commpress.

$ **gzip -c common\_collate\_model\_test\_case\_report\_jiva\_model.sh > single\_file\_com.gz**

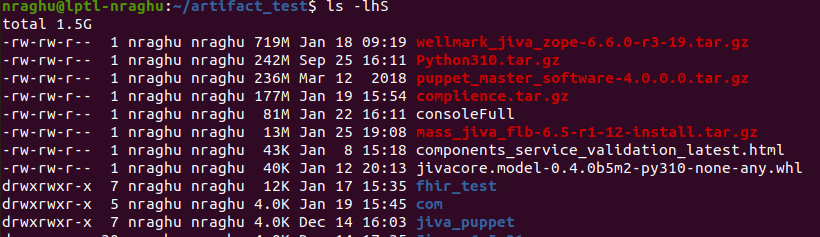
****

$ **gzip -dvf single\_file\_com.gz** --> decompress the file

**Sorting the files using ls command:**

$ ls -lhS --> options l is use long listing format,h is human readable,S is size.

Sort the files based on their file size



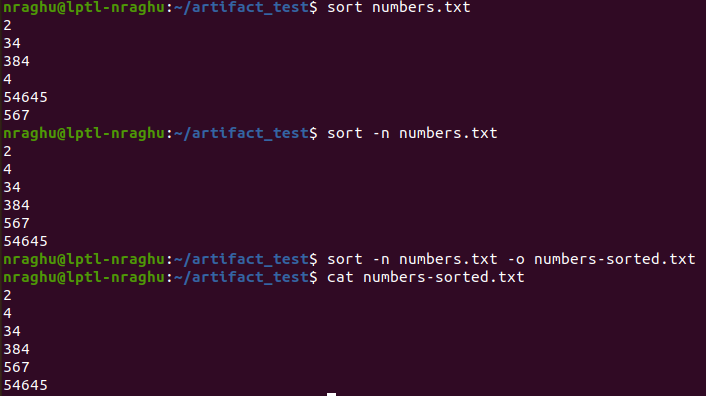
$ **ls -lh --sort=time or ls -lht** --> sort by time,newest is first

$ **ls -lhtr**  --> sort by time and reverse the order means newesr come first because of r option,reverse the list of files order.

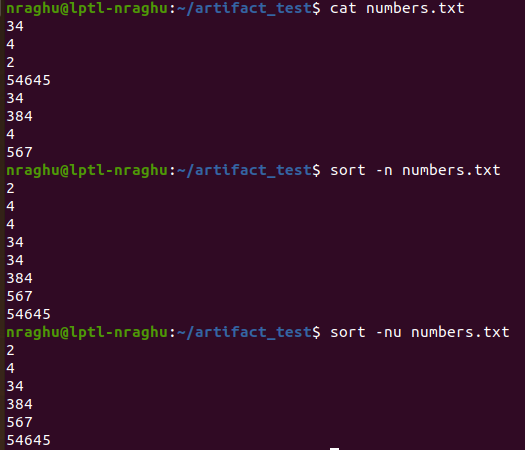
$ **ls --help** --> for more details

**12) Sort command:**

****

****

we need to use the -n option if we want to sort the numbers

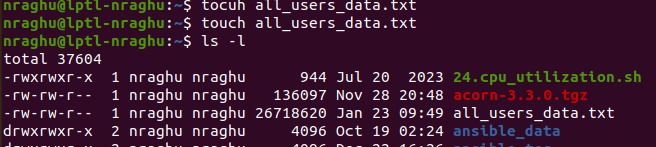
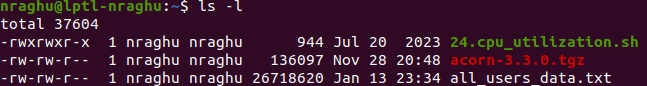


-u option to sort the unique values in file and -r option to reverse the sorted order --> sort -nur numbers.txt --> give descending order numbers

**14)** **touch command** --> Touch command is used to update the modification date and time of file into current date and time

**$ touch raghu.txt**

**before touch command:**

**after touch command:**

**15) mkdir command:**

$ **mkdir -p test2/test3** --> if test2 folder not present,mkdir with -p option create the parent directory(test2 is parent directory) and sub-directory(test3 is sub-directory)

**16) Sed command:**

**:%s/old\_string/new\_string/g** → it will replace the all old\_string with new\_string in opened file

ex: :%s/raghu/ravi/g → wherever raghu word is there, get replaced with ravi

**:s/old\_string/new\_string/g** → it will replace the first appear of old\_string with new\_string in opened file

**17) Chown and Chmod commands:**

**File and directory permission:**

Permissions are represented as a set of ten characters.

For example: **-rwxr-xr--**. Here's how to interpret this:

\*The first character indicates the type of file (e.g., - for a regular file, d for a directory)

\*The next three characters (rwx) show the permissions for the file owner (read, write, and execute).

\*The following three (r-x) show permissions for the users in same group (read and execute, but no write).

\*The final three (r--) show permissions for others users (read only, no write or execute).

4 stands for "read",

2 stands for "write",

1 stands for "execute",

0 stands for "no permission".

**Finding the permission decimal of any permission format:**

0 will represents the deny of persmission

1 will represents the allow of permission

Permissions Finding binary Converting binary into Decimal

-rwrr-wrw- --> 111 101 110 --> 756 is the permission number for permission pattern

**$ chmod 755 raghu.txt**  --> giving onwer have all,read and execute to users in same group and all other users

**$ chmod -x raghu.sh -->** removing the executable permission into all

**$ chmod +x raghu.sh -->** adding executable permission into all

**Important: Don’t assign 777 permission to any file or directory**

**Networking in linux:**

**$ hostname** --> give the hostname of system

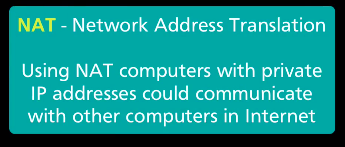
**$ hostname -i** --> ip address of system

**$ hostname -I** --> all p addresses asgined to host machine,due to other network interface,ip address are assigned to host machine

**$ ip address** --> it give the all network interfaces

**$ ifconfig** --> give all network interfaces with good format

**$ ip route** --> show the deafult rout,through that computer connect to internet.



**ping command** --> used to verify the connectivity with remote server from your host machine

**$ ping google.com** --> checking as access available to google.com from my host machine

**nslookup command** ---> bring the details from dns server for the hostname.

Reslove hostname into ip address manually using nslookup

**$ nslookup google.com**

**$ traceroute google.com**

**$whois google.com** -->whois command is used to check owner of domain name ,it’ registered details,expire details etc

**18) ssh command** is used to connect to remote server,ssh protocol establish the encripted communication between hosts over the public network.

Ssh work over tcp and uses the 22 port.

**$ apt-get install openssh-server** --> instaling the ssh

**$ systemctl start ssh** --> starting the ssh process

Connecting to remote server via ssh using root user is blocked by default as security reason.

ssh username@hostname\_or\_ip



Enabling the remote connection as root user:

We can connect remote server via ssh using root user but we need to change some configuration in **/*etc*/ssh/sshd\_config** file

Uncomment the permitroot login line and provide yes and save it



**$passwd root** --> change the password in remote server,usaually don’t change password of root user,get it from server owner

**$ systemctl restart sshd**

ssh [root@hostname\_or\_ip](mailto:root@hostname_or_tp) --> provide password.

Environment variables is variables, different values will be set to environment variable based on environment or user.

**$ env** --> show all environment variable set in current user and terminal.

**awk command:**

The awk command is used for text processing in Linux. Although, the sed command is also used for text processing, but it has some limitations, so the awk command becomes a handy option for text processing. It provides powerful control to the data.

The options can be:

**\* -f program files:**It reads the source code of the script written on the awk command

**\* -F fs:** It is used as the input field separator.

**List students with the specified pattern.**

**awk ' /CS/{print}' student.txt** → in student.txt scan for the student name container the CS and it his name

If we specify the column number on this command, it will print that line only

**awk ‘{print $1,$5}’student.txt**

**awk -F"<test" '{print $2}' student.txt**  -→in student.txt search for the <test and print next part starting after <test

ex: <testsuites> → print the suites>

**set -e and set +e command in shell script:**

The set -e command is used to enable the "exit on error" behavior, which causes the script to terminate immediately if any command returns a non-zero exit status (indicating an error). This is the default behavior for shell scripts.

The set +e command is used to disable the "exit on error" behavior, allowing the script to continue executing even if a command returns a non-zero exit status.

The set -e is very impatant to use in jenkins jobs.

**cat command:**

The cat(concatenate) command is mainly used to view the content of the file.

The cat command also used for folowing cases:

\*Display the text file or any file(yaml file) content → **cat raghu.txt**

\*Read text file → **Name=`cat raghu.txt | grep -i name`**

\*create the new text file → **cat > raghu.txt**

\*file concatenation → **cat score.txt name.txt > report.txt**

To view file: cat -n /*etc*/passwd → show content with line number

Create file: cat > Nagendra.txt

To view big file: more filename and less filename

combine two or more file: cat score.txt name.txt > report.txt

**head command:** head command is used read the first 10 line content by default, if you don’t pass the number

head Names.txt → show only first 10 lines

head -15 Names.txt --> show first 15 lines

head -1 Names.txt → only first line

**tail command:** opposite of head, read the last ten files of file by default

tail Names.txt → show only last 10 lines

tail -3 Names.txt → show only last 3 lines

tail -1 Names.txt → last line

**ps command:**

ps → give the processes running by current user in that terminal and terminal (bash), by default give bash and ps → bash is opned terminal,ps is commad running in that terminal

start any container in that terminal then do ps,you will get below output

PID TTY TIME CMD

1718962 pts/16 00:00:00 bash → terminal

1755239 pts/16 00:00:00 docker --→ process

1755957 pts/16 00:00:00 ps → processor

**Netstat command:**

is used to check on the network related details like which all posts are listening,listening connection type(tcp or udp) ect

$ netstat -tuln → give a list of all tcp,udp listening port details and show only on numerical address

-t → This shows TCP connections.

-u → This shows UDP connections.

-l → this display only listening sockets

-n → show only numerical address