System/OS related commands

To know the OS type:
`$ uname -o`

To check the kernel version:
`$ uname -r`

To list the system hardware:
`$ lshw`

To check system memory:
`$ free -h`

Free memory cache, dentries and inode (with root): `$ echo 3 > /proc/ sys /vm /dr op_ caches`

To search packages for installation:
`$ apt search <pa ckage name>`
` e.g.:`
` $ apt search python -boto`

To uninstall package:
$ sudo apt-get remove <package name

To mount the volume:
(create the directory first to mount volume)
`$ mkdir -p <di rectory path e..g /mount -vo l>`
` $ sudo mount <src path> <above created dir path>`

Find the file (search for a file):
`$ find <dir path> -name <fi len ame> -print`
` e.g. to find app.log in /var directory`
` $ find /var -name app.log –print`` `

File the text string from a given directory:
`$ grep -rIn <search text> <di rectory path>`

User admin Commands

To know the group/user exists on the system:	Check user added or not into system:
`$ getent group <group name>`	`$ id <us ern ame>`
`$ getent passwd <user name>`	` e.g. $ id clouduser1`

User admin Commands (cont)

To know the CPU architecture:
`$ sudo groupadd <group name>`
`e.g. $ sudo groupadd training`

To get the OS name, release, version:
To delete the existing group:
`$ sudo groupdel <group name>`
`e.g. $ sudo groupdel training`

To get the CPU details:
`$ lscpu`

To check the virtual memory stats:
`$ vmstat -S m`

To print the process specific memory utilizations:
`$ ps aux --sort =-%mem`

To installed package:
`$ sudo apt-get install <pa ckage name>`

Print the groups to which the current user is associated:
To list the mounted disk drives:
`$ groups`
`$ df -kh`

To list biggest files from directory (biggest 5):
`$ sudo du -a /dir/ | sort -n -r | head -n 5`
Change the group name:
` $ sudo groupmod –n <new group name> <old group name>`
` e.g. I want to change the groupname 'training' to '`

Search the text string in a directory and print filename containing that string:
`$ sudo groupmod -n cloudadmin training`
`$ file /var -type f -print | xargs grep <search text>`

Editor/Text manipulation commands

awk command for pattern scanning & processing:

```
1. Convert text from upper case to lower case
 $ echo " SAMPLE TEXT" | awk '{print tolowe r($0)}'
 2 Print the next word after found a pattern
 e.g. print the next work after 'reach:' appear in syslog
 $ awk '{for( i=1 ;i< =NF ;i+ +)i f($ i== " rea ch: " )pr int $(i +1) }'
 3. Trim the white spaces
 echo '    aws <co mma nd> help ' | awk '{gsub(/^ +| +$/ ,"")} 1'
 4. Print the selected columns from command output.
 E.g. from df command interested in only filesystem and use% column data
 $ df -kh |awk '{print $1 " " $5}'
 5. use regex as a field separator,
 e.g input field separator as / or = e.g.
 $ awk -F"= |:" '{print $2}'
 input text as 'dnsco nf= /et c/r eso lv.c onf' or 'dnsco nf: /et c/r eso lv.c onf' for both same comma
nd will work
```

Editor/Text manipulation commands (cont)

cut, cutting out the sections from lines:

```
$  cut -d "delim iter" -f <fi eld> <fi le.t xt>      file
 a) cut the line on space and print 1st to 4th field
 $ echo "my phone number is 887389 3" | cut -d " " -:
change the delimiter space with column
 $ echo "hello/wor ld" | cut -d " " -f 1-2 --outp ut-
%
```

Sort is to sort file, records, lists etc:

```
a) sort file contents of text file (-r option to reve
% sort file.txt
b) sort based on column number
$ df -kh | sort -k 5
```

Editor/Text manipulation commands (cont)

tee, is a command which reads the standard input and write into standard output and also to file. This is used to redirect logs or data to a file:

```
a) let we have two log files, file1.log & file2.log and we need to append file1.log to file2.log
$ cat file1.log | tee -a file2.log
b) redirect the command output to a log file
$ du --max- depth=1 -h | sort -hr 2>&1 | tee du.log
```

Network related commands

nslookup, Query internet domain name server
```
a) find the IP from fqdn
$ nslookup google.com
b) check the fqdn from ip address
$ nslookup 172.21 7.1 67.174
```

netstat, print the network stats, listening ports etc
```
a) print all listening ports
$ netstat -plunt
b) check if server is listening on port 8090 or not
$ netstat -plunt | grep 8090
c) list stats of all ports
$ netstat -s
d) display pid of listening ports
$ netstat -pt
e) list network interfaces
$ netstat -i
```

Network related commands (cont)

scp, secure copy from/to remote host

```
a) copy file from remote host
(syntax) scp <file name> @< remote
irpat h>
e.g. $ scp id_rsa.pem rakesh @19 2.1 68.5 6.12 ):/ho
b) copy local file to remote host
$ scp -i id_rsa.pem data.txt rakesh @19 2.1 68.5 6.
```

lsof, list open files by processes
```
a) list open files by specific user
lsof -u <us ern ame>
b) find processes running on specific port
lsof -i :9090
```

curl ifconfig.co, get the public ip of the machine
```
$ curl ifconf ig.co
```

ufw, manage firewall
```
a) check firewall status
$ sudo ufw status
b) enable /di sable firewall
$ sudo ufw enable /di sable
```

sed - stream
```
a) Find
b) sed
$ sed
c) del
$ sed
(check
```

sed - stream editor

Sed - perform basic transformations on an input stream i.e. a file or a stream input from the pipeline.

Example: replace all occurrences of TCP to UDP in network.log file

```
$ sed 's/TCP /UDP/' networ k.log > modifi ed- net wor k.log
```

Common sed command line options

-i : edit in place i.e. sed -i 's/TCP/UDP/' network.log

-n <line number>p e.g. print on line no 30 from network.log sed -n '30p' network.log

-e : expression e.g. sed -e 's/TCP/UDP/' network.log

[here 's' stand for substitute]

Basic regular expression overview

. : (dot) matches any single character

* : matches a sequence of zero or more instances e.g.

```
 $ echo 'hostn ame =lo cal hos t.m yor g.com' | sed 's/l.l/myap pse
 rver/' *
```

^ : indicates the beginning of the line

$: indicates the end of the line

[list] or **[^list]** :matches any single char in a list. e.g. [1-9] matches any digit from 1 to 9

\+ : As *, matches any single or multiple instances of chars

\?: As *, matches any zero or one instances of chars

\{i\}: matches exactly *i* sequences 'i is between 0 to 255'

\{i,\} : matches more than or equal to *i* sequences

regex1|regex2 : matches regular expression 1 or regular expression 2

[a-z0-9A-z]: matches any ASCII chars

====================================

Examples

----

find and replace any os name with Ubuntu

e.g.

1.

input: osname: centOS7

output: osname: Ubantu

2.

input: winOS: Windows-10

output: osname: Ubantu

3.

input: MacOS:Mac10

output: osname: Ubantu

Solution:

key=echo " <input string >" | cut -d " :" -f 1

echo "<input string" | sed -e 's/^$key:\s.$/$key: Ubantu/g'

first store the key i.e. left side label

^ - start of line

\s* - zero or more space charaters

.* - any zero or multiple charaters

sed - stream editor (cont)

$ - end of the line

Extract the line containing IP address from a file

sed -rn '/([0-9]{1,3}\.){3}[0-9]{1,3}/p' /etc/hosts