Cheatography

Linux Commands to know by DevOps Engineer Cheat Sheet

by rakesh zingade via cheatography.com/71377/cs/18096/

System/OS related commands

To know the OS type:

```
$ uname -o
```

To know the CPU architecture:

```
$ uname -m
```

To check the kernel version:

```
$ uname -r
```

To get the OS name, release, version:

```
$ cat /etc/os-release
```

To list the system hardware:

```
$ lshw
```

To get the CPU details:

```
$ lscpu
```

To check system memory:

```
$ free -h
```

To check the virtual memory stats:

```
$ vmstat -S m
```

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

To print the process specific memory utilizations:

```
$ ps aux --sort=-%mem
```

To search packages for installation:

```
$ apt search <package name>
e.g.:
$ apt search python-boto
```

To installed package:

```
$ sudo apt-get install <package name>
```

To uninstall package:

$ sudo apt-get remove <package name

To list the mounted disk drives:

```
$ df -kh
```

To mount the volume:

(create the directory first to mount volume)

```
$ mkdir -p <directory path e..g /mount-vol>
$ sudo mount <src path> <above created dir path>
```

To list biggest files from directory (biggest 5):

```
$ sudo du -a /dir/ | sort -n -r | head -n 5
```

Find the file (search for a file):

```
$ find <dir path> -name <filename> -print
e.g. to find app.log in /var directory
```

$ find /var -name app.log –print`

Search the text string in a directory and print filename containing that string:

```
$ file /var -type f -print | xargs grep <search text>
```

File the text string from a given directory:

```
$ grep -rIn <search text> <directory path>
```

User admin Commands

To know the group/user exists on the system:

```
$ getent group <group name>
$ getent passwd <user name>
```

Check user added or not into system:

```
$ id <username>
e.g. $ id clouduser1
```

To create a new group:

```
$ sudo groupadd <group name>
e.g. $ sudo groupadd training
```

Modify existing user, add user to group:

```
$ sudo usermod -aG <group name> <username>
e.g. $ sudo usermod –aG sudo clouduser1
```

To delete the existing group:

```
$ sudo groupdel <group name>
e.g. $ sudo groupdel training
```

Add user's home directory (example for clouduser1):

```
$ sudo mkdir -p /home/user1
$ sudo chown clouduser1:clouduser1 /home/user1 $ ls -l /home
drwxr-xr-x 2 clouduser1 clouduser1 4096 Nov 18 12:13 user1
$ sudo usermod -d /home/user1 clouduser1
$ id clouduser1
uid=1002(clouduser1) gid=1003(clouduser1) groups=1003(clouduser1),27(sudo)
$ su - clouduser1
$ pwd
/home/user1
```

Print the groups to which the current user is associated:

```
$ groups
```

Delete existing user with all files associated with user:

```
$ sudo userdel -r clouduser1
$ id clouduser1
id: 'clouduser1': no such user
```

User admin Commands (cont)

Change the group name:

```
$ sudo groupmod -n <new group name> <old group name>
e.g. I want to change the groupname 'training' to 'cloudadmin'
$ sudo groupmod -n cloudadmin training
```

Add user to system:

```
$ sudo adduser <user name>
e.g. add clouduser1 to system
$ sudo adduser clouduser1
```

Editor/Text manipulation commands

awk command for pattern scanning & processing:

```
1. Convert text from upper case to lower case
$ echo "SAMPLE TEXT" | awk '{print tolower($0)}'
2 Print the next word after found a pattern
e.g. print the next work after 'reach:' appear in syslog file
$ awk '{for(i=1;i<=NF;i++)if($i=="reach:")print $(i+1)}' /var/log/syslog
3. Trim the white spaces
echo ' aws <command> help ' | awk '{gsub(/^ +| +$/,"")}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 'dnsconf=/etc/resolv.conf' or 'dnsconf:/etc/resolv.conf' for both same command will work
```

diff, get the differences by comparing files line by line

```
$ diff file1.txt file2.txt
```

Editor/Text manipulation commands (cont)

cut, cutting out the sections from lines:

```
$ cut -d "delimiter" -f <field> <file.txt>
a) cut the line on space and print 1st to 4th field
$ echo "my phone number is 8873893" | cut -d " " -f 1-4 b)
change the delimiter space with column
$ echo "hello world" | cut -d " " -f 1-2 --output-delimiter=%
```

Uniq, is a command that filter out the duplicates

```
a) fetch repeated/duplicate lines from a file
$ uniq -d <file.txt>
b) get the count of uniq lines in a file {nl}} $ uniq -c <filename>
```

Sort is to sort file, records, lists etc:

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

tr is to translate or delete characters

```
a) translate all lowercase letters to upper case in a file
$ cat filename | tr "[:lower:]" "[:upper:]"
b) translate white spaces to tabs
$ cat filename | tr [:space:] '\t'
c) remove all digits from string
$ echo "my mob number 88039223" | tr -d [:digit:]
d) Just get the digits from string
$ echo "my mob number 88039223" | tr -cd [:digit:]
```

Editor/Text manipulation commands (cont)

tee, is a command which reads the standard input and write into standard output and also to a 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
```

sed - stream editor, it is used for filtering and transforming text

a) Find and replace text

```
$ echo 'Unix is multi-user OS' | sed 's/Unix/Linux/'
```

b) delete particular line from a file (e.g. 5th line)

```
$ sed '5d' file.txt
```

c) delete 5th to 10th line from a file

```
$ sed '5,10d' file.txt
```

(check more details in a separate block)

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 172.217.167.174
```

netstat, print the network stats, listening ports etc

a) print all listening ports

```
$ netstat -plunt
```

b) check if server is listening on port 8080 or not

```
$ netstat -plunt | grep 8080
```

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 remote host

a) copy file from remote host (syntax) scp -i <pem file> <username>@<remote ip>: <filepath> <local destination dirpath>

e.g. $ scp id_rsa.pem rakesh@192.168.56.120:/home/rakesh/data.txt .

b) copy local file to remote host

```
$ scp -i id_rsa.pem data.txt rakesh@192.168.56.120:/home/rakesh
```

nmap, check open ports on server, generally used as network exploration tool

a) check open ports on remote host

```
$ nmap 172.217.27.206
```

b) list out all machines from network that responds to ping

```
$ nmap -sP 192.168.56.0/24
```

c) scan and print ports, os & other details about remote host

```
$ sudo nmap -sS -A -T4 192.168.56.150
```

lsof, list open files by processes

a) list open files by specific user

```
lsof -u <username>
```

b) find processes running on specific port

```
$ lsof -i TCP:9090
```

netcat, debug and investigate network

a) start a dummy listening server on port 8080

```
$ netcat -l 8080
```

b) send data over some port to server

```
$ netcat <remote server ip> <port>
```

e.g.

```
$ netcat 192.168.56.120 8080
```

(press EOF CNTR+D at end)

Network related commands (cont)

curl ifconfig.co, get the public ip of the machine

```
$ curl ifconfig.co
```

route, show/=manipulate IP routing table

```
a) show current routing table
$ route -n
b) add route to particular network e.g.
make 10.10.76.0/24 accessible via gw 10.10.76.1
$ route add -net 10.10.76.0 netmask 255.255.255.0 gw 10.10.76.1
```

ufw, manage firewall

```
a) check firewall status
$ sudo ufw status
b) enable/disable firewall
$ sudo ufw enable/disable
```

hostname, provides hostname of a machine

```
a) get hostname
$ sudo hostname
```

sed - stream editor

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

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

```
$ sed 's/TCP/UDP/' network.log > modified-network.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 'hostname=localhost.myorg.com' | sed 's/l.l/myappserver/' *
```

^ : 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

sed - stream editor (cont)

\{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

$ - 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