**SSH**

**Connecting:**

Connect to a server (default port 22)

$ ssh root@192.168.1.5

Connect on a specific port

$ ssh root@192.168.1.5 -p 6222

Connect via pem file (0400 permissions)

$ ssh -i /path/file.pem [root@192.168.1.5](mailto:root@192.168.1.5)

**Executing:**

Executes remote command

$ ssh root@192.168.1.5 'ls -l'

Invoke a local script

$ ssh root@192.168.1.5 bash < script.sh

Compresses and downloads from a server

$ ssh root@192.168.1.5 "tar cvzf - ~/source" > output.tgz

**Config location:**

/etc/ssh/ssh\_config System-wide config

~/.ssh/config User-specific config

~/.ssh/id\_{type} Private key

~/.ssh/id\_{type}.pub Public key

~/.ssh/known\_hosts Logged in host

~/.ssh/authorized\_keys Authorized login key

**Config sample:**

Host server1

HostName 192.168.1.5

User root

Port 22

IdentityFile ~/.ssh/server1.key

Launch by alias

$ ssh server1

**SCP**

Copies from remote to local

$ scp user@server:/dir/file.ext dest/

Copies between two servers

$ scp user@server:/file user@server:/dir

Copies from local to remote

$ scp dest/file.ext user@server:/dir

Copies a whole folder

$ scp -r user@server:/dir dest/

Copies all files from a folder

$ scp user@server:/dir/\* dest/

Copies from a server folder to the current folder

$ scp user@server:/dir/\* .

**SCP Options:**

scp -r Recursively copy entire directories

scp -C Compresses data

scp -v Prints verbose info

scp -P 8080 Uses a specific Port

scp -B Batch mode (Prevents password)

scp -p Preserves times and modes

**ProxyJump**

$ ssh -J proxy\_host1 remote\_host2

$ ssh -J user@proxy\_host1 user@remote\_host2

Multiple jumps

$ ssh -J user@proxy\_host1:port1,user@proxy\_host2:port2 user@remote\_host3

**ssh-copy-id**

$ ssh-copy-id user@server

Copy to alias server

$ ssh-copy-id server1

Copy specific key

$ ssh-copy-id -i ~/.ssh/id\_rsa.pub user@servr

**SSH keygen:**

$ ssh-keygen -t rsa -b 4096 -C "your@mail.com"

-t Type of key

-b The number of bits in the key

-C Provides a new comment

Generate an RSA 4096 bit key with email as a comment

**Generate**

Generate a key interactively

$ ssh-keygen

Specify filename

$ ssh-keygen -f ~/.ssh/filename

Generate public key from private key

$ ssh-keygen -y -f private.key > public.pub

Change comment

$ ssh-keygen -c -f ~/.ssh/id\_rsa

Change private key passphrase

$ ssh-keygen -p -f ~/.ssh/id\_rsa

**known\_hosts**

Search from known\_hosts

$ ssh-keygen -F <ip/hostname>

Remove from known\_hosts

$ ssh-keygen -R <ip/hostname>

**Netstat**

**Statistics**

All connections on port 80

$ netstat -anp | grep :80

Netstat Help

$ netstat -h

**Listening**

netstat -ltunp All Listening ports

netstat -ltn Listening TCP ports

netstat -lun Listening UDP ports

netstat -lx Listening Unix ports

**Connections**

netstat -a All connections

netstat -at All TCP connections

netstat -au All UDP connections

**Statistics**

netstat -s Display statistics

netstat -st Display TCP statistics

netstat -su Display UDP statistics

**Networks**

netstat -i Show network interfaces

netstat -ie Show network interfaces extended info

**Routing**

netstat -r Show routing table

netstat -rn Show routing table, don't resolve hosts

LSOF

Lsof

lsof meaning LiSt Open Files is used to find out which files are open by which process

$ lsof

$ sudo lsof -u root

Port-specific

$ lsof -i :8080

$ lsof -i :80 -i :22

$ lsof -i TCP:22

$ lsof -i TCP:1-1024

$ lsof -i UDP

$ lsof -i @192.168.1.5

Process-specific

$ lsof -c mysql

$ lsof -c java

$ lsof -c ssh

$ lsof -c nginx

$ lsof -c ssh -c httpd

User-specific

$ lsof -u www-data

$ lsof -u www-data -u ubuntu

$ lsof -i -u ^root # Except certain user

Network-specific

$ lsof -i 4 # IPv4 only

$ lsof -i 6 # IPv6 only

PID-specific

$ lsof -p 1753

$ lsof -p ^3 # Except certain pids

Filename-specific

$ lsof /var/log/messages

$ lsof /etc/passwd

Directory-specific

$ lsof +D /var/log # Within a directory

Kill

$ kill -9 `lsof -t -u apache`

$ kill -9 $(lsof -t -i :8080)

GREP

grep command:

- Grep is a filter command, it is used to search a string in a given file.

- Simple grep command syntax:

grep [options] “string/pattern” file/files

cat file | grep [options] “string/pattern”

echo “some text” | grep [options] “string/pattern”

- Basic options: -i -w -v -o -n -c -A -B -C -r -l -h

- Advanced Options: -e -f and –E

Grep command with basic options:

grep

“string/pattern” file/files

grep [options] “string/pattern” file/files

Basic options: -i -w -v -o -n -c -A -B -C -r -l -h

-i To ignore case for matching/searching

-w To match a whole word

-v To display the lines which are not having given string or text

-o To print/display only matched parts from matched lines

-n To display the matched line numbers

-c To display matched number of lines

-A To display N lines after match (grep –A 3 “string” file)

-B To display N lines before match

-C To display N lines around match

-r To search under current directory and its sub-directory

-l To display only file names

-h To hide file names

-i grep -i ^DA demo.txt Forgets about case sensitivity

-w grep -w "of" demo.txt Search only for the full word

-A grep -A 3 'Exception' error.log Display 3 lines after matching string

-B grep -B 4 'Exception' error.log Display 4 lines before matching string

-C grep -C 5 'Exception' error.log Display 5 lines around matching string

-r grep -r 'quickref.me' /var/log/nginx/ Recursive search (within subdirs)

-v grep -v 'warning' /var/log/syslog Return all lines which don't match the pattern

-e grep -e '^al' filename Use regex (lines starting with 'al')

-E grep -E 'ja(s|cks)on' filename Extended regex (lines containing jason or jackson)

-c grep -c 'error' /var/log/syslog Count the number of matches

-l grep -l 'robot' /var/log/\* Print the name of the file(s) of matches

-o grep -o search\_string filename Only show the matching part of the string

-n grep -n "go" demo.txt Show the line numbers of the matches

Wildcards

. Any character.

? Optional and can only occur once.

\* Optional and can occur more than once.

+ Required and can occur more than once.

Quantifiers

{n} Previous item appears exactly n times.

{n,} Previous item appears n times or more.

{,m} Previous item appears n times maximum.

{n,m} Previous item appears between n and m times.

[:alpha:] Any lower and upper case letter.

[:digit:] Any number.

[:alnum:] Any lower and upper case letter or digit.

[:space:] Any whites­pace.

[A-Z­a-z] Any lower and upper case letter.

[0-9] Any number.

[0-9­A-Z­a-z] Any lower and upper case letter or digit.

^ Beginning of line.

$ End of line.

^$ Empty line.

\< Start of word.

\> End of word.

**CUT Command**

Cut command:

The 'cut‘ command is a powerful tool to extract parts of each line from a file.

 It is based on

 Byte Position

 Character Position

 Fields based on delimiter (by default delimiter is the tab)

Cut command syntax:

 cut [options] <positions(fields) /range of positions(fields)> <input\_file>

 cat file | cut [options] <positions(fields) /range of positions(fields)>

 Options: -b -c and -f

 Rages:

2 only second byte/character/filed

2- second byte/character/filed to last

-7 first to seven

3,5 third and fift

SED:

Syntax

$ sed [options] command [input-file]

With pipeline

$ cat report.txt | sed 's/Nick/John/g'

$ echo '123abc' | sed 's/[0-9]+//g'

-i sed -ibak 's/On/Off/' php.ini Backup and modify input file directly

-E sed -E 's/[0-9]+//g' input-file Use extended regular expressions

-n sed -n '3 p' config.conf Suppress default pattern space printing

-f sed -f script.sed config.conf Execute sed script file

-e sed -e 'command1' -e 'command2' input-file Execute multiple sed commands

$ echo "hello world" | sed -e 's/h/H/g' -e 's/w/W/g'

Hello World

$ echo 's/h/H/g' >> hello.sed

$ echo 's/w/W/g' >> hello.sed

$ echo "hello world" | sed -f hello.sed

Hello World

$ sed 's/old/new/g' file.txt

$ sed 's/old/new/g' file.txt > new.txt

$ sed 's/old/new/g' -i file.txt

$ sed 's/old/new/g' -i.backup file.txt

Command Example Description

p sed -n '1,4 p' input.txt Print lines 1-4

p sed -n -e '1,4 p' -e '6,7 p' input.txt Print lines 1-4 and 6-7

d sed '1,4 d' input.txt Print lines except 1-4

w sed -n '1,4 w output.txt' input.txt Write pattern space to file

a sed '2 a new-line' input.txt Append line after

i sed '2 i new-line' input.txt Insert line before

n Print pattern space, empty pattern space, and read next line

x Swap pattern space with hold space

h Copy pattern space to hold space

H Append pattern space to hold space

g Copy hold space to pattern space

G Append hold space to pattern space

$ sed 's/old/new/[flags]' [input-file]

g Global substitution

1,2... Substitute the nth occurrence

p Print only the substituted line

w Write only the substituted line to a file

I Ignore case while searching

e Substitute and execute in the command line

b label Branch to a label (for looping)

t label Branch to a label only on successful substitution

(for looping)

:label Label for the b and t commands (for looping)

N Append next line to pattern space

P Print 1st line in multi-line

D Delete 1st line in multi-line

/ | ^ @ ! # Substitution delimiter can be any character

& Gets the matched pattern

( ) \1 \2 \3 Group using ( and ).

Use \1, \2 in replacement to refer the group

Replace all occurrences of a string

$ sed 's/old/new/g' file.txt

Replace only the nth occurrence of a string

$ sed 's/old/new/2' file.txt

Replace replace a string only on the 5th line

$ sed '5 s/old/new/' file.txt

Replace "world" with "universe" but only if the line begins with "hello"

$ sed '/hello/s/world/universe/' file.txt

Remove "" from the end of each line

$ sed 's/\\$//' file.txt

Remove all whitespace from beginning of each line

$ sed 's/^\s\*//' file.txt

Remove comments. Even those that are at the end of a line

$ sed 's/#.\*$//' file.txt

Number line of a file (simple left alignment)

$ sed = file.txt | sed 'N;s/\n/\t/'

Number line of a file (number on left, right-aligned)

$ sed = file.txt | sed 'N; s/^/ /; s/ \*\(.\{6,\}\)\n/\1 /'

Number line of file, but only print numbers if line is not blank

$ sed '/./=' file.txt | sed '/./N; s/\n/ /'

Count lines (emulates "wc -l")

$ sed -n '$='

ontains "hello"

$ sed '/hello/i Example: ' file.txt

file.txt

Delete the last line in file

$ sed '$d' file.txt

Delete lines starting with "Hello"

$ sed '/^Hello/d' file.txt

Delete all empty lines

$ sed '/^$/d' file.txt

Delete lines starting with "#"

$ sed '/^#/d' file.txt

Double space

$ sed G

Delete all blank lines and double space

$ sed '/^$/d;G'

Triple space a file

$ sed 'G;G'

Undo double-spacing

$ sed 'n;d'

Insert a blank line above line which matches "regex"

$ sed '/regex/{x;p;x;}'

Insert a blank line below line which matches "regex"

$ sed '/regex/G'

Insert a blank line around line which matches "regex"

$ sed '/regex/{x;p;x;G;}'