Linux Fundamentals

Project: Info Extractor

Shaun Sng (S17)

Feb 2024

Table of Contents

1. Introduction

Summary - This project entails preparing a script to automate retrieval of system and network information of the machine from where it is run, for example public & internal IP, memory utilisation and active services.

Aim - This project aims to familiarise (and test) students on typical Linux commands to extract such information, text manipulation needed to isolate specific portions, and how to prepare this in a script.

The likely longer term intent of this project is provide a basic framework for more complex scripting subsequently - for instance when enumerating an unfamiliar network or conducting brute forcing of several machines.

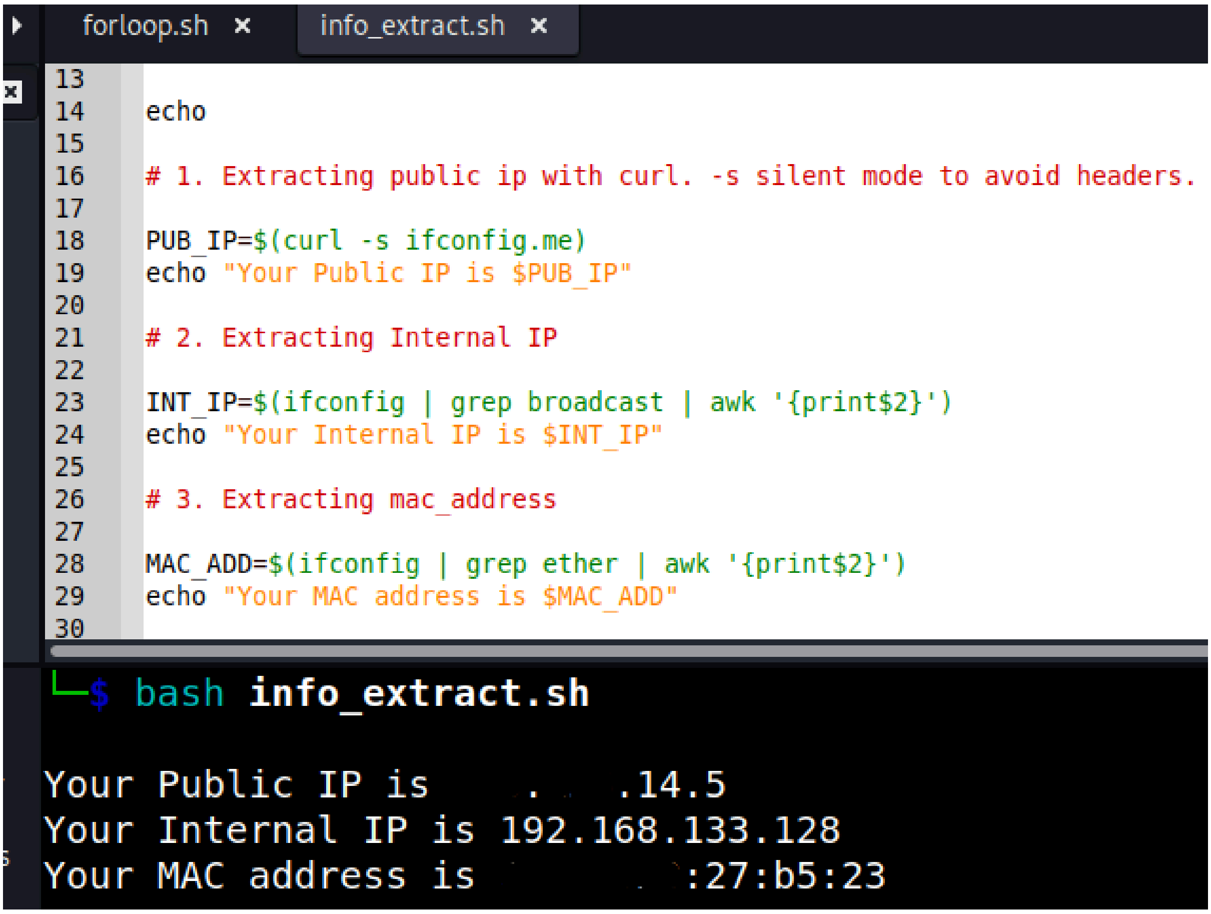
2. Methodologies

2.1 Public IP | Internal IP | MAC address

This section outlines how the script extracts the initial three pieces of information - public and internal IP, and MAC address.

curl with ifconfig.me is used to retrieve public IP, with -s silent mode flag to limit output to just the IP address.

Ifconfig piped to the “broadcast” keyword isolates the specific line we are interested in, with the awk pipe isolating the internal IP. A similar approach is used to extract MAC address, just with a different keyword of “ether”

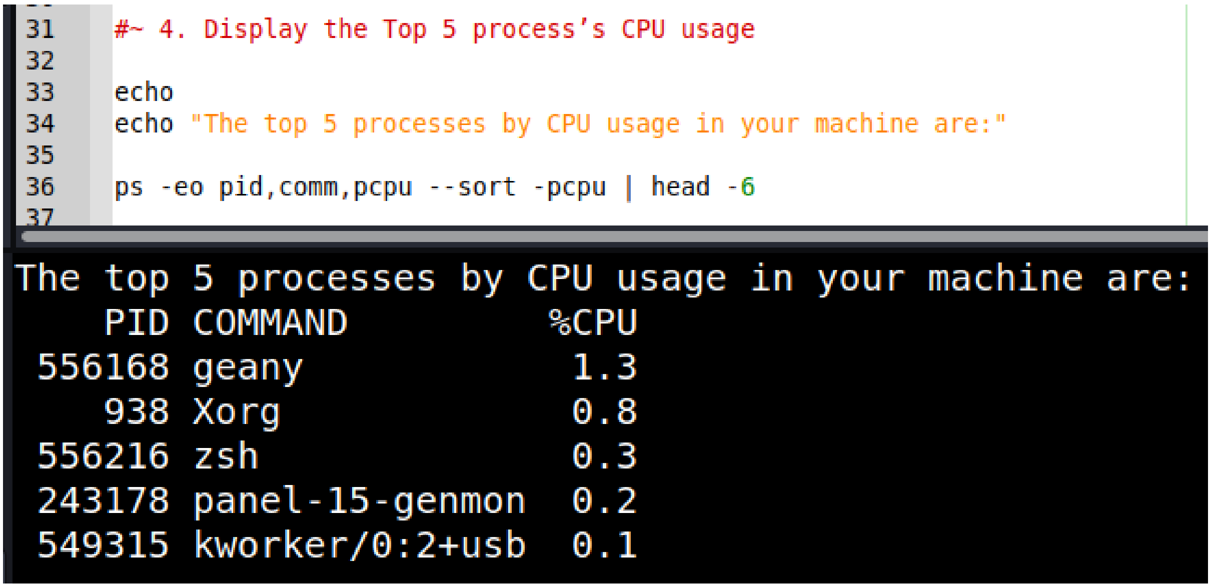


```
echo

# 1. Extracting public ip with curl. -s silent mode to avoid headers.

PUB_IP=$(curl -s ifconfig.me)
echo "Your Public IP is $PUB_IP"

# 2. Extracting Internal IP

INT_IP=$(ifconfig | grep broadcast | awk '{print$2}')
echo "Your Internal IP is $INT_IP"

# 3. Extracting mac_address

MAC_ADD=$(ifconfig | grep ether | awk '{print$2}')
echo "Your MAC address is $MAC_ADD"
```

```
└─$ bash info_extract.sh

Your Public IP is [redacted].14.5
Your Internal IP is 192.168.133.128
Your MAC address is [redacted]:27:b5:23
```

2.2 Top 5 process by CPU usage

As *ps* command provides a static snapshot of processes, it was used instead of *top*. Flag -o was used to limit the output to three columns of interest - the command name, CPU utilisation and process ID (which serves both as a unique identifier and label). Results were sorted by CPU utilisation and then head to limit output to the top 5 rows[1] (six with column headers).



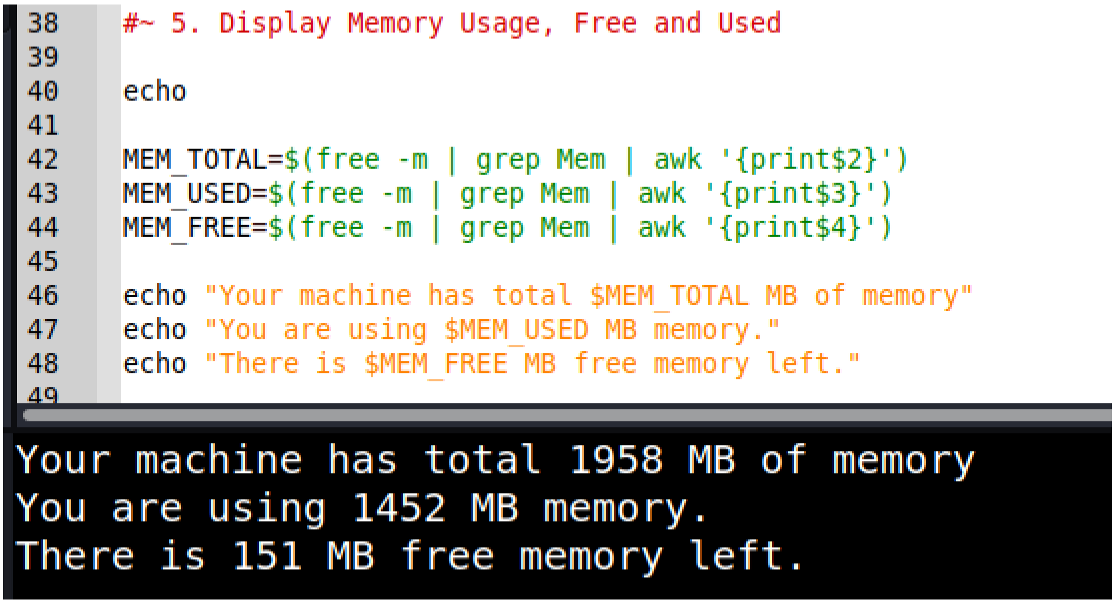
```
#~ 4. Display the Top 5 process’s CPU usage

echo
echo "The top 5 processes by CPU usage in your machine are:"

ps -eo pid,comm,pcpu --sort -pcpu | head -6
```

```
The top 5 processes by CPU usage in your machine are:
    PID COMMAND         %CPU
 556168 geany            1.3
    938 Xorg             0.8
 556216 zsh              0.3
 243178 panel-15-genmon  0.2
 549315 kworker/0:2+usb  0.1
```

[1] Networkworld article in reference provided the relevant ps flags.

2.3 Display Memory Usage

The *free* command provides memory (and swap file) utilisation in an already succinct output. Simple *grep* and *awk* manipulations isolate the specific row and column information we need. For completeness, this script provides total, used and free memory results.

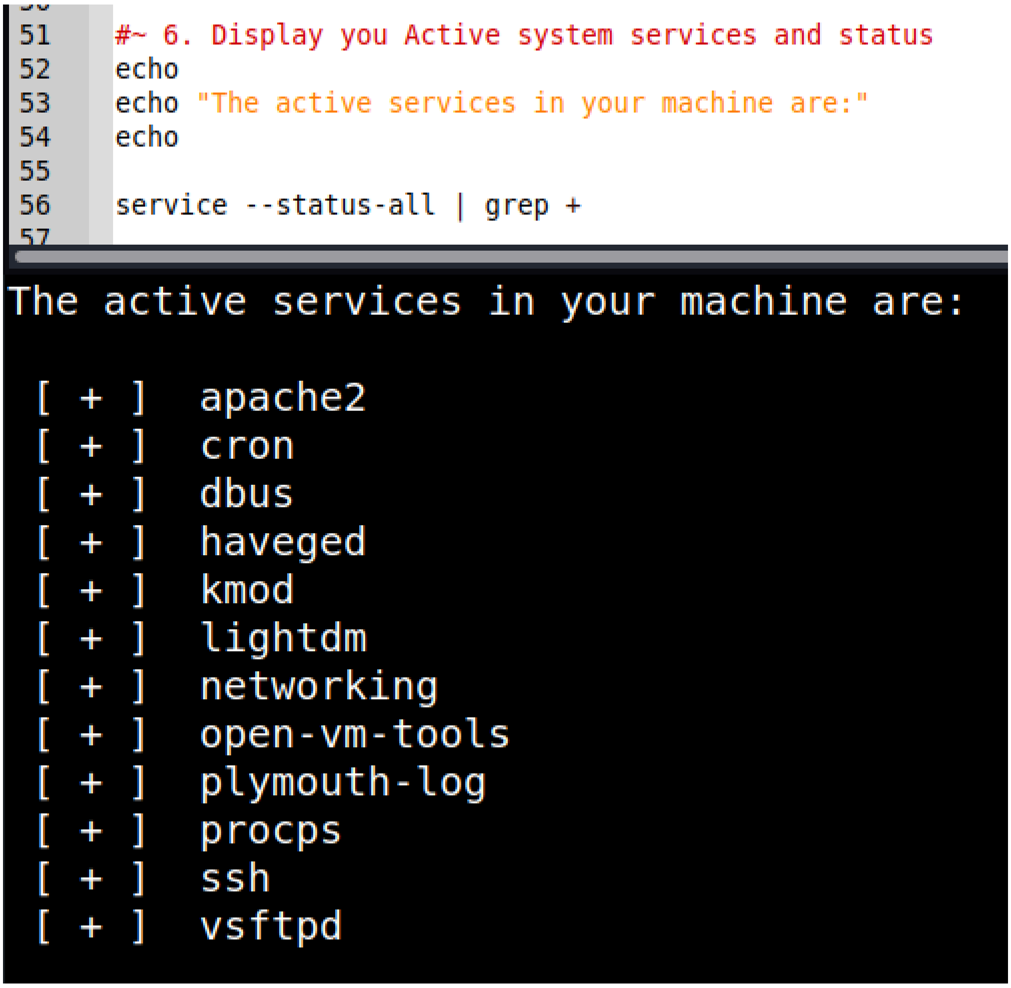


```
#~ 5. Display Memory Usage, Free and Used

echo

MEM_TOTAL=$(free -m | grep Mem | awk '{print$2}')
MEM_USED=$(free -m | grep Mem | awk '{print$3}')
MEM_FREE=$(free -m | grep Mem | awk '{print$4}')

echo "Your machine has total $MEM_TOTAL MB of memory"
echo "You are using $MEM_USED MB memory."
echo "There is $MEM_FREE MB free memory left."
```

```
Your machine has total 1958 MB of memory
You are using 1452 MB memory.
There is 151 MB free memory left.
```

2.4 Display Active system services and status

A *service* command is used, piped to *grep* **“+”** keyword used to output only the active services.

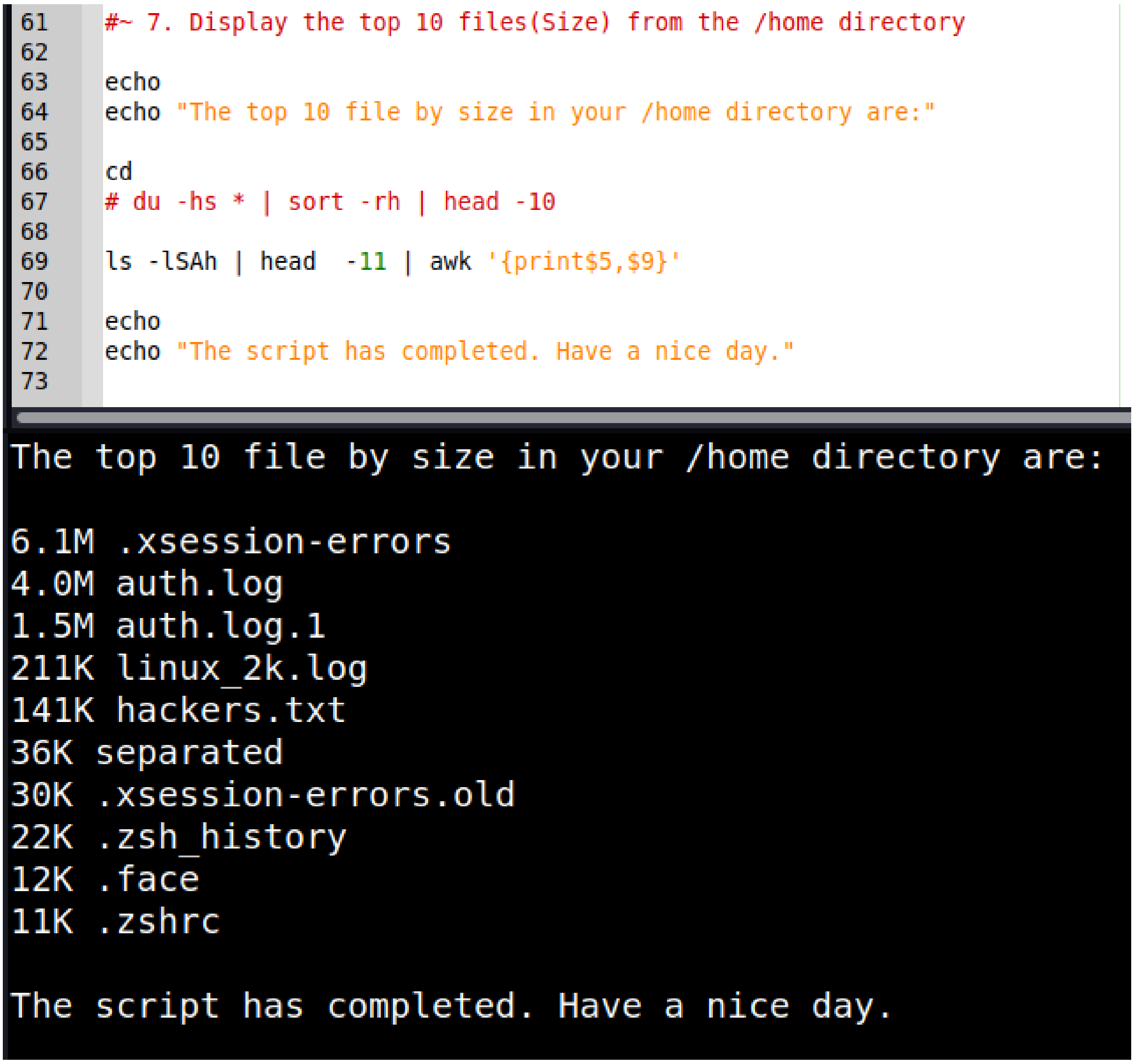


```
#~ 6. Display you Active system services and status
echo
echo "The active services in your machine are:"
echo

service --status-all | grep +
```

```
The active services in your machine are:

 [ + ]  apache2
 [ + ]  cron
 [ + ]  dbus
 [ + ]  haveged
 [ + ]  kmod
 [ + ]  lightdm
 [ + ]  networking
 [ + ]  open-vm-tools
 [ + ]  plymouth-log
 [ + ]  procps
 [ + ]  ssh
 [ + ]  vsftpd
```

2.5 Display the top 10 files by size from the /home directory

An *ls* command with several flags was used after *cd* into the home directory. Flags -l for listing with details, -S to sort by file size, -A for almost all (including hidden files) and -h for human readable memory sizes. We pipe to limit rows to top 10 (11 as there is a summary of numbers of results returned), with a final awk to simplify the output to just file size and file name.



```
#~ 7. Display the top 10 files(Size) from the /home directory

echo
echo "The top 10 file by size in your /home directory are:"

cd
# du -hs * | sort -rh | head -10

ls -lSAh | head  -11 | awk '{print$5,$9}'

echo
echo "The script has completed. Have a nice day."

```

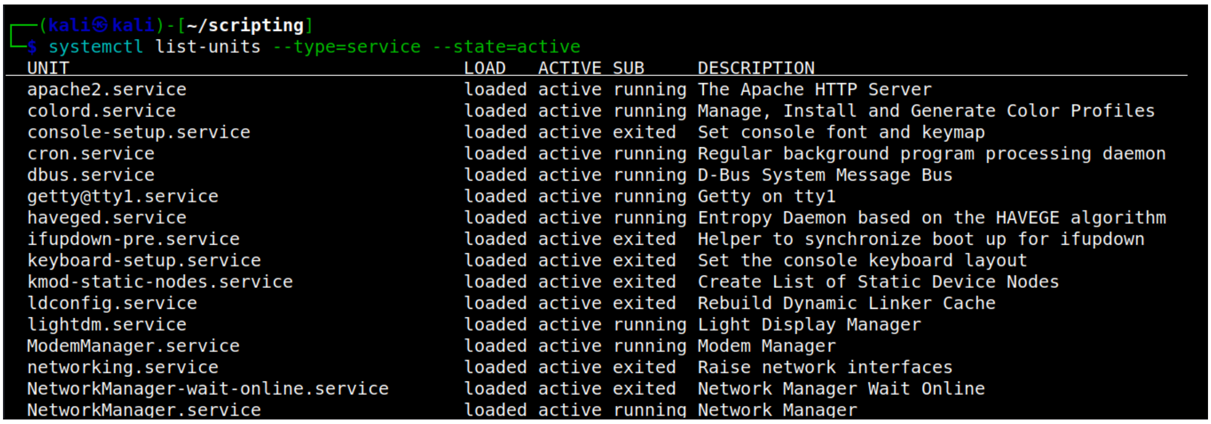
```
The top 10 file by size in your /home directory are:

6.1M .xsession-errors
4.0M auth.log
1.5M auth.log.1
211K linux_2k.log
141K hackers.txt
36K separated
30K .xsession-errors.old
22K .zsh_history
12K .face
11K .zshrc

The script has completed. Have a nice day.
```

3. Discussion

This section considers alternative approaches for Active system services and large file extractions.

3.1 System Services

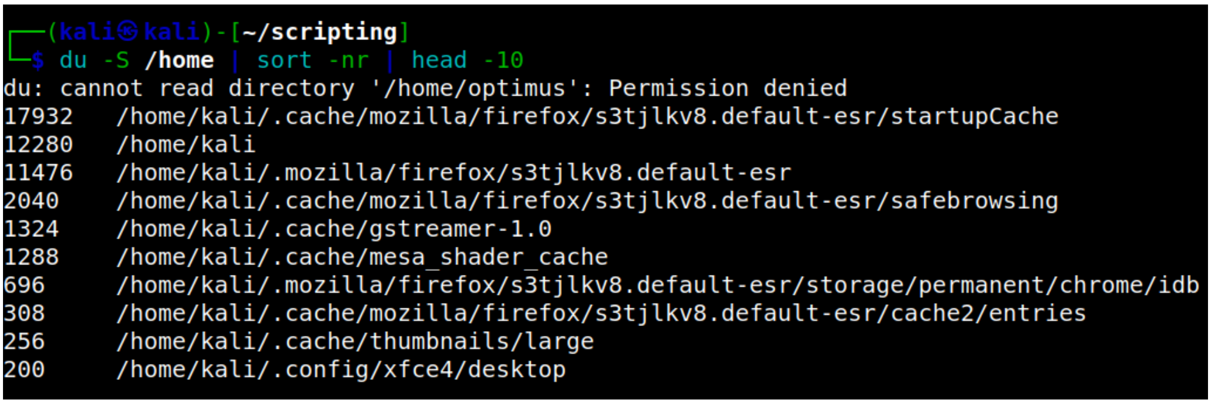
Instead of the simpler *service* command used in this script, more comprehensive results could be retrieved using the *systemctl* command. However, using this produced many results (55 in my system) which display over several pages, requiring the user to hit “enter” several times for the output to complete and script to proceed to the next section. This requirement for user intervention does not seem aligned with the intent of an automated script. As such it was determined using the simpler *service* command was more appropriate for this project.



```
┌──(kali㉿kali)-[~/scripting]
└─$ systemctl list-units --type=service --state=active
  UNIT                                   LOAD   ACTIVE SUB     DESCRIPTION
  apache2.service                        loaded active running The Apache HTTP Server
  colord.service                         loaded active running Manage, Install and Generate Color Profiles
  console-setup.service                  loaded active exited  Set console font and keymap
  cron.service                           loaded active running Regular background program processing daemon
  dbus.service                           loaded active running D-Bus System Message Bus
  getty@tty1.service                     loaded active running Getty on tty1
  haveged.service                        loaded active running Entropy Daemon based on the HAVEGE algorithm
  ifupdown-pre.service                   loaded active exited  Helper to synchronize boot up for ifupdown
  keyboard-setup.service                 loaded active exited  Set the console keyboard layout
  kmod-static-nodes.service              loaded active exited  Create List of Static Device Nodes
  ldconfig.service                       loaded active exited  Rebuild Dynamic Linker Cache
  lightdm.service                        loaded active running Light Display Manager
  ModemManager.service                   loaded active running Modem Manager
  networking.service                     loaded active exited  Raise network interfaces
  NetworkManager-wait-online.service     loaded active exited  Network Manager Wait Online
  NetworkManager.service                 loaded active running Network Manager
```

3.2 Large Files

As it was listed as one suggestion in the project brief, command *du* was explored to retrieve the top 10 large files in /home directory. The command already focuses on file size, and also provides full file paths.

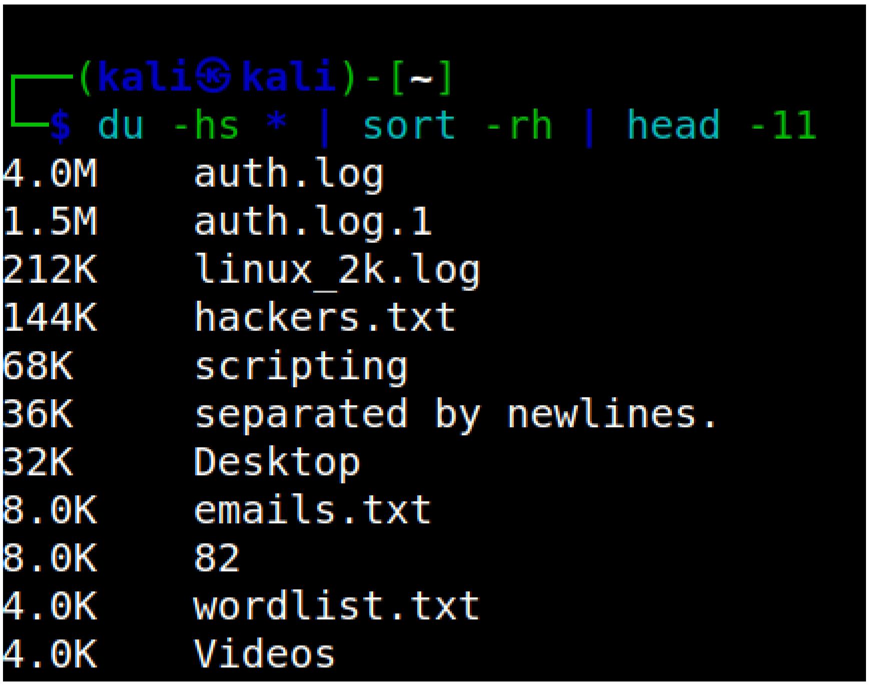


```
┌──(kali㉿kali)-[~/scripting]
└─$ du -S /home | sort -nr | head -10
du: cannot read directory '/home/optimus': Permission denied
17932   /home/kali/.cache/mozilla/firefox/s3tjlkv8.default-esr/startupCache
12280   /home/kali
11476   /home/kali/.mozilla/firefox/s3tjlkv8.default-esr
2040    /home/kali/.cache/mozilla/firefox/s3tjlkv8.default-esr/safebrowsing
1324    /home/kali/.cache/gstreamer-1.0
1288    /home/kali/.cache/mesa_shader_cache
696     /home/kali/.mozilla/firefox/s3tjlkv8.default-esr/storage/permanent/chrome/idb
308     /home/kali/.cache/mozilla/firefox/s3tjlkv8.default-esr/cache2/entries
256     /home/kali/.cache/thumbnails/large
200     /home/kali/.config/xfce4/desktop
```

The actual application of this portion is the script is arguably for users to identify large **files** if they may want to remove them to free up storage space. As such, it was deemed not useful to show the total or aggregate space taken up all files within a directory (e.g. the screenshot above showing 12280 kb in /home/kali).

We explored ways of processing *du* in order to omit such display, but was unsuccessful in achieving a desired result. For instance, while there is an exclude flag to omit directories, the specific directory name will need to be provided, which is not realistic for a script meant for generic use in another, unknown user's machine. Using the -s flag is imperfect as it still aggregates directories (although excluding sub directories).

One issue relating scope of the question is whether "within /home directory" is intended to include subdirectories - i.e. a complete search of all files. Du does so by default while ls would require addition of the appropriate flag. Ultimately it was decided not to use *ls* and interpret the requirement as focusing just the /home directory - but take into account hidden files.



```
┌──(kali㉿kali)-[~]
└─$ du -hs * | sort -rh | head -11
4.0M    auth.log
1.5M    auth.log.1
212K    linux_2k.log
144K    hackers.txt
68K     scripting
36K     separated by newlines.
32K     Desktop
8.0K    emails.txt
8.0K    82
4.0K    wordlist.txt
4.0K    Videos
```

4. Conclusion & Recommendations

This project was a useful exercise to familiarise oneself with the commands to retrieve key system and network information on a linux machine, and how to organise it as a script.

As outlined in the discussion portion, there are several approaches and commands that could be used to retrieve the same information. In particular for the more complex sections covering active services and file sizes, it would be beneficial to clarify scope and requirements as early as possible. These are useful points to be kept in mind for future projects.

5. References

Various websites and blogs that provided guidance and suggested approaches for this project are listed below.

1. https://www.networkworld.com/article/969352/how-to-sort-ps-output.html

2. https://www.tomshardware.com/how-to/find-large-files-linux

3. https://helpdeskgeek.com/linux-tips/6-easy-ways-to-check-memory-usage-on-linux/

4. https://www.baeldung.com/linux/process-name-from-pid

5. https://www.2daygeek.com/how-to-check-all-running-services-in-linux/

6. https://www.hostinger.com/tutorials/manage-and-list-services-in-linux

7. https://draculaservers.com/tutorials/sort-files-by-size-in-linux-unix/#:~:text=To%20sort%20files%20in%20Linux,and%20sort%20command%20in%20combination.&text=Sorting%20files%20in%20regard%20to,by%20adding%20specific%20flag%20options.