Contents

1. Introduction

This project involved the creation of a Bash script to capture and present various system metrics for a Linux operating system installed within user's current virtual environment (VMWare Workstation). The script provides insights into the Linux version, network details, private and public IP addresses, and the default gateway. It also computes disk statistics, showcasing overall size for the free and used space available within the system. Additionally, the script highlights the five largest directories and monitors CPU usage, updating these statistics every ten seconds (with every re-run of the bash script) to reflect the real-time system state.

2. Project Objective

The purpose of this project lies in the creation of a monitoring tool to provide depth in understanding how a Linux system performs and utilizes resources within a VMware Workstation environment (Kubernetes Monitoring: Metrics, Challenges & Best Practices). The following are the aims of the desired outcome:

- To identify the version of Linux being used.

- To retrieve network details such as private and public IP addresses and the default gateway, which are essential for understanding network connectivity within the virtual environment.

- To compute disk statistics, showing the overall free space, and used space. The information helps in monitoring storage consumption within the virtual machine.

- To identify the five largest directories, providing insights into storage allocation, and monitors CPU usage in real-time to identify potential performance bottlenecks to understand the system's state of resource utilization.

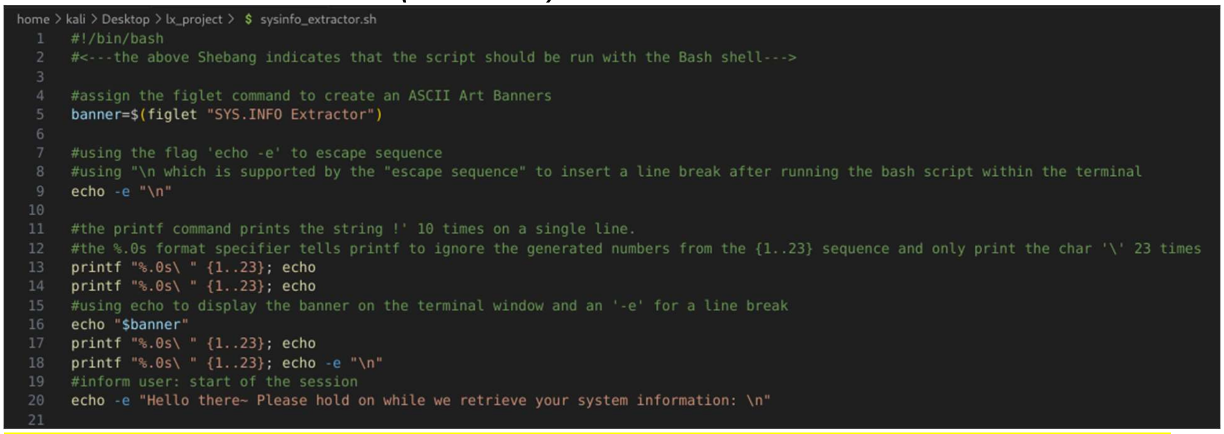
- To receive the latest CPU usage (updated every ten seconds) with every re-run of bash script, offering a dynamic view of system activity.

3. Methodologies

In order to achieve aim and outcome of the project the bash shell script titled: "***sysinfo_extractor.sh***" is created. The script is specifically written to retrieve and display the required system information, including external and internal IP addresses, MAC address, top CPU-consuming processes, memory usage, active system services, and the top ten largest files in the /home directory.

It formats the required information for easy readability and informs the user of each step in the process. The following are the screenshots and explanation of the purpose of each command/line written for the shell script:

3.1. Introduction and Banner (Lines 1-21)

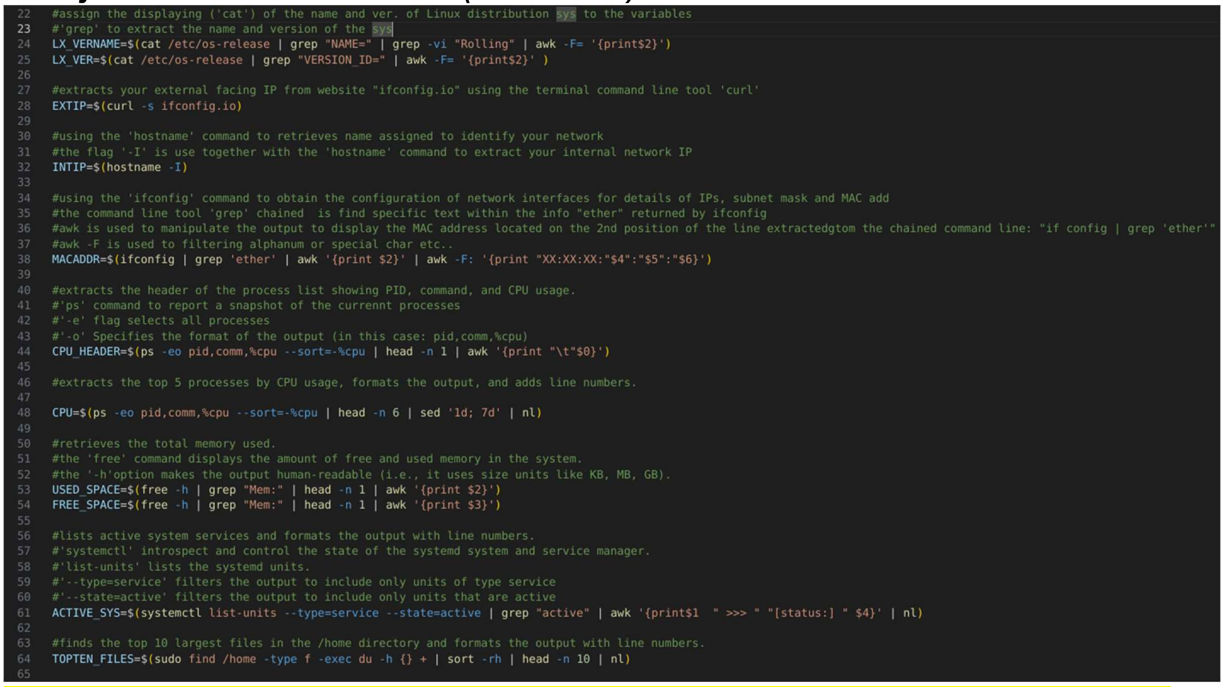


```
home > kali > Desktop > lx_project > $ sysinfo_extractor.sh
 1  #!/bin/bash
 2  #<---the above Shebang indicates that the script should be run with the Bash shell--->
 3
 4  #assign the figlet command to create an ASCII Art Banners
 5  banner=$(figlet "SYS.INFO Extractor")
 6
 7  #using the flag 'echo -e' to escape sequence
 8  #using "\n which is supported by the "escape sequence" to insert a line break after running the bash script within the terminal
 9  echo -e "\n"
10
11  #the printf command prints the string !' 10 times on a single line.
12  #the %.0s format specifier tells printf to ignore the generated numbers from the {1..23} sequence and only print the char '\' 23 times
13  printf "%.0s\ " {1..23}; echo
14  printf "%.0s\ " {1..23}; echo
15  #using echo to display the banner on the terminal window and an '-e' for a line break
16  echo "$banner"
17  printf "%.0s\ " {1..23}; echo
18  printf "%.0s\ " {1..23}; echo -e "\n"
19  #inform user: start of the session
20  echo -e "Hello there~ Please hold on while we retrieve your system information: \n"
21
```

[Note: Line-for-line explanation of the script can be obtained within the file: "sysinfo_extractor.sh" ***]***

- **Line 1:** The script starts with a Shebang line *#!/bin/bash* indicating it needs the Bash interpreter to run.
- **Line 5:** It defines a variable banner using *figlet* (assuming it's installed) to create an ASCII art title "***SYS.INFO Extractor***".
- **Line 13 – 20:** It prints some blank lines and a separator line using *printf* and *echo*.

3.2. System Information Retrieval (Lines 22-65)

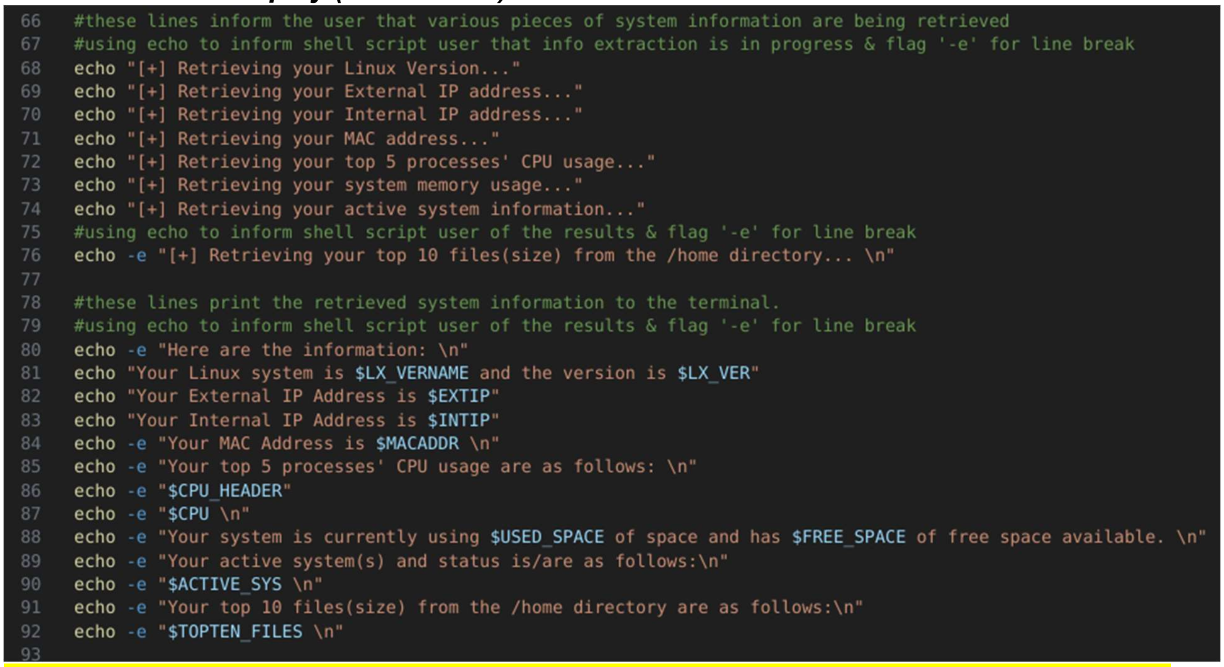


```
22  #assign the displaying ('cat') of the name and ver. of Linux distribution sys to the variables
23  #'grep' to extract the name and version of the sys
24  LX_VERNAME=$(cat /etc/os-release | grep "NAME=" | grep -vi "Rolling" | awk -F= '{print$2}')
25  LX_VER=$(cat /etc/os-release | grep "VERSION_ID=" | awk -F= '{print$2}' )
26
27  #extracts your external facing IP from website "ifconfig.io" using the terminal command line tool 'curl'
28  EXTIP=$(curl -s ifconfig.io)
29
30  #using the 'hostname' command to retrieves name assigned to identify your network
31  #the flag '-I' is use together with the 'hostname' command to extract your internal network IP
32  INTIP=$(hostname -I)
33
34  #using the 'ifconfig' command to obtain the configuration of network interfaces for details of IPs, subnet mask and MAC add
35  #the command line tool 'grep' chained  is find specific text within the info "ether" returned by ifconfig
36  #awk is used to manipulate the output to display the MAC address located on the 2nd position of the line extractedgtom the chained command line: "if config | grep 'ether'"
37  #awk -F is used to filtering alphanum or special char etc..
38  MACADDR=$(ifconfig | grep 'ether' | awk '{print $2}' | awk -F: '{print "XX:XX:XX:"$4":"$5":"$6}')
39
40  #extracts the header of the process list showing PID, command, and CPU usage.
41  #'ps' command to report a snapshot of the currennt processes
42  #'-e' flag selects all processes
43  #'-o' Specifies the format of the output (in this case: pid,comm,%cpu)
44  CPU_HEADER=$(ps -eo pid,comm,%cpu --sort=-%cpu | head -n 1 | awk '{print "\t"$0}')
45
46  #extracts the top 5 processes by CPU usage, formats the output, and adds line numbers.
47
48  CPU=$(ps -eo pid,comm,%cpu --sort=-%cpu | head -n 6 | sed '1d; 7d' | nl)
49
50  #retrieves the total memory used.
51  #the 'free' command displays the amount of free and used memory in the system.
52  #the '-h'option makes the output human-readable (i.e., it uses size units like KB, MB, GB).
53  USED_SPACE=$(free -h | grep "Mem:" | head -n 1 | awk '{print $2}')
54  FREE_SPACE=$(free -h | grep "Mem:" | head -n 1 | awk '{print $3}')
55
56  #lists active system services and formats the output with line numbers.
57  #'systemctl' introspect and control the state of the systemd system and service manager.
58  #'list-units' lists the systemd units.
59  #'--type=service' filters the output to include only units of type service
60  #'--state=active' filters the output to include only units that are active
61  ACTIVE_SYS=$(systemctl list-units --type=service --state=active | grep "active" | awk '{print$1  " >>> " "[status:] " $4}' | nl)
62
63  #finds the top 10 largest files in the /home directory and formats the output with line numbers.
64  TOPTEN_FILES=$(sudo find /home -type f -exec du -h {} + | sort -rh | head -n 10 | nl)
65
```

[Note: Line-for-line explanation of the script can be obtained within the file: "sysinfo_extractor.sh" ***]***

- **Line 24 – 25:** It retrieves the Linux distribution name and version using *cat, grep*, and *awk* commands, storing them in variables *LX_VERNAME* and *LX_VER*.
- **Line 26:** It gets the external IP address from the website "*ifconfig.io*" using *curl*.
- **Line32:** The internal IP is retrieved using *hostname -I*.
- **Line 38:** *ifconfig* and *grep* are used to find the MAC address. awk formats the output.
- **Line 48:** *ps* with options retrieves process information like PID, name, and CPU usage. It then uses head, *sed, awk*, and *nl* to format and display the top 5 processes by CPU usage.
- **Line 53 – 54:** Similar techniques are used to find total used and free memory with *free* and *-h* format to extract required output in a human readable format/unit such as kilobytes, megabytes or gigabytes.
- **Line 61:** *systemctl* retrieves a list of active system services and formats the output with line numbers
- **Line 64:** The script finds the top 10 largest files in the */home* directory using find, *du*, *sort*, *head*, and nl commands and stores the output.

3.3. Information Display (Lines 66-93)

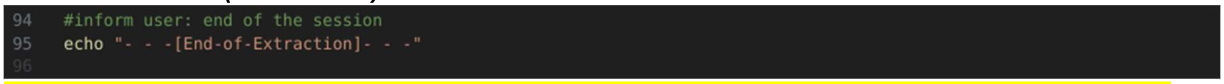


```
66  #these lines inform the user that various pieces of system information are being retrieved
67  #using echo to inform shell script user that info extraction is in progress & flag '-e' for line break
68  echo "[+] Retrieving your Linux Version..."
69  echo "[+] Retrieving your External IP address..."
70  echo "[+] Retrieving your Internal IP address..."
71  echo "[+] Retrieving your MAC address..."
72  echo "[+] Retrieving your top 5 processes' CPU usage..."
73  echo "[+] Retrieving your system memory usage..."
74  echo "[+] Retrieving your active system information..."
75  #using echo to inform shell script user of the results & flag '-e' for line break
76  echo -e "[+] Retrieving your top 10 files(size) from the /home directory... \n"
77
78  #these lines print the retrieved system information to the terminal.
79  #using echo to inform shell script user of the results & flag '-e' for line break
80  echo -e "Here are the information: \n"
81  echo "Your Linux system is $LX_VERNAME and the version is $LX_VER"
82  echo "Your External IP Address is $EXTIP"
83  echo "Your Internal IP Address is $INTIP"
84  echo -e "Your MAC Address is $MACADDR \n"
85  echo -e "Your top 5 processes' CPU usage are as follows: \n"
86  echo -e "$CPU_HEADER"
87  echo -e "$CPU \n"
88  echo -e "Your system is currently using $USED_SPACE of space and has $FREE_SPACE of free space available. \n"
89  echo -e "Your active system(s) and status is/are as follows:\n"
90  echo -e "$ACTIVE_SYS \n"
91  echo -e "Your top 10 files(size) from the /home directory are as follows:\n"
92  echo -e "$TOPTEN_FILES \n"
93
```

[Note: Line-for-line explanation of the script can be obtained within the file: "sysinfo_extractor.sh" ***]***

- **Line 68 – 67:** The script informs the user it's retrieving system information
- **Line 80 – 92:** Finally, it displays the retrieved information using *echo* for various details like Linux version, IP addresses, MAC address, CPU usage, memory usage, active services, and top 10 largest files. The script informs the user the information retrieval is complete.

3.4. Conclusion (Line 94-96)



```
94  #inform user: end of the session
95  echo "- - -[End-of-Extraction]- - -"
96
```

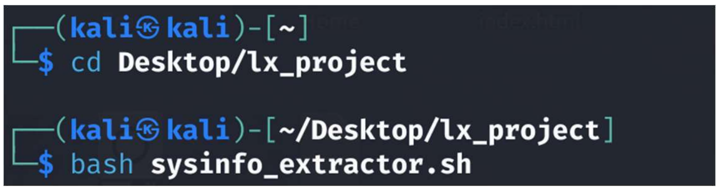
[Note: Line-for-line explanation of the script can be obtained within the file: "sysinfo_extractor.sh" ***]***

- **Line 94 – 95:** The script concludes by informing the user it has finished extracting information.

Overall, this script is a useful tool to gather and display various system information on a Linux system.

4. Discussion

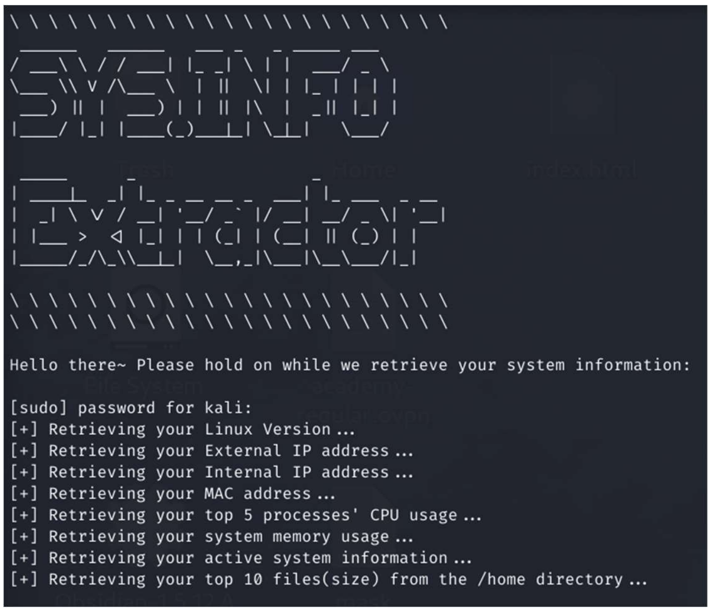
With the scripted and debugged. It was being executed within the terminal of the Kali Linux environment with the execution of the valid file path:



```
┌──(kali㉿kali)-[~]
└─$ cd Desktop/lx_project

┌──(kali㉿kali)-[~/Desktop/lx_project]
└─$ bash sysinfo_extractor.sh
```

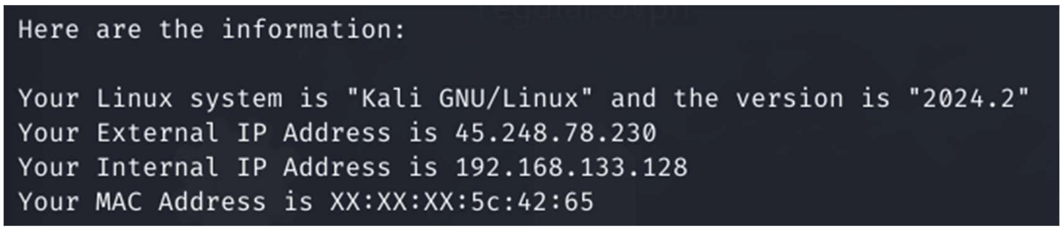
4.1. Welcome Banner for script execution

With the execution of the bash <file name> command, it was observed that the script was ran as expected with the script informing user that it is performing the taskings as assigned (based on the pre-requisite of the project):



```
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \

 ______   ______    ___ _   _ _____ ___  
/ ___\ \ / / ___|  |_ _| \ | |  ___/ _ \ 
\___ \\ V /\___ \   | ||  \| | |_ | | | |
 ___) || |  ___) |  | || |\  |  _|| |_| |
|____/ |_| |____(_)|___|_| \_|_|   \___/ 

 _____      _                  _             
| ____|_  _| |_ _ __ __ _  ___| |_ ___  _ __ 
|  _| \ \/ / __| '__/ _` |/ __| __/ _ \| '__|
| |___ >  <| |_| | | (_| | (__| || (_) | |   
|_____/_/\_\\__|_|  \__,_|\___|\__\___/|_|   

\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \

Hello there~ Please hold on while we retrieve your system information:

[sudo] password for kali:
[+] Retrieving your Linux Version ...
[+] Retrieving your External IP address ...
[+] Retrieving your Internal IP address ...
[+] Retrieving your MAC address ...
[+] Retrieving your top 5 processes' CPU usage ...
[+] Retrieving your system memory usage ...
[+] Retrieving your active system information ...
[+] Retrieving your top 10 files(size) from the /home directory ...
```

4.2. Linux Version, IP & MAC Addresses

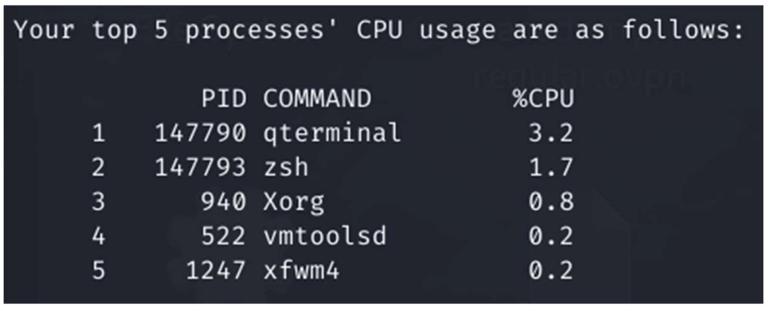


```
Here are the information:

Your Linux system is "Kali GNU/Linux" and the version is "2024.2"
Your External IP Address is 45.248.78.230
Your Internal IP Address is 192.168.133.128
Your MAC Address is XX:XX:XX:5c:42:65
```

The system began with the return of the relevant information based on the sequences of the script. With reference to the screenshot below, it was observed that name and version of the Linux system along with the external IP (Public IP), internal IP and masked MAC address was successfully returned:

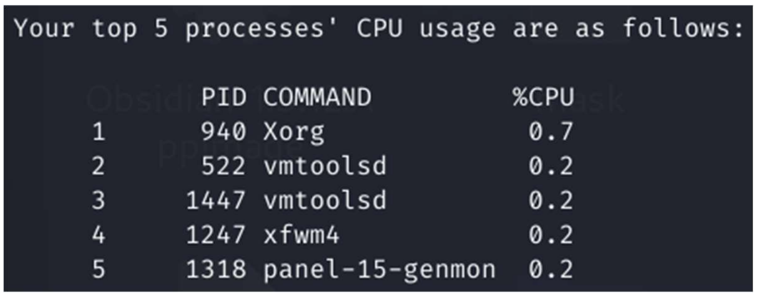
4.3. CPU Usage

The next information displayed shows the CPU usage of the top 5 processes running on the system as shown in the following screen shot:



```
Your top 5 processes' CPU usage are as follows:

           PID COMMAND         %CPU
    1   147790 qterminal        3.2
    2   147793 zsh              1.7
    3      940 Xorg             0.8
    4      522 vmtoolsd         0.2
    5     1247 xfwm4            0.2
```

Refresh of the bash script was conducted 5minutes later to ensure that the command to extract latest CPU stats are fully functional (which refreshes itself every ten seconds using the command line '*ps*'). The screenshot appended below indicated that the intended command was functional:



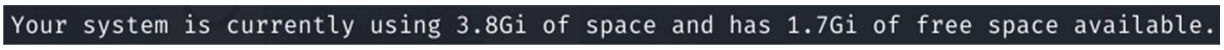
```
Your top 5 processes' CPU usage are as follows:

           PID COMMAND         %CPU
    1      940 Xorg             0.7
    2      522 vmtoolsd         0.2
    3     1447 vmtoolsd         0.2
    4     1247 xfwm4            0.2
    5     1318 panel-15-genmon  0.2
```

As observed, the header provides the following information:

- **PID:** This is the Process Identifier, a unique number assigned to each running process.
- **COMMAND:** This is the name of the program or command associated with the process.
- **%CPU:** This indicates the percentage of CPU time each process is consuming.

Next, the values displayed, shows the top 5 processes based on their CPU usage, with the highest CPU usage rate ranked at the top. For example, process 1 with PID 147790 named "qterminal" is using the most CPU (3.2%) at the time this information was captured. Other processes like "zsh" (shell), "Xorg" (graphics server), "vmtoolsd" (VMware related process), and "xfwm4" (window manager) are using less CPU (between 0.2% and 1.7%).

4.4. Memory Usage



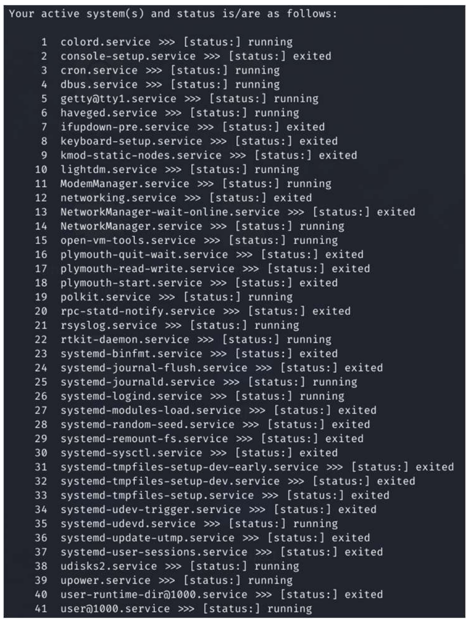
```
Your system is currently using 3.8Gi of space and has 1.7Gi of free space available.
```

With reference to the displayed information extracted, the amount of storage space being used on the system's disk drive was observed. The result indicated that during the point of extraction, the system has stored 3.8 Gigabytes (Gi) of data, with 1.7 Gigabytes of free space remaining on the disk drive. It is also noteworthy that "Gi" is the unit of storage capacity in this process of extraction.

4.5. Status of Active System

41 active systems was observed as shown in the screenshot appended below. Each line shows a service name followed by its status. For example, "colord.service >>> [status:] running" indicates the service named "colord.service" is currently running. The ">>>" symbol is likely used as a separator for better readability.

The script identifies two main service statuses: "running" and "exited". “running” means the service is currently active and functioning. “exited” indicates that the service has finished its task and is no longer running. Some services might be designed to start, perform a specific action, and then exit.



```
Your active system(s) and status is/are as follows:

     1  colord.service >>> [status:] running
     2  console-setup.service >>> [status:] exited
     3  cron.service >>> [status:] running
     4  dbus.service >>> [status:] running
     5  getty@tty1.service >>> [status:] running
     6  haveged.service >>> [status:] running
     7  ifupdown-pre.service >>> [status:] exited
     8  keyboard-setup.service >>> [status:] exited
     9  kmod-static-nodes.service >>> [status:] exited
    10  lightdm.service >>> [status:] running
    11  ModemManager.service >>> [status:] running
    12  networking.service >>> [status:] exited
    13  NetworkManager-wait-online.service >>> [status:] exited
    14  NetworkManager.service >>> [status:] running
    15  open-vm-tools.service >>> [status:] running
    16  plymouth-quit-wait.service >>> [status:] exited
    17  plymouth-read-write.service >>> [status:] exited
    18  plymouth-start.service >>> [status:] exited
    19  polkit.service >>> [status:] running
    20  rpc-statd-notify.service >>> [status:] exited
    21  rsyslog.service >>> [status:] running
    22  rtkit-daemon.service >>> [status:] running
    23  systemd-binfmt.service >>> [status:] exited
    24  systemd-journal-flush.service >>> [status:] exited
    25  systemd-journald.service >>> [status:] running
    26  systemd-logind.service >>> [status:] running
    27  systemd-modules-load.service >>> [status:] exited
    28  systemd-random-seed.service >>> [status:] exited
    29  systemd-remount-fs.service >>> [status:] exited
    30  systemd-sysctl.service >>> [status:] exited
    31  systemd-tmpfiles-setup-dev-early.service >>> [status:] exited
    32  systemd-tmpfiles-setup-dev.service >>> [status:] exited
    33  systemd-tmpfiles-setup.service >>> [status:] exited
    34  systemd-udev-trigger.service >>> [status:] exited
    35  systemd-udevd.service >>> [status:] running
    36  systemd-update-utmp.service >>> [status:] exited
    37  systemd-user-sessions.service >>> [status:] exited
    38  udisks2.service >>> [status:] running
    39  upower.service >>> [status:] running
    40  user-runtime-dir@1000.service >>> [status:] exited
    41  user@1000.service >>> [status:] running
```

Based on the results, essential services to run the system and key services to start the system were identified as follows:

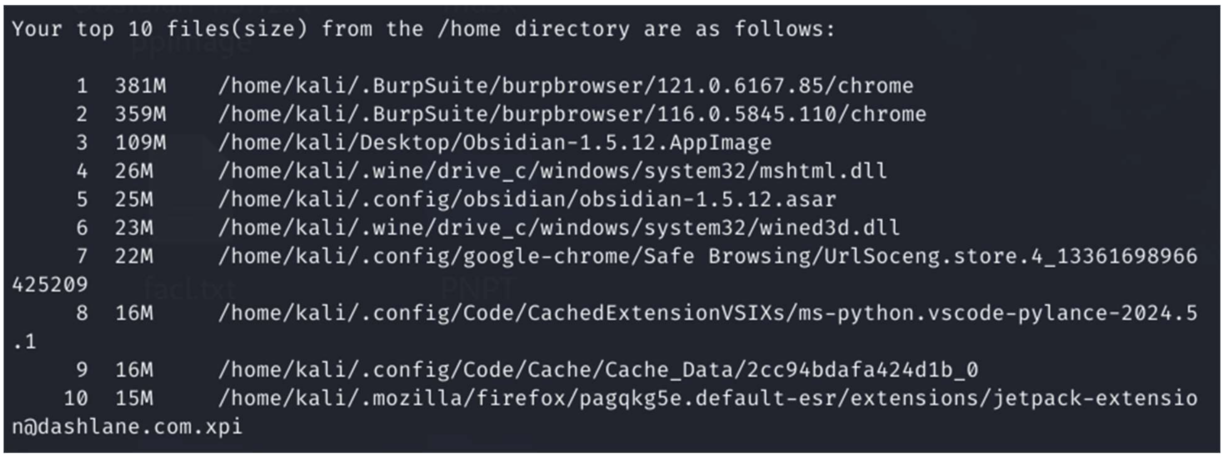
4.5.1. Essential Services (running)

- cron.service: This service is responsible for running scheduled tasks on your system.
- dbus.service: This service is a core component for communication between applications.
- networking.service: This service manages your network connection (although it might be currently exited if you're not actively connected).
- rsyslog.service: This service handles system logging.
- systemd-journald.service: This service is responsible for system journaling, which keeps track of system events.
- udisks2.service: This service manages disk drives.
- upower.service: This service manages power management.

4.5.2. Services related to starting the system (exited)

- console-setup.service: This service is involved in setting up the console during system startup.
- keyboard-setup.service: This service is responsible for configuring the keyboard at startup.
- plymouth-* services: These services are likely related to the boot splash screen that user see during system startup.
- systemd-* services (some exited): These services is responsible for various system initialization tasks during startup and may exit once their specific functions are complete.

4.6. Top 10 Home Directory Files (Size)



```
Your top 10 files(size) from the /home directory are as follows:

     1  381M    /home/kali/.BurpSuite/burpbrowser/121.0.6167.85/chrome
     2  359M    /home/kali/.BurpSuite/burpbrowser/116.0.5845.110/chrome
     3  109M    /home/kali/Desktop/Obsidian-1.5.12.AppImage
     4  26M     /home/kali/.wine/drive_c/windows/system32/mshtml.dll
     5  25M     /home/kali/.config/obsidian/obsidian-1.5.12.asar
     6  23M     /home/kali/.wine/drive_c/windows/system32/wined3d.dll
     7  22M     /home/kali/.config/google-chrome/Safe Browsing/UrlSoceng.store.4_13361698966
425209
     8  16M     /home/kali/.config/Code/CachedExtensionVSIXs/ms-python.vscode-pylance-2024.5
.1
     9  16M     /home/kali/.config/Code/Cache/Cache_Data/2cc94bdafa424d1b_0
    10  15M     /home/kali/.mozilla/firefox/pagqkg5e.default-esr/extensions/jetpack-extensio
n@dashlane.com.xpi
```

This section shows the ten largest files found in your /home directory, based on their size. Here's a breakdown:

4.7. File Paths and Sizes:

Each line displays the complete path and size of a file. For example: "*/home/kali/Desktop/Obsidian-1.5.12.AppImage 4 26M*" indicates a file named "*Obsidian-1.5.12.AppImage*" located on the user's Desktop with a size of 26 Megabytes (MB).

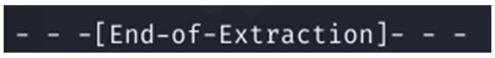
4.8. Analysis of Listed Files:

The list includes a mix of program files, application data, and browser data. Here are some possible explanations for some of the entries:

- /home/kali/Desktop/Obsidian-1.5.12.AppImage: This is the installer file for an application called Obsidian.
- Files starting with /home/kali/.wine/: These are related to the Wine compatibility layer used to run Windows programs on Linux.
- The specific files mshtml.dll and wined3d.dll are Windows system libraries.
- Files starting with /home/kali/.config/: These are configuration files for various applications.
- /home/kali/.config/obsidian/obsidian-1.5.12.asar: This are application data related to Obsidian.

- /home/kali/.config/google-chrome/: This directory stores Chrome configuration data.
- The file UrlSoceng.store.4_13361698966425209 could be part of Chrome's Safe Browsing feature.
- /home/kali/.config/Code/: This directory stores configuration data for Visual Studio Code. The files listed might be cached extensions or data related to the Python extension
- /home/kali/.mozilla/firefox/: This directory stores configuration data for Firefox.
- The file jetpack-extension@dashlane.com.xpi is an extension for Dashlane password manager.

4.9. End of Script



```
- - -[End-of-Extraction]- - -
```

System was observed to print out the ending line as indicated in the screenshot above, upon the execution of the final command for extraction of the 'top ten home directory file (size)', confirming the success and completion of the 'script execution'.

5. Conclusion:

The script provides a system overview and performs like a system information dashboard. It compiles various details of the Linux system in one place, such as the operating system version, kernel version, system uptime, and memory usage. This can be useful in the process of system and system troubleshoot and performance monitoring (Debugging and docker containers), or simply understanding user's current system configuration. With a thorough walkthrough of the project, it was observed that the information obtained for each aspects provides lead-in data for in-depth proactive review of an individual and organization's cybersecurity posture for risk analysis, mitigation and system hardening of the analyzed system, which can be translated in the following ways:

- **Network Insights:** The script provides a wealth of network information, including private and public IP addresses, and the default gateway. This knowledge equips user with the ability to set up network connections, diagnose connectivity problems, and stay informed about user(s) position on the network, fostering a sense of awareness and control

- **Storage Management:** The script shows disk usage statistics, including total size, free space, and used space. This helps user monitor storage consumption and identify potential low-space situations.

- **Resource Utilization:** By highlighting the five largest directories, it can give user clues about where current storage space is being used. Additionally, monitoring CPU usage in real-time allows user to identify potential performance bottlenecks.

- **Automation and Efficiency:** The script takes the burden off user by automating the task of collecting system information. Instead of manually running multiple commands, a single script execution provides a quick snapshot. This not only saves time but also ensures consistency in the information gathered, relieving user of manual work and enhancing efficiency

- **Customization Potential:** Bash scripts are flexible. user can modify this script to capture additional information relevant to your specific needs. For example, user can add a command to display the system's current users or the status of running services. This allows individual to tailor the script to their unique system monitoring requirements.

6. Recommendations

From a personal perspective. The analysis has provided insights which enable to further decide on the next course of actions for my system:

1. From the angle of system service functionality, the bash script has provided me with actionable data into the various services that are running or have recently run on my system. It can be helpful for troubleshooting system and services issues, or understanding which services are essential for my system's operation (How To Monitor All User Activities In Your Team: Tips and Tricks | Servers free - hosting news).

2. The ability to retrieve the “Top 10 Home Directory File (Size)” through the bash script helps me to identify the largest files consuming storage space. It highlights a significant size increase in the Obsidian installer file, which I would want to investigate further on my need to:
    a. Keep the files
    b. Delete the files
    c. Move the less frequently used files to an external storage device, etc.