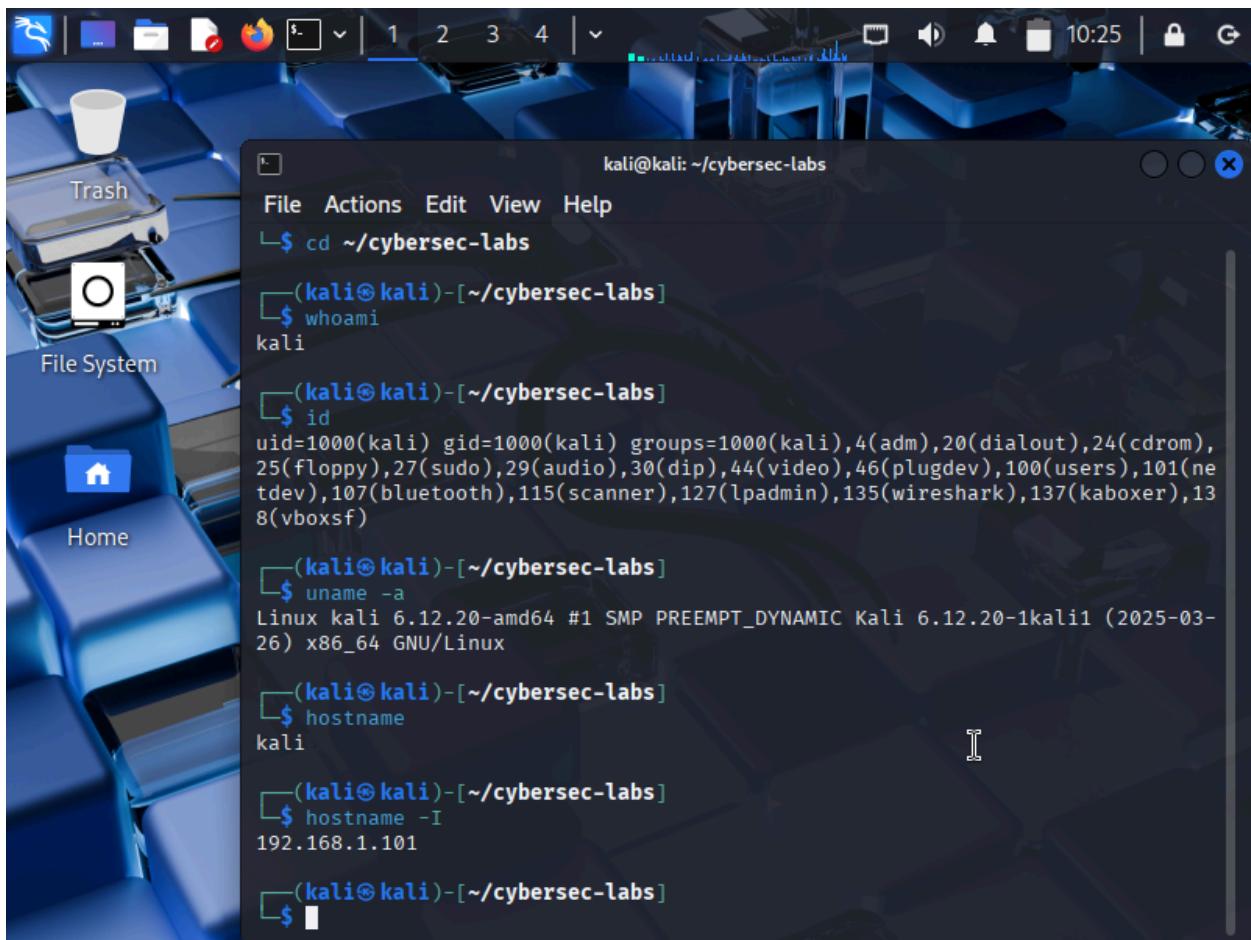


Title: Lab 2: Linux Basics Part 2 - Text Processing and System Information

Matthew Cox, COMP325-01, term (Fall 2025)

Introduction: This lab builds upon Lab 1's foundation, introducing powerful text processing tools and system information commands that are essential for security analysis workflows. In this lab, I'll demonstrate processes and analyze text files using essential Linux text processing tools, extract specific information from command outputs and log files, use piping to chain commands for complex data processing, and gather system information for reconnaissance and enumeration.

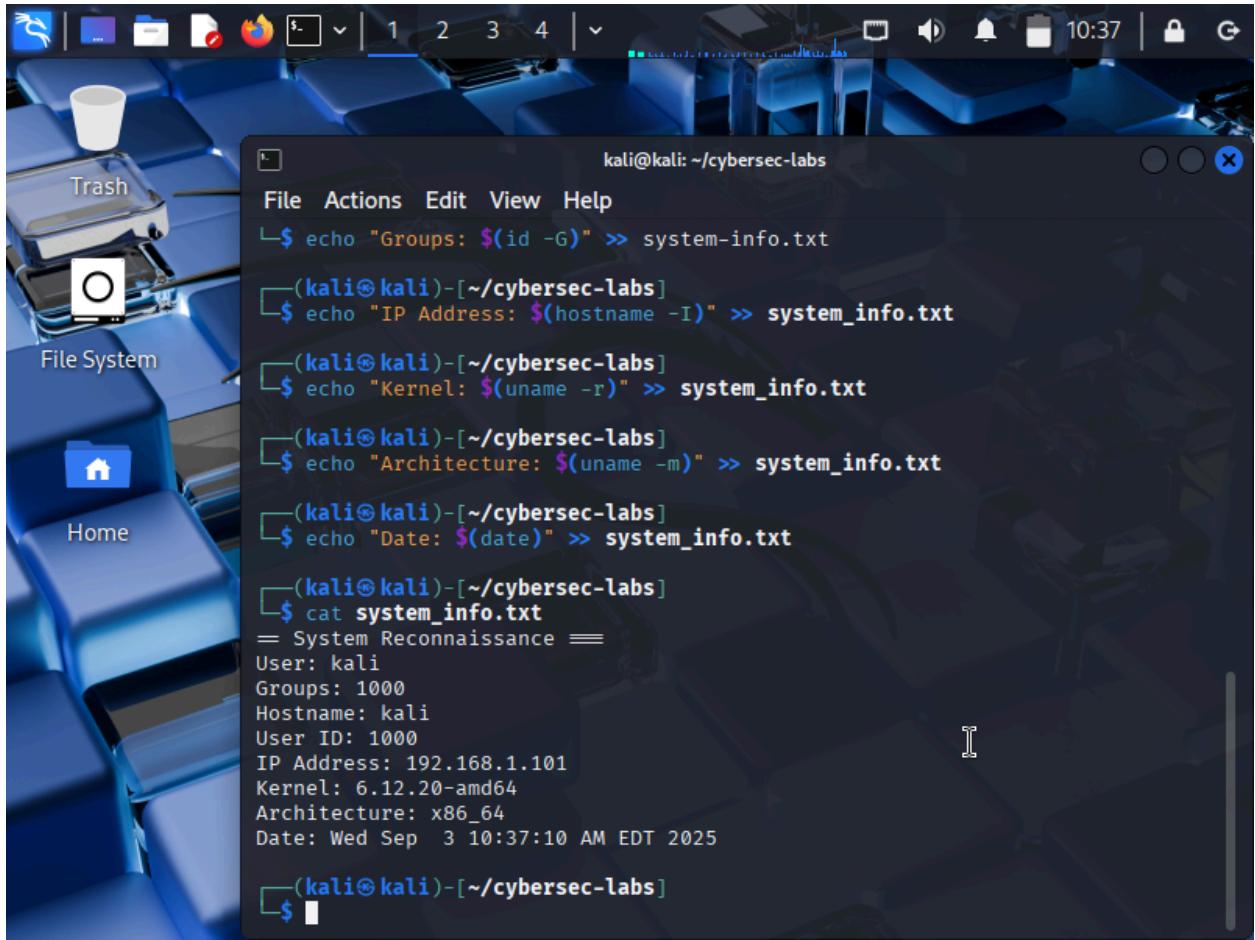
Body:



A screenshot of a Kali Linux desktop environment. On the left, there is a dock with icons for Trash, File System, and Home. A terminal window is open in the center, showing the following session:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
└$ cd ~/cybersec-labs
[(kali㉿kali)-~/cybersec-labs]
└$ whoami
kali
[(kali㉿kali)-~/cybersec-labs]
└$ id
uid=1000(kali) gid=1000(kali) groups=1000(kali),4(adm),20(dialout),24(cdrom),
25(floppy),27(sudo),29(audio),30(dip),44(video),46(plugdev),100(users),101(ne
tdev),107(bluetooth),115(scanner),127(lpadmin),135(wireshark),137(kaboxer),13
8(vboxsf)
[(kali㉿kali)-~/cybersec-labs]
└$ uname -a
Linux kali 6.12.20-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.20-1kali1 (2025-03-
26) x86_64 GNU/Linux
[(kali㉿kali)-~/cybersec-labs]
└$ hostname
kali
[(kali㉿kali)-~/cybersec-labs]
└$ hostname -I
192.168.1.101
[(kali㉿kali)-~/cybersec-labs]
└$ ]
```

This just shows that I am starting up my environment and checking the host.



A screenshot of a Kali Linux desktop environment. A terminal window is open in the foreground, showing the command-line process of generating a system information file. The terminal output is as follows:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
└$ echo "Groups: $(id -G)" >> system-info.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ echo "IP Address: $(hostname -I)" >> system_info.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ echo "Kernel: $(uname -r)" >> system_info.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ echo "Architecture: $(uname -m)" >> system_info.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ echo "Date: $(date)" >> system_info.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ cat system_info.txt
= System Reconnaissance =
User: kali
Groups: 1000
Hostname: kali
User ID: 1000
IP Address: 192.168.1.101
Kernel: 6.12.20-amd64
Architecture: x86_64
Date: Wed Sep 3 10:37:10 AM EDT 2025
└(kali㉿kali)-[~/cybersec-labs]
└$
```

Here, I created the system information and have it displayed for myself to look over.

The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal window title is "kali@kali: ~/cybersec-labs". The terminal content displays several command-line sessions:

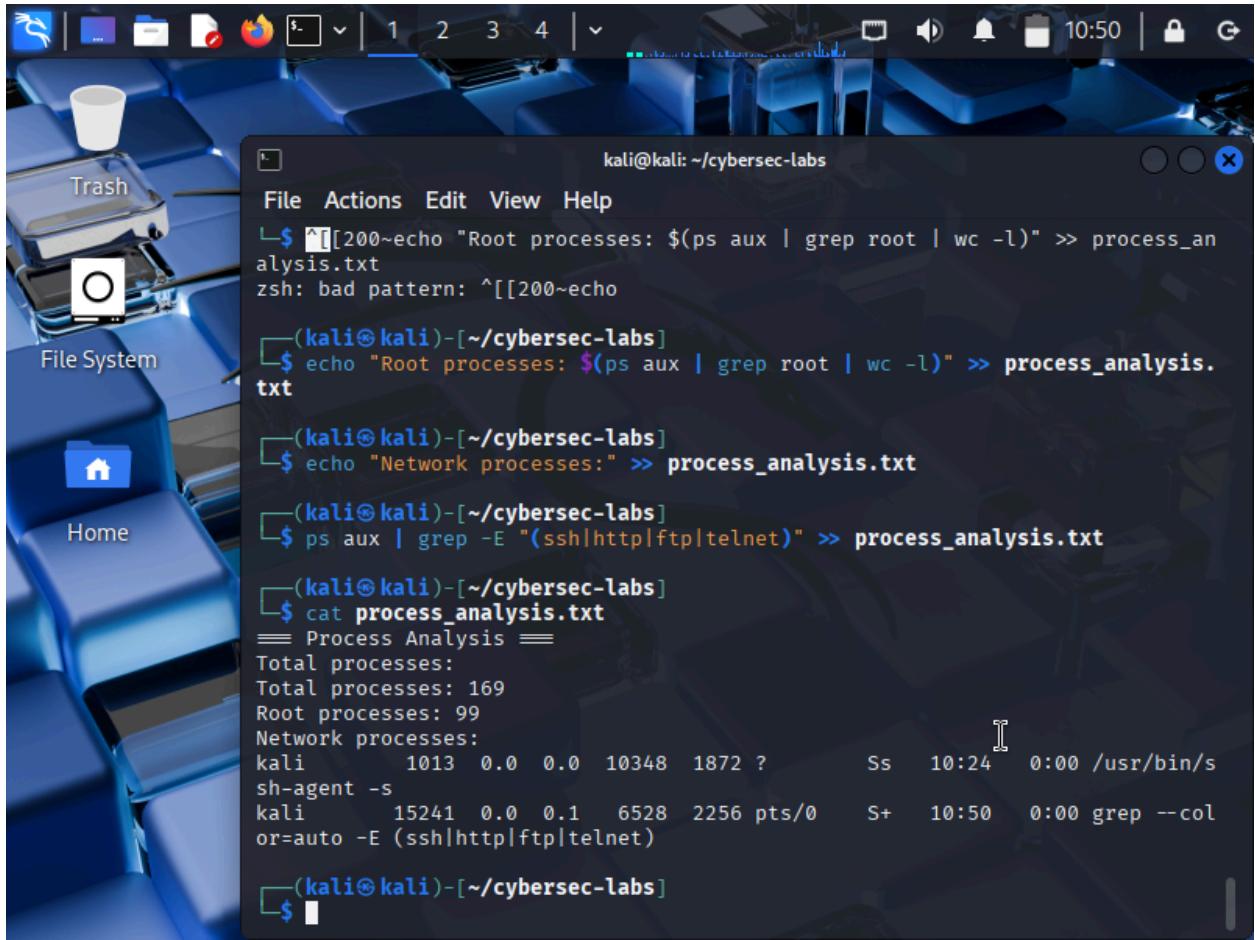
```
_rude_kthread]
root      16  0.0  0.0      0      0 ?      I   10:23  0:00 [rcu_tasks
_trace_kthread]
root      17  0.0  0.0      0      0 ?      S   10:23  0:00 [ksoftirqd
/0]
root      18  0.0  0.0      0      0 ?      I   10:23  0:00 [rcu_prem
pt]
root      19  0.0  0.0      0      0 ?      S   10:23  0:00 [rcu_exp_p
ar_gp_kthread_worker/0]
root      20  0.0  0.0      0      0 ?      S   10:23  0:00 [rcu_exp_g
p_kthread_worker]
root      21  0.0  0.0      0      0 ?      S   10:23  0:00 [migration
/0]

[(kali㉿kali)-[~/cybersec-labs]
$ ps aux | grep firefox
kali      9714  0.0  0.1  6528 2296 pts/0    S+  10:39  0:00 grep --col
or=auto firefox

[(kali㉿kali)-[~/cybersec-labs]
$ ps aux | grep ssh
kali     1013  0.0  0.0 10348 1872 ?        Ss  10:24  0:00 /usr/bin/s
sh-agent -s
kali     9855  0.0  0.1  6528 2160 pts/0    S+  10:40  0:00 grep --col
or=auto ssh

[(kali㉿kali)-[~/cybersec-labs]
$ ]
```

Out of sight, I used the command to view all running processes. In the screenshot, you can see that I used the first twenty processes and then specifically singled out Firefox and the last seek processes using the string ssh.



A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. The main area shows a terminal window titled "kali@kali: ~/cybersec-labs". The terminal history includes:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
└$ ^[[200~echo "Root processes: $(ps aux | grep root | wc -l)" >> process_analysis.txt
zsh: bad pattern: ^[[200~echo

[(kali㉿kali)-~/cybersec-labs]
└$ echo "Root processes: $(ps aux | grep root | wc -l)" >> process_analysis.txt

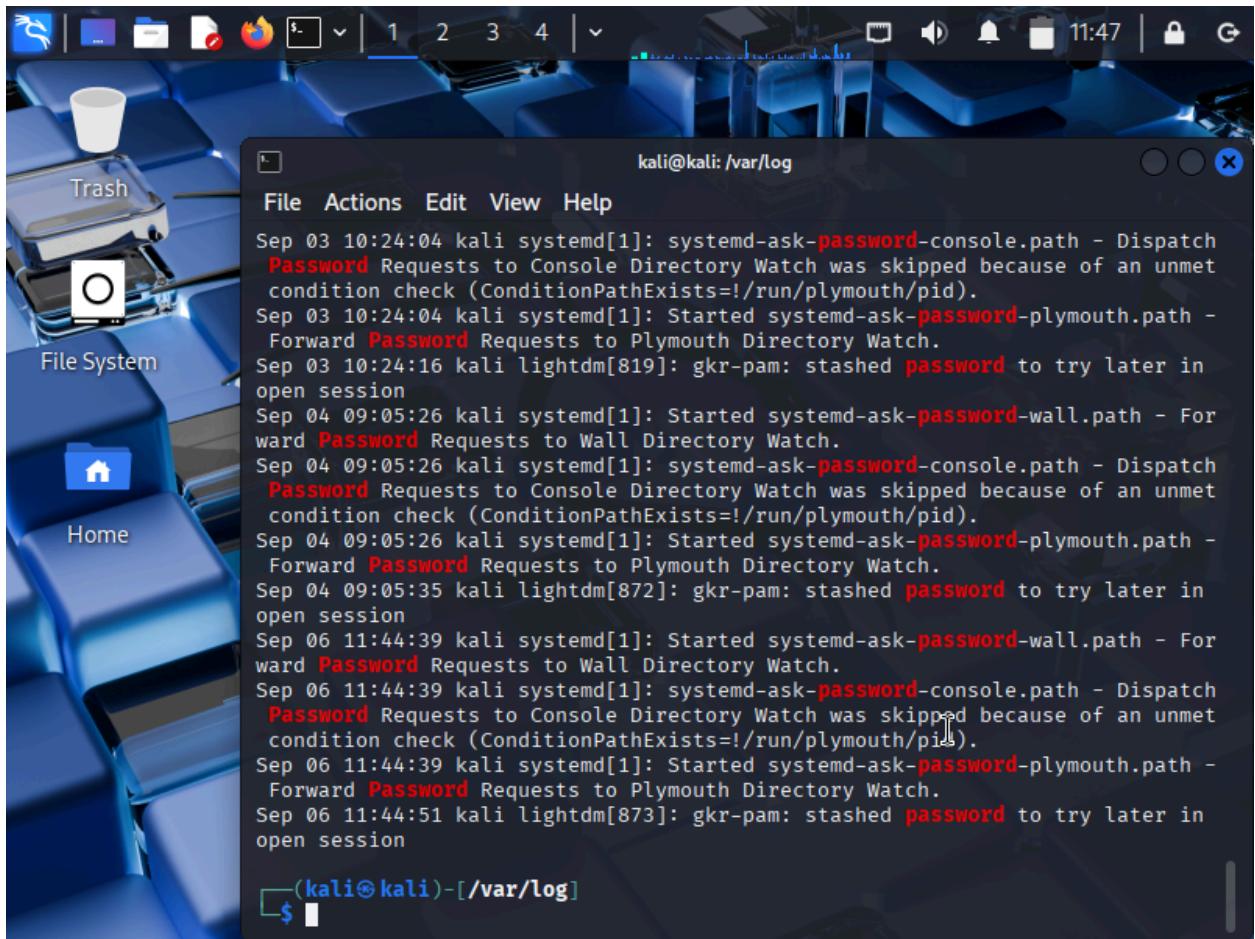
[(kali㉿kali)-~/cybersec-labs]
└$ echo "Network processes:" >> process_analysis.txt

[(kali㉿kali)-~/cybersec-labs]
└$ ps aux | grep -E "(ssh|http|ftp|telnet)" >> process_analysis.txt

[(kali㉿kali)-~/cybersec-labs]
└$ cat process_analysis.txt
==== Process Analysis ====
Total processes:
Total processes: 169
Root processes: 99
Network processes:
kali      1013  0.0  0.0  10348  1872 ?          Ss   10:24   0:00 /usr/bin/s
sh-agent -s
kali      15241  0.0  0.1  6528   2256 pts/0      S+   10:50   0:00 grep --color=auto -E (ssh|http|ftp|telnet)

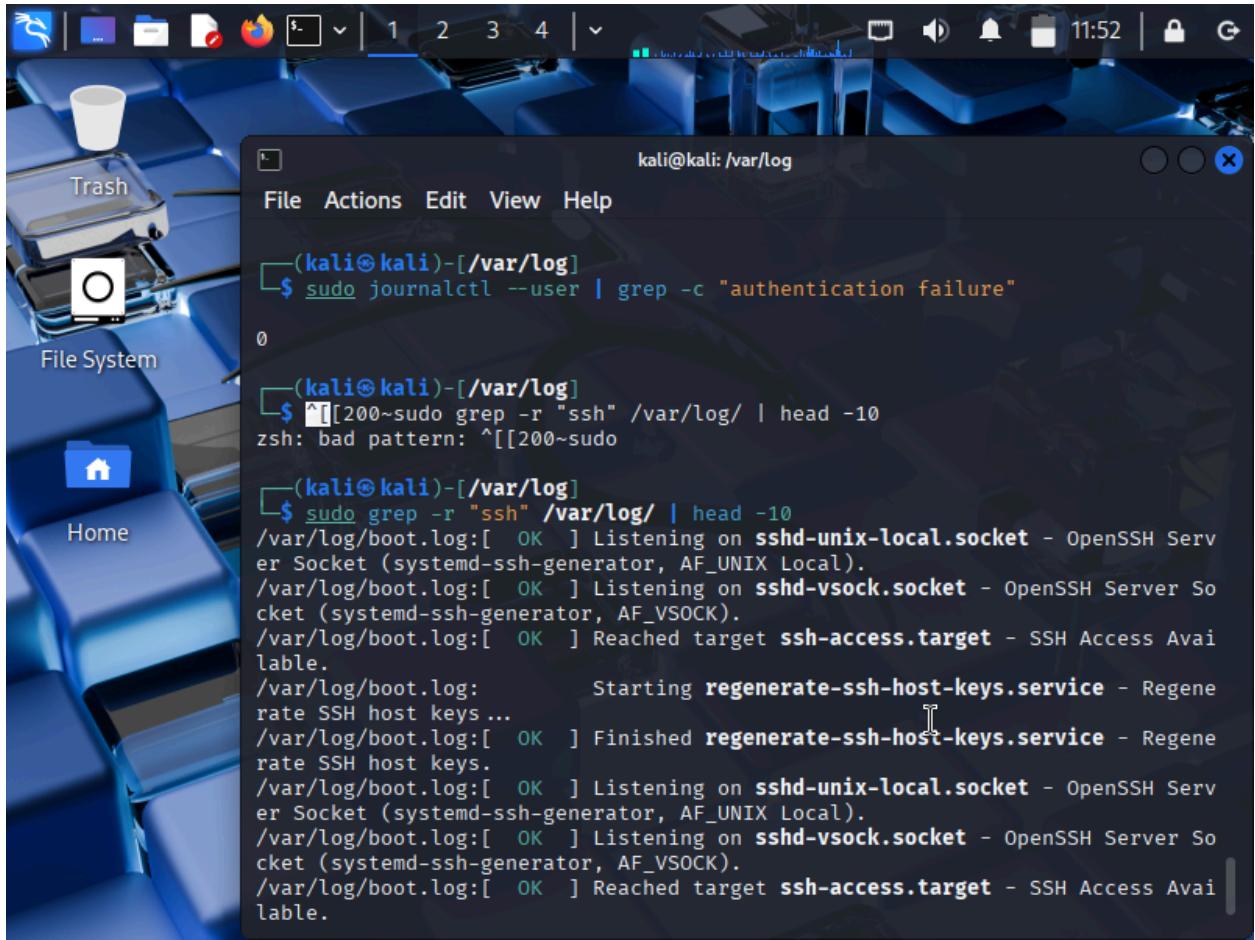
[(kali㉿kali)-~/cybersec-labs]
└$
```

Here, I showed the first 10 processes running as the root user. Then, I created a process monitoring file. This is where I stopped during class recitation. This is the conclusion of Exercise 2.



```
kali@kali: /var/log
File Actions Edit View Help
Sep 03 10:24:04 kali systemd[1]: systemd-ask-password-console.path - Dispatch
>Password Requests to Console Directory Watch was skipped because of an unmet
condition check (ConditionPathExists=!/run/plymouth/pid).
Sep 03 10:24:04 kali systemd[1]: Started systemd-ask-password-plymouth.path -
Forward Password Requests to Plymouth Directory Watch.
Sep 03 10:24:16 kali lightdm[819]: gkr-pam: stashed password to try later in
open session
Sep 04 09:05:26 kali systemd[1]: Started systemd-ask-password-wall.path - For
ward Password Requests to Wall Directory Watch.
Sep 04 09:05:26 kali systemd[1]: systemd-ask-password-console.path - Dispatch
>Password Requests to Console Directory Watch was skipped because of an unmet
condition check (ConditionPathExists=!/run/plymouth/pid).
Sep 04 09:05:26 kali systemd[1]: Started systemd-ask-password-plymouth.path -
Forward Password Requests to Plymouth Directory Watch.
Sep 04 09:05:35 kali lightdm[872]: gkr-pam: stashed password to try later in
open session
Sep 06 11:44:39 kali systemd[1]: Started systemd-ask-password-wall.path - For
ward Password Requests to Wall Directory Watch.
Sep 06 11:44:39 kali systemd[1]: systemd-ask-password-console.path - Dispatch
>Password Requests to Console Directory Watch was skipped because of an unmet
condition check (ConditionPathExists=!/run/plymouth/pid).
Sep 06 11:44:39 kali systemd[1]: Started systemd-ask-password-plymouth.path -
Forward Password Requests to Plymouth Directory Watch.
Sep 06 11:44:51 kali lightdm[873]: gkr-pam: stashed password to try later in
open session
(kali㉿kali)-[~/var/log]
```

Here you can specifically see me using the journal and group command to find the entries for “password”, out of view, I used those commands to find the five entries for “error”.



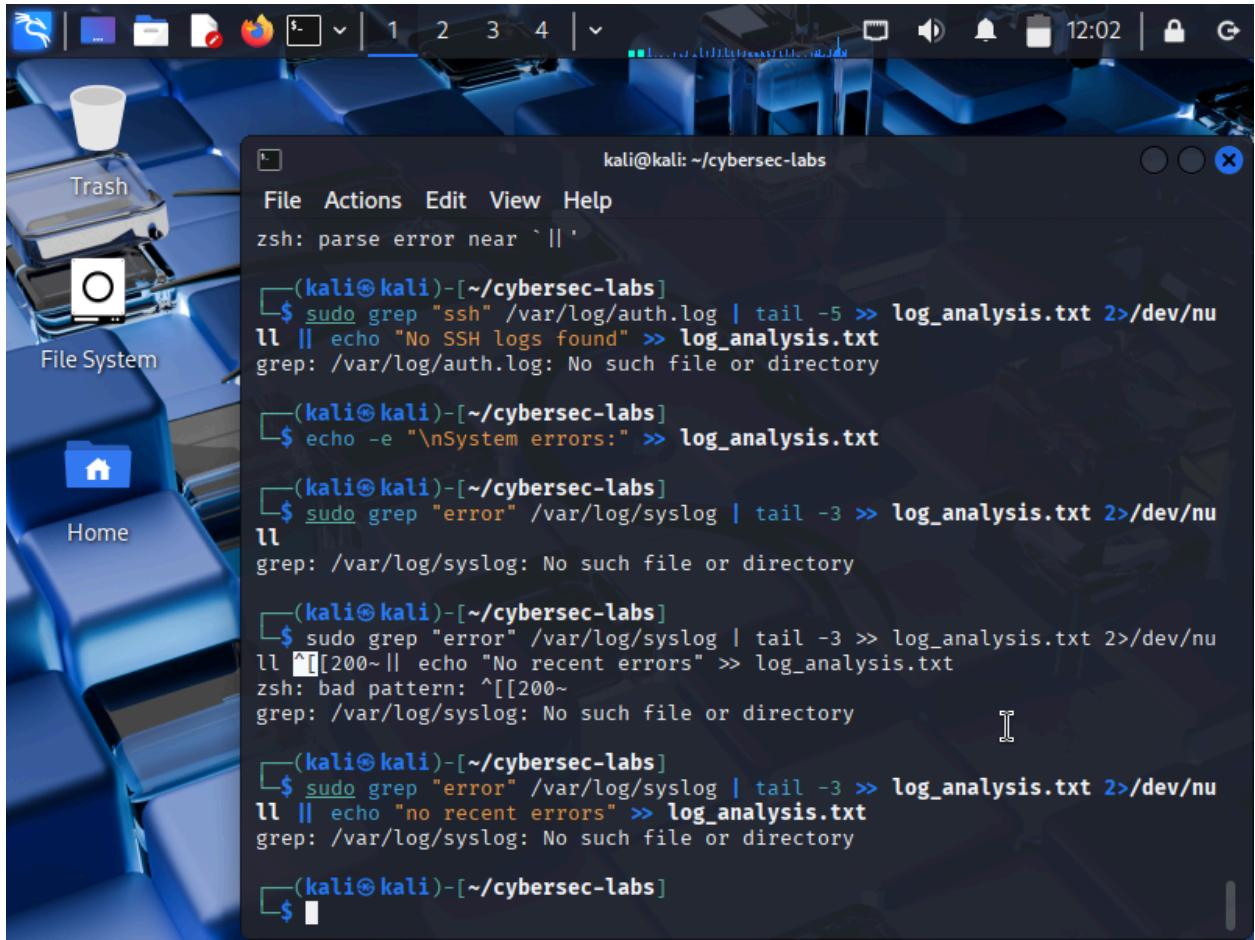
A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. The main area shows a terminal window titled "kali@kali: /var/log". The terminal displays several commands and their outputs related to log file analysis:

```
(kali㉿kali)-[~/var/log]
$ sudo journalctl --user | grep -c "authentication failure"
0

(kali㉿kali)-[~/var/log]
$ ^[[200~sudo grep -r "ssh" /var/log/ | head -10
zsh: bad pattern: ^[[200~sudo

(kali㉿kali)-[~/var/log]
$ sudo grep -r "ssh" /var/log/ | head -10
/var/log/boot.log:[ OK ] Listening on sshd-unix-local.socket - OpenSSH Server Socket (systemd-ssh-generator, AF_UNIX Local).
/var/log/boot.log:[ OK ] Listening on sshd-vsock.socket - OpenSSH Server Socket (systemd-ssh-generator, AF_VSOCK).
/var/log/boot.log:[ OK ] Reached target ssh-access.target - SSH Access Available.
/var/log/boot.log:           Starting regenerate-ssh-host-keys.service - Regenerate SSH host keys ...
/var/log/boot.log:[ OK ] Finished regenerate-ssh-host-keys.service - Regenerate SSH host keys.
/var/log/boot.log:[ OK ] Listening on sshd-unix-local.socket - OpenSSH Server Socket (systemd-ssh-generator, AF_UNIX Local).
/var/log/boot.log:[ OK ] Listening on sshd-vsock.socket - OpenSSH Server Socket (systemd-ssh-generator, AF_VSOCK).
/var/log/boot.log:[ OK ] Reached target ssh-access.target - SSH Access Available.
```

Here, I am counting the occurrences for specific entries and then searching multiple files.



A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. The main area shows a terminal window titled "kali@kali: ~/cybersec-labs". The terminal displays the following command-line session:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
zsh: parse error near `|| '
[(kali㉿kali)-~/cybersec-labs]
$ sudo grep "ssh" /var/log/auth.log | tail -5 >> log_analysis.txt 2>/dev/n
ll || echo "No SSH logs found" >> log_analysis.txt
grep: /var/log/auth.log: No such file or directory

[(kali㉿kali)-~/cybersec-labs]
$ echo -e "\nSystem errors:" >> log_analysis.txt

[(kali㉿kali)-~/cybersec-labs]
$ sudo grep "error" /var/log/syslog | tail -3 >> log_analysis.txt 2>/dev/n
ll
grep: /var/log/syslog: No such file or directory

[(kali㉿kali)-~/cybersec-labs]
$ sudo grep "error" /var/log/syslog | tail -3 >> log_analysis.txt 2>/dev/n
ll ^[[200~|| echo "No recent errors" >> log_analysis.txt
zsh: bad pattern: ^[[200~
grep: /var/log/syslog: No such file or directory

[(kali㉿kali)-~/cybersec-labs]
$ sudo grep "error" /var/log/syslog | tail -3 >> log_analysis.txt 2>/dev/n
ll || echo "no recent errors" >> log_analysis.txt
grep: /var/log/syslog: No such file or directory

[(kali㉿kali)-~/cybersec-labs]
$
```

Tried testing this log analysis, but it's not compatible with Kali Linux.

The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal window has a dark background and displays three separate command examples. The first example shows the output of the 'ps aux' command with specific columns selected using 'cut'. The second example shows the output of 'cat /etc/passwd' with the first five lines and columns 1 through 10 selected. The third example shows the output of 'cat /etc/passwd' with the first five lines and columns 1, 3, and 6 selected. The terminal window also shows the standard file menu (File, Actions, Edit, View, Help) and a status bar indicating the current user and directory.

```
kali@kali: ~/cybersec-labs
File  Actions  Edit  View  Help

[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | head -10 | cut -d ' ' -f1,2,11
USER %CPU
root

[(kali㉿kali)-[~/cybersec-labs]] $ cat /etc/passwd | head -5 | cut -c1-10
root:x:0:0
daemon:x:1
bin:x:2:2
sys:x:3:3
sync:x:4:6

[(kali㉿kali)-[~/cybersec-labs]] $ cat /etc/passwd | head -5 | cut -d':' -f1,3,6
root:0:/root
daemon:1:/usr/sbin
bin:2:/bin
sys:3:/dev
```

Here, I am extracting various things such as users, specific characters, and more in a specific order.

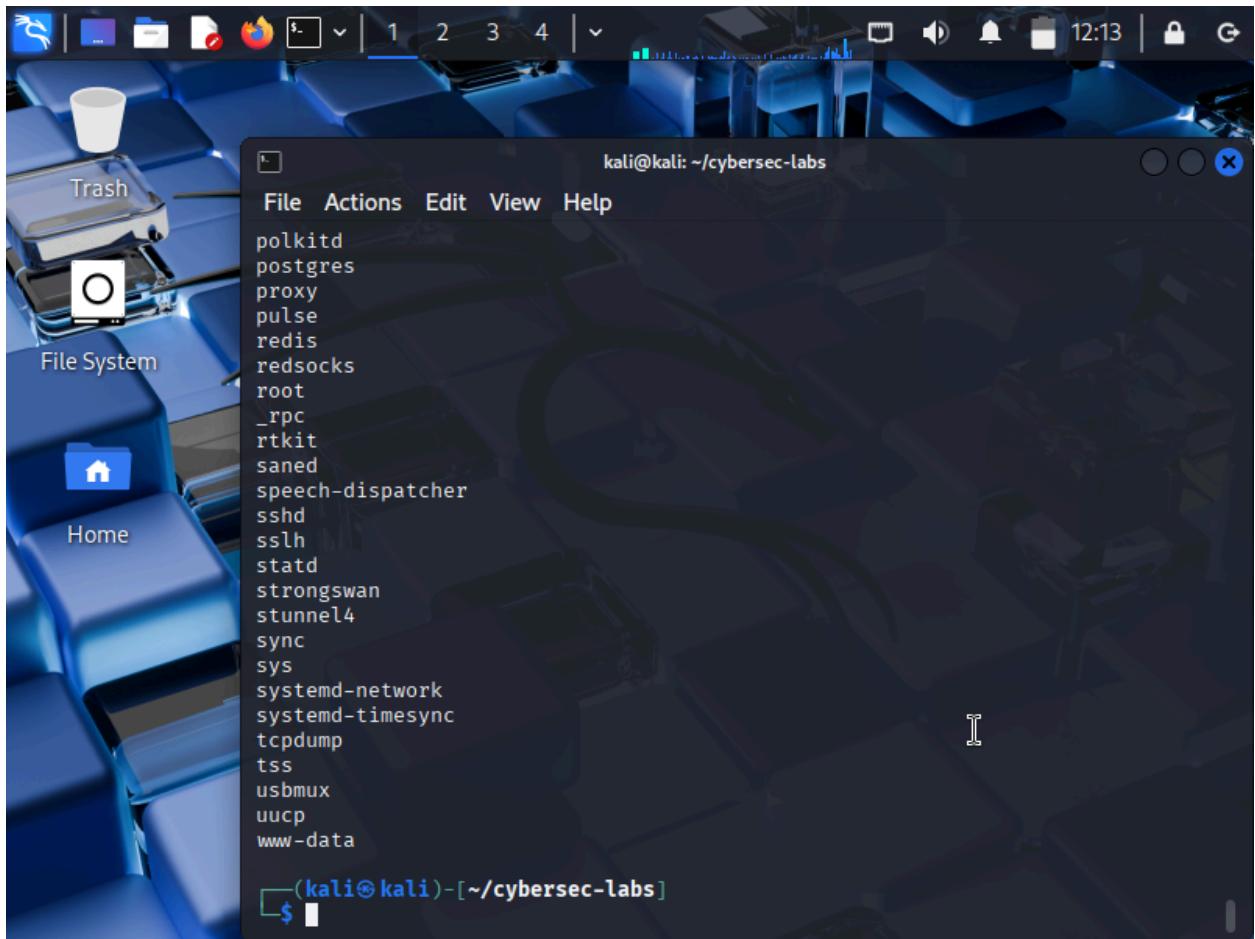
A screenshot of a Kali Linux desktop environment. The desktop background is a blue-toned image of a keyboard. A terminal window is open in the foreground, showing the command output from the command `cat /etc/passwd | awk -F: '{print \$1 ":" \$3}'`.

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
stunnel4:990
_rpc:118
geoclue:119
Debian-snmp:120
sslh:121
ntpsec:122
cups-pk-helper:123
redsocks:124
_gophish:125
iodine:126
miredo:127
statd:128
redis:129
postgres:130
mosquitto:131
inetsim:132
_gvm:133
kali:1000

Service accounts:
daemon:1
bin:2
sys:3
sync:4
games:5

(kali㉿kali)-[~/cybersec-labs]
$
```

Here, I created a user analysis. This displays the username and user ID for all system users and system accounts in /etc/passwd



A screenshot of a Kali Linux desktop environment. The desktop background is a blue keyboard image. On the left, there's a dock with icons for Trash, File System, and Home. A terminal window is open in the center, titled "kali@kali: ~/cybersec-labs". The terminal shows the following command and its output:

```
speech-dispatcher
sshd
sslh
statd
strongswan
stunnel4
sync
sys
systemd-network
systemd-timesync
tcpdump
tss
usbmux
uucp
www-data

(kali㉿kali)-[~/cybersec-labs]
$ ps aux | grep -v grep | cut -d ' ' -f1 | sort | uniq -c | sort -nr
 99 root
 66 kali
   1 USER
   1 rtkit
   1 polkitd
   1 message+
   1 colord
```

In this image, I executed some basic piping. Then I executed a Multi-stage processing where this counts processes by user and sorts by frequency.

The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal window has a dark background and displays the following command-line session:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help

[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | grep -v grep | cut -d ' ' -f1 | sort | uniq -c | sort -nr
99 root
66 kali
1 USER
1 rtkit
1 polkitd
1 message+
1 colord

[(kali㉿kali)-[~/cybersec-labs]] $ echo " 192.168.1.100
dquote> 192.168.1.101
dquote> 192.168.1.102
dquote> 10.0.0.50
dquote> 10.0.0.51
dquote> 172.16.1.100" > ip_list.txt

[(kali㉿kali)-[~/cybersec-labs]] $ cat ip_list.txt | cut -d '.' -f1-3 | sort | uniq -c
2 10.0.0
1 172.16.1
1 192.168.1
2 192.168.1

[(kali㉿kali)-[~/cybersec-labs]] $
```

In this portion of the exercise, I got the machine to analyze the IP address ranges.

The screenshot shows a Kali Linux desktop environment with a terminal window open. The terminal window title is "kali@kali: ~/cybersec-labs". The terminal content displays a log file analysis pipeline:

```
dquote> 192.168.1.101
dquote> 192.168.1.102
dquote> 10.0.0.50
dquote> 10.0.0.51
dquote> 172.16.1.100" > ip_list.txt

(kali㉿kali)-[~/cybersec-labs]
$ cat ip_list.txt | cut -d '.' -f1-3 | sort | uniq -c
 2 10.0.0
 1 172.16.1
 1 192.168.1
 2 192.168.1

(kali㉿kali)-[~/cybersec-labs]
$ echo "≡≡ Network Traffic Analysis ≡≡" > traffic_analysis.txt

(kali㉿kali)-[~/cybersec-labs]
$ echo "Unique IP subnets:" >> traffic_analysis.txt

(kali㉿kali)-[~/cybersec-labs]
$ cat ip_list.txt | cut -d '.' -f1-3 | sort | uniq >> traffic_analysis.txt

(kali㉿kali)-[~/cybersec-labs]
$ echo -e "\nSubnet frequency:" >> traffic_analysis.txt

(kali㉿kali)-[~/cybersec-labs]
$ cat ip_list.txt | cut -d '.' -f1-3 | sort | uniq -c | sort -nr >> traffic_analysis.txt
```

In this, I create a log file analysis pipeline.

A screenshot of a Kali Linux desktop environment. The desktop background is a blue keyboard image. A terminal window is open in the foreground, titled "kali@kali: ~/cybersec-labs". The terminal shows the following command and output:

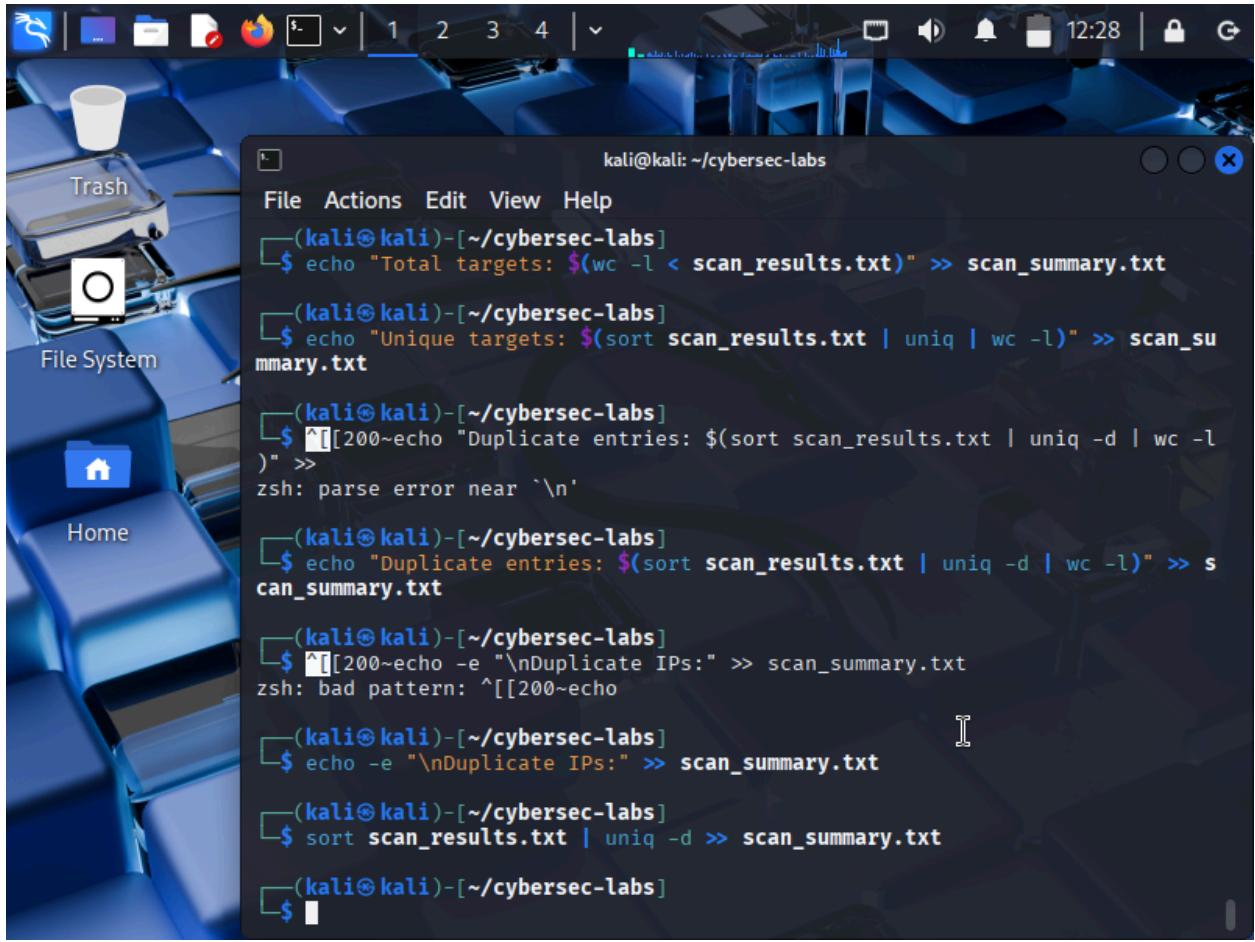
```
7
8
9
10
13
33
34
38
39
42
100
101
102

[(kali㉿kali)-[~/cybersec-labs]] $ echo -e "admin\nuser\nadmin\nroot\nuser" | sort | uniq
admin
root
user

[(kali㉿kali)-[~/cybersec-labs]] $ echo -e "admin\nuser\nadmin\nroot\nuser" | sort | uniq -c
2 admin
1 root
2 user

[(kali㉿kali)-[~/cybersec-labs]] $
```

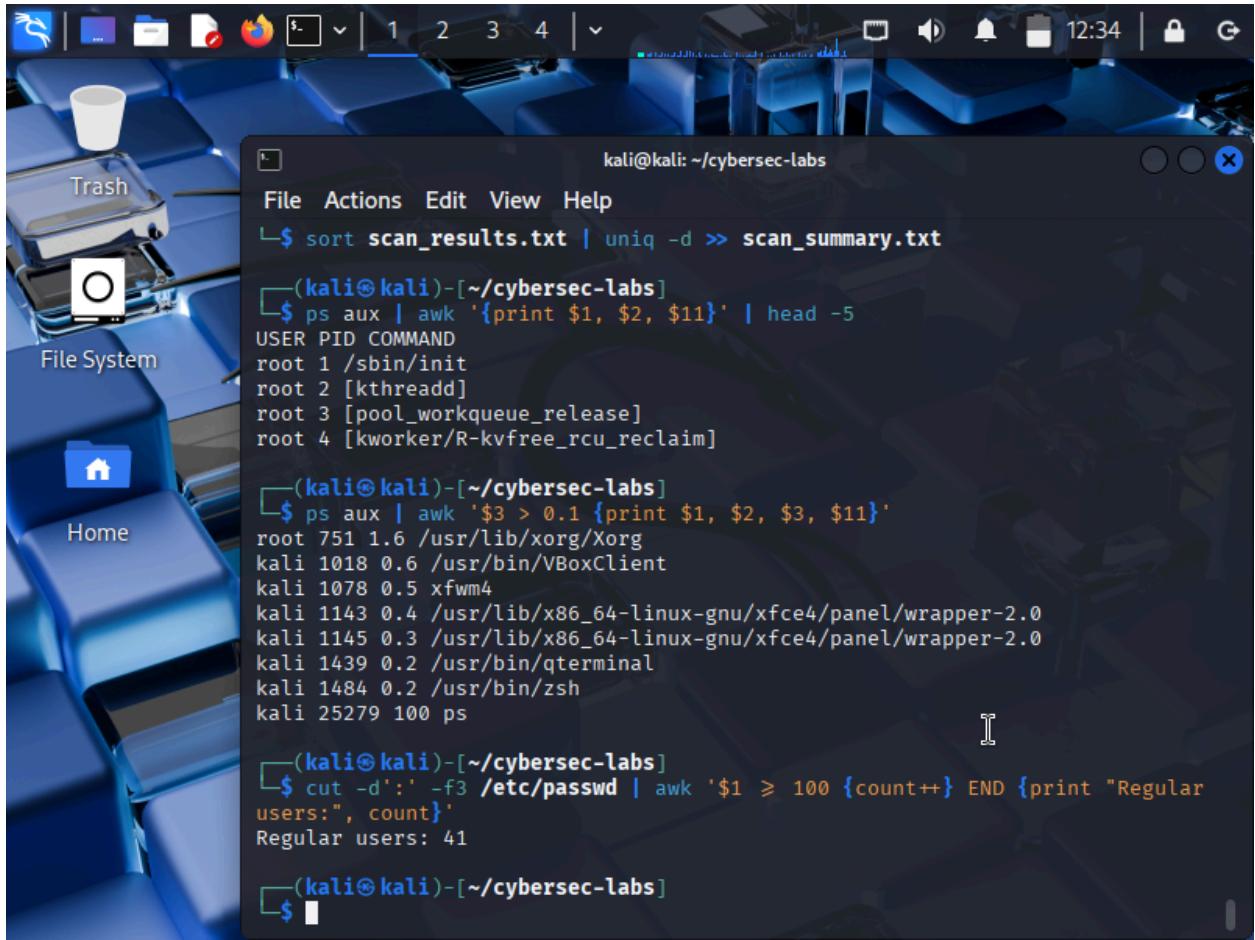
Here, I performed some basic sorting, numerical sorting, and then removed duplicates.



A screenshot of a Kali Linux desktop environment. The terminal window is open and shows a series of shell commands being run. The terminal title is "kali@kali: ~/cybersec-labs". The commands are as follows:

```
kali@kali: ~/cybersec-labs
$ echo "Total targets: $(wc -l < scan_results.txt)" >> scan_summary.txt
$ echo "Unique targets: $(sort scan_results.txt | uniq | wc -l)" >> scan_summary.txt
$ ^[[200~echo "Duplicate entries: $(sort scan_results.txt | uniq -d | wc -l)" >> scan_summary.txt
zsh: parse error near `\'n'
$ echo "Duplicate entries: $(sort scan_results.txt | uniq -d | wc -l)" >> scan_summary.txt
$ ^[[200~echo -e "\nDuplicate IPs:" >> scan_summary.txt
$ echo -e "\nDuplicate IPs:" >> scan_summary.txt
$ sort scan_results.txt | uniq -d >> scan_summary.txt
$
```

Finally, in exercise 6, I found the two duplicates, then created a comprehensive analysis.



A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. The main area shows a terminal window titled "kali@kali: ~/cybersec-labs". The terminal displays the following command-line session:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
└$ sort scan_results.txt | uniq -d >> scan_summary.txt

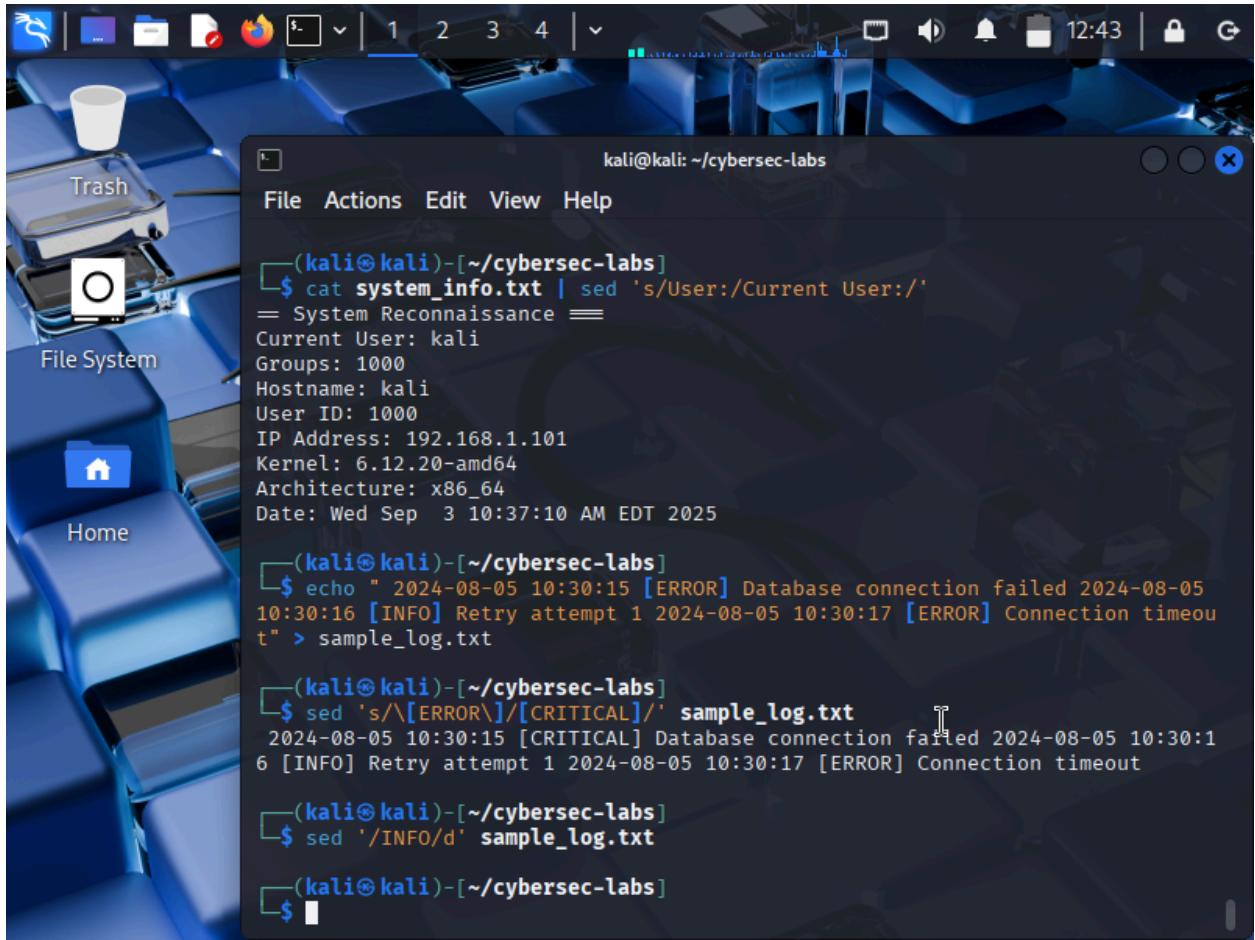
[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | awk '{print $1, $2, $11}' | head -5
USER PID COMMAND
root 1 /sbin/init
root 2 [kthreadd]
root 3 [pool_workqueue_release]
root 4 [kworker/R-kvfree_rcu_reclaim]

[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | awk '$3 > 0.1 {print $1, $2, $3, $11}'
root 751 1.6 /usr/lib/xorg/Xorg
kali 1018 0.6 /usr/bin/VBoxClient
kali 1078 0.5 xfwm4
kali 1143 0.4 /usr/lib/x86_64-linux-gnu/xfce4/panel/wrapper-2.0
kali 1145 0.3 /usr/lib/x86_64-linux-gnu/xfce4/panel/wrapper-2.0
kali 1439 0.2 /usr/bin/qterminal
kali 1484 0.2 /usr/bin/zsh
kali 25279 100 ps

[(kali㉿kali)-[~/cybersec-labs]] $ cut -d':' -f3 /etc/passwd | awk '$1 > 100 {count++} END {print "Regular users:", count}'
Regular users: 41

[(kali㉿kali)-[~/cybersec-labs]] $
```

In these few commands, I printed specific columns with clear outputs rather than a cut. Then I used a command to provide me with an output that shows processes using more than 0.1% CPU. Then I used an awk calculation command that counts the number of regular users by seeking user IDs starting at 1000.



A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. The main area shows a terminal window titled "kali@kali: ~/cybersec-labs". The terminal displays the following session:

```
(kali㉿kali)-[~/cybersec-labs]
$ cat system_info.txt | sed 's/User:/Current User:/'
= System Reconnaissance =
Current User: kali
Groups: 1000
Hostname: kali
User ID: 1000
IP Address: 192.168.1.101
Kernel: 6.12.20-amd64
Architecture: x86_64
Date: Wed Sep 3 10:37:10 AM EDT 2025

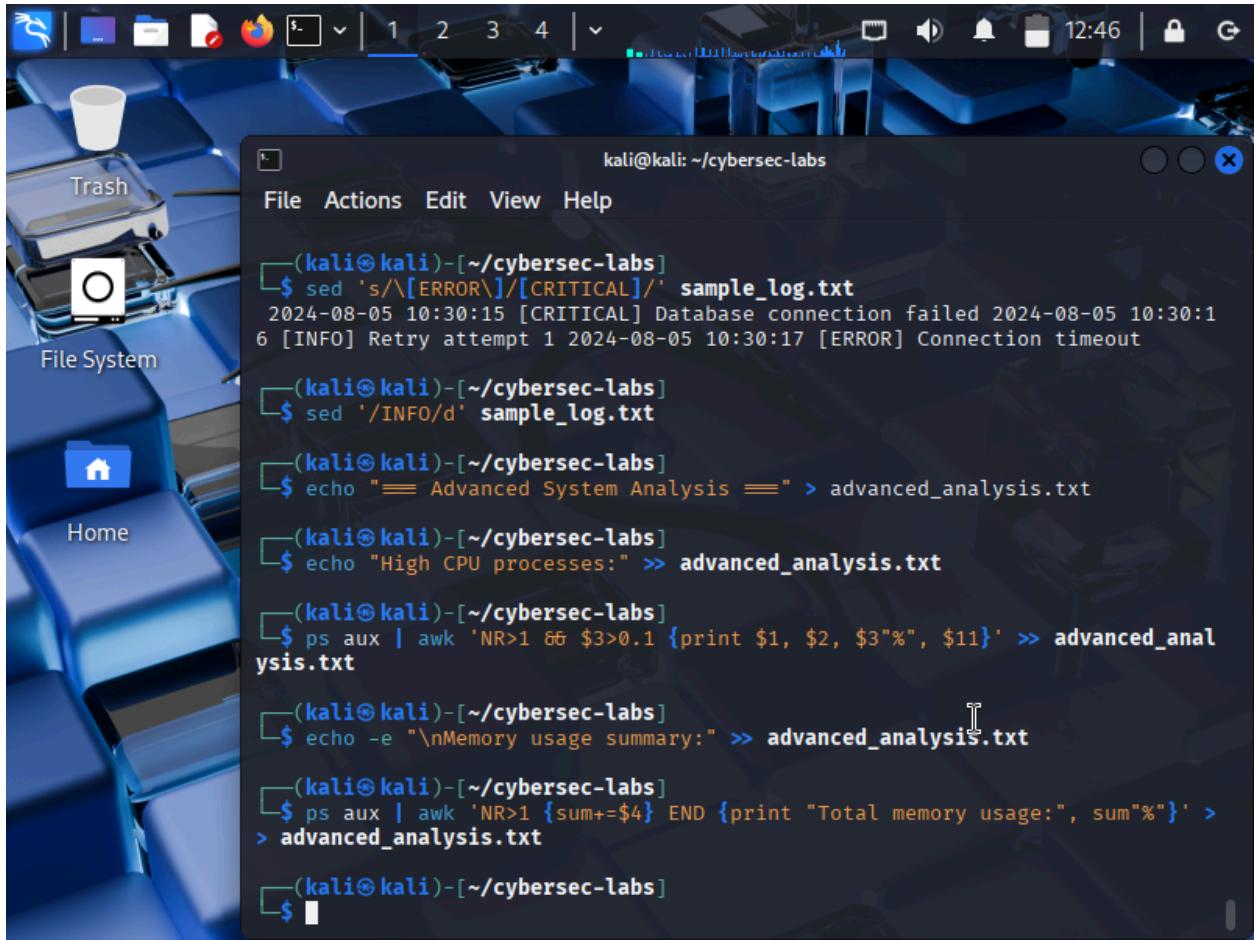
(kali㉿kali)-[~/cybersec-labs]
$ echo " 2024-08-05 10:30:15 [ERROR] Database connection failed 2024-08-05
10:30:16 [INFO] Retry attempt 1 2024-08-05 10:30:17 [ERROR] Connection timeout
t" > sample_log.txt

(kali㉿kali)-[~/cybersec-labs]
$ sed 's/\[ERROR\]/\[CRITICAL\]/' sample_log.txt
2024-08-05 10:30:15 [CRITICAL] Database connection failed 2024-08-05 10:30:16 [INFO] Retry attempt 1 2024-08-05 10:30:17 [ERROR] Connection timeout

(kali㉿kali)-[~/cybersec-labs]
$ sed '/INFO/d' sample_log.txt

(kali㉿kali)-[~/cybersec-labs]
$
```

Used a simple string substitution. Then I used a sed command for log cleaning.



A screenshot of a Kali Linux desktop environment. The terminal window shows a series of command-line operations to analyze log files:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help

[(kali㉿kali)-[~/cybersec-labs]] $ sed 's/\[ERROR\]/\[CRITICAL\]/' sample_log.txt
2024-08-05 10:30:15 [CRITICAL] Database connection failed 2024-08-05 10:30:1
6 [INFO] Retry attempt 1 2024-08-05 10:30:17 [ERROR] Connection timeout

[(kali㉿kali)-[~/cybersec-labs]] $ sed '/INFO/d' sample_log.txt

[(kali㉿kali)-[~/cybersec-labs]] $ echo "≡ Advanced System Analysis ≡" > advanced_analysis.txt

[(kali㉿kali)-[~/cybersec-labs]] $ echo "High CPU processes:" >> advanced_analysis.txt

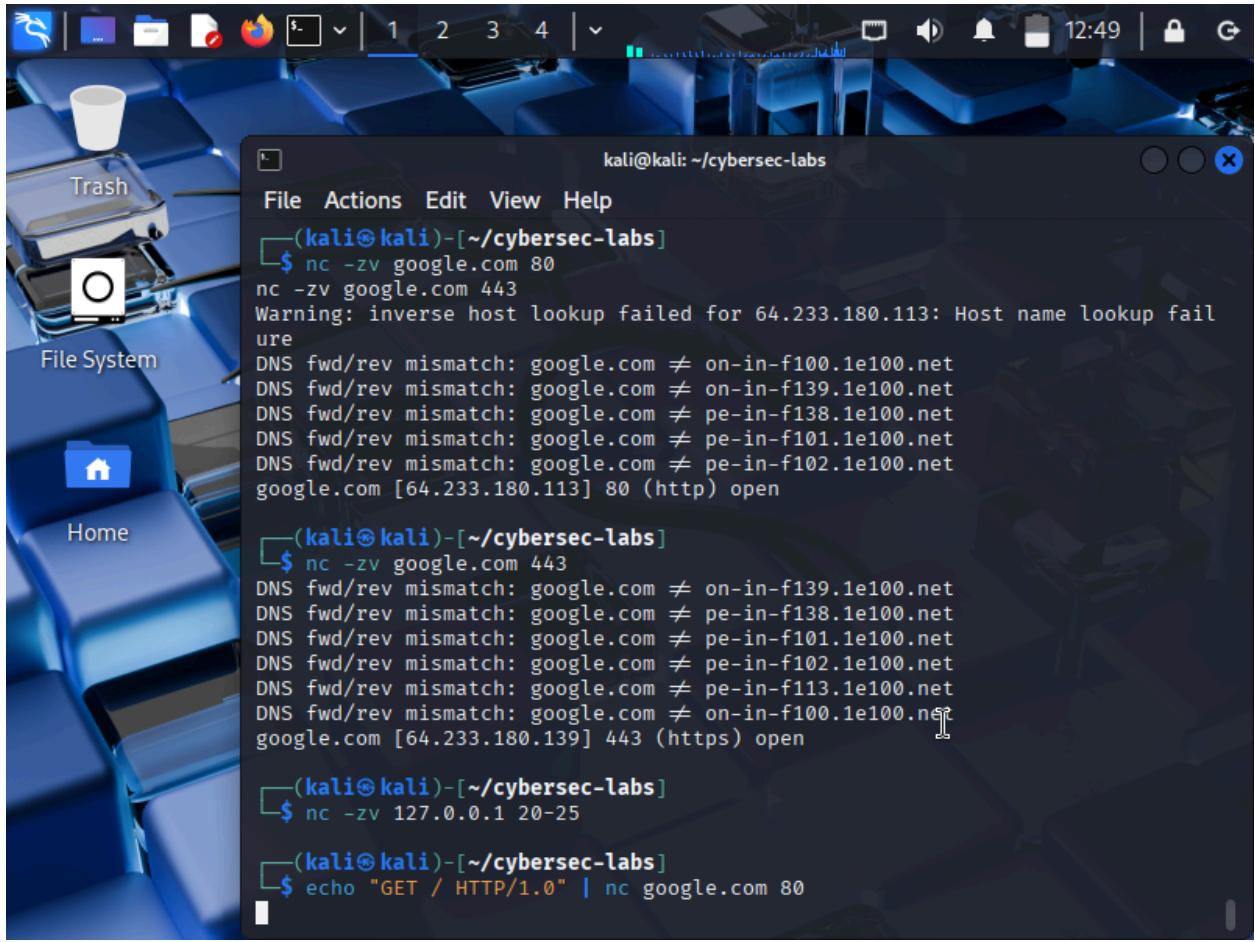
[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | awk 'NR>1 && $3>0.1 {print $1, $2, $3"%", $11}' >> advanced_analysis.txt

[(kali㉿kali)-[~/cybersec-labs]] $ echo -e "\nMemory usage summary:" >> advanced_analysis.txt

[(kali㉿kali)-[~/cybersec-labs]] $ ps aux | awk 'NR>1 {sum+=$4} END {print "Total memory usage:", sum%"}' >
> advanced_analysis.txt

[(kali㉿kali)-[~/cybersec-labs]] $
```

Here, I created an advanced analysis.



A screenshot of a Kali Linux desktop environment. The terminal window shows the following command-line session:

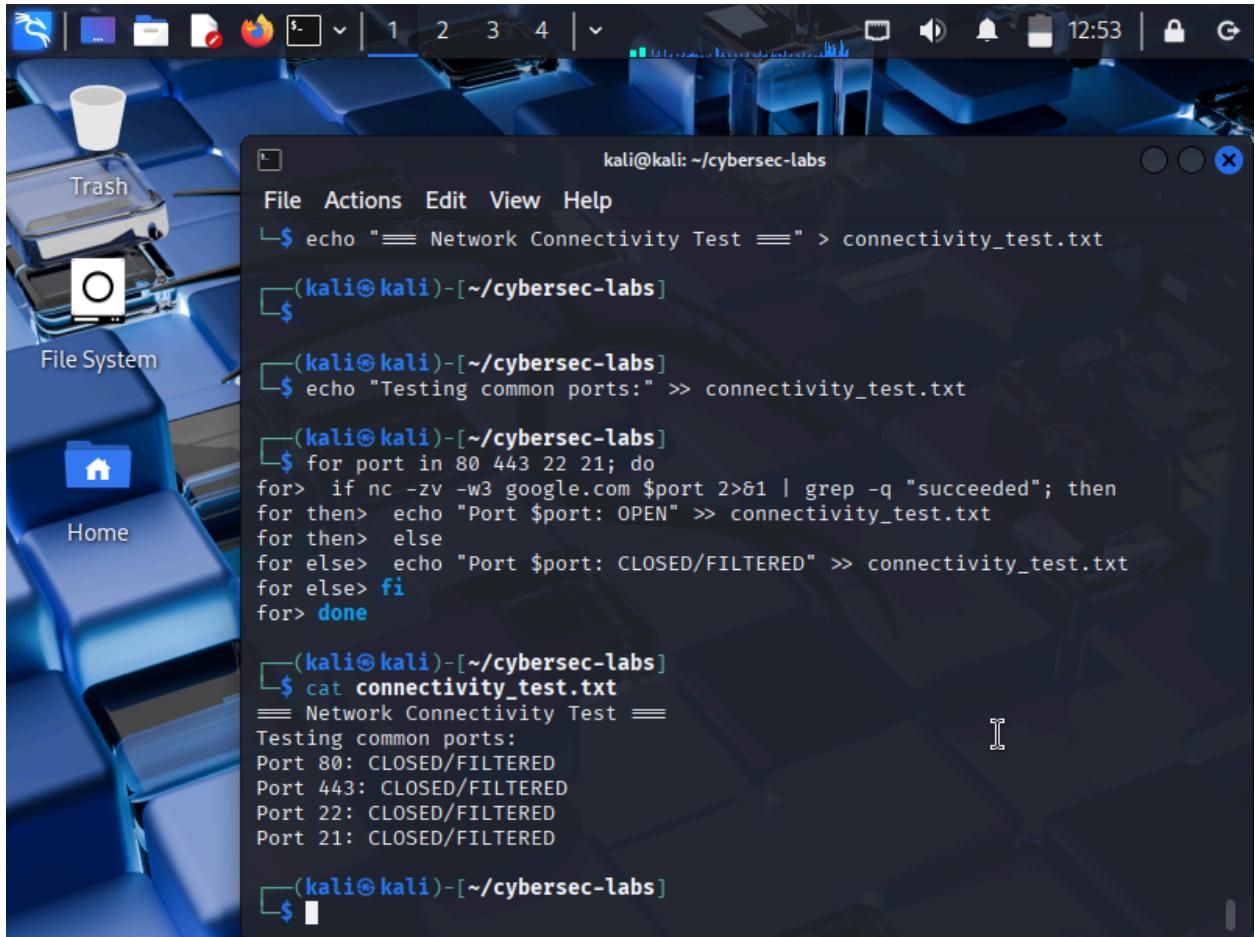
```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
(kali㉿kali)-[~/cybersec-labs]
$ nc -zv google.com 80
nc -zv google.com 443
Warning: inverse host lookup failed for 64.233.180.113: Host name lookup failure
DNS fwd/rev mismatch: google.com ≠ on-in-f100.1e100.net
DNS fwd/rev mismatch: google.com ≠ on-in-f139.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f138.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f101.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f102.1e100.net
google.com [64.233.180.113] 80 (http) open

(kali㉿kali)-[~/cybersec-labs]
$ nc -zv google.com 443
DNS fwd/rev mismatch: google.com ≠ on-in-f139.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f138.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f101.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f102.1e100.net
DNS fwd/rev mismatch: google.com ≠ pe-in-f113.1e100.net
DNS fwd/rev mismatch: google.com ≠ on-in-f100.1e100.net
google.com [64.233.180.139] 443 (https) open

(kali㉿kali)-[~/cybersec-labs]
$ nc -zv 127.0.0.1 20-25

(kali㉿kali)-[~/cybersec-labs]
$ echo "GET / HTTP/1.0" | nc google.com 80
```

In this exercise, I tested two ports to see if they were open on Google. Then I scanned through ports 20-25 on the local host.



A screenshot of a Kali Linux desktop environment. On the left, there's a dock with icons for Trash, File System, and Home. A terminal window is open in the center, titled 'kali@kali: ~/cybersec-labs'. The terminal shows the following command history:

```
kali@kali: ~/cybersec-labs
File Actions Edit View Help
└$ echo "==== Network Connectivity Test ===" > connectivity_test.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ 
└(kali㉿kali)-[~/cybersec-labs]
└$ echo "Testing common ports:" >> connectivity_test.txt
└(kali㉿kali)-[~/cybersec-labs]
└$ for port in 80 443 22 21; do
for> if nc -zv -w3 google.com $port 2>&1 | grep -q "succeeded"; then
for then> echo "Port $port: OPEN" >> connectivity_test.txt
for then> else
for else> echo "Port $port: CLOSED/FILTERED" >> connectivity_test.txt
for else> fi
for> done
└(kali㉿kali)-[~/cybersec-labs]
└$ cat connectivity_test.txt
==== Network Connectivity Test ===
Testing common ports:
Port 80: CLOSED/FILTERED
Port 443: CLOSED/FILTERED
Port 22: CLOSED/FILTERED
Port 21: CLOSED/FILTERED
└(kali㉿kali)-[~/cybersec-labs]
└$
```

In this assessment, I completed a banner grab, followed by creating a connectivity test. I can't say that I understand it completely or did it correctly. I may have messed up the indentation on the command fi.

Knowledge Assessment:

Question 1: True or False: The command ps aux | grep ssh | wc -l counts the number of SSH-related processes currently running.

False

Question 2: Which command combination would extract only the usernames from the/etc/passwd file?

- a) **cat /etc/passwd | cut -d':' -f1**

Question 3: True or False: The uniq command removes duplicate lines even if the input is not sorted.

False

Question 4: What does the command nc -zv google.com 80 accomplish?

b) Tests if port 80 is open on google.com

Question 5: Which command would show processes using more than 1% CPU?

b) ps aux | awk '\$3 > 1'

Conclusion:

With the conclusion of lab 2, I am now well acquainted with Kali Linux. After building off of Lab 1, I am now comfortable with the previous commands and exercises committed in Lab 1. After the conclusion of Lab 2, I have a greater understanding of extracting and processing text data using grep, cut, awk, and sed. Gather comprehensive system information, use piping to create complex data processing workflows, and analyze processes and network connectivity.