2ND APRIL 2025

ASSIGNMENT

311-BASIC COMPUTER SKILLS FOR DIGITAL FORENSIS

SUBMITTED BY
Ayilara Busari Dare

IDEAS/24/28133

Basic Computer Skills for Digital Forensics – Lab 4

Introduction to Operating Systems and File Systems in Linux

Operating System (OS) Overview: An operating system like Linux manages hardware resources and provides services for applications. It handles processes, memory, device management, security, and file management. Linux is known for being open-source, secure, and flexible.

Linux File System: A file system organizes and stores files on storage devices. Linux uses a hierarchical file system starting from the root directory (/). Common file systems in Linux include:

- Ext4: Popular and reliable.
- XFS: Used for large-scale data.
- Btrfs: Focuses on scalability and data integrity.

File Permissions: Linux uses a permission system for files, including read, write, and execute permissions, which can be assigned to the owner, group, and others.

Mounting and Unmounting: Storage devices must be mounted to access them and unmounted to ensure data integrity.

In short, Linux provides a secure and flexible OS with a hierarchical file system, efficient file management, and strong access controls.

Task:

o Write a brief summary of at least two advantages and two challenges of using Linux for digital forensics.

ANS:

Advantages of Using Linux for Digital Forensics:

1. Open Source: Linux's open-source nature allows forensic professionals to inspect, modify, and adapt tools for specific needs, ensuring transparency and reliability.

2. Powerful Command-Line Tools: Linux provides a variety of command-line utilities, like dd and grep, that enable efficient and precise forensic investigations.

Challenges of Using Linux for Digital Forensics:

1. File System Compatibility: Linux may face issues when working with nonnative file systems, such as NTFS, making it harder to analyze certain data types.

2. Learning Curve: For investigators new to Linux, the system's command-line interface and file management can be challenging to master, requiring additional time to learn.

Part 2: Virtual File System and File Structure

1. Virtual File System (VFS):
    - o Explain what a Virtual File System is and its role in Linux.

ANS:

A Virtual File System (VFS) is an abstraction layer in the Linux kernel that provides a uniform interface for interacting with different file systems. Its role is to allow the kernel to access various file systems without needing to understand the specific details of each one. VFS acts as a bridge between user applications and the underlying file systems.
Role of VFS in Linux:

1. Abstraction: VFS hides the complexity of different file systems (e.g., Ext4, NTFS, XFS) and provides a standard interface for all file operations. Applications and system processes can work with files without worrying about the specifics of the file system used.
2. Uniform Interface: It allows the kernel to handle system calls like open, read, write, and close uniformly, regardless of the underlying file system type. For example, whether you're accessing an Ext4 file system or an NTFS file system, the process remains the same from the kernel's perspective.
3. File System Management: VFS manages the mounting and unmounting of file systems. When a new file system is added (e.g., a USB drive with NTFS), VFS ensures that Linux can interact with it using a standardized approach, even though the file system is not native to Linux.
4. Performance Optimization: VFS helps in optimizing file system interactions by providing an efficient caching mechanism. It reduces the number of disk accesses by caching file data, which improves overall performance.

In short, VFS is crucial for enabling Linux to support multiple file systems seamlessly, providing a standardized interface for file operations, and simplifying file system management.

- Discuss how VFS allows Linux to support multiple file systems seamlessly.

ANS:

The Virtual File System (VFS) in Linux allows seamless support for multiple file systems by providing an abstraction layer between user applications and the underlying file systems. VFS standardizes file operations (like read, write) across different file systems (e.g., Ext4, NTFS). It uses specific file system drivers to handle each type, ensuring consistent interaction. VFS also allows multiple file systems to be mounted simultaneously and optimizes performance through caching, making it possible for Linux to work with various file systems without modification to the core system.

2. File Structure:
   - Describe the hierarchical file structure in Linux and the significance of the root directory (/).

ANS:

Hierarchical File Structure in Linux

Linux uses a hierarchical file structure where all files and directories are organized in a tree-like format, starting from a single root directory.

- Root Directory (/): The top-level directory in the hierarchy. All other directories and files are contained within it, directly or indirectly.
  - Example: /home, /bin, /etc, etc., are subdirectories under the root directory.

Key Directories in Linux:

1. /home: Contains user-specific directories, where each user's personal files are stored (e.g., /home/user).
2. /bin: Stores essential system binaries (programs) required for basic operations, like ls and cp.
3. /etc: Contains configuration files for the system and software (e.g., /etc/passwd).
4. /var: Holds variable files such as logs and spools (e.g., /var/log for system logs).
5. /tmp: Temporary files used by applications.
6. /usr: Contains user programs and data, usually installed software.
7. /dev: Contains device files that represent hardware devices (e.g., /dev/sda for storage devices).
8. /lib: Contains shared libraries essential for system programs.

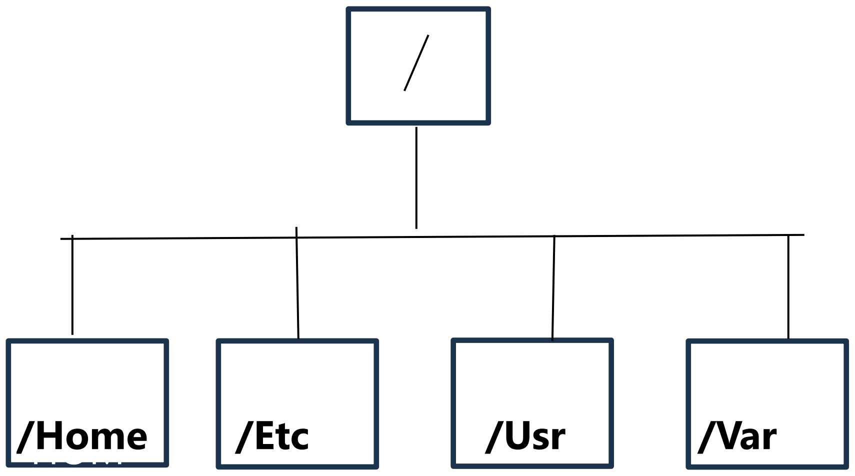
Significance of the Root Directory (/):

- Central Starting Point: The root directory is the root of the entire file system, and all other files and directories branch off from it. It's where the file system begins, acting as the starting point for all paths.
- System Integrity: The root directory holds critical system files and directories that are essential for booting and running the operating system.
- Unique to Unix-like Systems: Unlike some OSes, Linux does not use drive letters (e.g., C: drive) but a single unified file system starting from /.

In summary, the root directory (/) is the foundation of the Linux file structure, where everything is organized and from which all paths originate. The hierarchical structure ensures a well-organized system where files and directories are logically grouped.

Task:

- Create a diagram representing the Linux file structure, highlighting key directories such as /home, /etc, /usr, and /var.



Part 3: Path and Path Variable

1. Path Variable:
   o Define what the path variable is in Linux and its importance in executing commands.

ANS:

The Path variable is an environment variable in Linux that specifies a list of directories in which the operating system searches for executable files (commands). It is essential for the shell (like Bash) to locate and execute commands without requiring the full path to be specified.

When you type a command in the terminal, the system checks the directories listed in the Path variable in order, looking for an executable file that matches the command. If the command is found, it is executed. If not, an error message is displayed.

Importance:
Efficiency: Without the Path variable, you would need to provide the full path (like /usr/bin/ls) every time you want to run a command. The Path variable simplifies this by allowing you to type just the command (like ls).

Customization: You can customize the Path variable to include directories containing your own scripts or programs, making them accessible globally without needing to specify the full path each time.

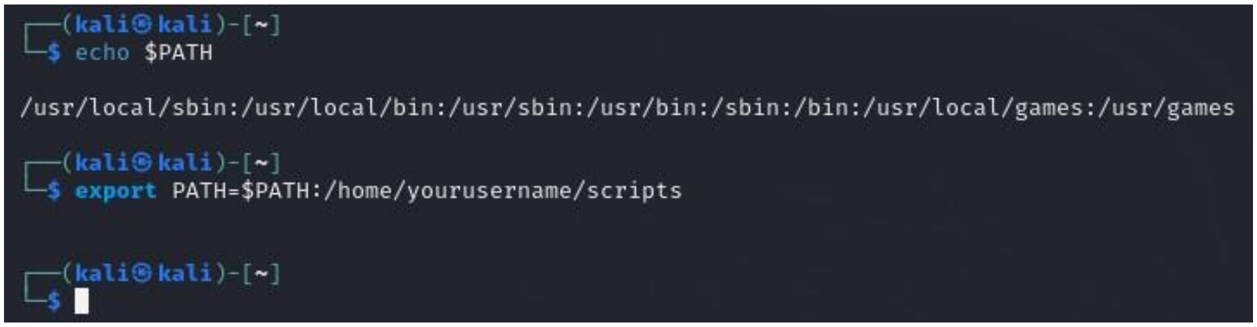
Security: It allows system administrators to control which directories are searched for executables, which can help in preventing the execution of potentially harmful commands or software.

Task:

- Display the current path variable by executing the command: echo $PATH

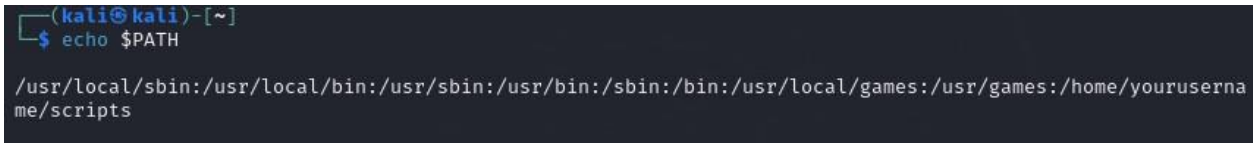
- Add a new directory (e.g., /home/yourusername/scripts) to the path variable temporarily for the session:

  export PATH=$PATH:/home/yourusername/scripts



```
┌──(kali㉿kali)-[~]
└─$ echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games

┌──(kali㉿kali)-[~]
└─$ export PATH=$PATH:/home/yourusername/scripts


┌──(kali㉿kali)-[~]
└─$ █
```

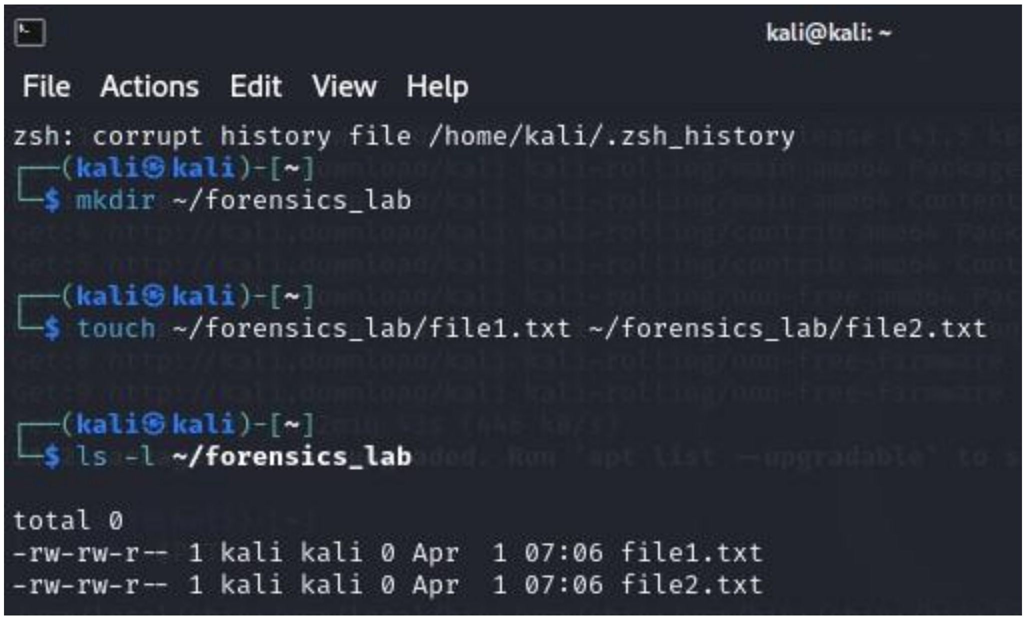
Task: o Confirm the new path by executing the echo $PATH command again.



```
┌──(kali㉿kali)-[~]
└─$ echo $PATH

/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games:/home/yourusername/scripts
```

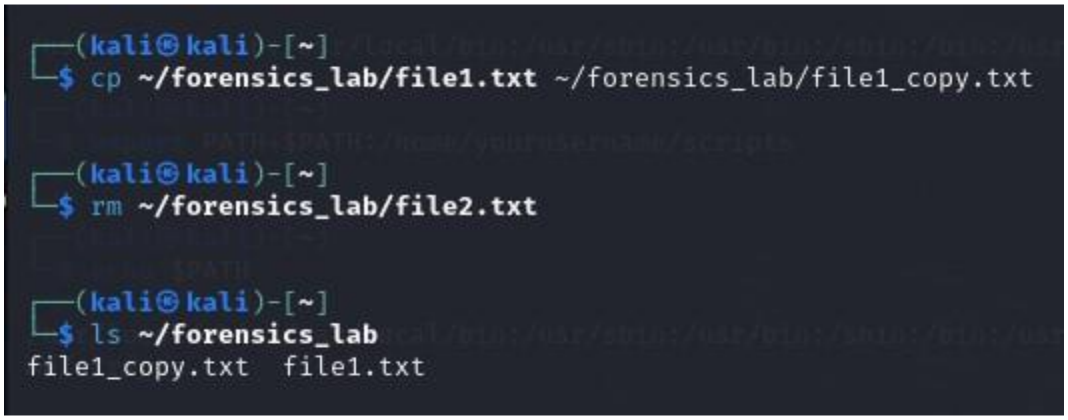
Part 4: Essential Linux Commands

1. Creating Folders and Files:



```
zsh: corrupt history file /home/kali/.zsh_history
┌──(kali㉿kali)-[~]
└─$ mkdir ~/forensics_lab

┌──(kali㉿kali)-[~]
└─$ touch ~/forensics_lab/file1.txt ~/forensics_lab/file2.txt

┌──(kali㉿kali)-[~]
└─$ ls -l ~/forensics_lab

total 0
-rw-rw-r-- 1 kali kali 0 Apr  1 07:06 file1.txt
-rw-rw-r-- 1 kali kali 0 Apr  1 07:06 file2.txt
```

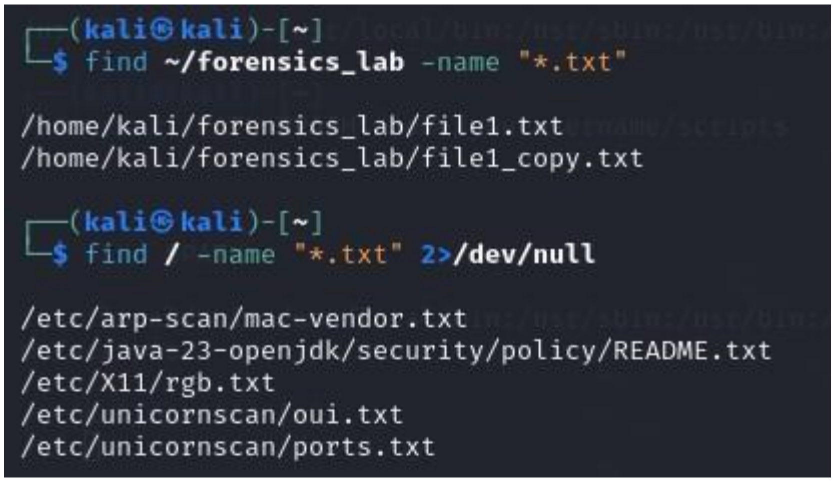
2. File Copying and Deletion:
   - Copy file1.txt to a new file named file1_copy.txt:
3. Task:
   - Verify that the copy and deletion were successful by listing the contents of the forensics_lab directory:



```
┌──(kali㉿kali)-[~]
└─$ cp ~/forensics_lab/file1.txt ~/forensics_lab/file1_copy.txt

┌──(kali㉿kali)-[~]
└─$ rm ~/forensics_lab/file2.txt

┌──(kali㉿kali)-[~]
└─$ ls ~/forensics_lab
file1_copy.txt  file1.txt
```

Part 5: Searching for Information

1. Searching for Files:



```
┌──(kali㉿kali)-[~]
└─$ find ~/forensics_lab -name "*.txt"

/home/kali/forensics_lab/file1.txt
/home/kali/forensics_lab/file1_copy.txt

┌──(kali㉿kali)-[~]
└─$ find / -name "*.txt" 2>/dev/null

/etc/arp-scan/mac-vendor.txt
/etc/java-23-openjdk/security/policy/README.txt
/etc/X11/rgb.txt
/etc/unicornscan/oui.txt
/etc/unicornscan/ports.txt
```

2. Task:
   o Document the output of the command and explain what it shows.

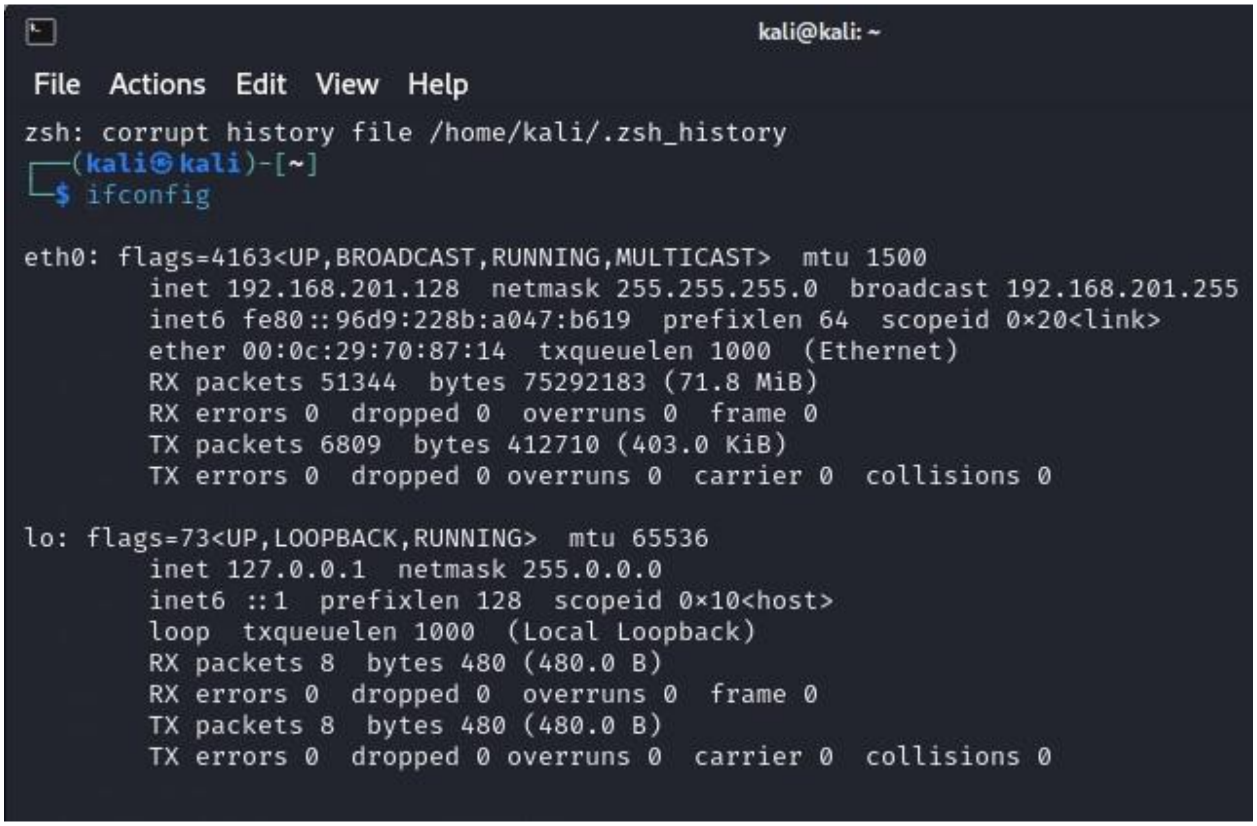
Explanation:
The find command is used to search for files and directories. In this case:
- ~/forensics_lab specifies the directory to search in.
- -name "*.txt" tells find to look for files with the .txt extension. The output lists all .txt files found inside the forensics_lab directory:

1. /home/kali/forensics_lab/file1.txt → The original text file. 2. /home/kali/forensics_lab/file1_copy.txt A copied version of file1.txt.

This confirms that both .txt files exist in the specified directory.

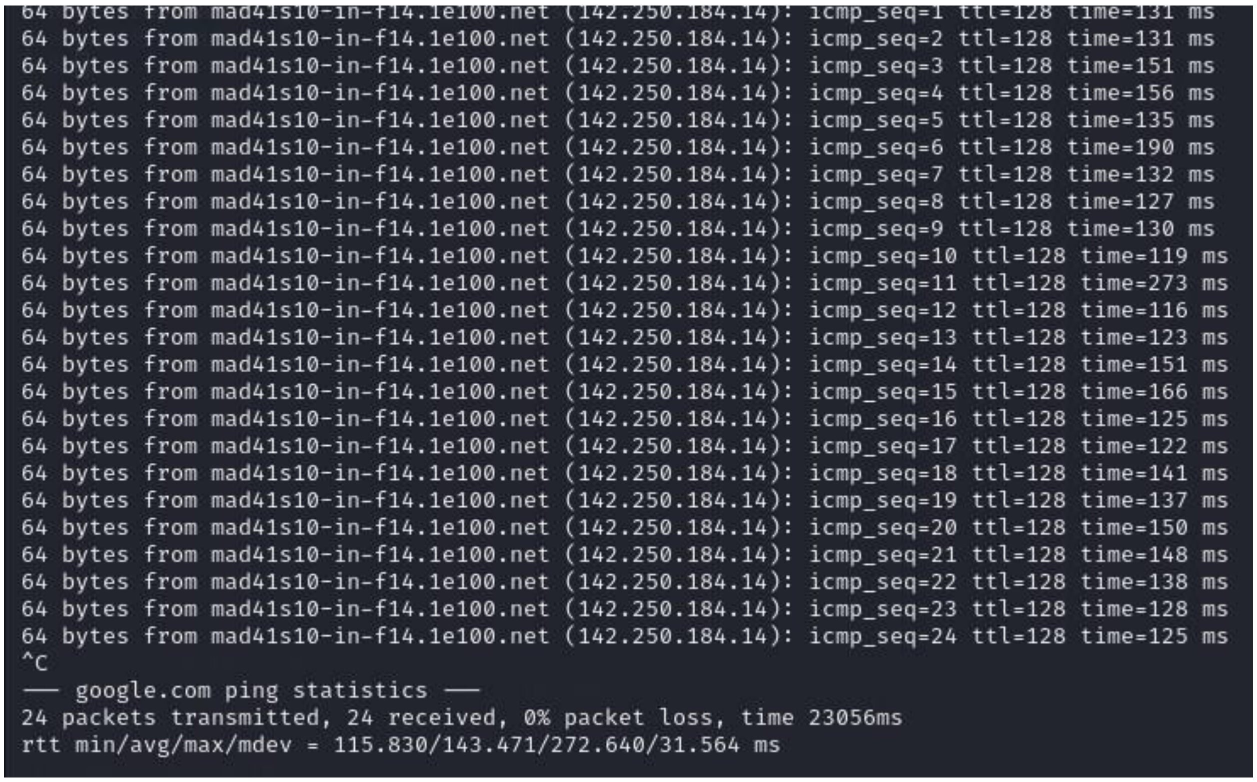
Part 6: Networking in Linux

1. Basic Networking Commands:
   - Execute the following commands to gather network information: ▪ Display network interfaces and their configurations.



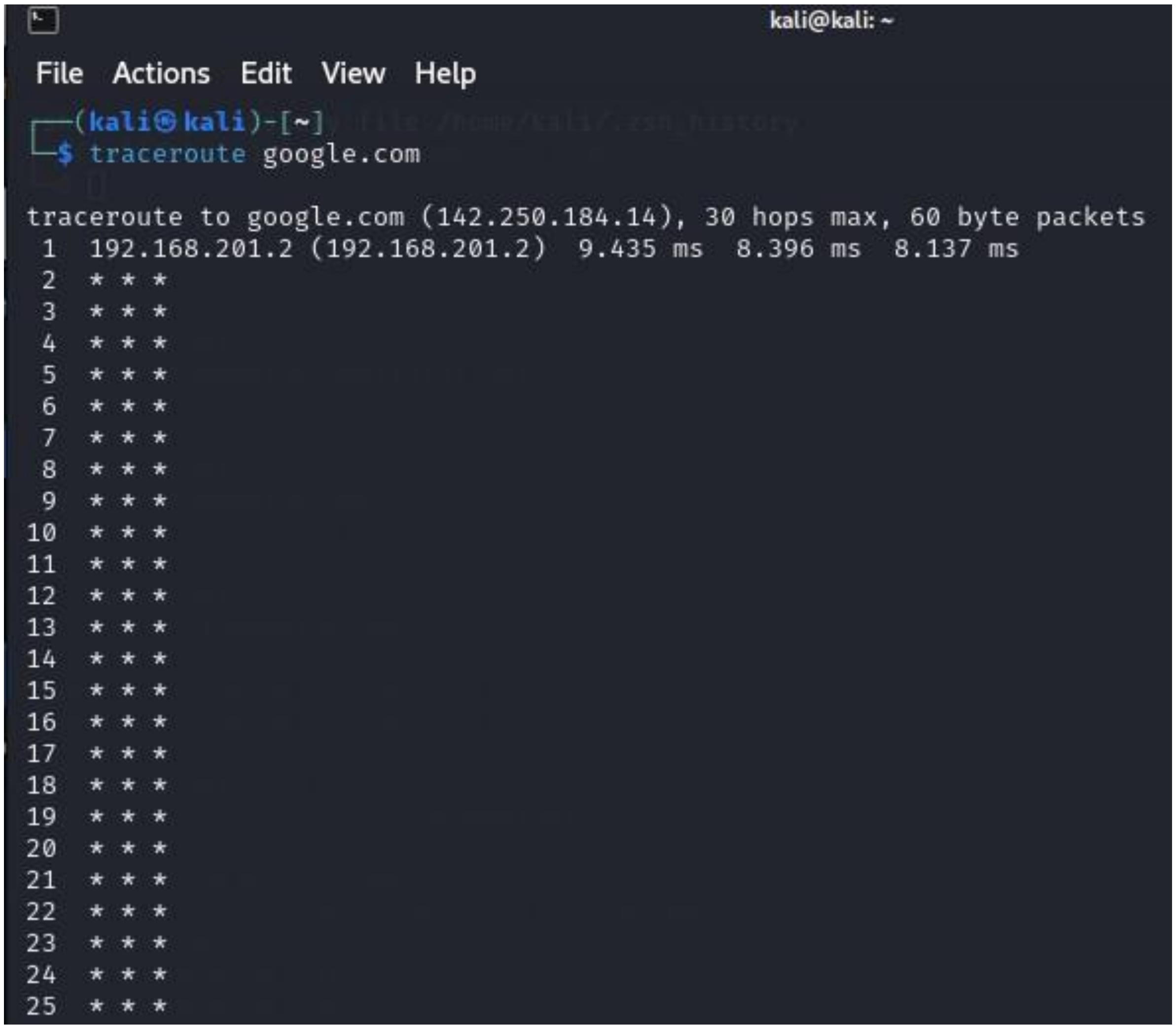
```
zsh: corrupt history file /home/kali/.zsh_history
┌──(kali㉿kali)-[~]
└─$ ifconfig

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.201.128  netmask 255.255.255.0  broadcast 192.168.201.255
        inet6 fe80::96d9:228b:a047:b619  prefixlen 64  scopeid 0×20<link>
        ether 00:0c:29:70:87:14  txqueuelen 1000  (Ethernet)
        RX packets 51344  bytes 75292183 (71.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6809  bytes 412710 (403.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 480 (480.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 480 (480.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

- Check connectivity to Google's servers.



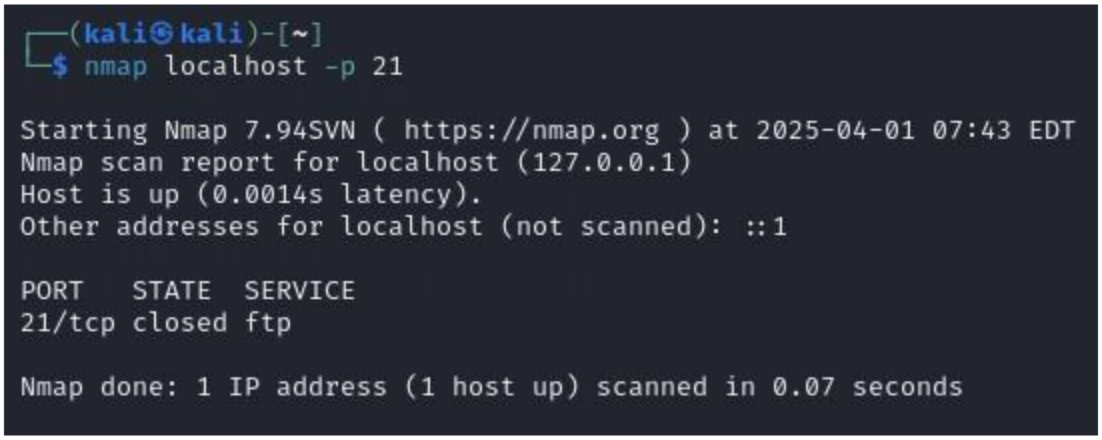
```
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=1 ttl=128 time=131 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=2 ttl=128 time=131 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=3 ttl=128 time=151 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=4 ttl=128 time=156 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=5 ttl=128 time=135 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=6 ttl=128 time=190 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=7 ttl=128 time=132 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=8 ttl=128 time=127 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=9 ttl=128 time=130 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=10 ttl=128 time=119 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=11 ttl=128 time=273 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=12 ttl=128 time=116 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=13 ttl=128 time=123 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=14 ttl=128 time=151 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=15 ttl=128 time=166 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=16 ttl=128 time=125 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=17 ttl=128 time=122 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=18 ttl=128 time=141 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=19 ttl=128 time=137 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=20 ttl=128 time=150 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=21 ttl=128 time=148 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=22 ttl=128 time=138 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=23 ttl=128 time=128 ms
64 bytes from mad41s10-in-f14.1e100.net (142.250.184.14): icmp_seq=24 ttl=128 time=125 ms
^C
─── google.com ping statistics ───
24 packets transmitted, 24 received, 0% packet loss, time 23056ms
rtt min/avg/max/mdev = 115.830/143.471/272.640/31.564 ms
```

- Trace the route to Google.



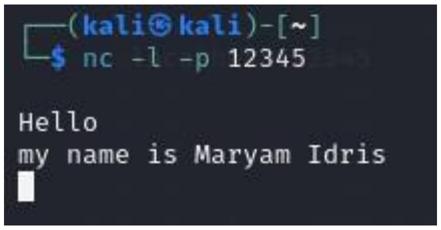
```
┌──(kali㉿kali)-[~]
└─$ traceroute google.com

traceroute to google.com (142.250.184.14), 30 hops max, 60 byte packets
 1  192.168.201.2 (192.168.201.2)  9.435 ms  8.396 ms  8.137 ms
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  * * *
 9  * * *
10  * * *
11  * * *
12  * * *
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * * *
21  * * *
22  * * *
23  * * *
24  * * *
25  * * *
```

- Use nmap to scan the localhost for port 21:

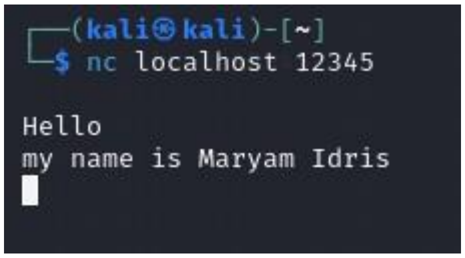


```
┌──(kali㉿kali)-[~]
└─$ nmap localhost -p 21

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-01 07:43 EDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0014s latency).
Other addresses for localhost (not scanned): ::1

PORT   STATE  SERVICE
21/tcp closed ftp

Nmap done: 1 IP address (1 host up) scanned in 0.07 seconds
```

- Use netcat to set up a simple TCP connection:

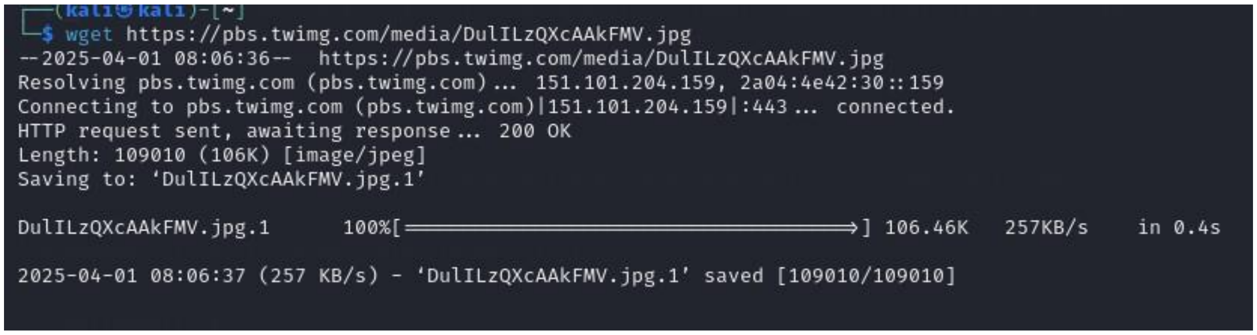


```
┌──(kali㉿kali)-[~]
└─$ nc -l -p 12345

Hello
my name is Maryam Idris
```

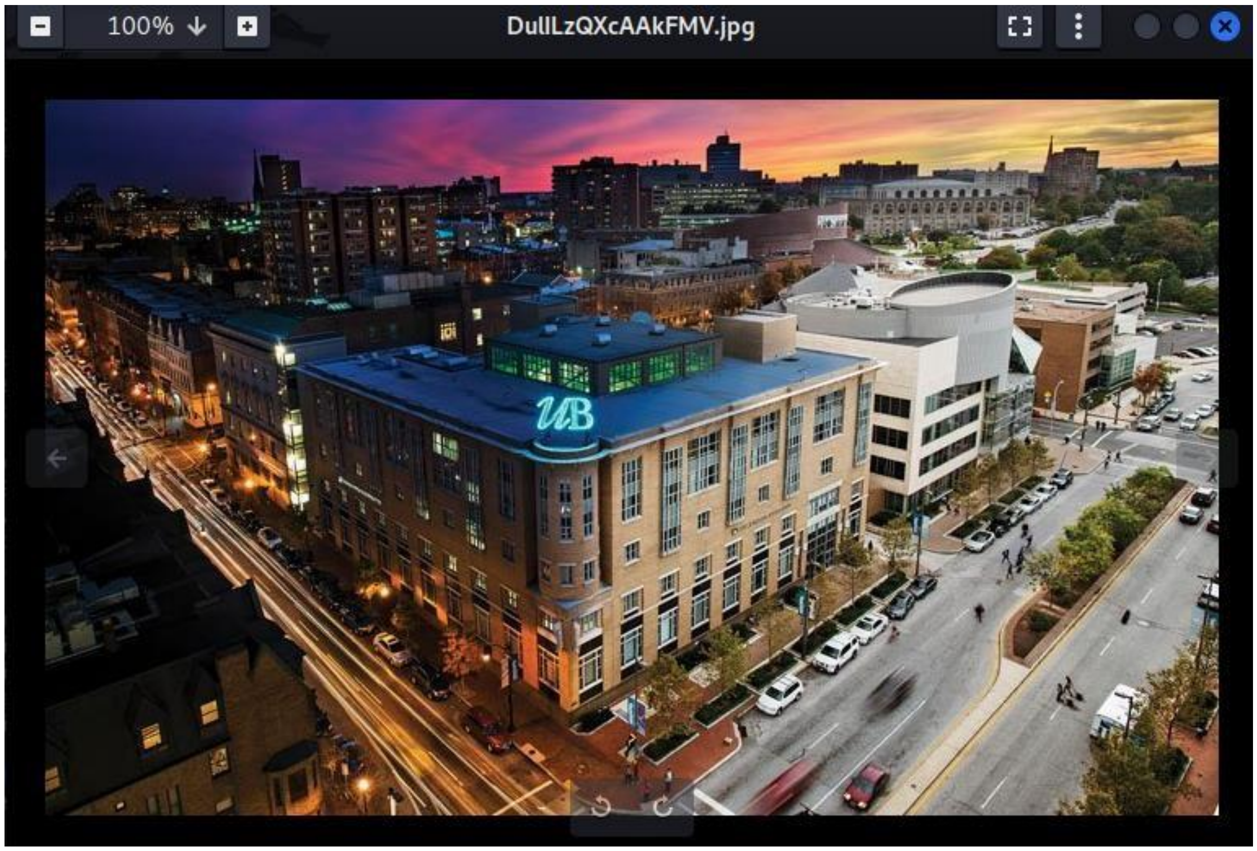


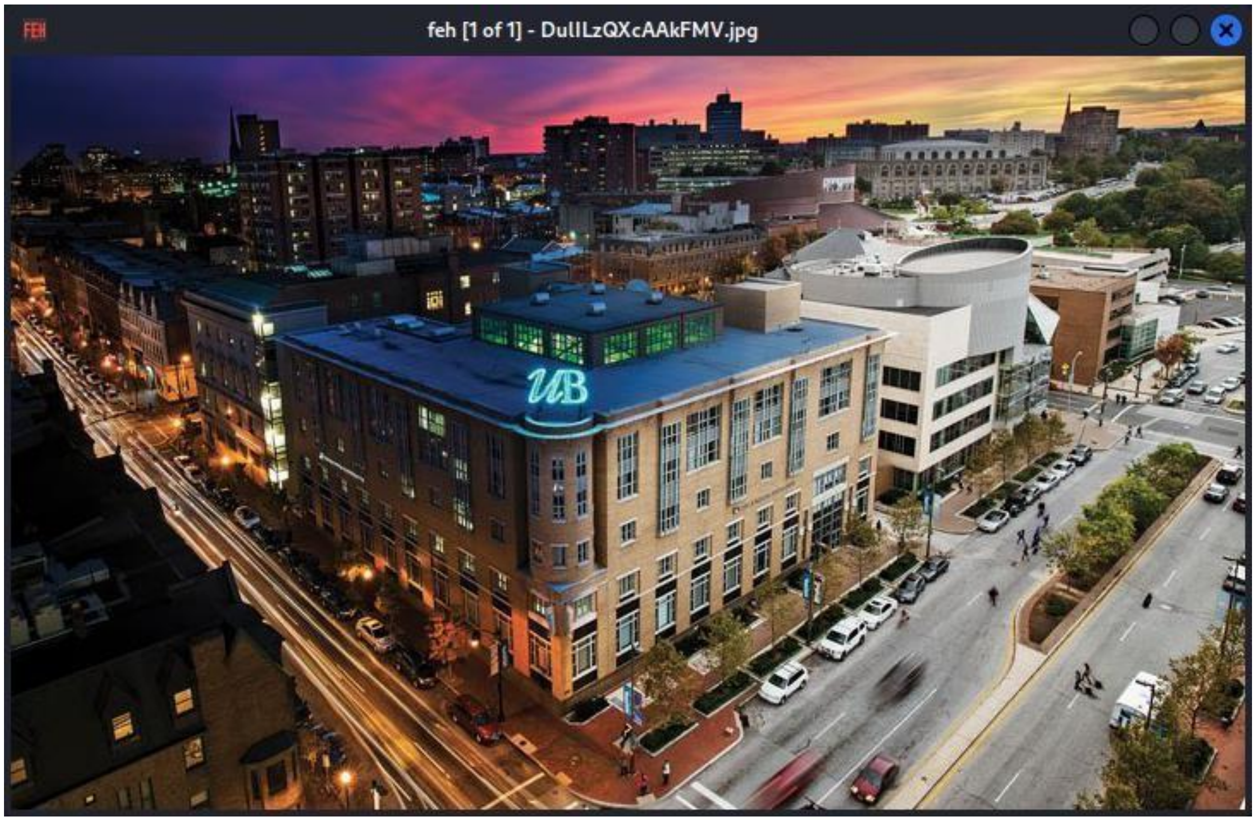
```
┌──(kali㉿kali)-[~]
└─$ nc localhost 12345

Hello
my name is Maryam Idris
```

- Use wget to download an image:



```
┌──(kali㉿kali)-[~]
└─$ wget https://pbs.twimg.com/media/DulILzQXcAAkFMV.jpg
--2025-04-01 08:06:36--  https://pbs.twimg.com/media/DulILzQXcAAkFMV.jpg
Resolving pbs.twimg.com (pbs.twimg.com)... 151.101.204.159, 2a04:4e42:30::159
Connecting to pbs.twimg.com (pbs.twimg.com)|151.101.204.159|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 109010 (106K) [image/jpeg]
Saving to: 'DulILzQXcAAkFMV.jpg.1'

DulILzQXcAAkFMV.jpg.1      100%[===================================>] 106.46K  257KB/s    in 0.4s

2025-04-01 08:06:37 (257 KB/s) - 'DulILzQXcAAkFMV.jpg.1' saved [109010/109010]
```


100%
DulILzQXcAAkFMV.jpg
UB


feh [1 of 1] - DuIILzQXcAAkFMV.jpg
UB

2. Task:
   - Document the outputs of these commands and explain the significance of the information provided.

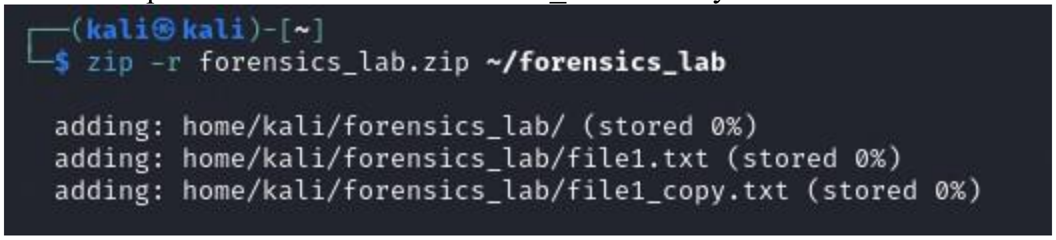
ifconfig helps to inspect the network interfaces and their configurations. ping and traceroute check connectivity and trace the route to remote destinations.
nmap scans for open ports, allowing you to see if specific services (like FTP) are running. nc (Netcat) is used to establish simple network connections.
wget helps you download files from the internet, useful for automating file retrieval.

nc -l -p 12345 Output:
(There's no immediate output when running this command as it simply listens for incoming connections.)

nc localhost 12345 Output:
(There's no immediate output when running this command either, but once connected, you can send data back and forth.) Explanation:
- The command connects to the Netcat listener running on port 12345 on the localhost (same machine).
- After connection, i typed my hello,my name is Maryam idris on terminal one and its automatically sent to listener terminal, allowing simple two-way communication.
- The connection is established over TCP, which is a reliable connection protocol used for communication between computers or services.

Part 7: File Management

1. Zipping and Unzipping Files:
   o Create a compressed archive of the forensics_lab directory:



```
┌──(kali㉿kali)-[~]
└─$ zip -r forensics_lab.zip ~/forensics_lab

  adding: home/kali/forensics_lab/ (stored 0%)
  adding: home/kali/forensics_lab/file1.txt (stored 0%)
  adding: home/kali/forensics_lab/file1_copy.txt (stored 0%)
```

Explanation:

- The zip command is used to create compressed archives in .zip format.

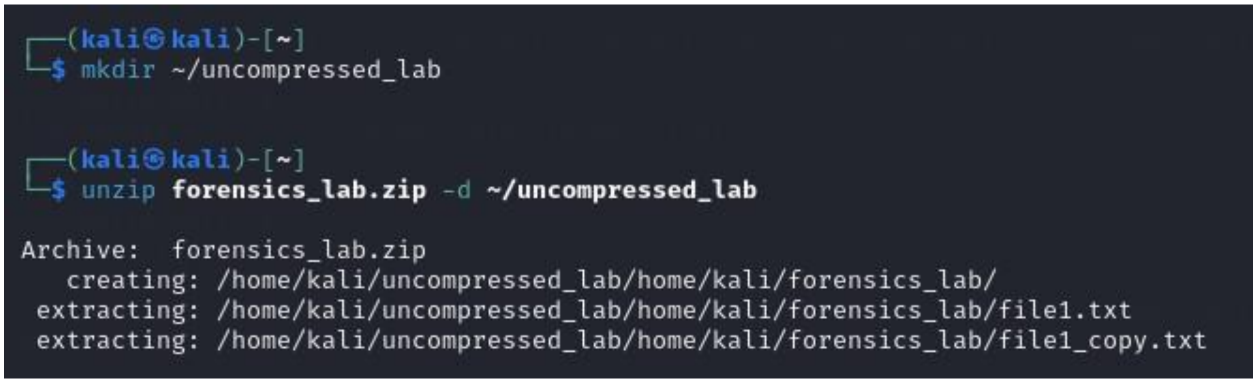
- The -r flag means "recursive," meaning it will zip the entire contents of the forensics_lab directory, including subdirectories and files.

- The command will create a zip file named forensics_lab.zip in the current working directory, containing all files and subdirectories from ~/forensics_lab.

Significance:

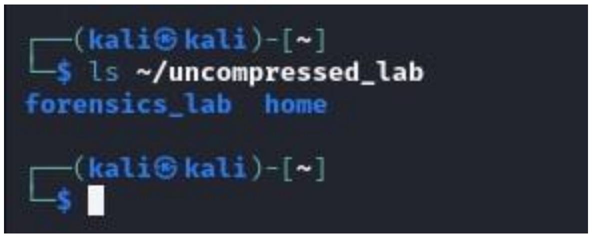
- This command creates a compressed archive that can be easily transferred or stored, reducing file size. It is a common way to package and share directories in a compressed format.

o Unzip the created archive into a new directory:



```
┌──(kali㉿kali)-[~]
└─$ mkdir ~/uncompressed_lab

┌──(kali㉿kali)-[~]
└─$ unzip forensics_lab.zip -d ~/uncompressed_lab

Archive:  forensics_lab.zip
   creating: /home/kali/uncompressed_lab/home/kali/forensics_lab/
  extracting: /home/kali/uncompressed_lab/home/kali/forensics_lab/file1.txt
  extracting: /home/kali/uncompressed_lab/home/kali/forensics_lab/file1_copy.txt
```

2. Task:
   o Verify the contents of the uncompressed directory to ensure all files were restored correctly:

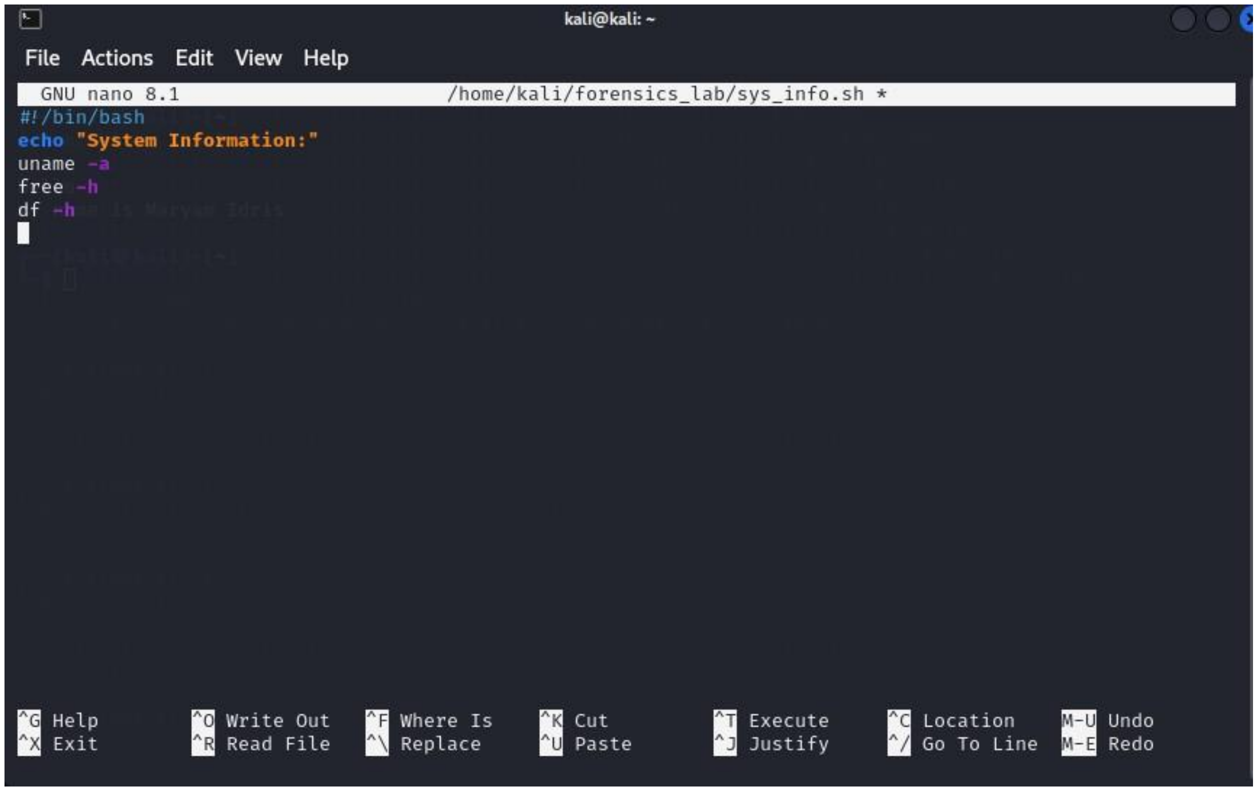


```
┌──(kali㉿kali)-[~]
└─$ ls ~/uncompressed_lab
forensics_lab  home

┌──(kali㉿kali)-[~]
└─$ 
```

Part 8: Creating a Shell Script

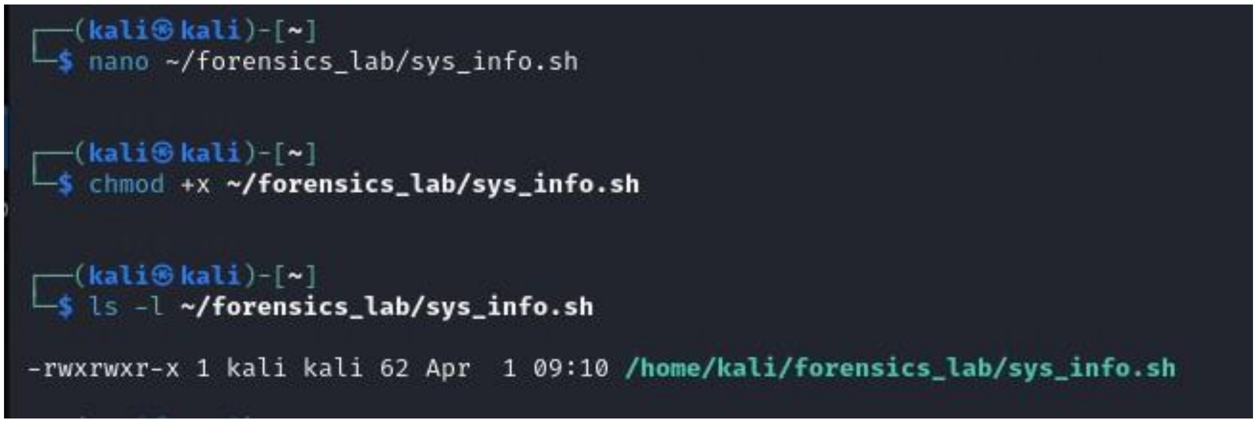
1. Creating a Shell Script:
   - Create a shell script named sys_info.sh with the following content:
2. #!/bin/bash
3. echo "System Information:"
4. uname -a
5. free -h df -h

o Make the script executable:



```
  GNU nano 8.1                     /home/kali/forensics_lab/sys_info.sh *
#!/bin/bash
echo "System Information:"
uname -a
free -h
df -h

^G Help       ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo
```

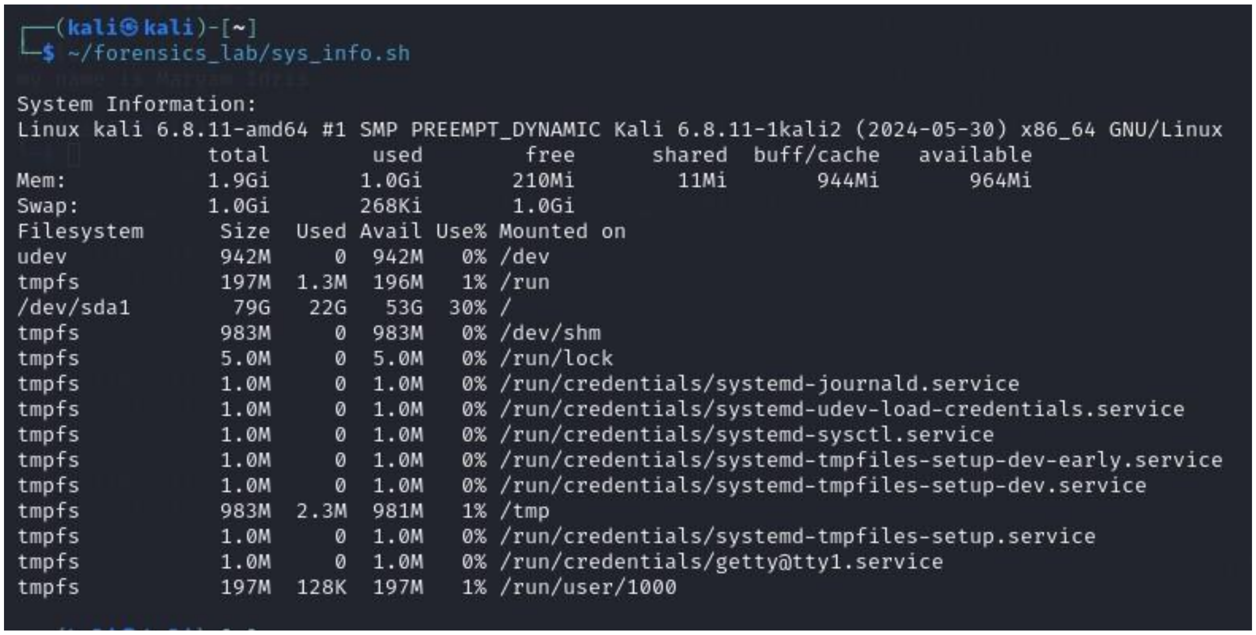
chmod +x ~/forensics_lab/sys_info.sh



```
┌──(kali㉿kali)-[~]
└─$ nano ~/forensics_lab/sys_info.sh

┌──(kali㉿kali)-[~]
└─$ chmod +x ~/forensics_lab/sys_info.sh

┌──(kali㉿kali)-[~]
└─$ ls -l ~/forensics_lab/sys_info.sh
-rwxrwxr-x 1 kali kali 62 Apr  1 09:10 /home/kali/forensics_lab/sys_info.sh
```

Task:

- Run the script from the terminal:



```
┌──(kali㉿kali)-[~]
└─$ ~/forensics_lab/sys_info.sh
my name is Maryam Idris
System Information:
Linux kali 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) x86_64 GNU/Linux
               total        used        free      shared  buff/cache   available
Mem:           1.9Gi       1.0Gi       210Mi        11Mi       944Mi       964Mi
Swap:          1.0Gi       268Ki       1.0Gi
Filesystem      Size  Used Avail Use% Mounted on
udev            942M     0  942M   0% /dev
tmpfs           197M  1.3M  196M   1% /run
/dev/sda1        79G   22G   53G  30% /
tmpfs           983M     0  983M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-journald.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-udev-load-credentials.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-sysctl.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev-early.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev.service
tmpfs           983M  2.3M  981M   1% /tmp
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-tmpfiles-setup.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/getty@tty1.service
tmpfs           197M  128K  197M   1% /run/user/1000
```

- Document the output observed from executing the script.

When executing ~/forensics_lab/sys_info.sh, the script displayed system information, memory usage, and disk usage.

- The script successfully retrieved system information, memory usage, and disk usage.
- The system has adequate free RAM and disk space, with low swap usage.
- It confirms Kali Linux is running on kernel version 6.8.11.

MEMORY USAGE:

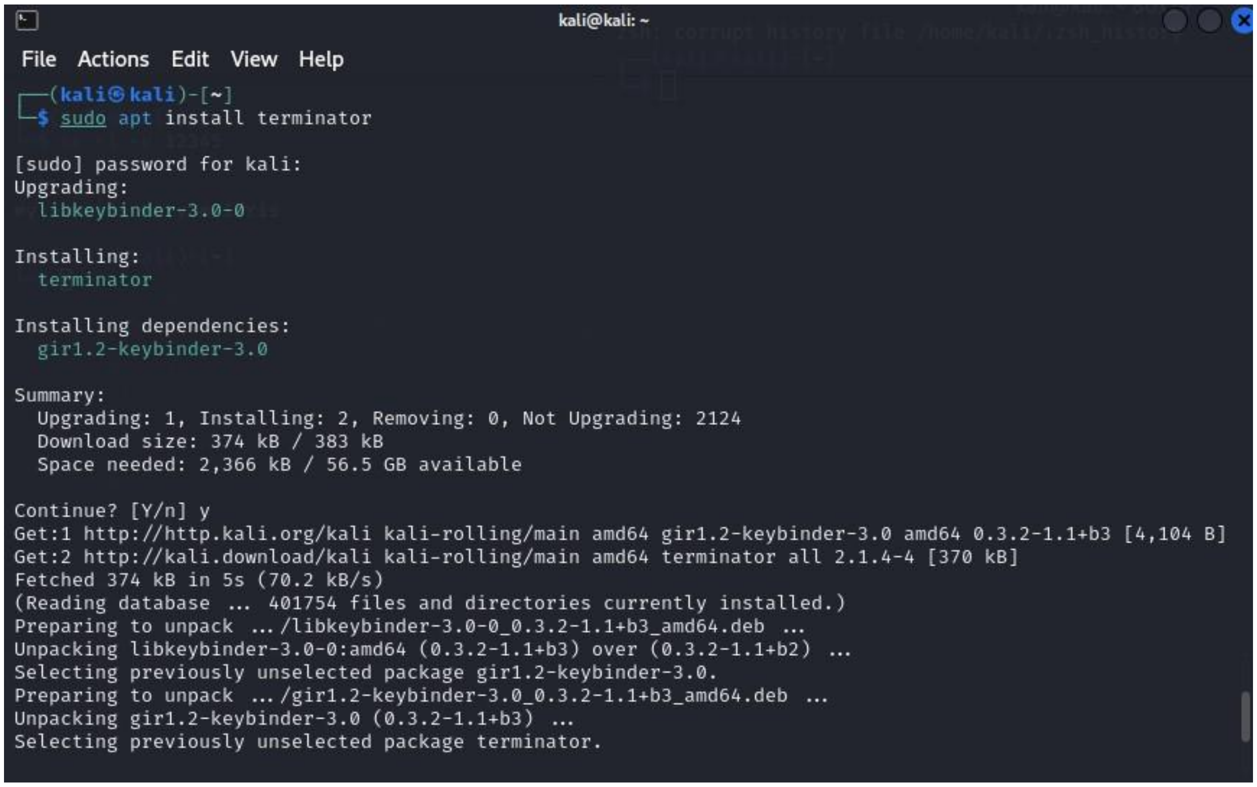
- Total Memory: 1.9Gi (around 2GB RAM installed).
- Used Memory: 1.0Gi (about 1GB of RAM in use).
- Free Memory: 210Mi (about 210MB free).
- Shared Memory: 11Mi (memory shared between processes).
- Buffer/Cache: 944Mi (memory temporarily used by the system to speed up operations).
- Available Memory: 964Mi (amount of memory still available for new processes).
- Swap Space: 1.0Gi (swap memory is 1GB, and only 268Ki is used).

Significance:

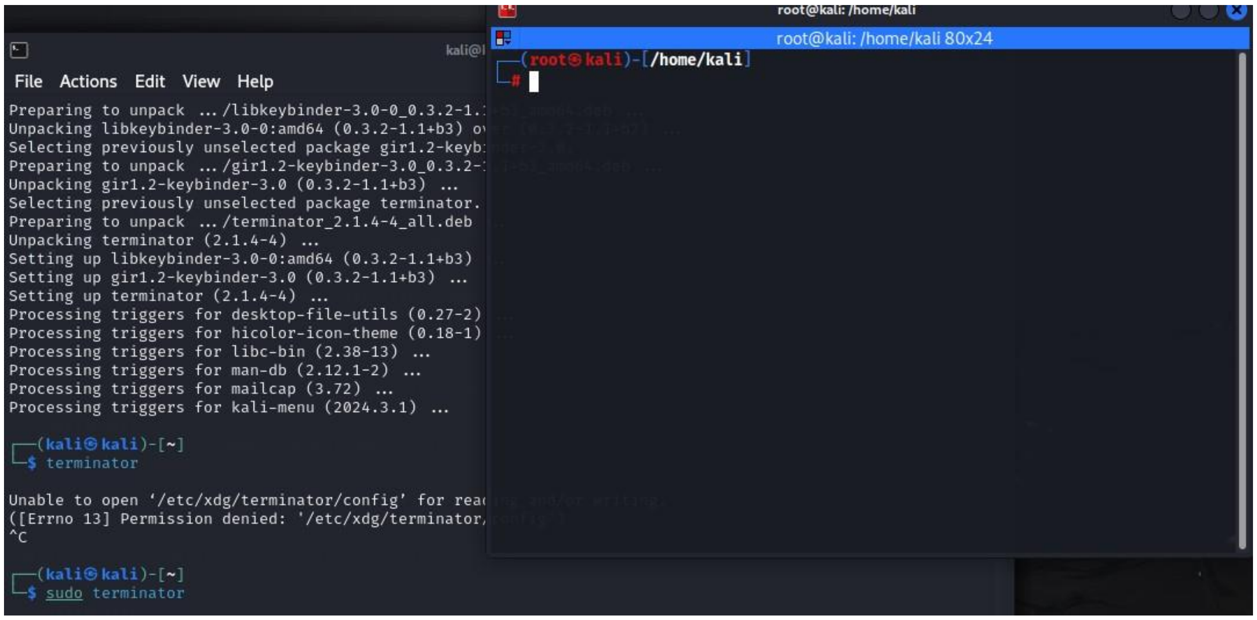
- The system is using about 1GB of RAM and has 210MB free, meaning there is still some room for applications.
- Swap memory is hardly used, which suggests the system is not running out of RAM.

Part 9: Installing Software

1. Installing Terminator:
   - Use the following command to install Terminator, a terminal emulator:

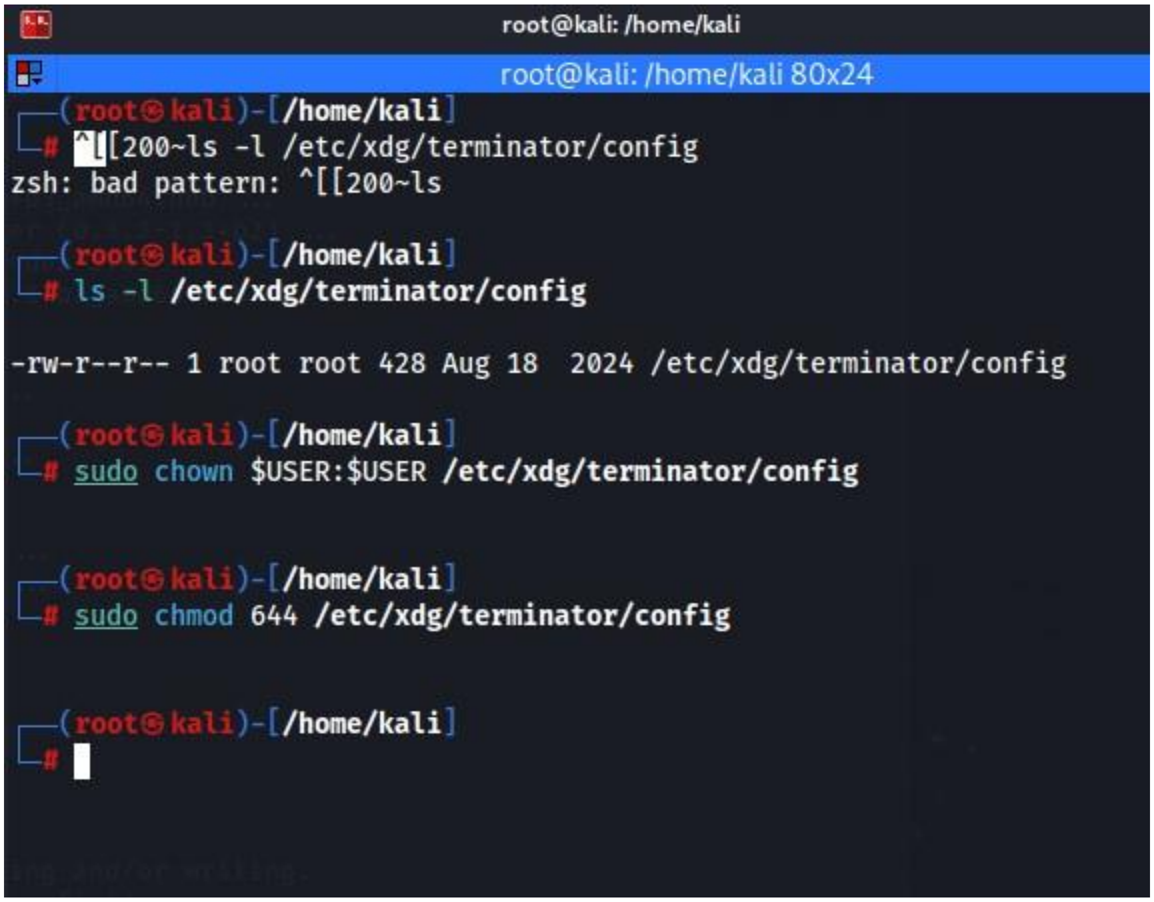


```
┌──(kali㉿kali)-[~]
└─$ sudo apt install terminator

[sudo] password for kali:
Upgrading:
  libkeybinder-3.0-0

Installing:
  terminator

Installing dependencies:
  gir1.2-keybinder-3.0

Summary:
  Upgrading: 1, Installing: 2, Removing: 0, Not Upgrading: 2124
  Download size: 374 kB / 383 kB
  Space needed: 2,366 kB / 56.5 GB available

Continue? [Y/n] y
Get:1 http://http.kali.org/kali kali-rolling/main amd64 gir1.2-keybinder-3.0 amd64 0.3.2-1.1+b3 [4,104 B]
Get:2 http://kali.download/kali kali-rolling/main amd64 terminator all 2.1.4-4 [370 kB]
Fetched 374 kB in 5s (70.2 kB/s)
(Reading database ... 401754 files and directories currently installed.)
Preparing to unpack .../libkeybinder-3.0-0_0.3.2-1.1+b3_amd64.deb ...
Unpacking libkeybinder-3.0-0:amd64 (0.3.2-1.1+b3) over (0.3.2-1.1+b2) ...
Selecting previously unselected package gir1.2-keybinder-3.0.
Preparing to unpack .../gir1.2-keybinder-3.0_0.3.2-1.1+b3_amd64.deb ...
Unpacking gir1.2-keybinder-3.0 (0.3.2-1.1+b3) ...
Selecting previously unselected package terminator.
```



```
Preparing to unpack .../libkeybinder-3.0-0_0.3.2-1.
Unpacking libkeybinder-3.0-0:amd64 (0.3.2-1.1+b3) o
Selecting previously unselected package gir1.2-keyb
Preparing to unpack .../gir1.2-keybinder-3.0_0.3.2-
Unpacking gir1.2-keybinder-3.0 (0.3.2-1.1+b3) ...
Selecting previously unselected package terminator.
Preparing to unpack .../terminator_2.1.4-4_all.deb
Unpacking terminator (2.1.4-4) ...
Setting up libkeybinder-3.0-0:amd64 (0.3.2-1.1+b3)
Setting up gir1.2-keybinder-3.0 (0.3.2-1.1+b3) ...
Setting up terminator (2.1.4-4) ...
Processing triggers for desktop-file-utils (0.27-2)
Processing triggers for hicolor-icon-theme (0.18-1)
Processing triggers for libc-bin (2.38-13) ...
Processing triggers for man-db (2.12.1-2) ...
Processing triggers for mailcap (3.72) ...
Processing triggers for kali-menu (2024.3.1) ...

┌──(kali㉿kali)-[~]
└─$ terminator

Unable to open '/etc/xdg/terminator/config' for read
([Errno 13] Permission denied: '/etc/xdg/terminator/
^C

┌──(kali㉿kali)-[~]
└─$ sudo terminator
```

```
root@kali: /home/kali 80x24
┌──(root㉿kali)-[/home/kali]
└─#
```



```
┌──(root㉿kali)-[/home/kali]
└─# ^[[200~ls -l /etc/xdg/terminator/config
zsh: bad pattern: ^[[200~ls

┌──(root㉿kali)-[/home/kali]
└─# ls -l /etc/xdg/terminator/config

-rw-r--r-- 1 root root 428 Aug 18  2024 /etc/xdg/terminator/config

┌──(root㉿kali)-[/home/kali]
└─# sudo chown $USER:$USER /etc/xdg/terminator/config


┌──(root㉿kali)-[/home/kali]
└─# sudo chmod 644 /etc/xdg/terminator/config


┌──(root㉿kali)-[/home/kali]
└─# 
```

Task:

- Document the installation steps and any outputs observed during the installation process.

Documentation:
I updated and installed Terminator. During the installation, the system retrieved the necessary packages and completed the setup successfully.
I verified the installation to ensure Terminator was properly installed. Initially, I encountered a permission error preventing it from accessing the configuration file.
To resolve the issue, I adjusted the file permissions. After making the necessary changes, I launched Terminator successfully.

INT311 - Basic Computer Skills for Digital Forensics – Lab 5

Lab Title: Remote Evidence Acquisition and Digital Forensics Techniques

Part 1: Understanding Standard Input/Output/Error Devices

1. Standard Devices:
    o Explain the concepts of standard input, standard output, and standard error devices in Linux.

ANS:

Standard Devices in Linux

Linux uses three standard devices for handling input and output operations:

1. Standard Input (stdin) - File Descriptor 0 o This is the default source of input data for programs.
    - By default, it takes input from the keyboard unless redirected from a file or another command.
2. Standard Output (stdout) - File Descriptor 1 o This is the default destination for normal command output. o By default, output is displayed on the terminal screen unless redirected to a file or another command.
3. Standard Error (stderr) - File Descriptor 2 o This is used for displaying error messages and diagnostics.
    - By default, it sends error messages to the terminal screen, ensuring errors are visible even if standard output is redirected.

These devices help in managing input and output streams efficiently, allowing redirection and piping to enhance command-line operations.

- Discuss how network utilities utilize these standard devices to communicate and process data.

ANS:

How Network Utilities Use Standard Devices in Linux

1. Standard Input (stdin) – Some utilities like netcat accept user input to send data over a network.
2. Standard Output (stdout) – Commands like ping and nmap display results on the screen or redirect them to a file.
3. Standard Error (stderr) – Errors from tools like wget are sent to stderr, allowing troubleshooting without mixing with standard output.

Redirection & Piping:

- ping google.com > output.txt saves output to a file.
- nmap localhost -p 21 2> error.log logs errors separately.

These mechanisms enhance efficiency in network diagnostics and automation.

Task:

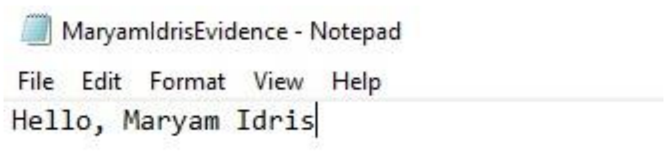
- Write a short paragraph explaining how standard devices work in the context of command-line operations and networking.

ANS:

Standard devices in Linux—stdin (standard input), stdout (standard output), and stderr (standard error)—play a crucial role in command-line operations and networking. Stdin allows commands to receive input from the keyboard or files, while stdout displays results or redirects them to files for further processing. Stderr ensures that error messages are displayed separately, preventing them from mixing with normal output. In networking, tools like ping, nmap, and wget use stdout to display results and stderr to report errors, making it easier to diagnose issues. Redirection (>, 2>) and piping (|) further enhance automation and data handling in network operations.

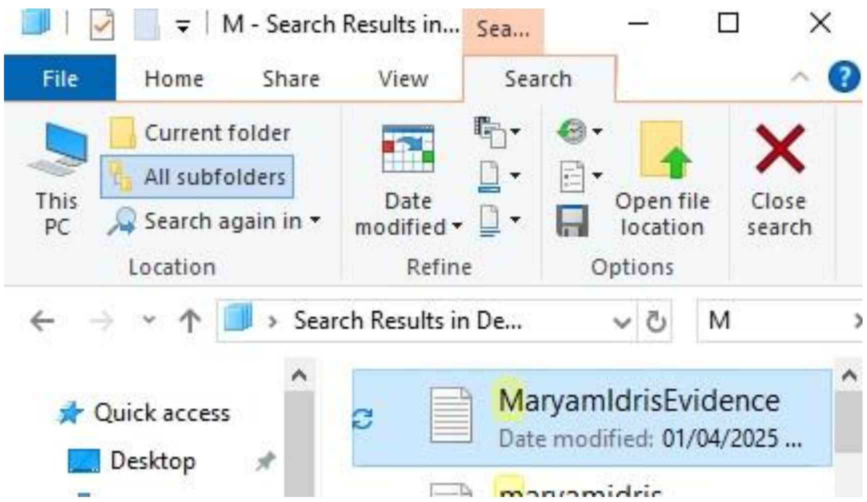
Part 2: Setting Up Windows for Remote Access

1. Creating an Evidence File in Windows:



```
Hello, Maryam Idris
```

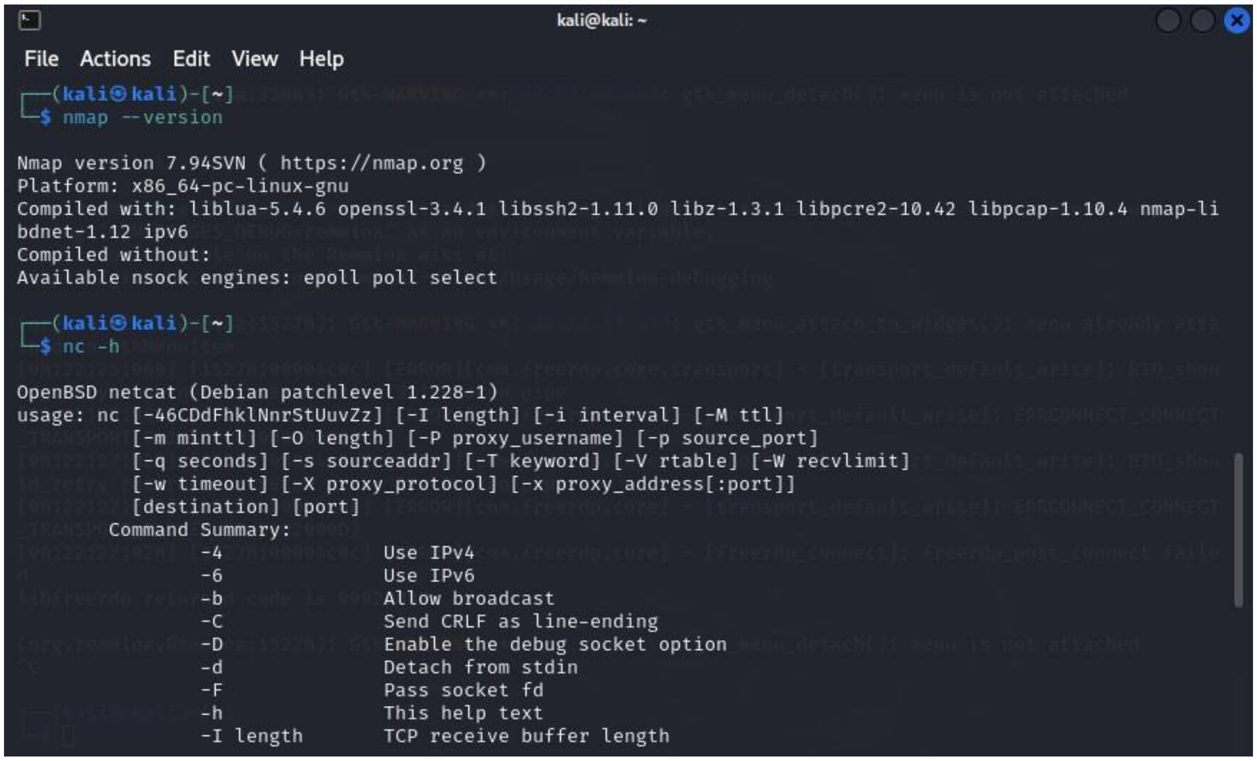
Task: o Verify the file has been created successfully with the correct content.



MaryamIdrisEvidence
Date modified: 01/04/2025 ...

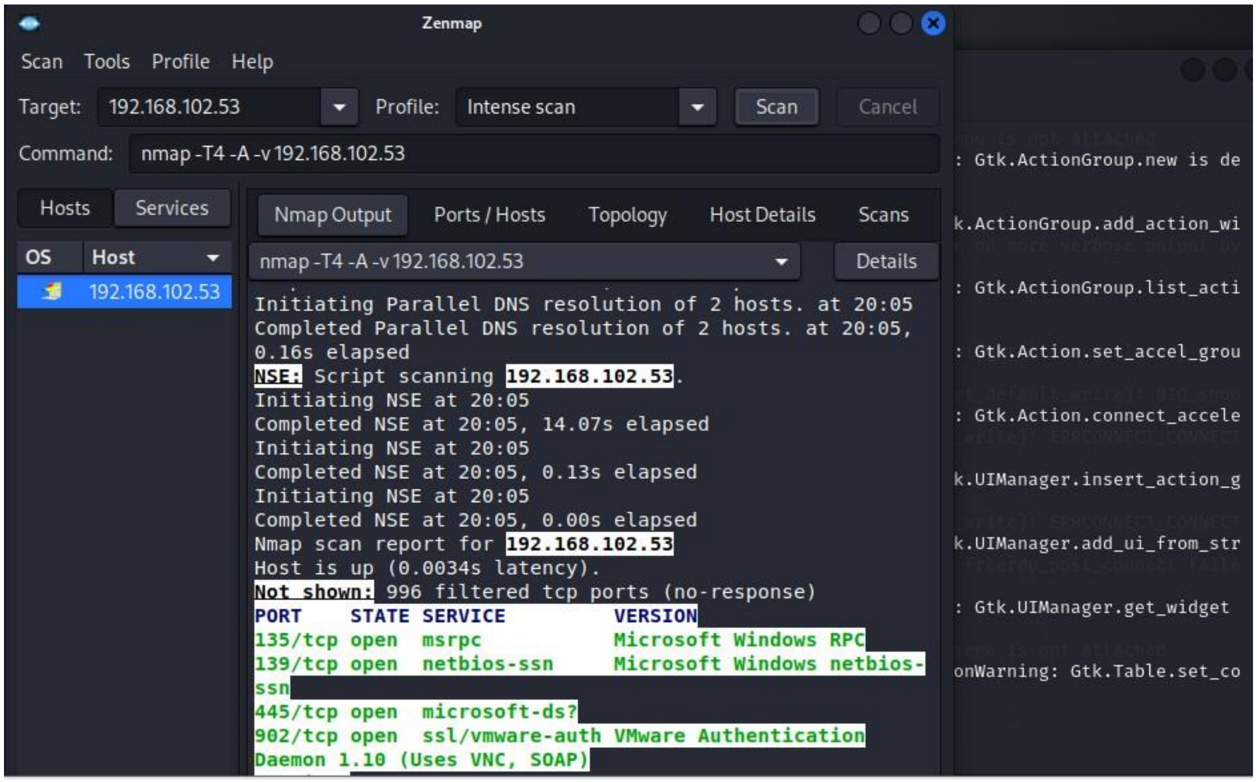
Part 3: Setting Up Kali Linux for Remote Access

1. Installing Necessary Tools:
    - Ensure that nmap and netcat are installed on your Kali Linux system. If not, install them using:



```
┌──(kali㉿kali)-[~]
└─$ nmap --version

Nmap version 7.94SVN ( https://nmap.org )
Platform: x86_64-pc-linux-gnu
Compiled with: liblua-5.4.6 openssl-3.4.1 libssh2-1.11.0 libz-1.3.1 libpcre2-10.42 libpcap-1.10.4 nmap-li
bdnet-1.12 ipv6
Compiled without:
Available nsock engines: epoll poll select

┌──(kali㉿kali)-[~]
└─$ nc -h

OpenBSD netcat (Debian patchlevel 1.228-1)
usage: nc [-46CDdFhklNnrStUuvZz] [-I length] [-i interval] [-M ttl]
          [-m minttl] [-O length] [-P proxy_username] [-p source_port]
          [-q seconds] [-s sourceaddr] [-T keyword] [-V rtable] [-W recvlimit]
          [-w timeout] [-X proxy_protocol] [-x proxy_address[:port]]
          [destination] [port]
        Command Summary:
                -4              Use IPv4
                -6              Use IPv6
                -b              Allow broadcast
                -C              Send CRLF as line-ending
                -D              Enable the debug socket option
                -d              Detach from stdin
                -F              Pass socket fd
                -h              This help text
                -I length       TCP receive buffer length
```

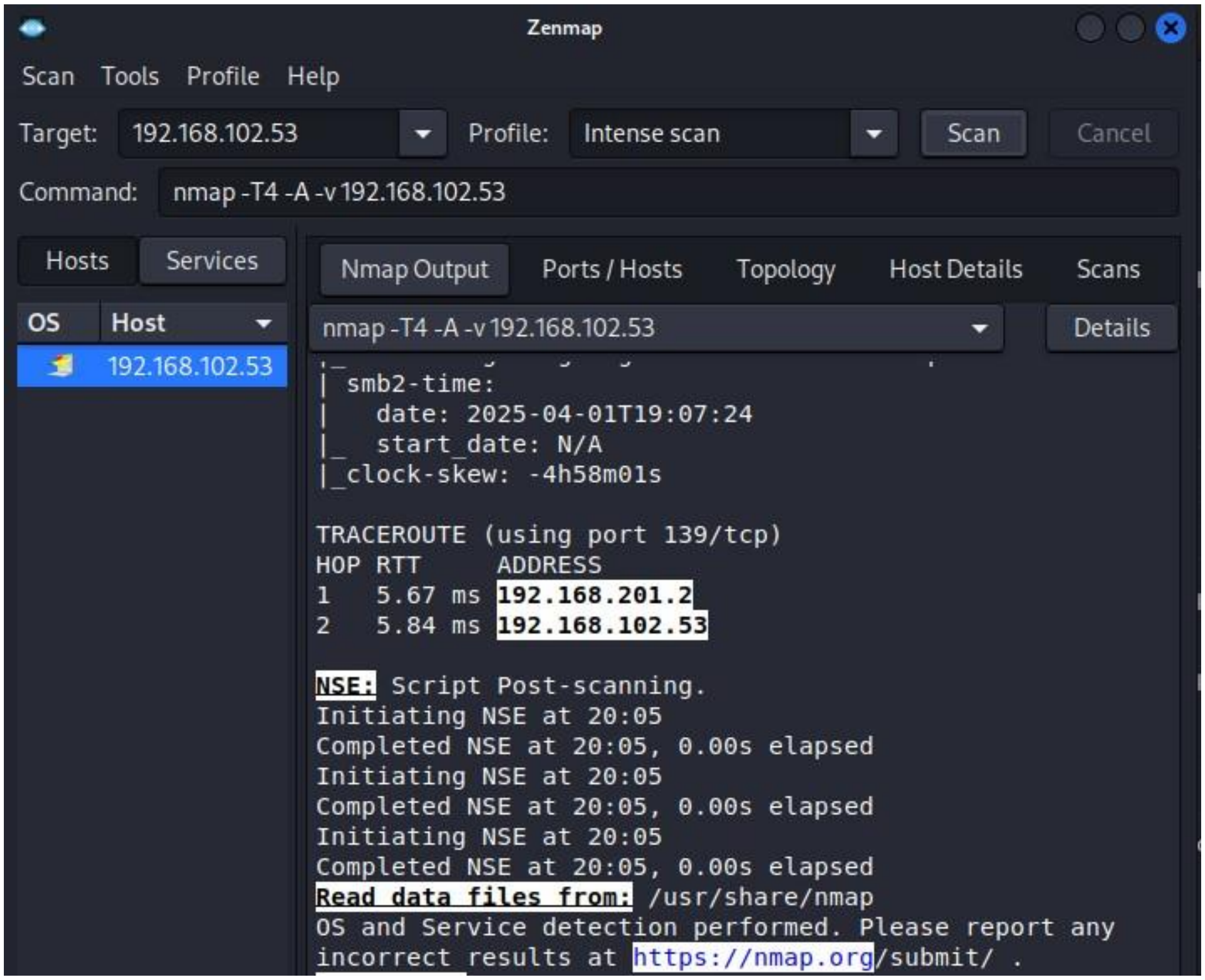
2. Using Zenmap:
   - Open Zenmap on Kali Linux to perform a scan of the Windows machine to identify open ports and services. Input the Windows machine's IP address and scan for common ports.



Command: nmap -T4 -A -v 192.168.102.53

```
nmap -T4 -A -v 192.168.102.53
Initiating Parallel DNS resolution of 2 hosts. at 20:05
Completed Parallel DNS resolution of 2 hosts. at 20:05,
0.16s elapsed
NSE: Script scanning 192.168.102.53.
Initiating NSE at 20:05
Completed NSE at 20:05, 14.07s elapsed
Initiating NSE at 20:05
Completed NSE at 20:05, 0.13s elapsed
Initiating NSE at 20:05
Completed NSE at 20:05, 0.00s elapsed
Nmap scan report for 192.168.102.53
Host is up (0.0034s latency).
Not shown: 996 filtered tcp ports (no-response)
PORT     STATE SERVICE          VERSION
135/tcp open  msrpc            Microsoft Windows RPC
139/tcp open  netbios-ssn      Microsoft Windows netbios-
ssn
445/tcp open  microsoft-ds?
902/tcp open  ssl/vmware-auth VMware Authentication
Daemon 1.10 (Uses VNC, SOAP)
```

```
: Gtk.ActionGroup.new is de
k.ActionGroup.add_action_wi
: Gtk.ActionGroup.list_acti
: Gtk.Action.set_accel_grou
: Gtk.Action.connect_accele
k.UIManager.insert_action_g
k.UIManager.add_ui_from_str
: Gtk.UIManager.get_widget
onWarning: Gtk.Table.set_co
```



```
| smb2-time:
|   date: 2025-04-01T19:07:24
|_  start_date: N/A
|_clock-skew: -4h58m01s

TRACEROUTE (using port 139/tcp)
HOP RTT     ADDRESS
1   5.67 ms 192.168.201.2
2   5.84 ms 192.168.102.53

NSE: Script Post-scanning.
Initiating NSE at 20:05
Completed NSE at 20:05, 0.00s elapsed
Initiating NSE at 20:05
Completed NSE at 20:05, 0.00s elapsed
Initiating NSE at 20:05
Completed NSE at 20:05, 0.00s elapsed
Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any
incorrect results at https://nmap.org/submit/ .
```

Task:

- Document the scan results, focusing on any open ports related to file sharing or remote access (e.g., port 139 for SMB, port 445 for SMB over TCP).

Scan Results Documentation

I performed the scan on my Windows machine with IP address 192.168.102.53 to identify open ports and services. Below are the results:

Scan Details:

- Scan Type: SYN Stealth Scan (1000 ports)
- Target IP: 192.168.102.53

Open Ports Identified:

- Port 139/tcp: The SMB (Server Message Block) service is running.

- Port 135/tcp: The Microsoft RPC (Remote Procedure Call) service is running.

Traceroute Results:

A traceroute was performed using port 139/tcp. The following is the path and round-trip time (RTT) to the target machine:

- Hop 1: IP Address: 192.168.201.2, RTT: 5.67 ms
- Hop 2: Target IP Address: 192.168.102.53, RTT: 5.84 ms

Nmap Scan Summary:

- The scan started with a Ping Scan to identify active hosts. The target machine at 192.168.102.53 responded, confirming it is online.
- Following the Ping Scan, a SYN Stealth Scan was conducted on 1000 common ports. Two open ports were discovered: Port 135 (RPC) and Port 139 (SMB).
- The Traceroute scan confirmed that there is a two-hop network path from my Kali machine to the target, with a minimal RTT of 5.67 ms on the first hop and 5.84 ms on the second hop.

Conclusion:

- Ports 135 and 139 are open on the target machine, which are typically used by Windows services (RPC and SMB).
- The traceroute confirms that the target is reachable within two hops, and the RTT is relatively low.

This scan provides essential information regarding the active services and network path to the target machine. Let me know if more details are needed!

Digital Forensics and Remote Evidence Acquisition

1. Discussing Remote Evidence Acquisition:
    - o Define what remote evidence acquisition means in the context of digital forensics. Highlight the importance of maintaining evidence integrity and legal implications.

Remote Evidence Acquisition in Digital Forensics

Remote evidence acquisition refers to the process of collecting digital evidence from a remote system or device, without physically accessing the device itself. This method is often used when the target system is located in a different geographical location or when physical access is not feasible due to time or security constraints. In digital forensics, remote evidence acquisition is essential for several reasons:

- Timeliness: It allows investigators to collect evidence from a live system quickly, reducing the risk of evidence being altered, lost, or overwritten.
- Non-intrusiveness: Remote acquisition can be performed without disturbing the target system, minimizing the risk of destroying or compromising crucial evidence.
- Preserving Chain of Custody: By remotely acquiring evidence, investigators can ensure that the chain of custody is maintained, which is critical for the legal admissibility of evidence in court.

Importance of Maintaining Evidence Integrity

The integrity of digital evidence is paramount in forensics. Any alterations, whether intentional or accidental, can invalidate the evidence, making it inadmissible in court.

To preserve the integrity of evidence during remote acquisition:

- Write Protection: Evidence should be collected in a way that prevents any changes to the original data. For example, forensic tools can create bitby-bit copies (known as forensic images) to ensure that the original system remains untouched.
- Hashing: Hashing algorithms (e.g., SHA-256) are used to generate a unique value for the original evidence. The hash value is compared before and after the acquisition to verify that the evidence has not been altered during the process.
- Logs: Detailed logs should be kept of all actions taken during the remote acquisition process to prove that no tampering occurred and to maintain a clear chain of custody.

Legal Implications

The remote acquisition of evidence has significant legal implications:

- Authorization: Investigators must obtain proper legal authorization, such as a warrant, before acquiring evidence remotely. Unauthorized access can lead to legal issues and the potential exclusion of evidence from court.
- Jurisdiction: Remote evidence acquisition may involve accessing systems in different jurisdictions, raising concerns about the laws governing digital evidence and the admissibility of evidence across borders.
- Privacy Considerations: Digital forensics professionals must be aware of privacy laws and regulations, ensuring that sensitive or private data is not accessed or disclosed unnecessarily during the acquisition process.

Remote evidence acquisition is a critical practice in digital forensics, offering the ability to collect evidence from remote systems while preserving its integrity and adhering to legal standards. Proper techniques and documentation are essential to ensure that evidence remains admissible in court.

Task:

o Write a brief explanation (150-200 words) discussing the challenges and best practices of remote evidence acquisition.

Remote evidence acquisition presents several challenges and requires strict adherence to best practices to ensure the integrity and legality of the process.

One major challenge is ensuring evidence integrity during acquisition. Since evidence is collected over a network, there is a risk of data corruption or tampering during transmission. To mitigate this, investigators use forensic imaging tools that create bit-by-bit copies of data, preserving its original state. Additionally, hashing techniques are employed to verify that the data hasn't changed during the process.

Another challenge is the legal and ethical implications of accessing remote systems. Remote evidence acquisition must be performed within the confines of the law, which often requires obtaining proper authorization through a warrant or permission. Investigators must also be mindful of jurisdictional issues, particularly when accessing data across borders.

To address these challenges, best practices include maintaining a chain of custody, documenting all actions during the acquisition process, and using writeblocking tools to prevent changes to the original system. Additionally, investigators should use secure communication channels to ensure the confidentiality of the data being transferred.

By following these best practices, remote evidence acquisition can be conducted effectively, ensuring the reliability and admissibility of digital evidence in legal proceedings.