EXPT NO: 05 ROLL NO: 220701216

# LINUX FILE SYSTEM ANALYSIS

---

AIM:

To explore and apply live forensic file system analysis techniques on a compromised Linux environment. This includes investigating users, system logs, binaries, permissions, and digital artefacts to reconstruct the attack timeline and identify evidence of compromise.

PROCEDURE:

1. Isolate the compromised system and load clean binaries via USB for trusted analysis.

2. Modify the `PATH` and `LD_LIBRARY_PATH` to ensure only clean binaries are used.

3. Investigate suspicious uploads and artifacts under `/var/www/html/`.

4. Extract metadata, timestamps, and file integrity using tools like `stat`, `exiftool`, and checksum utilities.

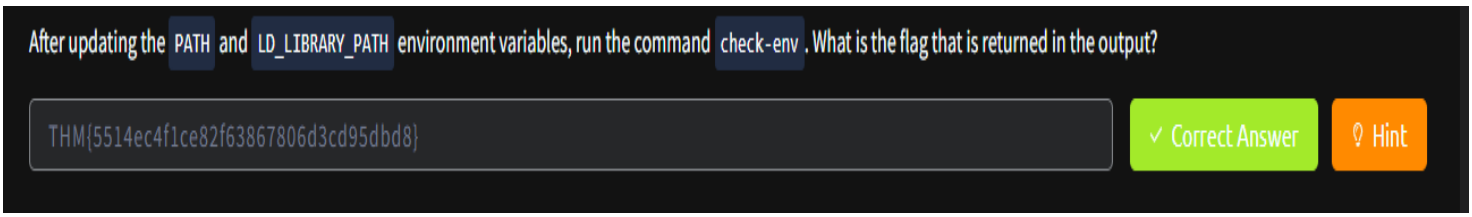
5. Identify and investigate unusual user accounts, group IDs, and `sudoers` entries.

6. Review user history and SSH configurations for backdoors.

7. Examine SUID binaries, unverified executables, and detect rootkits.

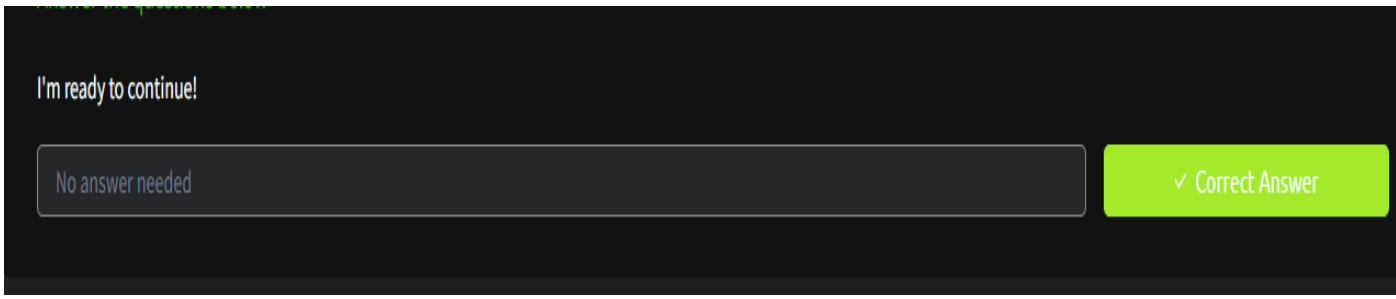
TASK 1 – INTRODUCTION

- Introduced the importance of live file system forensic analysis in Linux environments.
- Emphasized the goal of identifying digital artefacts and compromise indicators.
- Clarified that remediation should not be done on live systems during initial analysis.
- Highlighted the focus on detecting unauthorized access, data tampering, and rootkits.
- Stressed the relevance of understanding logs, users, file structures, and permissions.
- Recommended restoring from backups after analysis, not reusing compromised systems.




**TASK 2 – INVESTIGATION SETUP**

- Mounted a USB containing clean Debian-based binaries and libraries on the compromised system.
- Copied /bin, /sbin, /lib, and /lib64 folders to /mnt/usb for a trusted toolset.
- Updated PATH and LD_LIBRARY_PATH to prioritize clean binaries for forensic commands.
- Ensured the environment uses only verified binaries to avoid tampered results.
- Verified clean environment setup using the check-env script.
- Provided a secure and controlled setup for conducting further forensic analysis.



TASK 3 – FILES, PERMISSIONS & TIMESTAMPS

- Detected uploaded web shell `b2c8e1f5.phtml` via upload vulnerability.

- Found and analyzed reverse shell binary `reverse.elf`.

- Retrieved its metadata (MIME type), timestamps (`stat`), and computed hashes (MD5 & SHA256).

- Verified indicators via VirusTotal for malware classification.

- Practiced `find` command to identify files created by user `bob`.


TASK 4 – USERS AND GROUPS

- Used `/etc/passwd`, `getent`, and `cat /etc/group` to identify suspicious users.

- Discovered backdoor UID 0 user.

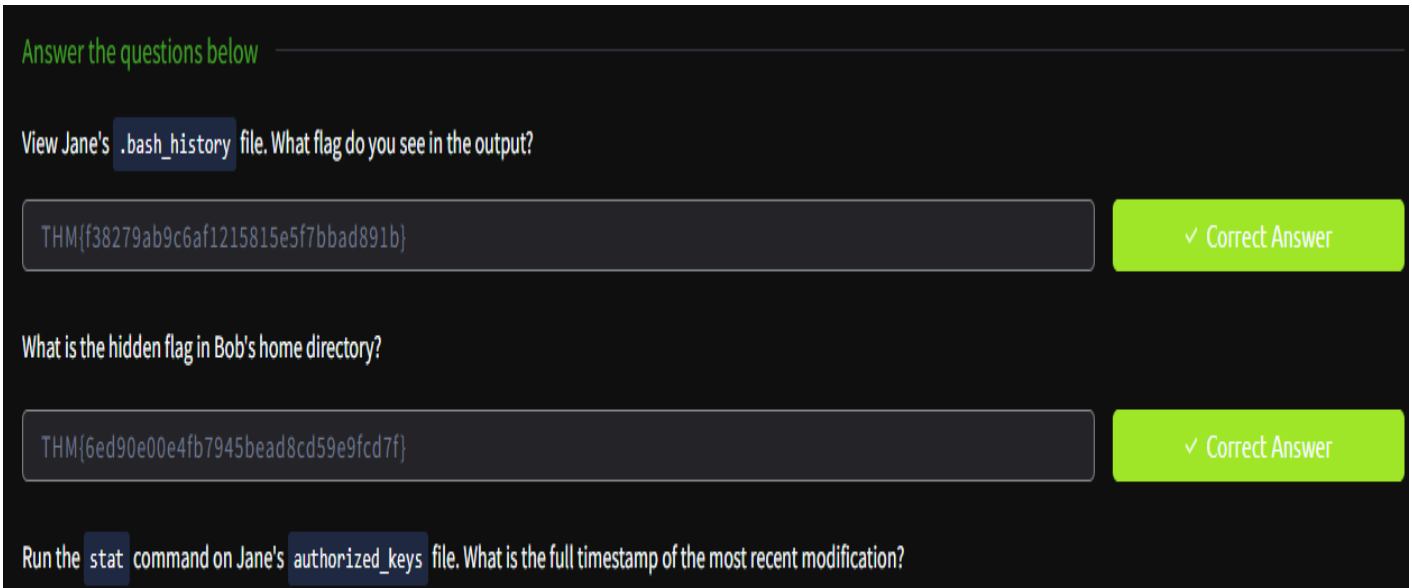
- Identified group with GID 46.

- Inspected `/etc/sudoers` file to find binaries accessible to Jane.

- Observed that Jane could use `/sbin/ifconfig` with `sudo`.


TASK 5 – USER DIRECTORIES & SSH ACCESS

- Explored hidden files in home directories such as `.bash_history` and `.ssh/authorized_keys`.
- Found a backdoor SSH key in Jane's authorized_keys.
- Discovered flag in Jane's bash history.
- Located a hidden flag in Bob's home directory.
- Extracted modification timestamp for Jane's `.ssh/authorized_keys` using `stat`.



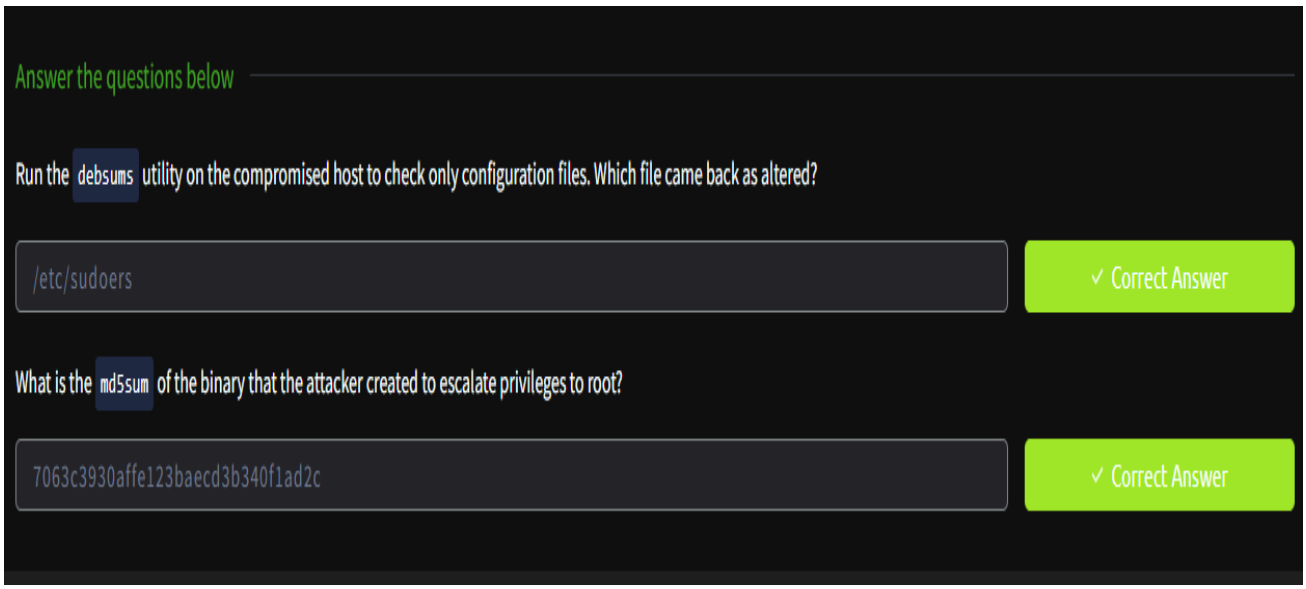
TASK 6 – BINARIES & EXECUTABLES

- Used `find` and `debsums` to identify unauthorized root-owned binaries and config file modifications.

- Used `md5sum` and `strings` for integrity and behavior analysis.

- Identified altered system config files.

- Found attacker-created binary in `/var/tmp/bash` with suspicious MD5 hash.

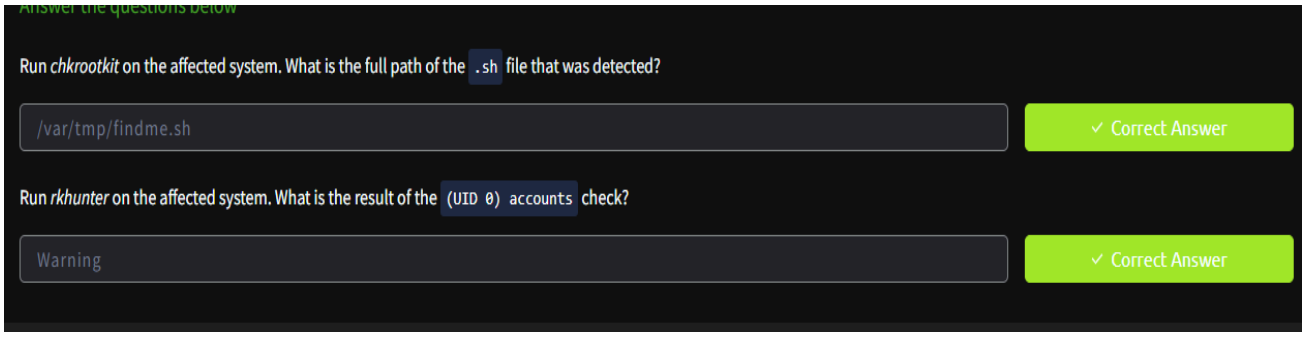



TASK 7 – ROOTKIT DETECTION

- Ran `chkrootkit` and detected a suspicious `.sh` script.

- Used `rkhunter` to scan for deeper system integrity checks.

- Confirmed UID 0 account anomaly through rkhunter summary.



RESULT:

Successfully identified indicators of compromise, backdoor accounts, and manipulated binaries. Demonstrated capability to use live forensics methodology in incident response and Linux system compromise investigations.