

## LINUX PRIVELEGE ESCALATION

---

### AIM:

This experiment aims to explore Linux privilege escalation techniques including kernel exploits, misconfigured sudo privileges, and SUID binary exploitation. It provides hands-on experience in identifying and exploiting real-world vulnerabilities to escalate privileges on Linux systems for better understanding of secure configurations and attacker methodologies.

### PROCEDURE:

1. Enumerate Linux kernel version and search for CVEs.
2. Transfer and compile kernel exploits on target machines.
3. Use `sudo -l` to check misconfigured binaries and escalate via GTF0Bins.
4. Exploit vulnerable SUID binaries to read sensitive files or spawn root shells.
5. Dump password hashes using unshadow and crack using John the Ripper.
6. Bypass protections using LD\_PRELOAD or service misconfiguration

## TASK 1 – INTRODUCTION

- Introduced the fundamentals of privilege escalation in Linux systems.
- Highlighted the significance of escalating from a low-privilege user to root.
- Emphasized enumeration as a critical first step using commands like `uname -a`, `id`, and `sudo -l`.
- Outlined common escalation vectors such as kernel exploits, misconfigured binaries, and SUID bits.
- Reinforced the goal of identifying misconfigurations that could allow unauthorized access or privilege abuse.

Answer the questions below

Read the above.

No answer needed

✓ Correct Answer

## TASK 2 – ENUMERATION

- Used system commands (`hostname`, `uname -a`, `id`, `env`, `ps`, `ls -la`, `ifconfig`) to gather system-level information.
- Listed current user's permissions, environment variables, and active processes.
- Inspected the `/etc/passwd` file and running services for hints on misconfigurations.
- Checked for installed software and outdated packages that could be exploited.

- Identified enumeration as an essential precondition before attempting privilege escalation.

Answer the questions below

Read the above.

No answer needed

✓ Correct Answer

### TASK 3 – KERNEL EXPLOITS

- Identified kernel version using `uname -a` and determined it was vulnerable.
- Researched public CVEs and found a suitable kernel exploit (CVE-2015-1328).
- Downloaded the exploit source code from Exploit-DB (37292.c).
- Transferred the exploit to the target machine using `wget` via a Python HTTP server.
- Compiled the exploit with `gcc` and executed it to escalate privileges.
- Successfully gained root access and retrieved the first flag (flag1.txt).

- Demonstrated how kernel vulnerabilities can directly lead to root access in CTF/lab scenarios.

Answer the questions below

What is the hostname of the target system?

wade7363 ✓ Correct Answer

What is the Linux kernel version of the target system?

3.13.0-24-generic ✓ Correct Answer

What Linux is this?

Ubuntu 14.04 LTS ✓ Correct Answer

What version of the Python language is installed on the system?

2.7.6 ✓ Correct Answer

What vulnerability seem to affect the kernel of the target system? (Enter a CVE number)

CVE-2015-1328 ✓ Correct Answer

#### TASK 4 – SUDO MISCONFIGURATION

- Used `sudo -l` to enumerate commands the user karen could run as root.
- Identified `find` and `nmap` as exploitable binaries using GTF0Bins.
- Spawned a root shell using `sudo find . -exec /bin/sh \; -quit`.
- Gained root privileges again via interactive mode in `nmap`: `!sh`.
- Accessed and read `/etc/shadow` file and extracted password hashes.
- Verified access by opening the `flag2.txt` located in a privileged user's home directory.
- Showed how improper `sudo` access to certain binaries can lead to full privilege escalation.

#### TASK 5 – KERNEL EXPLOITS

Answer the questions below

find and use the appropriate kernel exploit to gain root privileges on the target system.

No answer needed

✓ Correct Answer

🔍 Hint

What is the content of the flag1.txt file?

THM-28392872729920

✓ Correct Answer

- Identified kernel version using ``uname -a`` and searched CVE.
- Found exploit CVE-2015-1328 and downloaded code from Exploit-DB.
- Transferred code to ``/tmp`` via Python SimpleHTTPServer and ``wget``.
- Compiled using ``gcc 37292.c -o kernelexploit`` and executed.
- Successfully escalated to root and accessed ``flag1.txt``.
- Emphasized stability risks with kernel exploits in real-world environments.

Answer the questions below

Install and try a few automated enumeration tools on your local Linux distribution

No answer needed

✓ Correct Answer

## TASK 6 – SUDO MISCONFIGURATION

- Used ``sudo -l`` to list programs executable by user ``karen``.
- Used GTF0Bins to find escalation path via ``find`` and ``nmap``.
- Spawned root shell with: ``sudo find . -exec /bin/sh \; -quit``
- Used ``nmap`` in interactive mode to get shell: ``nmap --interactive``.
- Accessed ``/etc/shadow`` and extracted hash for user frank.
- Confirmed root access by reading ``flag2.txt`` in privileged user's directory.

Answer the questions below

Which user shares the name of a great comic book writer?

gerryconway

✓ Correct Answer

What is the password of user2?

Password1

✓ Correct Answer

What is the content of the flag3.txt file?

THM-3847834

✓ Correct Answer

## TASK 7 – SUID BINARY ABUSE

- Listed all SUID binaries: `find / -type f -perm -04000 -ls 2>/dev/null`
- Identified base64 with SUID bit and used GTF0Bins for file read.
- Read `/etc/shadow` via SUID-enabled base64 decoding method.
- Combined `/etc/passwd` and `/etc/shadow` using `unshadow`.

Answer the questions below

How many programs can the user "karen" run on the target system with sudo rights?

3

✓ Correct Answer

What is the content of the flag2.txt file?

THM-402028394

✓ Correct Answer

How would you use Nmap to spawn a root shell if your user had sudo rights on nmap?

sudo nmap --interactive

✓ Correct Answer

What is the hash of frank's password?

5652.sUUDsOLipXKxcr5eImtgFEyr2Is4jsghdD3DHLHHP9X50Iv.jNmwo/BJpphrPRJWjeIWEz2HH.joV14aDEwW1c3CahzB1uaqI

✓ Correct Answer

- Cracked hashes using John the Ripper and rockyou.txt wordlist.
- Accessed `flag3.txt` using base64 decode method from GTF0Bins.

## TASK 8 – SUDOERS MISCONFIGURATION

- Analyzed `/etc/sudoers` file and identified misconfigured entries.
- Found user allowed to run specific commands as root without a password.
- Exploited command such as `/usr/bin/less` or `/bin/bash` via sudo.

- Gained a root shell and validated access by checking `/root/`` directory.
- Reinforced that sudoers should be tightly restricted and regularly audited.

Answer the questions below

Complete the task described above on the target system

No answer needed

✓ Correct Answer

How many binaries have set capabilities?

6

✓ Correct Answer

What other binary can be used through its capabilities?

view

✓ Correct Answer

What is the content of the flag4.txt file?

THM-9349843

✓ Correct Answer

## TASK 9 – ENVIRONMENTAL VARIABLE ABUSE

- Identified vulnerable scripts or binaries relying on insecure `$PATH`.
- Created a fake executable with same name as trusted binary and placed it earlier in `$PATH`.
- Triggered privilege escalation when root ran the vulnerable script.
- Showed how `PATH` hijacking can escalate privileges due to lazy script writing.
- Stressed best practices for securing scripts with full path references.

Answer the questions below

How many user-defined cron jobs can you see on the target system?

4

✓ Correct Answer

What is the content of the flag5.txt file?

THM-383000283

✓ Correct Answer

What is Matt's password?

123456

✓ Correct Answer

## TASK 10 – BACKUP MISCONFIGURATION

- Discovered world-writable backup directories with sensitive data.
- Analyzed and modified backups to insert payloads.
- Restored the manipulated backup or exploited symlink attacks to gain code execution.
- Confirmed elevated access and retrieved associated flags.
- Revealed real-world missteps in backup script design and access control.
- Stressed best practices for securing scripts with full path references.

Answer the questions below

What is the odd folder you have write access for?

✓ Correct Answer [Hint](#)

Exploit the \$PATH vulnerability to read the content of the flag6.txt file.

✓ Correct Answer [Hint](#)

What is the content of the flag6.txt file?

✓ Correct Answer

## TASK 11 – SERVICE MISCONFIGURATION

- Found custom services with writeable binary paths or configs.
- Modified service binary or configuration file to execute reverse shell.
- Restarted service and gained root access.
- Validated exploitation via service misconfiguration and confirmed with a root flag.
- Outlined how improper service permissions are common real-world attack

Answer the questions below

How many mountable shares can you identify on the target system?

✓ Correct Answer

How many shares have the "no\_root\_squash" option enabled?

✓ Correct Answer

Gain a root shell on the target system

✓ Correct Answer

What is the content of the flag7.txt file?

✓ Correct Answer



vectors.

## TASK 12 – CRON JOB EXPLOITATION

- Listed scheduled cron jobs using `crontab -l` and inspected global cron directories.
- Identified scripts run by root with insecure write permissions.
- Injected payloads or replaced the original cron script.
- Waited for cron job execution and confirmed root shell access.
- Captured final flag and validated escalation path.

Answer the questions below

What is the content of the flag1.txt file?

THM-42828719920544

✓ Correct Answer

What is the content of the flag2.txt file?

THM-168824782390238

✓ Correct Answer

## RESULT:

Successfully escalated privileges through multiple Linux techniques and retrieved all required flags. Demonstrated understanding of local privilege escalation through hands-on exploitation and tool usage on simulated targets.