PROJECT

Linux Privilege Escalation Automation Toolkit

AYYAVARI VAMSHI KRISHNA

Unified Mentor

21 December 2025

1. ABSTRACT

Linux systems are widely used in enterprise environments, servers, and cloud infrastructures due to their stability and open-source nature. However, improper system configuration, weak permission management, and outdated software can introduce serious security vulnerabilities. One of the most critical threats in such environments is **privilege escalation**, where a normal user gains unauthorized elevated (root) privileges.

This project focuses on identifying potential privilege escalation vectors in a Linux system using an automated approach. A Python-based tool is developed to perform systematic enumeration of system configurations such as SUID binaries, cron jobs, sudo permissions, and kernel information. These components are commonly exploited by attackers if misconfigured.

The objective of this project is not exploitation, but **detection and analysis** of misconfigurations that could lead to privilege escalation. By automating the enumeration process, the tool reduces manual effort and ensures consistency in security assessment. The project was carried out in a controlled Kali Linux environment strictly for learning and ethical purposes.

2. INTRODUCTION

Linux privilege escalation refers to a situation where a user with limited permissions is able to gain higher privileges, typically root access, due to system weaknesses. These weaknesses often arise from misconfigured permissions, insecure scheduled tasks, vulnerable binaries, or kernel-level issues.

In real-world systems, privilege escalation is one of the final stages of an attack lifecycle. Once an attacker gains elevated access, they can modify system files, install persistent malware, or compromise sensitive data. Therefore, identifying privilege escalation vectors at an early stage is crucial for maintaining system security.

This project aims to understand how such vulnerabilities exist and how they can be detected using automated techniques. Instead of manually running multiple Linux commands, a Python script is used to automate the enumeration process. This approach aligns with modern security practices where automation plays a key role in efficient security auditing.

3. OBJECTIVES OF THE PROJECT

The main objectives of this project are:

- To understand the concept of Linux privilege escalation and its security implications
- To identify common misconfigurations that may lead to privilege escalation
- To automate system enumeration using Python scripting
- To analyze SUID binaries, cron jobs, sudo permissions, and kernel details
- To provide a structured and repeatable method for privilege escalation detection
- To improve practical knowledge of Linux system security

4. SYSTEM ENVIRONMENT AND TOOLS USED

The project was implemented in a controlled laboratory environment to ensure safety and ethical compliance.

Operating System:

- Kali Linux (Debian-based security distribution)

Programming Language:

- Python 3

Tools and Utilities:

- Linux built-in commands (`find`, `ls`, `sudo`, `uname`)
- Terminal interface
- Python `os` module

Kali Linux was chosen because it provides a secure and isolated environment for security experimentation and learning.

5. PROJECT METHODOLOGY

The methodology followed in this project consists of the following steps:

1. **Environment Setup:**
   A Kali Linux system was configured and verified for proper functioning.
2. **Manual Understanding:**
   Core Linux privilege escalation concepts were studied, including SUID binaries, cron jobs, and sudo permissions.
3. **Automation Design:**
   A Python script was designed to execute essential enumeration commands automatically.
4. **Execution:**
   The script was executed in the terminal to collect system information.
5. **Observation and Analysis:**
   The output was analyzed to identify potential privilege escalation vectors.
6. **Documentation:**
   Screenshots and explanations were documented for reporting and presentation purposes.

6. IMPLEMENTATION DETAILS (PYTHON AUTOMATION)

To automate the enumeration process, a Python script was developed using the `os.system()` function. This script sequentially executes important Linux commands required for privilege escalation analysis.

Python Script Used:

```
import os

print("Linux Privilege Escalation Scanner Started")

print("\n[+] Checking SUID binaries")
os.system("find / -perm -4000 2>/dev/null")

print("\n[+] Checking cron jobs")
os.system("ls -la /etc/cron*")

print("\n[+] Checking sudo permissions")
os.system("sudo -l")

print("\n[+] Kernel information")
os.system("uname -a”)
```

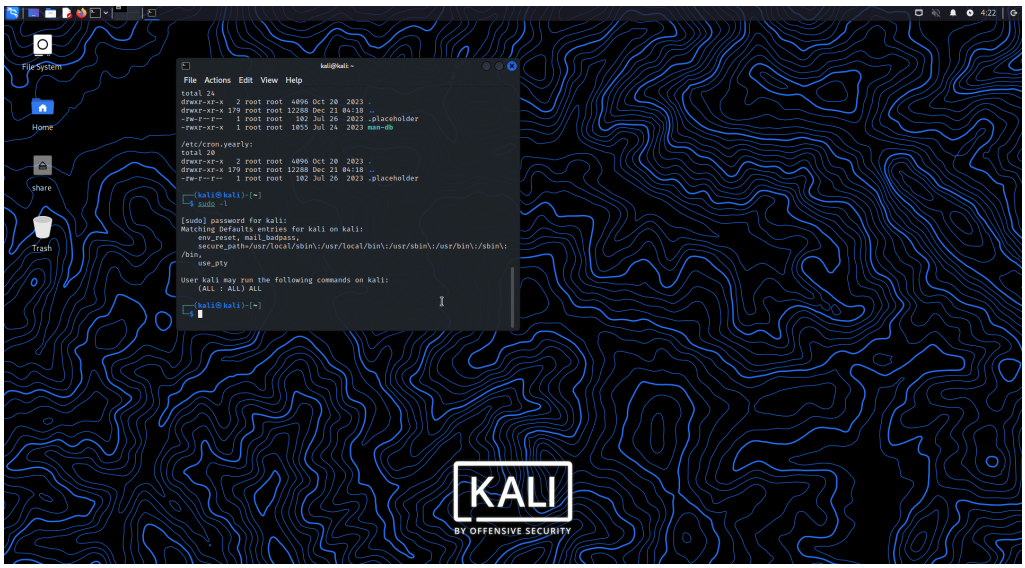
Explanation:

The script automates system enumeration by executing predefined commands that inspect privilege-related configurations. It eliminates the need for manually running each command and ensures that all checks are performed in a structured manner.

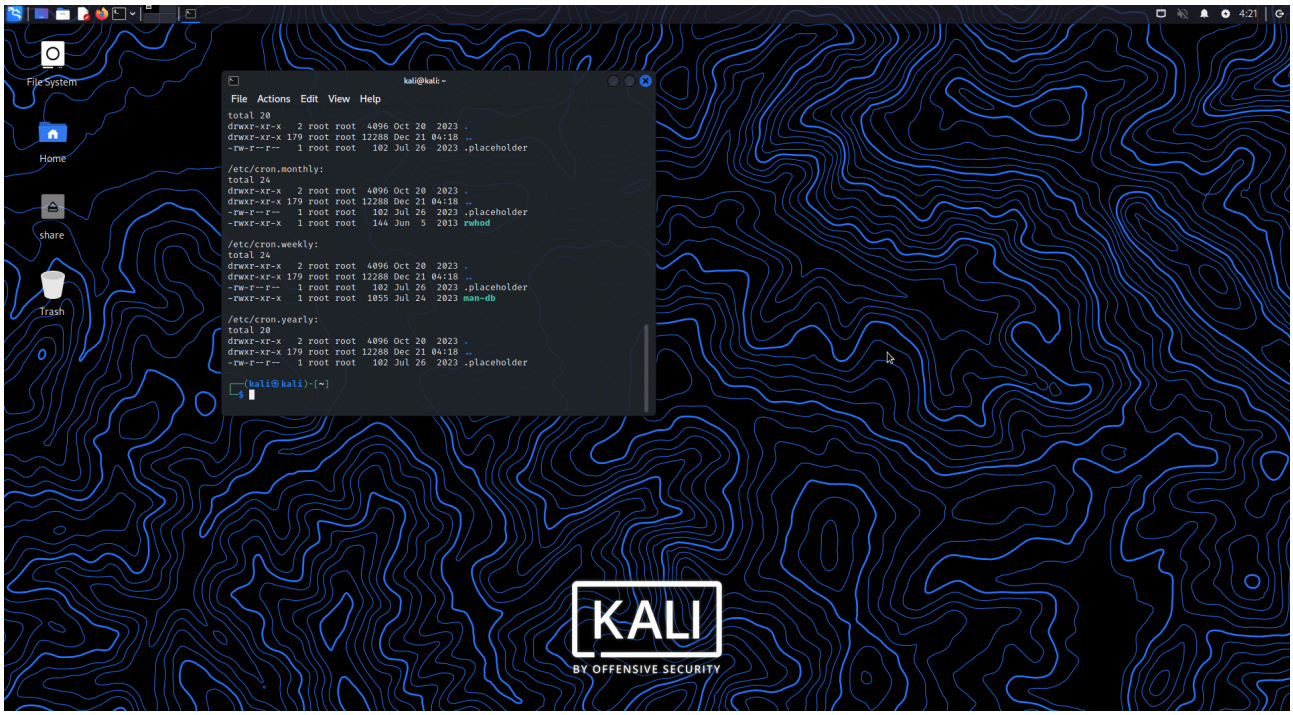
7. RESULTS AND OBSERVATIONS

During execution, the script successfully retrieved the following information:

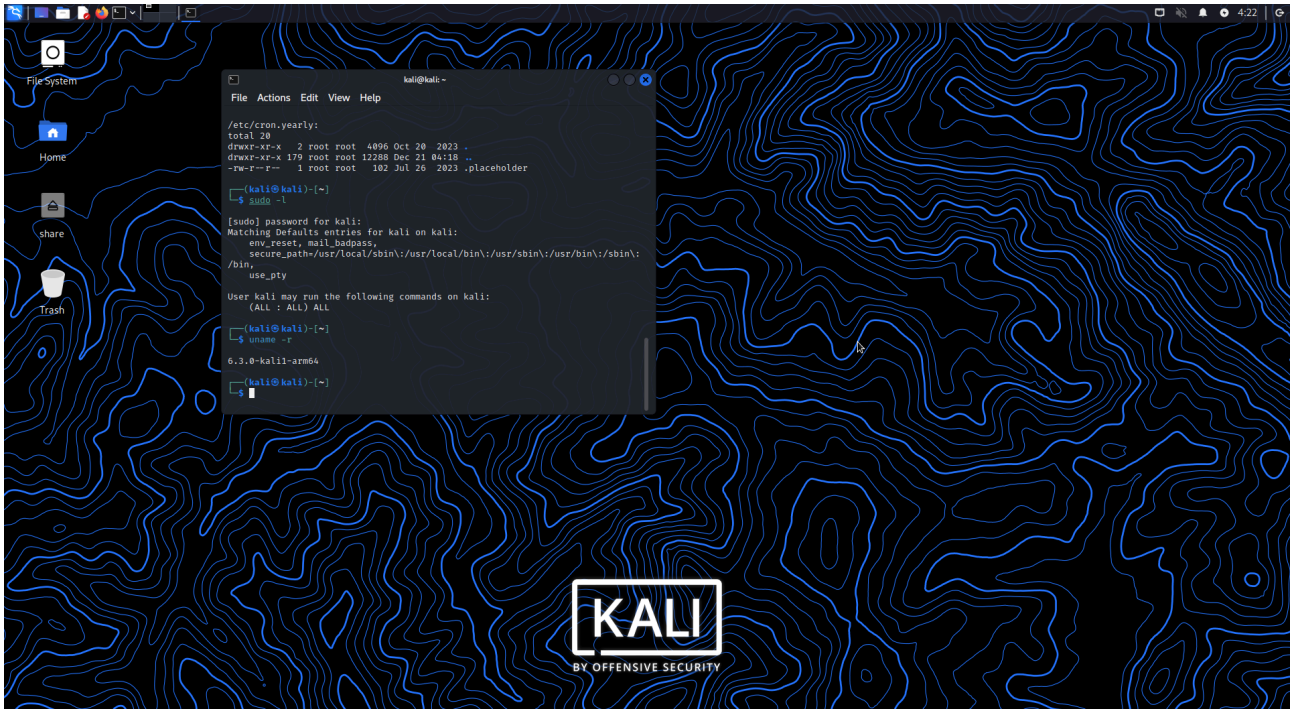
- **SUID Binaries:**
  Identified binaries that run with elevated privileges and may pose a risk if misused.



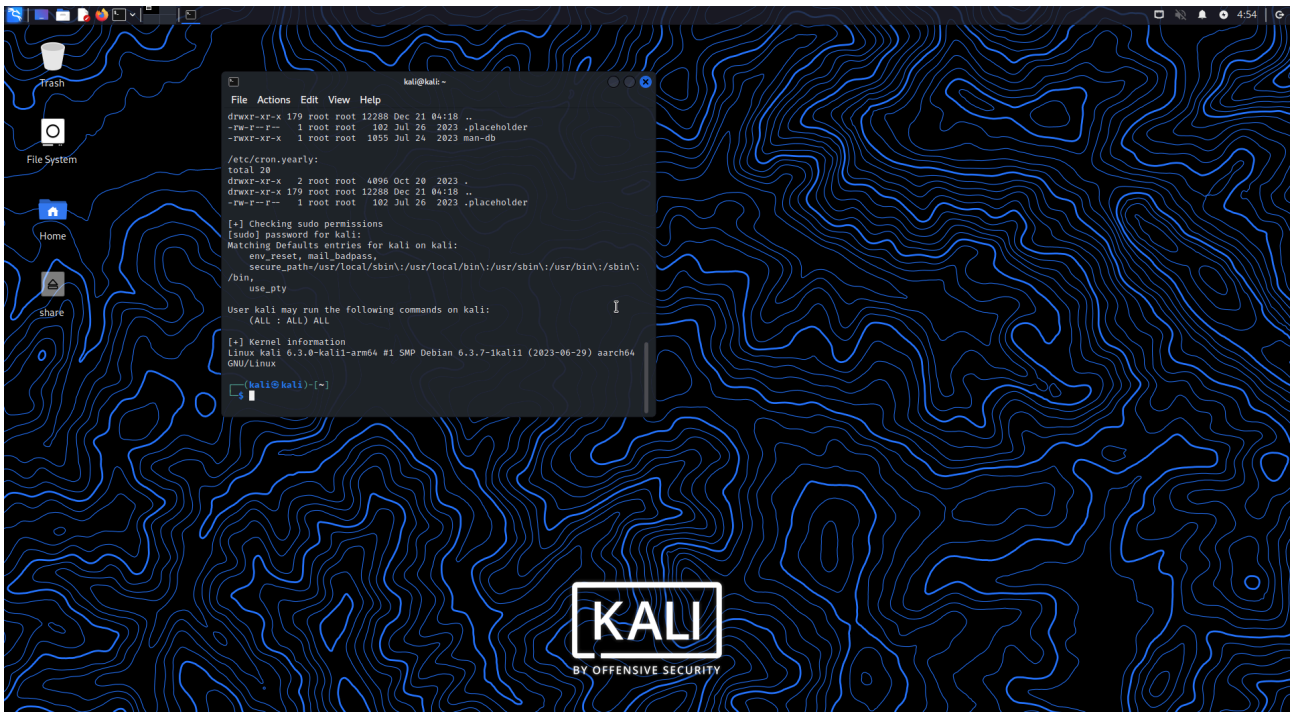
Cron Jobs:
Listed scheduled system tasks that could be exploited if writable by non-root users.



Sudo Permissions:
Displayed commands that the current user is allowed to execute with elevated privileges.



Kernel Information:
Provided kernel version details useful for identifying known vulnerabilities.



8. CONCLUSION

This project demonstrates how Linux privilege escalation vectors can be identified using a simple yet effective Python automation approach. By combining Linux command-line utilities with Python scripting, the project highlights the importance of automation in security auditing.

The project helped in gaining practical knowledge of Linux internals, privilege management, and security assessment techniques. It successfully fulfills the objectives of the internship by providing hands-on exposure to real-world system security concepts.

THANKYOU