

Project 8: Linux Hardening Audit Tool

Intern Name: Mohammed Fadil PK

Domain: Cybersecurity / System Administration

1. Introduction

System hardening is the process of securing a system by reducing its surface of vulnerability. In a manual environment, administrators must check each server configuration individually, which is time-consuming and prone to human error. This project automates that process using a custom-built Python tool.

2. Abstract

The "Linux Hardening Audit Tool" is a Python-based utility that automatically verifies the security posture of a Linux server. It performs critical checks—such as verifying firewall status, SSH root login configurations, and network forwarding rules—and provides immediate feedback (PASS/FAIL) along with a saved text report for auditing purposes.

3. Tools & Technologies Used

- **Operating System:** Kali Linux (Debian-based).
- **Language:** Python 3.
- **Libraries:** subprocess (to execute system commands), os (for file handling).
- **Target System:** Localhost (Virtual Machine).

4. Steps Involved in Building the Project

Step 1: Environment Setup

- Initialised a Kali Linux virtual environment.
- Configured the development workspace and created the audit_tool.py script.

Step 2: Developing the Core Logic

- **Firewall Check:** Implemented a function using subprocess to query ufw status. The tool parses the output to ensure the firewall is "active".

- **SSH Security:** Wrote logic to read the `/etc/ssh/sshd_config` file and verify that `PermitRootLogin` is set to `no`, preventing high-privileged unauthorised access.
- **Network Hardening:** Added a check for `/proc/sys/net/ipv4/ip_forward` to ensure the server is not acting as a router/gateway (Value must be 0).

Step 3: Reporting Mechanism

- Created a logging function that prints colour-coded results (Green for PASS, Red for FAIL) to the terminal for real-time feedback.
- Implemented file handling to append all results into a permanent log file named `audit_report.txt`.

Step 4: Testing & Validation

- Executed the script with `sudo` privileges.
- Verified that the tool correctly identified the hardened state of the system (Firewall ON, SSH Root Login OFF).

5. Results (Evidence)

Screenshot 1: Tool Execution & Terminal Output

A terminal window with a dark background and a green, abstract pattern on the left side. The prompt is `(melvin@kali) - [~/Project-8-Linux-Audit]`. The user enters `$ sudo python3 audit_tool.py`. The output shows the script starting a Linux Hardening Audit. It checks the Firewall Status (ACTIVE), SSH Configuration (Root Login DISABLED), and IP Forwarding (DISABLED), all with PASS status. The audit is complete, and the report is saved to `audit_report.txt`.

```
(melvin@kali) - [~/Project-8-Linux-Audit]
$ sudo python3 audit_tool.py

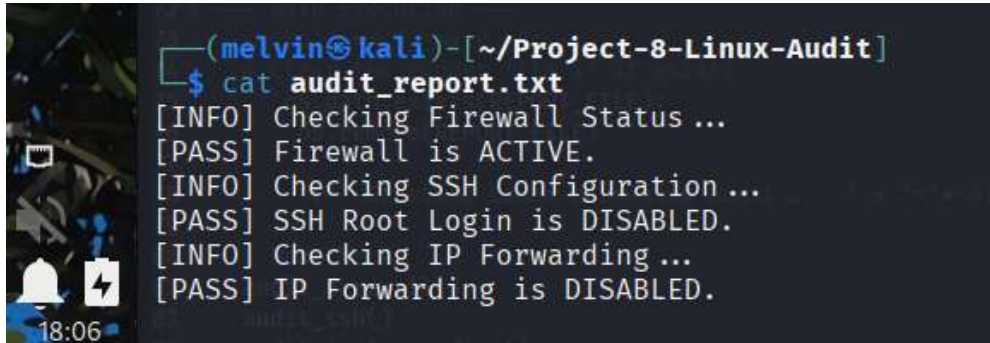
Starting Linux Hardening Audit ...

[INFO] Checking Firewall Status ...
[PASS] Firewall is ACTIVE.
[INFO] Checking SSH Configuration ...
[PASS] SSH Root Login is DISABLED.
[INFO] Checking IP Forwarding ...
[PASS] IP Forwarding is DISABLED.

Audit Complete! Report saved to audit_report.txt
```

Figure 1: Execution of the `audit_tool.py` script, displaying real-time system checks and successful validation (PASS) of the firewall, SSH configuration, and IP forwarding rules.

Screenshot 2: Generated Log File

A terminal window with a dark background. The prompt is `(melvin@kali)-[~/Project-8-Linux-Audit]`. The command `$ cat audit_report.txt` has been executed. The output shows three checks: Firewall Status (PASS, ACTIVE), SSH Configuration (PASS, Root Login DISABLED), and IP Forwarding (PASS, DISABLED). The terminal has a sidebar on the left with icons for a window, a folder, a bell, and a lightning bolt. The time `18:06` is visible at the bottom left.

```
(melvin@kali)-[~/Project-8-Linux-Audit]
$ cat audit_report.txt
[INFO] Checking Firewall Status ...
[PASS] Firewall is ACTIVE.
[INFO] Checking SSH Configuration ...
[PASS] SSH Root Login is DISABLED.
[INFO] Checking IP Forwarding ...
[PASS] IP Forwarding is DISABLED.
```

Figure 2: Verification of the automated reporting feature, displaying the contents of the `audit_report.txt` file, which stores the audit results for future review.

6. Conclusion

This project successfully demonstrates how automation can enhance cybersecurity operations. The tool provides a reliable, repeatable method for auditing Linux servers, ensuring they comply with basic hardening standards before being deployed in a production environment. Future improvements could include checks for password complexity and automatic remediation of failed checks.