Linux IAM & System Hardening

Name: **Sanjana Thawait**
ERP: **6604710**
Course: **B.Tech CSE (Cybersecurity)**
Semester: **5th**
Section: **CY5A**
Date: **05/11/2025**

1. INTRODUCTION

This project demonstrates how Identity and Access Management (IAM) and system hardening are implemented on a Linux-based server environment. IAM ensures that users have the correct amount of access depending on their role, while hardening focuses on reducing vulnerabilities. We simulate a real organization scenario with:
• Admins – full administrative privileges
• Developers – limited access only to required commands
• Auditors – read-only monitoring access

2. PROJECT OBJECTIVE

• Build a secure user access model in Linux using IAM concepts
• Apply least privilege through sudo and ACLs
• Create a vulnerable lab intentionally and exploit it using Kali
• Fix vulnerabilities and produce a remediation report

3. THEORY OF PROJECT

IAM ensures the right user gets the right access. Linux implements IAM using:
• Users and Groups (RBAC)
• Sudo permissions (Least Privilege)
• POSIX/ACL permissions
• auditd monitoring

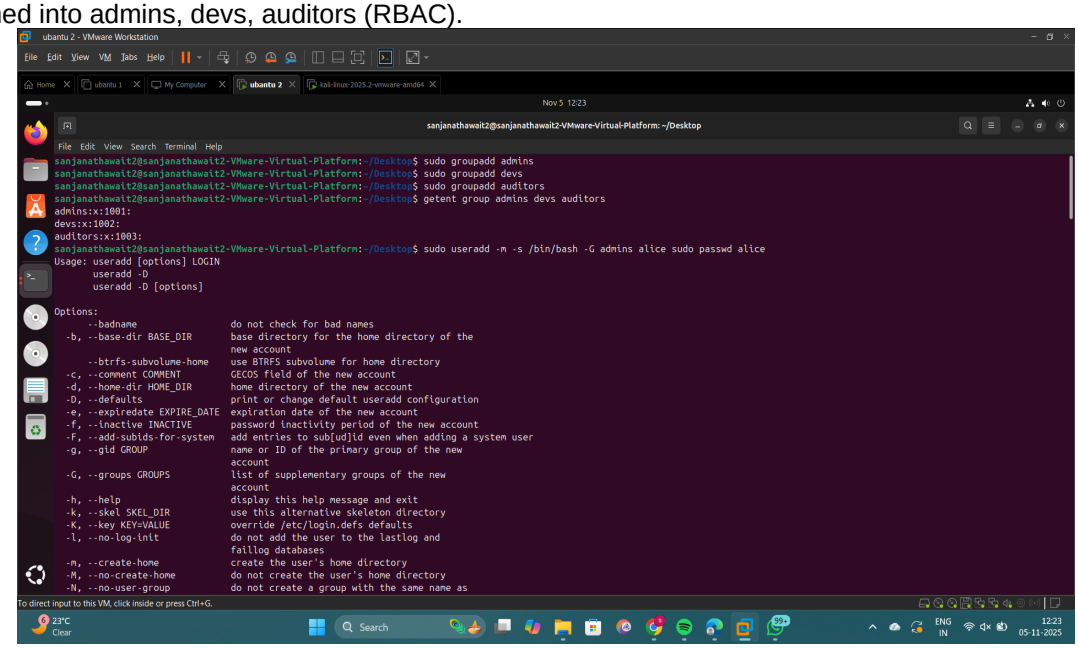
Hardening removes misconfigurations that allow privilege escalation.

4. TOOLS USED

• Ubuntu VM (Target)
• Kali Linux VM (Attacker)
• ACL, POSIX Permissions
• auditd (log monitoring)
• SSH & nmap for attack simulation

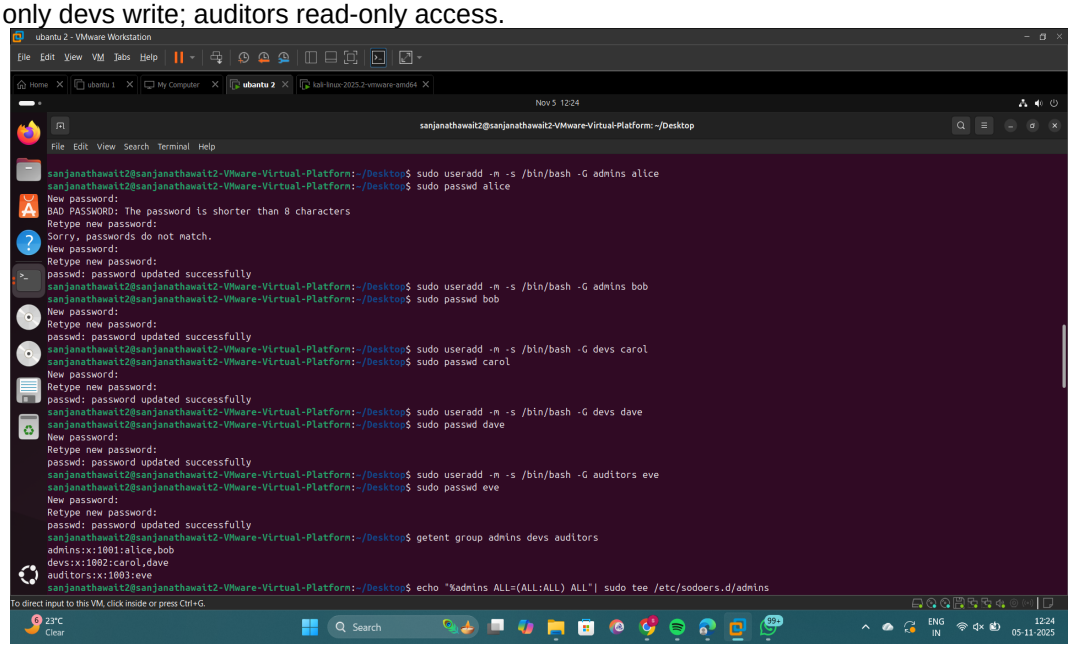
Creating Users & Assigning Groups

Users assigned into admins, devs, auditors (RBAC).



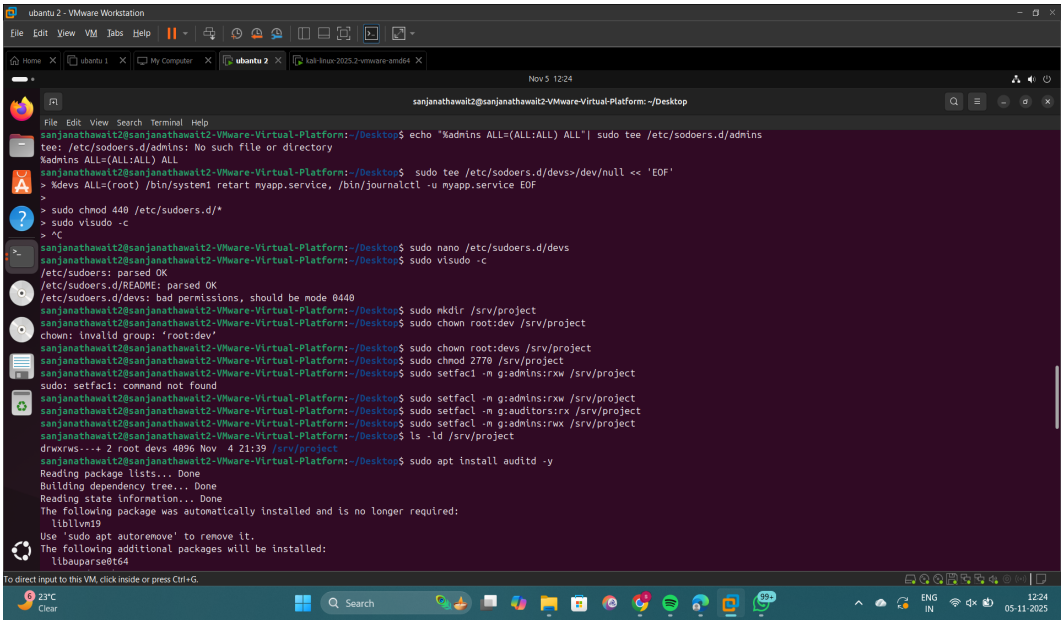
Applying ACL Permissions on /srv/project

ACL ensures only devs write; auditors read-only access.



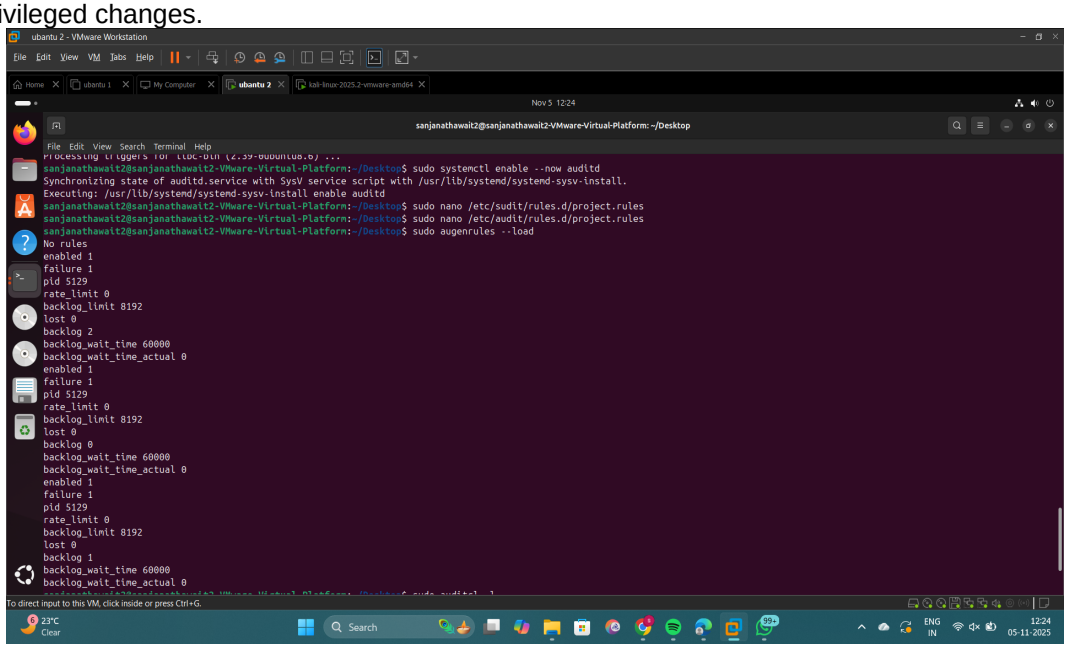
Configuring Restricted Sudo Permissions

Developers allowed only specific sudo actions (least privilege).



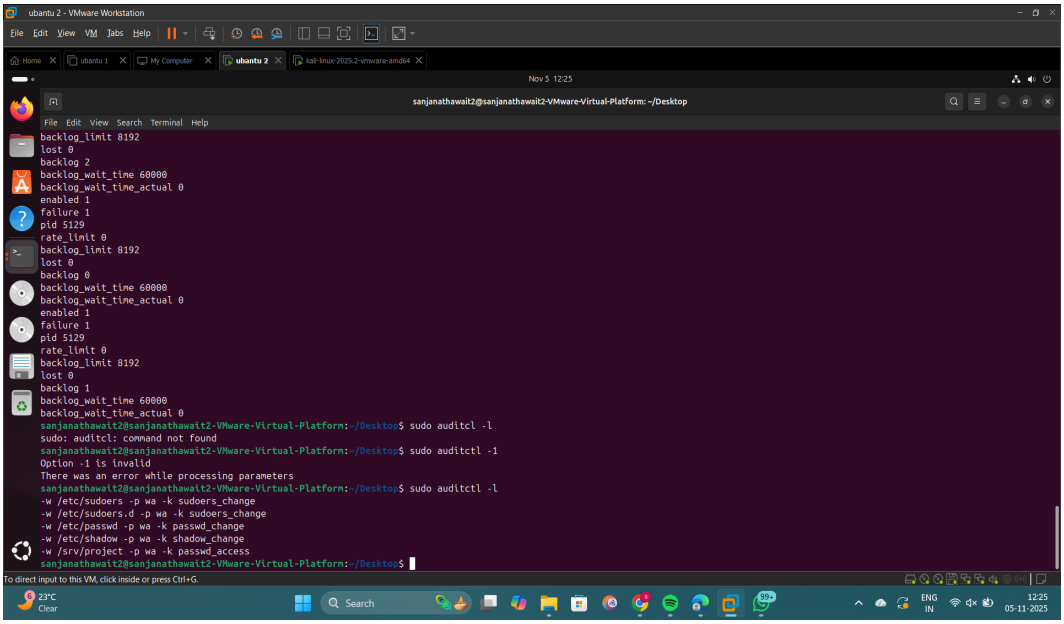
Enabling auditd Monitoring

auditd logs privileged changes.



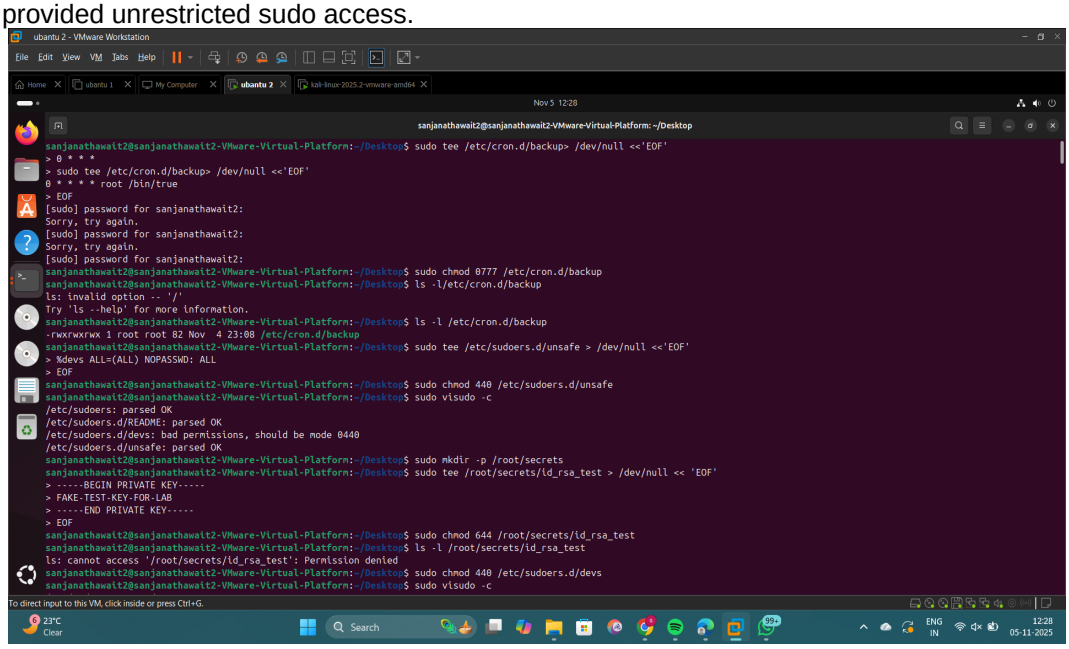
World-Writable Cron Vulnerability

Cron was intentionally made writable (vulnerability).



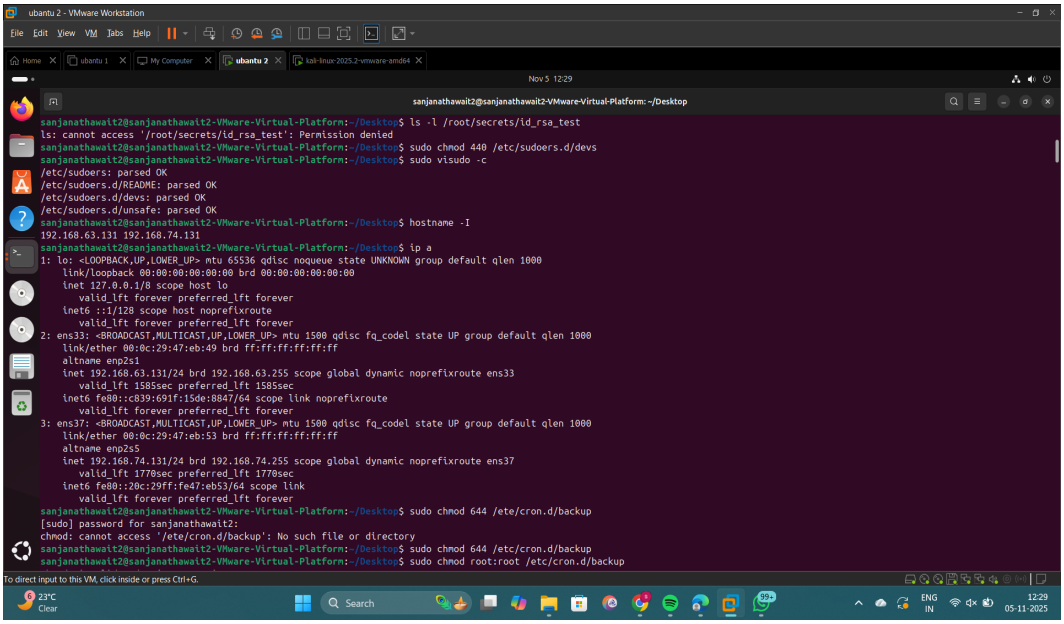
NOPASSWD sudo Misconfiguration

NOPASSWD provided unrestricted sudo access.



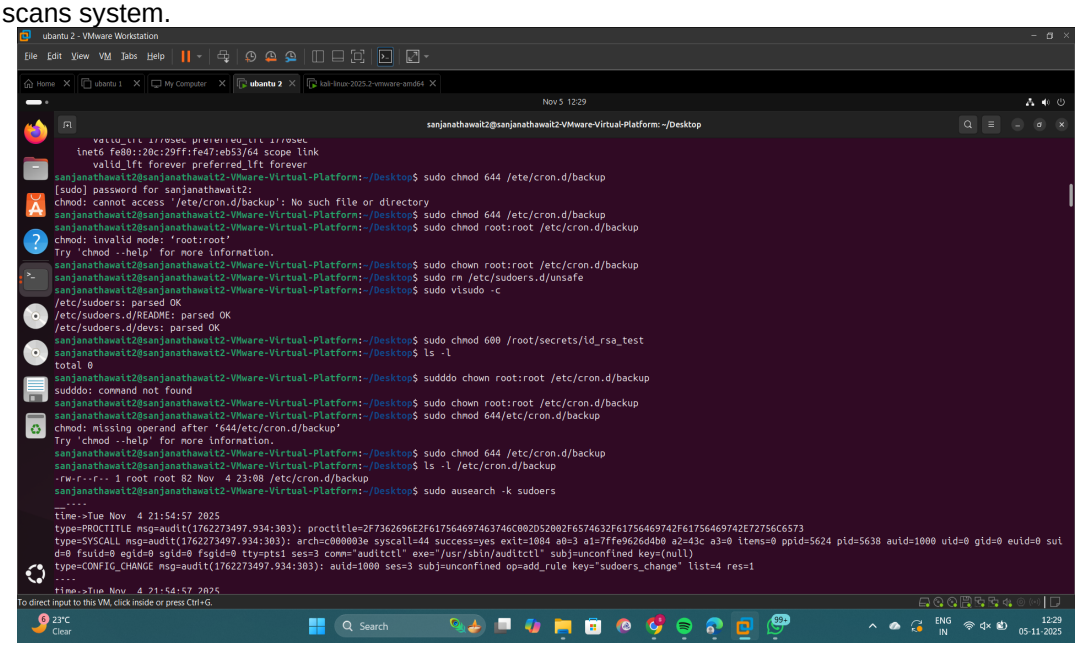
Weak SSH Private Key Permissions

Weak key permissions allowed unauthorized access.



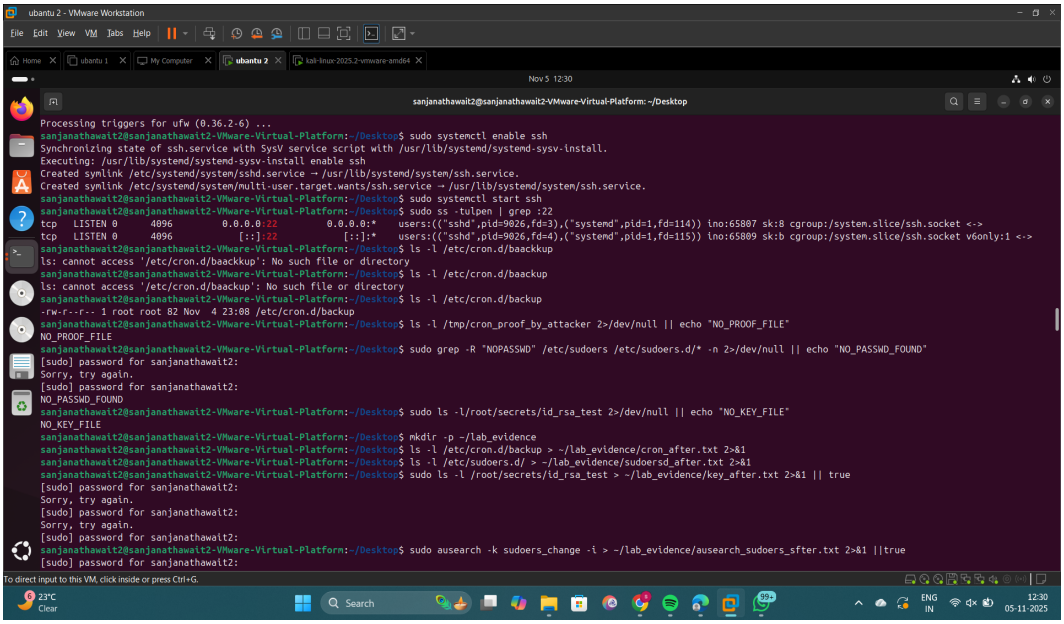
Kali Scanning using nmap

Kali attacker scans system.



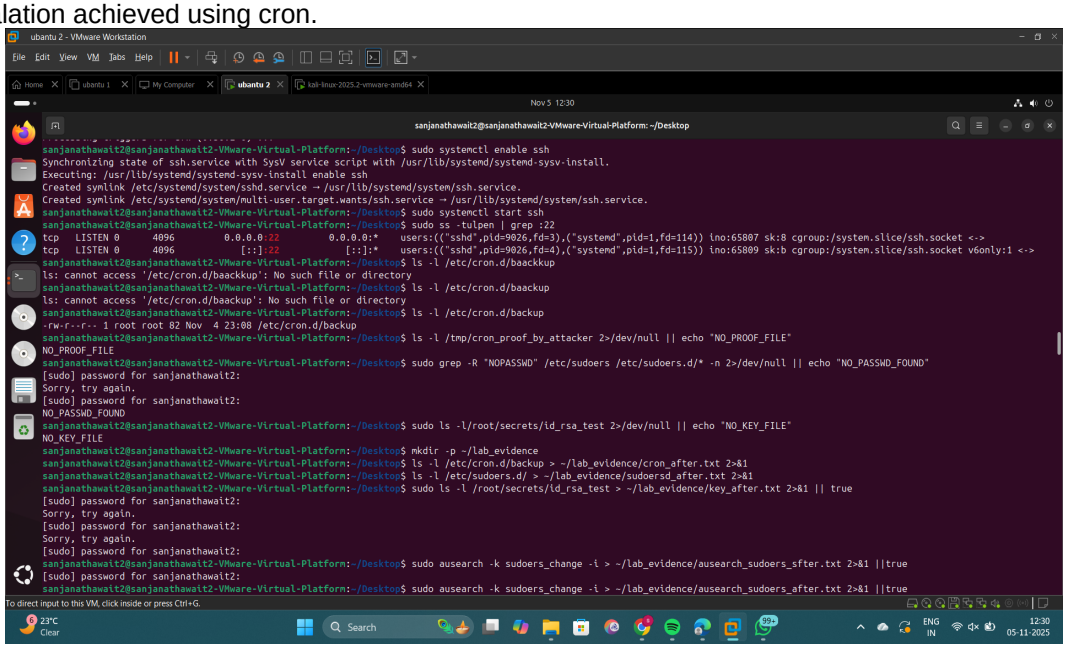
SSH Access Exploitation

Attacker connects via SSH.



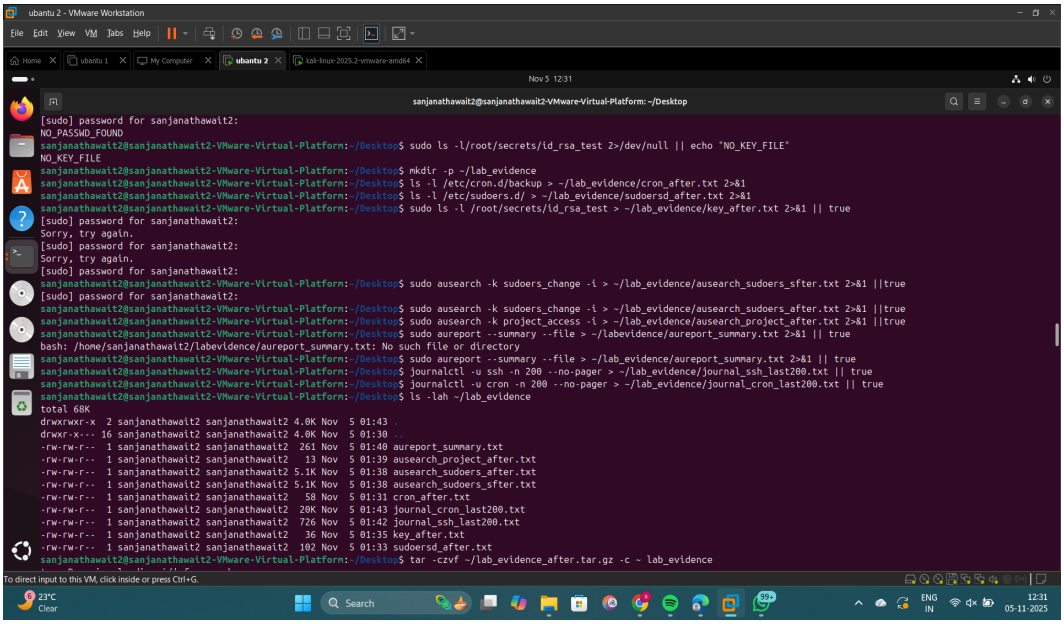
Privilege Escalation via cron

Privilege escalation achieved using cron.



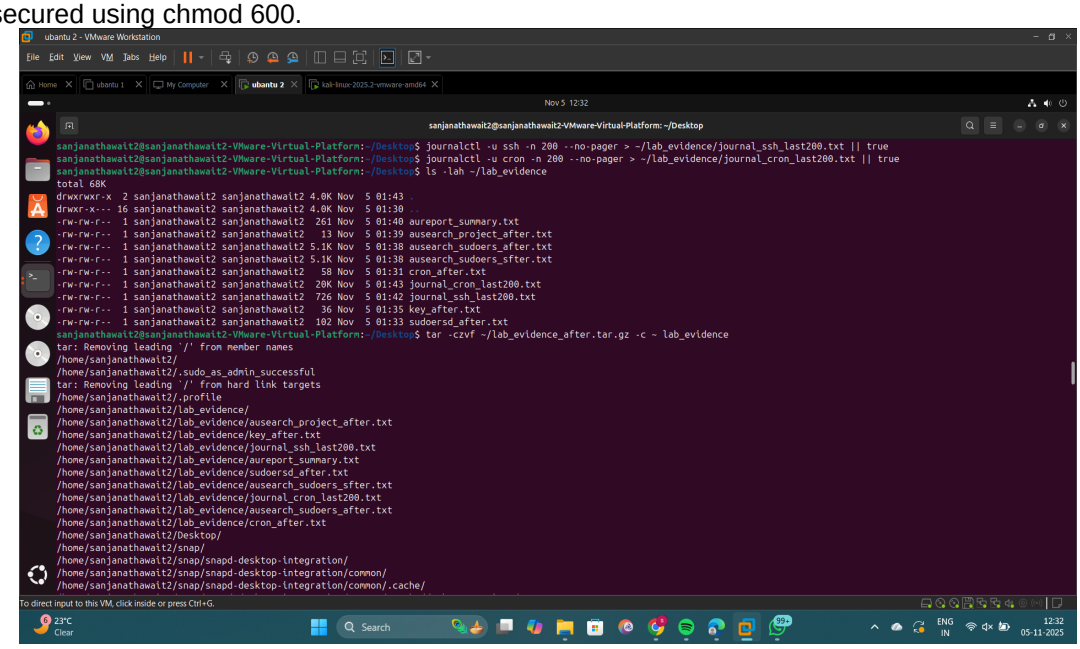
Fix: Correcting Cron Permissions

Corrected ownership and permissions.



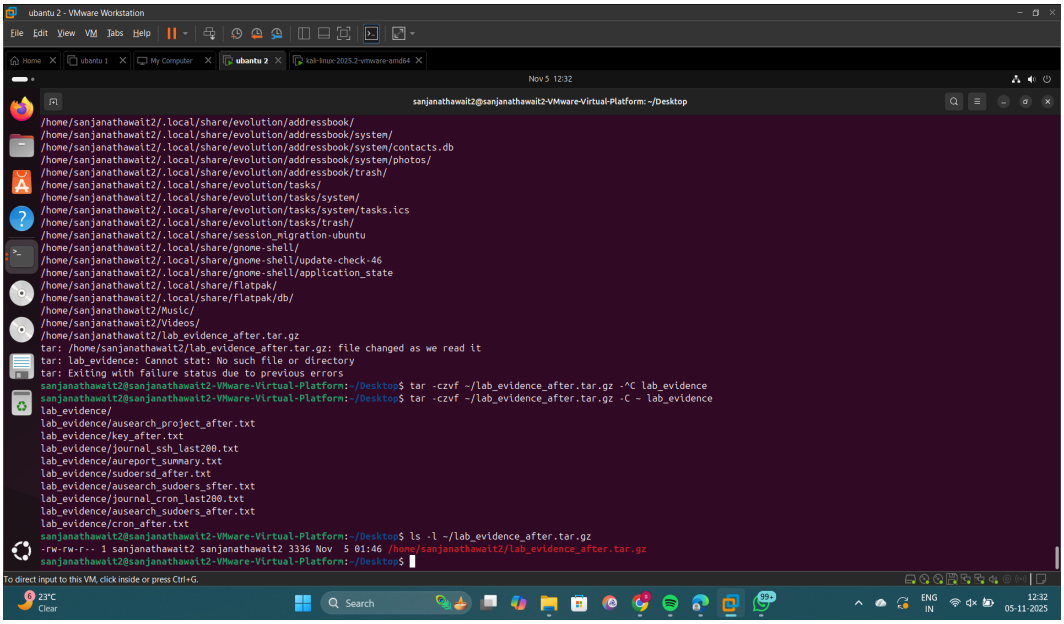
Fix: Securing Private Key

Private key secured using chmod 600.



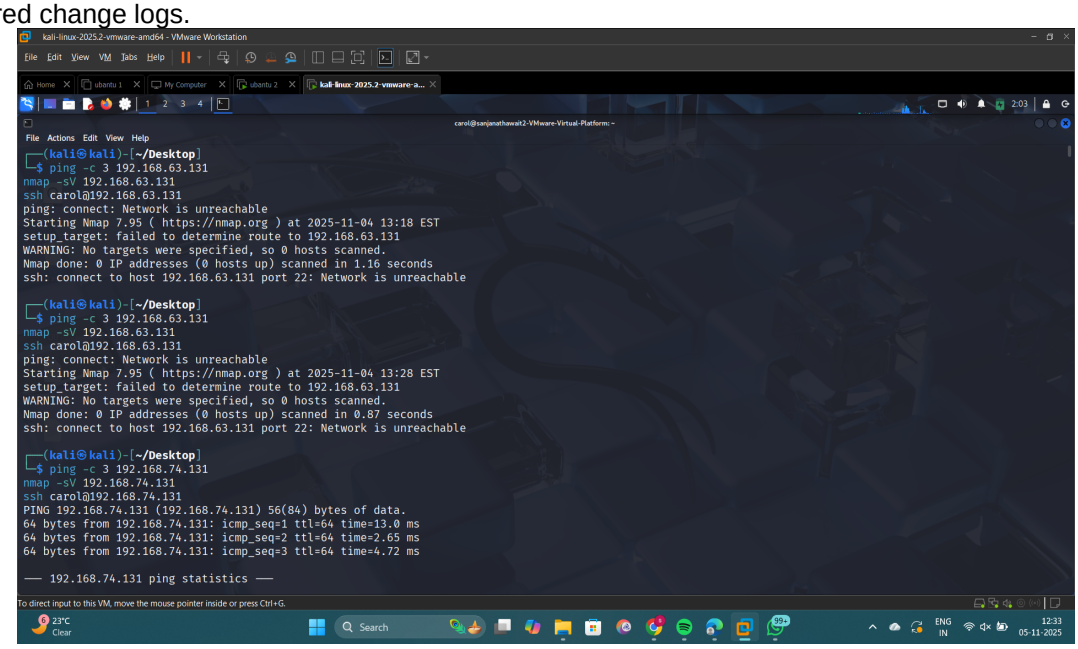
Fix: Validating sudo rules

sudo validated using visudo.



Audit Log Evidence

auditd captured change logs.



Conclusion

This project demonstrated the complete lifecycle of Linux IAM and System Hardening.

First, a secure baseline was implemented:
• Role Based Access Control was enforced using users and groups
• Least privilege principle applied through restricted sudo rules
• ACLs secured the project directory allowing controlled access
• auditd was configured for monitoring high risk configuration changes

Second, security weaknesses were intentionally introduced:
• Writable cron allowed privilege escalation
• NOPASSWD enabled unrestricted sudo privilege
• Weak permissions on a private key allowed unauthorized access

Third, using Kali Linux as an attacker:
• Reconnaissance was performed (nmap scan)
• Exploitation succeeded, proving privilege escalation to root

Finally, remediation:
• Misconfigurations were fixed (chmod, removing NOPASSWD)
• auditd logs confirmed tracking of privileged operations

Learning Outcome:
This project reinforced how misconfigurations, not software vulnerabilities, are the biggest root cause of security breaches.
By enforcing IAM, least privilege, ACL, and auditing, the attack surface drastically reduces.
Securing Linux is not only about tools — it is about discipline and correct configuration.