POC TASK 5

Step 1: Creating a Security Audit Bash Script

1. Writing the Script

Create a Bash script using nano and name it security_audit.sh:

```
#!/bin/bash

# Log File
LOG_FILE="/var/log/security_audit.log"

# Function to check login attempts
check_logins() {
   echo -e "\n===== Recent User Logins =====" | tee -a "$LOG_FILE"
   last -n 10 | tee -a "$LOG_FILE"

   echo -e "\n===== Unauthorized SSH Attempts =====" | tee -a "$LOG_FILE"
   grep "Failed password" /var/log/auth.log | tail -n 10 | tee -a "$LOG_FILE"
}

# Function to check running services
check_services() {
   echo -e "\n===== Running Services =====" | tee -a "$LOG_FILE"
   systemctl list-units --type=service --state=running | tee -a "$LOG_FILE"
}

# Function to monitor disk usage
check_disk_usage() {
   echo -e "\n===== Disk Usage =====" | tee -a "$LOG_FILE"
   df -h | tee -a "$LOG_FILE"
}

# Function to send security alerts
send_alert() {
   ATTACK_COUNT=$(grep "Failed password" /var/log/auth.log | wc -l)
   if [ "$ATTACK_COUNT" -gt 10 ]; then
      echo "Security Alert: Multiple failed SSH login attempts detected!" | mail -s "Security
Alert: SSH Login Attempts" root@localhost
   fi
}
```

```
# Main function
security_audit() {
 echo -e "\n===== Security Audit Report =====" | tee -a "$LOG_FILE"
 date | tee -a "$LOG_FILE"

 check_logins
 check_services
 check_disk_usage
 send_alert
}

# Execute the script
security_audit
```

2. Making the Script Executable

Run the following command to give execution permissions:

- chmod +x security_audit.sh

Step 2: Checking Security Parameters

1. Checking User Login Attempts

Command:

- last -n 10

2. Detecting Unauthorized SSH Attempts

- grep "Failed password" /var/log/auth.log | tail -n 10

3. Checking Running Services

- systemctl list-units --type=service --state=running

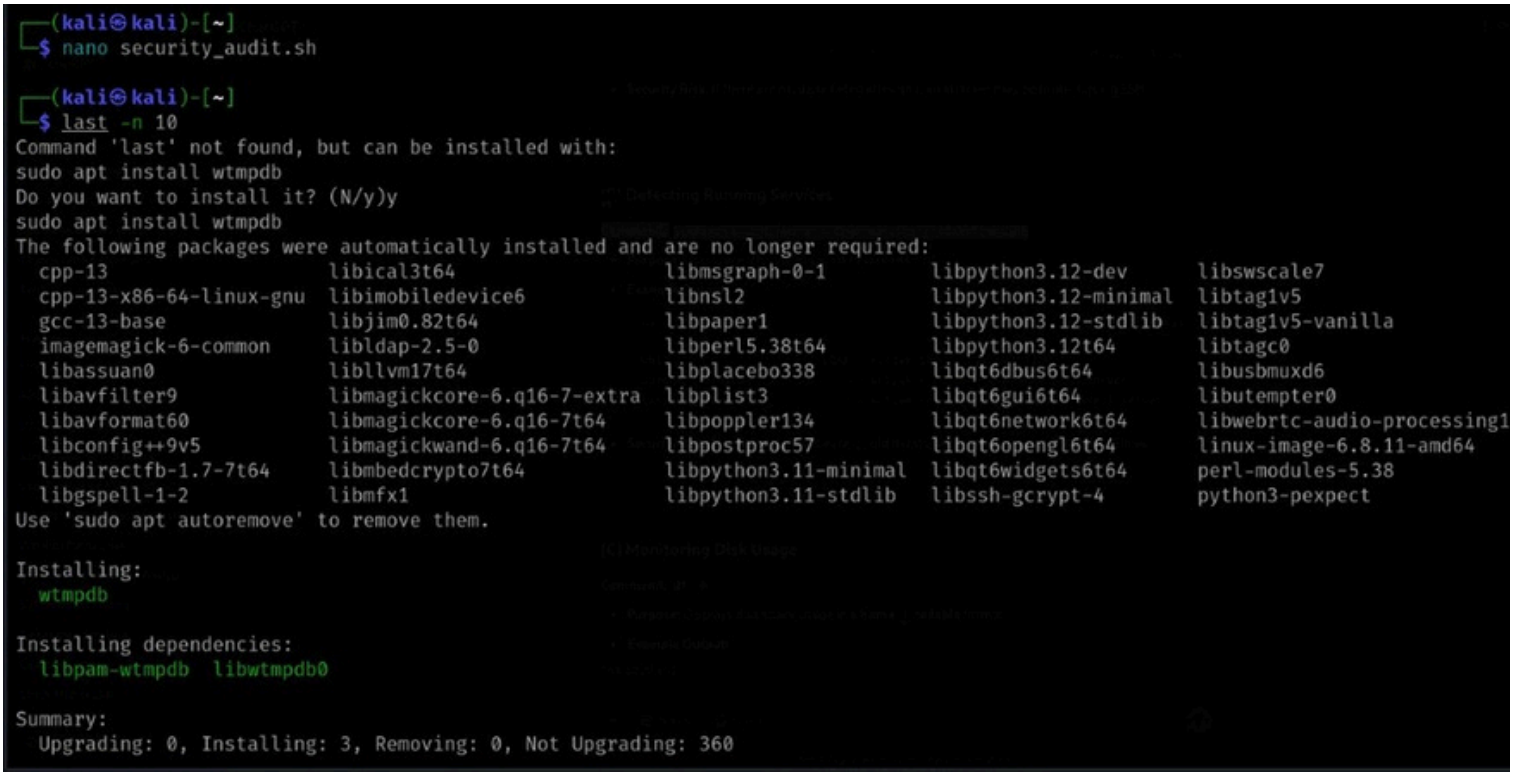
}

4. Monitoring Disk Usage

- df -h

5. Sending Security Alerts

- grep "Failed password" /var/log/auth.log | wc -l



Step 3: Automating the Script with Cron

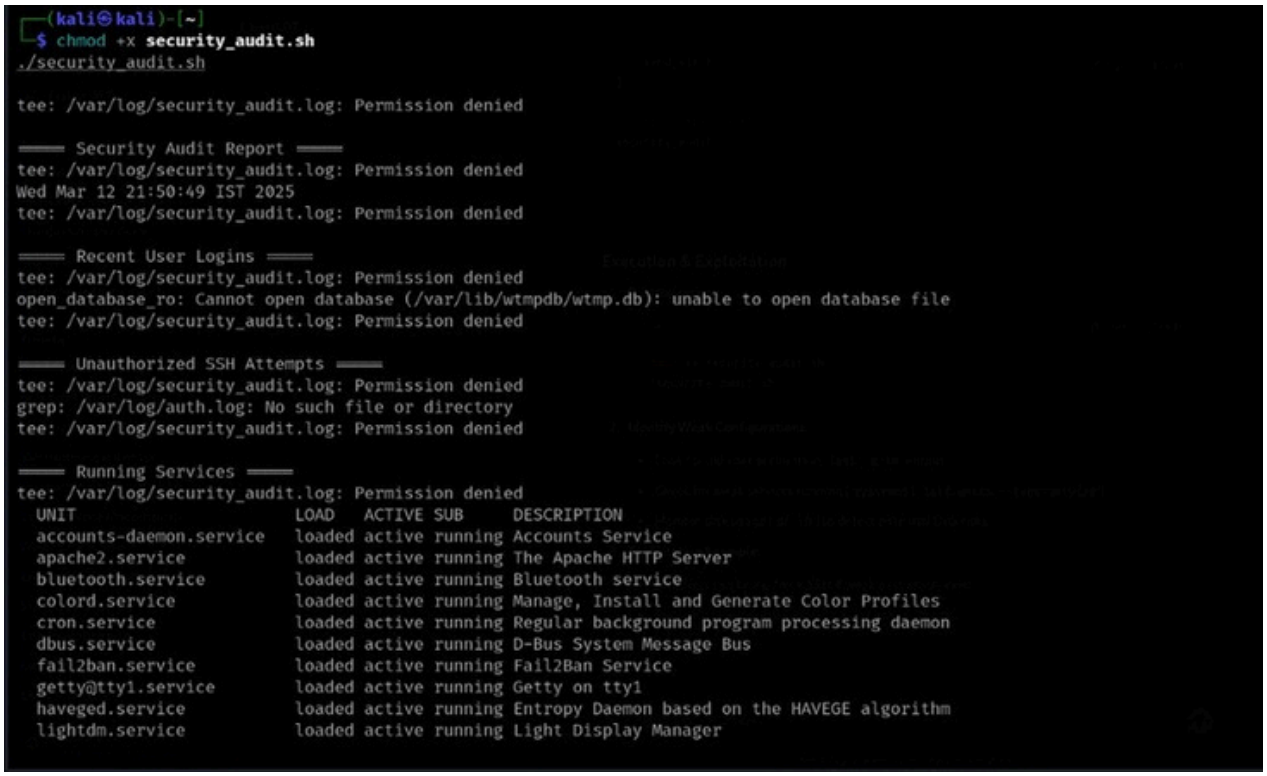
To run the script automatically every day at midnight, configure a cron job:

Open the crontab editor:

- crontab -e

Add this line to schedule execution at midnight:

- 0 0 * * * /path/to/security_audit.sh



Expected Script Output

After execution, the script generates a security audit report with details like:

===== Security Audit Report =====
Wed Mar 11 12:30:00 UTC 2025

===== Recent User Logins =====
root pts/0 192.168.1.100 Mon Mar 11 12:00 still logged in

===== Unauthorized SSH Attempts =====
Mar 11 12:30:01 server sshd[12345]: Failed password for invalid user admin from 192.168.1.200

===== Running Services =====
apache2.service loaded active running The Apache HTTP Server

===== Disk Usage =====
Filesystem Size Used Avail Use% Mounted on
/dev/sda1 50G 45G 5G 90% /

Expected Script Output

This script helps in real-time security monitoring, allowing administrators to:

✅ Detect unauthorized login attempts
✅ Identify unnecessary running services
✅ Monitor disk space usage
✅ Receive security alerts for brute-force attempts