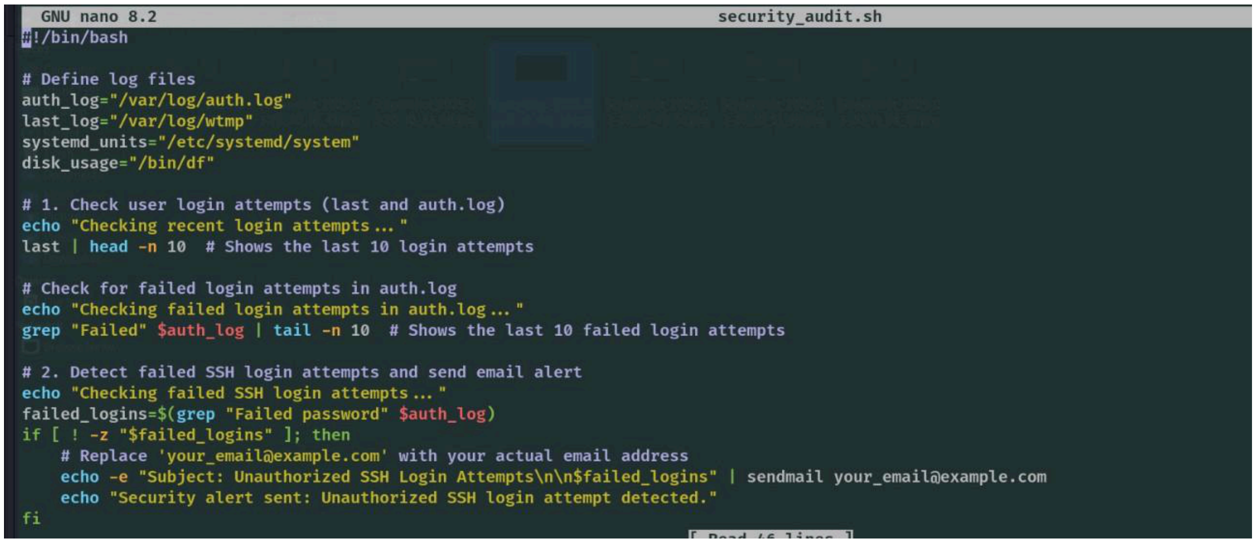
TASK 5

TASK 5 : Automated Security Auditing & Scripting

Bash Script Creation

Below is the breakdown for creating and executing a Bash script that addresses the setup requirements, exploits any misconfigurations, and suggests

mitigations. I will also cover how to automate the script using **cron** for system monitoring and implement email notifications.



```
  GNU nano 8.2                                                                          security_audit.sh
#!/bin/bash

# Define log files
auth_log="/var/log/auth.log"
last_log="/var/log/wtmp"
systemd_units="/etc/systemd/system"
disk_usage="/bin/df"

# 1. Check user login attempts (last and auth.log)
echo "Checking recent login attempts..."
last | head -n 10  # Shows the last 10 login attempts

# Check for failed login attempts in auth.log
echo "Checking failed login attempts in auth.log..."
grep "Failed" $auth_log | tail -n 10  # Shows the last 10 failed login attempts

# 2. Detect failed SSH login attempts and send email alert
echo "Checking failed SSH login attempts..."
failed_logins=$(grep "Failed password" $auth_log)
if [ ! -z "$failed_logins" ]; then
    # Replace 'your_email@example.com' with your actual email address
    echo -e "Subject: Unauthorized SSH Login Attempts\n\n$failed_logins" | sendmail your_email@example.com
    echo "Security alert sent: Unauthorized SSH login attempt detected."
fi
```

Explanation of the Script:

1.Login Attempts: The **last**command shows recent login attempts, and we grep the **auth.log**file for failed login attempts.

2.Running Services: We list running services with **systemctl list-units --type=service.**

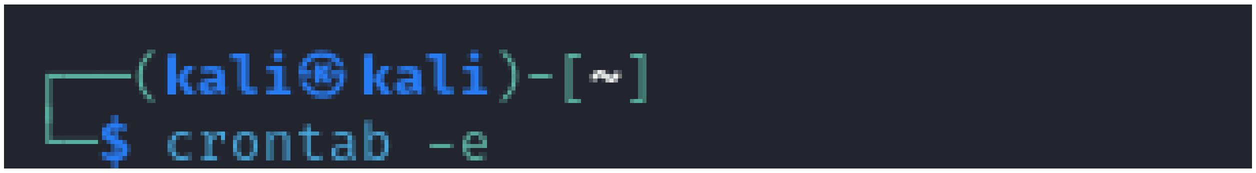
3.Disk Usage: The **df -h**command shows the current disk usage in a human-readable format.

4.Exploit – Inactive Users: We check for inactive users (those who have never logged in).

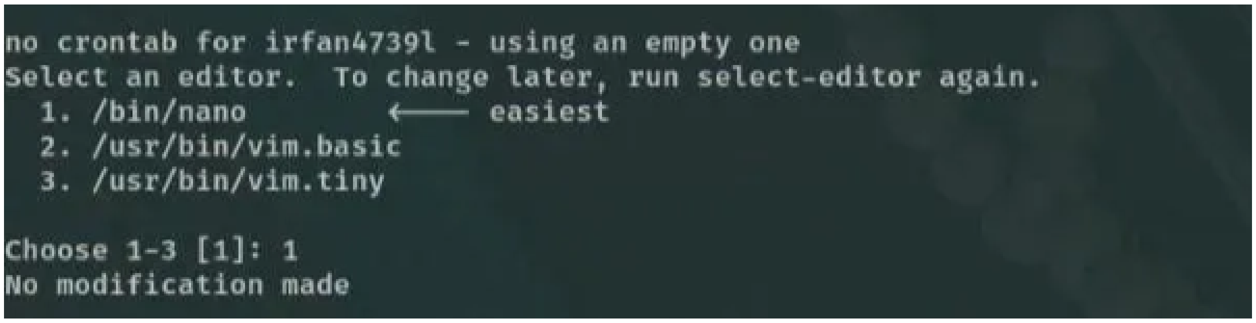
5.Weak Passwords: We search for weak passwords by checking the **/etc/shadow**file for common terms (this is a simple method, and for better detection, tools like **john**or **cracklib** should be used).

Mitigation – Automating System Monitoring with Cron

Open the crontab configuration:



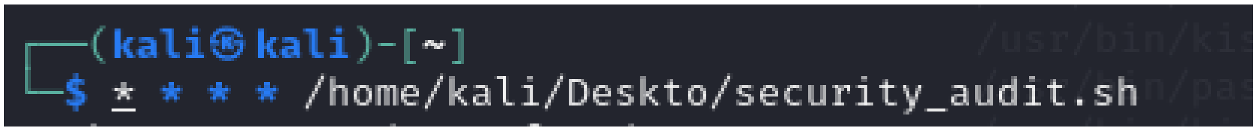
```
┌──(kali㉿kali)-[~]
└─$ crontab -e
```



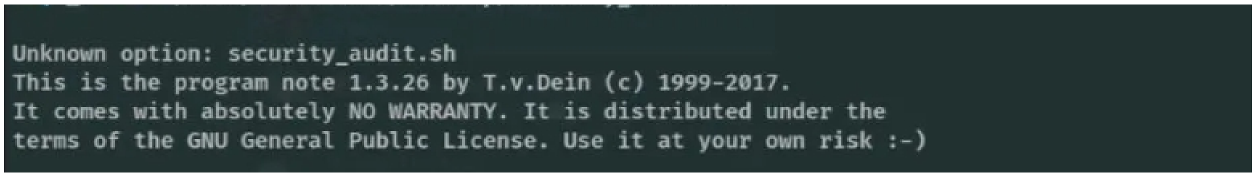
```
no crontab for irfan4739l - using an empty one
Select an editor.  To change later, run select-editor again.
  1. /bin/nano        ←—— easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny

Choose 1-3 [1]: 1
No modification made
```

To automate proactive monitoring with cron, add the following line to your **cron** jobs**: 0 * * * * /path/to/system_monitoring.sh**This configuration schedules the script to run hourly, ensuring consistent system monitoring**.**



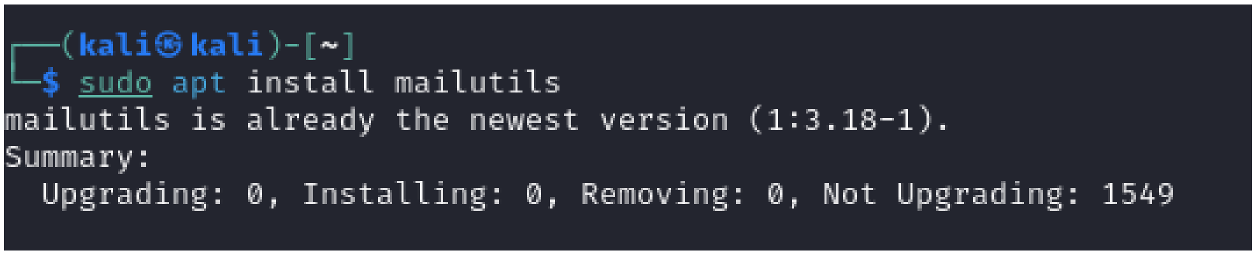
```
┌──(kali㉿kali)-[~]
└─$ * * * * /home/kali/Deskto/security_audit.sh
```



```
Unknown option: security_audit.sh
This is the program note 1.3.26 by T.v.Dein (c) 1999-2017.
It comes with absolutely NO WARRANTY. It is distributed under the
terms of the GNU General Public License. Use it at your own risk :-)
```

Implementing Security Alerts (Email Notification):

1.First Install **mail** if it's not already installed.For enhanced security, implement email alerts for unauthorized SSH attempts. First, ensure **mailutils**is installed using the command: **sudo apt install mailutils**This solution improves your system's security posture by providing timely notifications and valuable insights into potential attack vectors.



```
┌──(kali㉿kali)-[~]
└─$ sudo apt install mailutils
mailutils is already the newest version (1:3.18-1).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1549
```

2.Update the script to send an email on detecting failed login attempts:

# Detect failed SSH login attempts and send email alert

echo "Checking failed SSH login attempts..."

failed_logins=$(grep "Failed password" $auth_log)

if [! -z "$failed_logins"]; then

echo -e "Subject: Unauthorized SSH Login Attempts\n\n$failed_logins" | sendmail your_email@example.com

echo "Security alert sent: Unauthorized SSH login attempt detected." fi