

Proof of Concept (PoC) Report

Task 5: Automating Security Audits with Scripting

1. Overview

This report illustrates how a Bash script can be leveraged to automate security assessments, focusing on login activity tracking, active service status, and storage capacity monitoring. The script is designed to detect security weaknesses (such as outdated services or excessive open connections) and suggest preventive actions, including scheduled scans and real-time alerts.

2. Goals

Configuration: Develop a Bash script that will review:

- User authentication logs (`last`, `/var/log/auth.log`).
- Active system services (`systemctl`).
- Current disk space utilization (`df -h`).

Analysis: Identify vulnerabilities like outdated services or excessive resource consumption.

Remediation: Implement automation through `cron` and configure alert mechanisms.

3. Implementation

3.1 Bash Script (`security_audit.sh`)

This script compiles a security summary report.

```
#!/bin/bash
```

```
REPORT_FILE="security_audit_report.txt"
echo "Security Review Report - $(date)" > $REPORT_FILE
```

```
# Capture recent login records
echo -e "\n⇒ User Login History ==>" >> $REPORT_FILE
last -n 10 >> $REPORT_FILE # Displays last 10 logins
grep "sshd" /var/log/auth.log | tail -n 10 >> $REPORT_FILE # Last 10 SSH connection
attempts
```

```
# List currently running services
echo -e "\n⇒ Active Services ==>" >> $REPORT_FILE
systemctl list-units --type=service --state=running >> $REPORT_FILE
```

```
# Check disk usage
```

```
echo -e "\n⇒ Disk Utilization ==" >> $REPORT_FILE
df -h >> $REPORT_FILE
```

```
echo -e "\nAudit completed. Report saved to $REPORT_FILE"
```

A terminal window on a Kali Linux system showing the execution of a script named 'security_audit.sh'. The prompt is '(kali㉿kali)-[~]'. The user enters '\$ cat security_audit.sh'. The script content is displayed, including a header 'Security Audit Report', three sections for checking login attempts, running services, and disk usage, and a final separator line. The prompt returns to '(kali㉿kali)-[~]'.

```
(kali㉿kali)-[~]
$ cat security_audit.sh
#!/bin/bash

echo "===== Security Audit Report ====="

# 1. Check User Login Attempts
echo -e "\n[+] User Login Attempts:"
last -a | head -n 10

echo -e "\n[+] Failed SSH Login Attempts:"
journalctl -u sshd --no-pager | grep "Failed password" | tail -n 10

# 2. Detect Running Services
echo -e "\n[+] Active Running Services:"
systemctl list-units --type=service --state=running | awk '{print $1, $2, $3, $4}'

# 3. Monitor Disk Usage
echo -e "\n[+] Disk Usage Information:"
df -h

echo -e "\n===== "

(kali㉿kali)-[~]
$
```

4. Execution & Analysis

4.1 Running the Script

Execute the script to analyze system vulnerabilities:

(kali@kali)-[~]

\$ bash security_audit.sh

Security Audit Report

[+] User Login Attempts:

lightdm	tty8	Mon Mar 24 11:46 - 11:53	(00:06)	:1
lightdm	tty8	Mon Mar 24 11:27 - 11:30	(00:02)	:1
lightdm	tty8	Mon Mar 24 10:27 - 11:04	(00:37)	:1
kali	tty7	Mon Mar 24 10:21 - still logged in		:0
lightdm	tty7	Mon Mar 24 10:20 - 10:21	(00:00)	:0
lightdm	tty8	Mon Mar 24 02:11 - 02:36	(00:25)	:1
lightdm	tty8	Mon Mar 24 01:10 - 01:13	(00:03)	:1
lightdm	tty8	Mon Mar 24 00:44 - 00:47	(00:02)	:1
kali	tty7	Sun Mar 23 23:09 - 05:19	(06:10)	:0
lightdm	tty7	Sun Mar 23 23:08 - 23:09	(00:00)	:0

[+] Failed SSH Login Attempts:

[+] Active Running Services:

UNIT LOAD ACTIVE SUB

accounts-daemon.service loaded active running
apache2.service loaded active running
colord.service loaded active running
cron.service loaded active running
dbus.service loaded active running
fail2ban.service loaded active running
getty@tty1.service loaded active running
haveged.service loaded active running
lightdm.service loaded active running
ModemManager.service loaded active running
NetworkManager.service loaded active running
polkit.service loaded active running
rtkit-daemon.service loaded active running
ssh.service loaded active running
systemd-journald.service loaded active running
systemd-logind.service loaded active running
systemd-udevd.service loaded active running
udisks2.service loaded active running
upower.service loaded active running
user@1000.service loaded active running
virtualbox-guest-utils.service loaded active running

Legend: LOAD → Reflects

ACTIVE → The high-level

SUB → The low-level

21 loaded units listed.

```

Legend: LOAD → Reflects
ACTIVE → The high-level
SUB → The low-level

21 loaded units listed.

[+] Disk Usage Information:
Filesystem      Size  Used Avail Use% Mounted on
udev            926M    0  926M   0% /dev
tmpfs           198M  988K  197M   1% /run
/dev/sda1       79G   17G   59G  22% /
tmpfs           988M   4.0K  988M   1% /dev/shm
tmpfs           5.0M    0   5.0M   0% /run/lock
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-journald.service
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-udev-load-credentials.service
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-sysctl.service
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev-early.service
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev.service
tmpfs           988M  184K  987M   1% /tmp
tmpfs           1.0M    0   1.0M   0% /run/credentials/systemd-tmpfiles-setup.service
tmpfs           1.0M    0   1.0M   0% /run/credentials/getty@tty1.service
tmpfs           198M  120K  198M   1% /run/user/1000

```

```

(kali@kali)-[~]
$

```

4.2 Identified Issues

- **Login Records:** Unsecured SSH logs were detected.
- **Service Status:** Presence of unnecessary services (e.g., `apache2`).
- **Disk Utilization:** Partition space insights obtained.

5. Hardening Strategies

5.1 Scheduling Automated Checks

To execute security scans daily, schedule a `cron` job:

```

(kali@kali)-[~]
$ crontab -e
no crontab for kali - 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

```

Add the following line:

```
0 3 * * * /path/to/security_audit.sh # Runs the script every day at 3 AM
```

5.2 Disabling Unused Services

Shut down services that are not required, such as **apache2** and **ssh** (if remote access is unnecessary):

```
(kali㉿kali)-[~]
$ sudo systemctl stop ssh

(kali㉿kali)-[~]
$ sudo systemctl stop apache2

(kali㉿kali)-[~]
$
```

5.3 Enhancing SSH Security with Fail2Ban

To prevent brute-force attacks:

```
(kali㉿kali)-[~]
$ sudo systemctl start fail2ban

(kali㉿kali)-[~]
$ sudo systemctl enable fail2ban
Synchronizing state of fail2ban.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable fail2ban

(kali㉿kali)-[~]
$
```

6. Conclusion

The security audit script effectively highlights risks such as publicly accessible SSH logs and redundant active services. Countermeasures such as automated task scheduling (**cron**) and service hardening (**fail2ban**) bolster system security.