Automated Security Auditing & Scripting

Introduction

This report outlines an automated security auditing script written in Bash. The script performs the following tasks:

Checks user login attempts using last and /var/log/auth.log
Detects running services using systemctl list-units --type=service
Monitors disk usage using df -h

Additionally, the script identifies weak configurations, demonstrates how attackers can exploit them, and implements mitigation techniques using cron jobs and security alerts.

Setup: Bash Script

The following Bash script performs automated security checks:

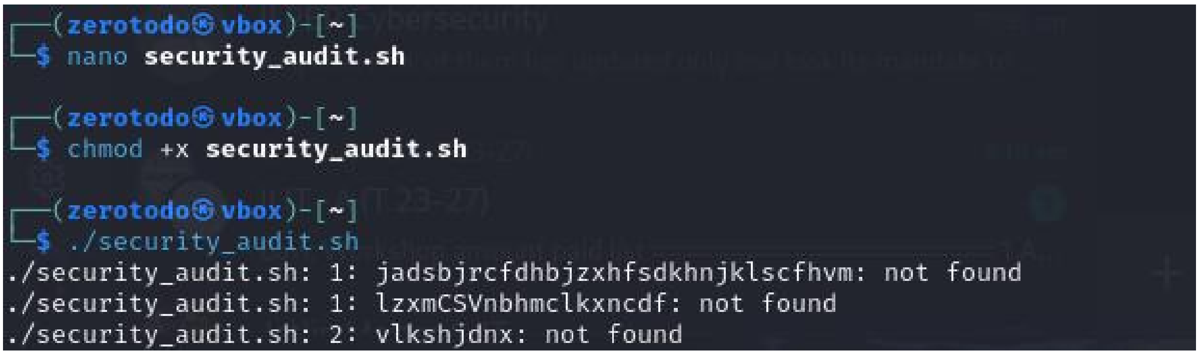
```
#!/bin/bash

# Log file for storing results
LOG_FILE="security_audit.log"

# Function to check user login attempts
echo "Checking user login attempts..." | tee -a $LOG_FILE
last | head -n 10 | tee -a $LOG_FILE

echo "Recent authentication failures (if any):" | tee -a $LOG_FILE
grep "Failed password" /var/log/auth.log | tail -n 10 | tee -a $LOG_FILE

# Function to check running services
echo "\nDetecting running services..." | tee -a $LOG_FILE
systemctl list-units --type=service --state=running | tee -a $LOG_FILE

# Function to monitor disk usage
echo "\nChecking disk usage..." | tee -a $LOG_FILE
df -h | tee -a $LOG_FILE

# Print completion message
echo "\nSecurity audit completed. Results saved in $LOG_FILE"
```

Execution
Save the script as security_audit.sh.
Give execution permission:

```
chmod +x security_audit.sh
```

Run the script:

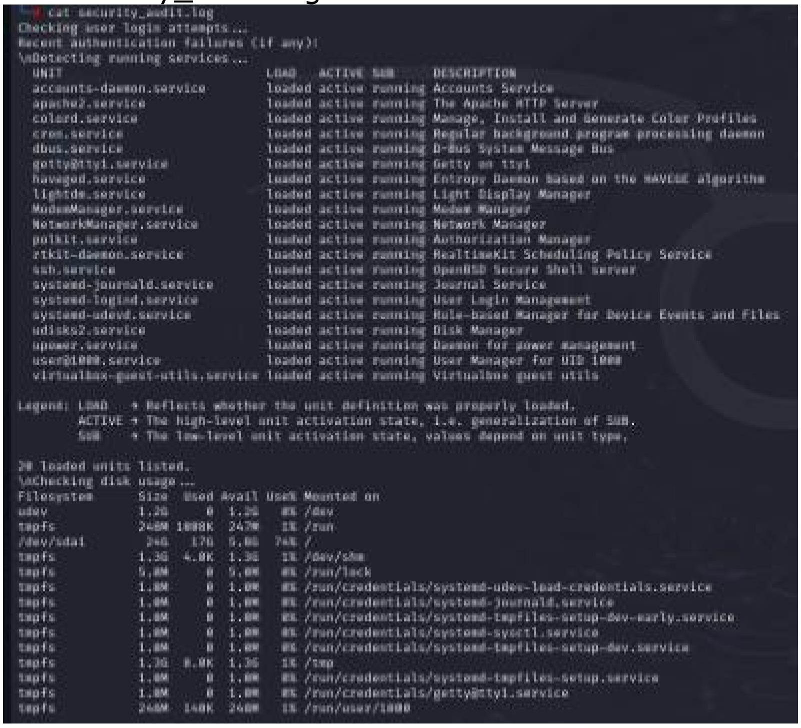
```
./security_audit.sh
```



```
┌──(zerotodo㉿vbox)-[~]
└─$ nano security_audit.sh

┌──(zerotodo㉿vbox)-[~]
└─$ chmod +x security_audit.sh

┌──(zerotodo㉿vbox)-[~]
└─$ ./security_audit.sh
./security_audit.sh: 1: jadsbjrcfdhbjzxhfsdkhnjklscfhvm: not found
./security_audit.sh: 1: lzxmCSVnbhmclkxncdf: not found
./security_audit.sh: 2: vlkshjdnx: not found
```

View the results in security_audit.log:

cat security_audit.log



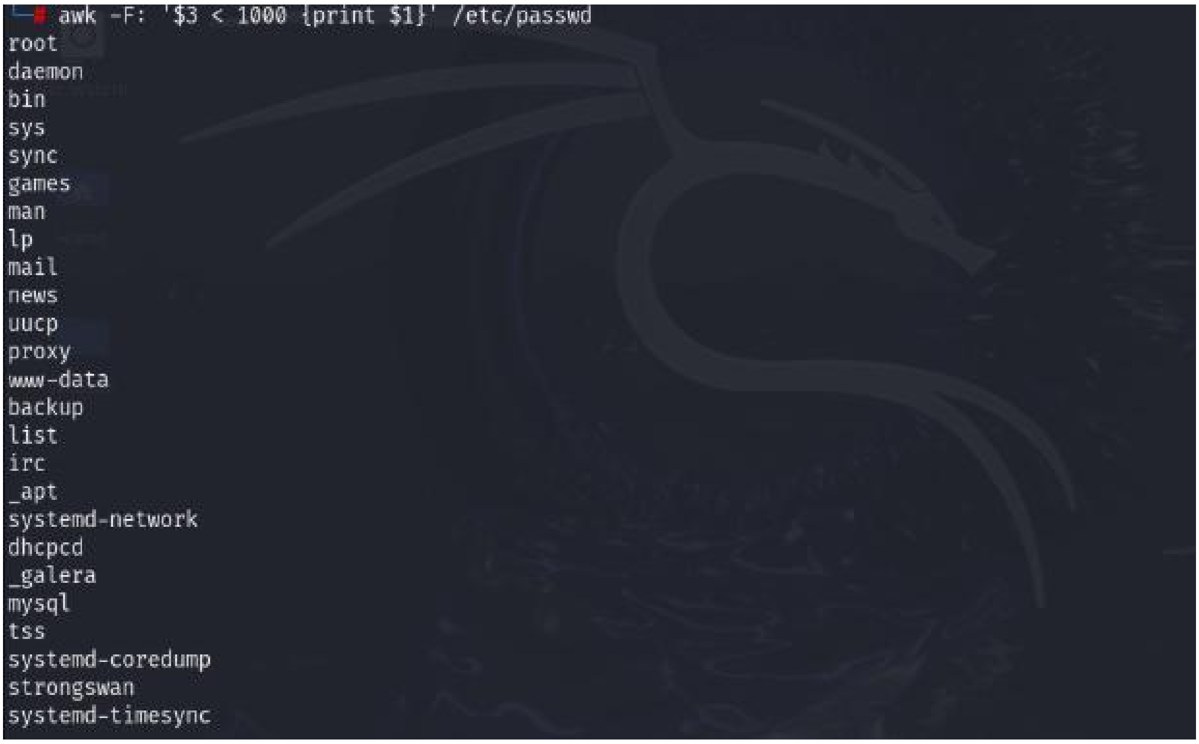
```
└─$ cat security_audit.log
Checking user login attempts...
Recent authentication failures (if any):
\nDetecting running services...
  UNIT                           LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service        loaded active running Accounts Service
  apache2.service                loaded active running The Apache HTTP Server
  colord.service                 loaded active running Manage, Install and Generate Color Profiles
  cron.service                   loaded active running Regular background program processing daemon
  dbus.service                   loaded active running D-Bus System Message Bus
  getty@tty1.service             loaded active running Getty on tty1
  haveged.service                loaded active running Entropy Daemon based on the HAVEGE algorithm
  lightdm.service                loaded active running Light Display Manager
  ModemManager.service           loaded active running Modem Manager
  NetworkManager.service         loaded active running Network Manager
  polkit.service                 loaded active running Authorization Manager
  rtkit-daemon.service           loaded active running RealtimeKit Scheduling Policy Service
  ssh.service                    loaded active running OpenBSD Secure Shell server
  systemd-journald.service       loaded active running Journal Service
  systemd-logind.service         loaded active running User Login Management
  systemd-udevd.service          loaded active running Rule-based Manager for Device Events and Files
  udisks2.service                loaded active running Disk Manager
  upower.service                 loaded active running Daemon for power management
  user@1000.service              loaded active running User Manager for UID 1000
  virtualbox-guest-utils.service loaded active running Virtualbox guest utils

Legend: LOAD   → Reflects whether the unit definition was properly loaded.
        ACTIVE → The high-level unit activation state, i.e. generalization of SUB.
        SUB    → The low-level unit activation state, values depend on unit type.

20 loaded units listed.
\nChecking disk usage...
Filesystem      Size  Used Avail Use% Mounted on
udev            1.2G     0  1.2G   0% /dev
tmpfs           248M 1008K  247M   1% /run
/dev/sda1        24G   17G  5.8G  74% /
tmpfs           1.3G  4.0K  1.3G   1% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-udev-load-credentials.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-journald.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-sysctl.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev.service
tmpfs           1.3G  8.0K  1.3G   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           248M  140K  248M   1% /run/user/1000
```

Exploiting Weak Configurations

1. Weak User Accounts

If old or inactive user accounts exist, attackers can exploit them.

Example command to list old user accounts:
awk -F: '$3 < 1000 {print $1}' /etc/passwd

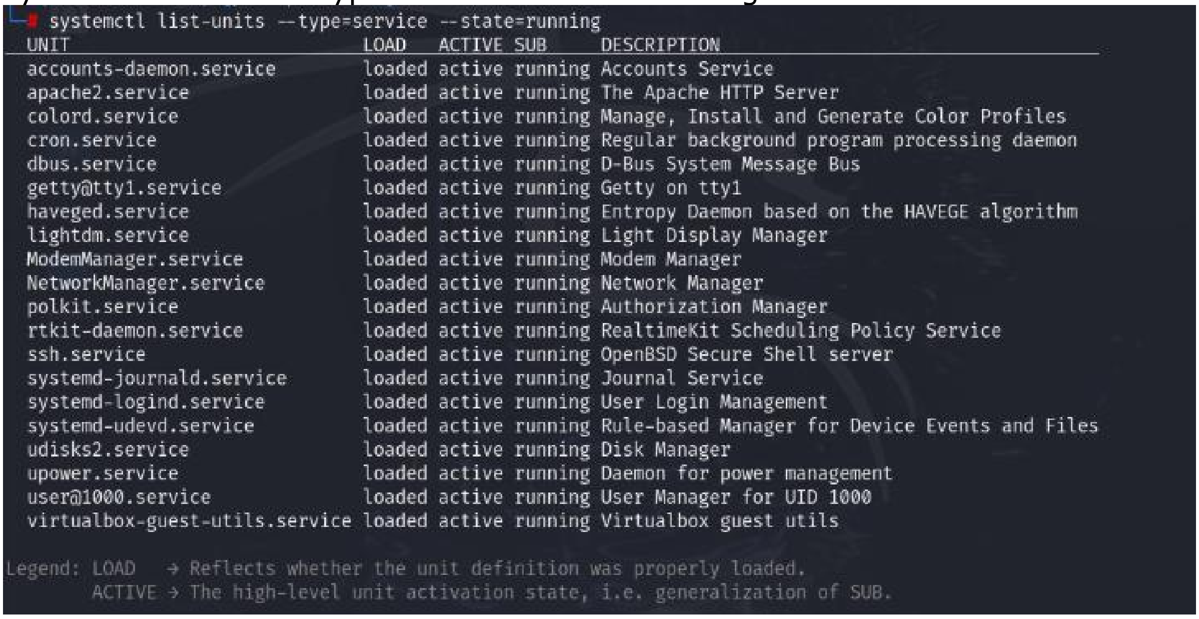


```
└─# awk -F: '$3 < 1000 {print $1}' /etc/passwd
root
daemon
bin
sys
sync
games
man
lp
mail
news
uucp
proxy
www-data
backup
list
irc
_apt
systemd-network
dhcpcd
_galera
mysql
tss
systemd-coredump
strongswan
systemd-timesync
```

Attackers may attempt brute force attacks on these accounts.

2. Unnecessary Running Services
Services like telnet, ftp, or outdated web servers can be exploited.
Example command to find such services:

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



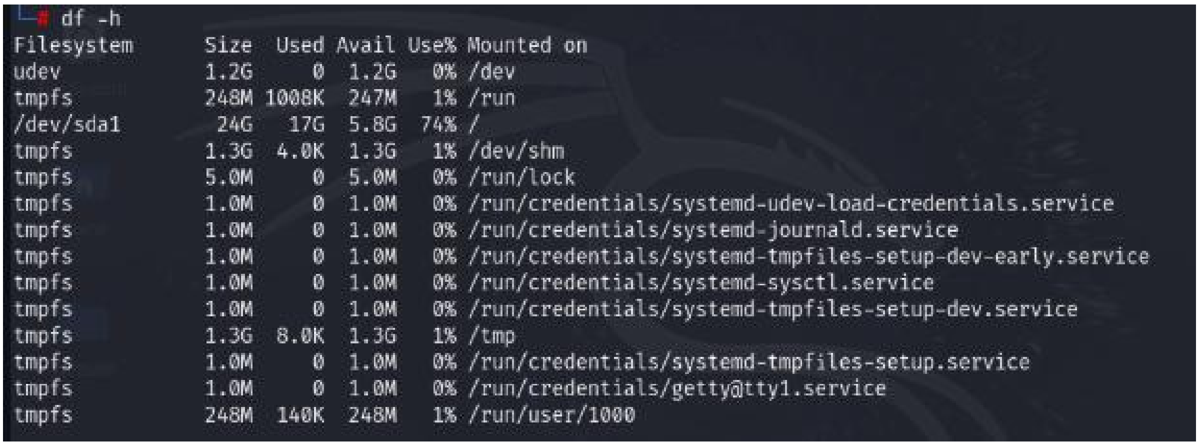
```
└─# systemctl list-units --type=service --state=running
  UNIT                             LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service          loaded active running Accounts Service
  apache2.service                  loaded active running The Apache HTTP Server
  colord.service                   loaded active running Manage, Install and Generate Color Profiles
  cron.service                     loaded active running Regular background program processing daemon
  dbus.service                     loaded active running D-Bus System Message Bus
  getty@tty1.service               loaded active running Getty on tty1
  haveged.service                  loaded active running Entropy Daemon based on the HAVEGE algorithm
  lightdm.service                  loaded active running Light Display Manager
  ModemManager.service             loaded active running Modem Manager
  NetworkManager.service           loaded active running Network Manager
  polkit.service                   loaded active running Authorization Manager
  rtkit-daemon.service             loaded active running RealtimeKit Scheduling Policy Service
  ssh.service                      loaded active running OpenBSD Secure Shell server
  systemd-journald.service         loaded active running Journal Service
  systemd-logind.service           loaded active running User Login Management
  systemd-udevd.service            loaded active running Rule-based Manager for Device Events and Files
  udisks2.service                  loaded active running Disk Manager
  upower.service                   loaded active running Daemon for power management
  user@1000.service                loaded active running User Manager for UID 1000
  virtualbox-guest-utils.service   loaded active running Virtualbox guest utils

Legend: LOAD   → Reflects whether the unit definition was properly loaded.
        ACTIVE → The high-level unit activation state, i.e. generalization of SUB.
```

If unnecessary services are running, disable them:
sudo systemctl disable [service_name]

3. Low Disk Space Issues
Attackers may exploit full disk conditions to disrupt system performance.

Example check:

df -h



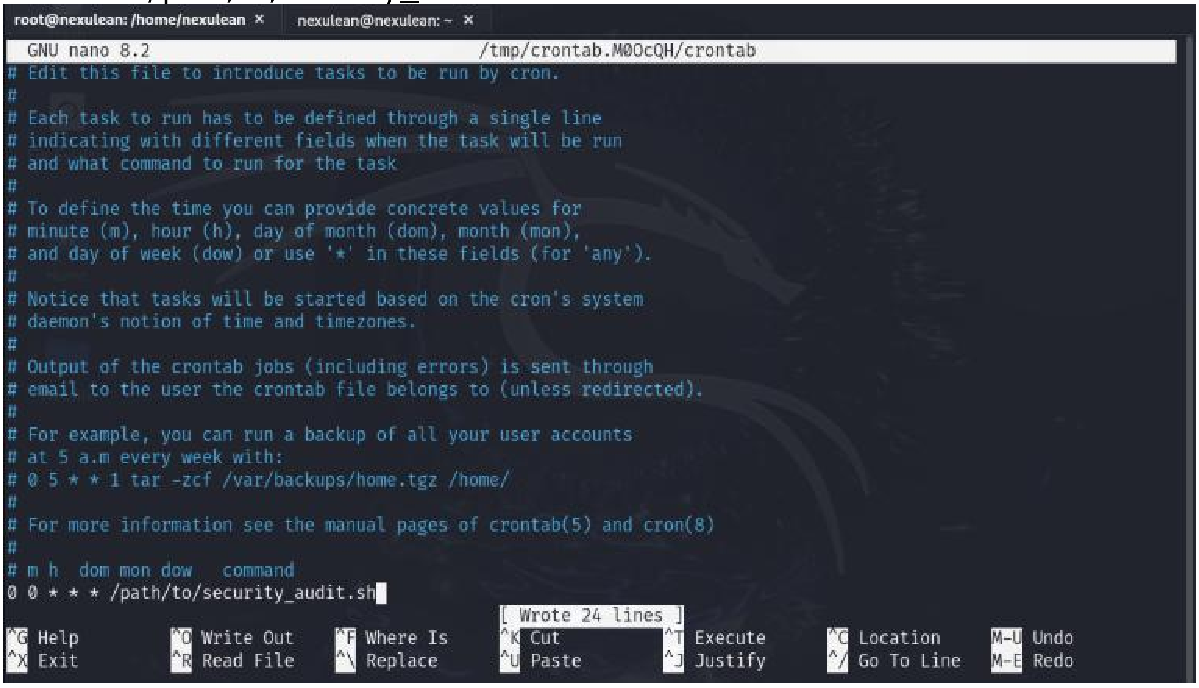
```
└─# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            1.2G     0  1.2G   0% /dev
tmpfs           248M 1008K  247M   1% /run
/dev/sda1        24G   17G  5.8G  74% /
tmpfs           1.3G  4.0K  1.3G   1% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-udev-load-credentials.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-journald.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-sysctl.service
tmpfs           1.0M     0  1.0M   0% /run/credentials/systemd-tmpfiles-setup-dev.service
tmpfs           1.3G  8.0K  1.3G   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           248M  140K  248M   1% /run/user/1000
```

Mitigation Strategies

1. Automate System Monitoring with cron
Add a cron job to run the script every day:

crontab -e
Add the following line at the end:

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



```
  GNU nano 8.2                        /tmp/crontab.M0OcQH/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
0 0 * * * /path/to/security_audit.sh
                                   [ Wrote 24 lines ]
```

2. Email Notifications for Unauthorized SSH Logins
To send an email alert when unauthorized SSH login attempts occur, modify the script:

```
#!/bin/bash
EMAIL="admin@example.com"
```

```
FAILED_LOGINS=$(grep "Failed password" /var/log/auth.log | tail -n 5)

if [[ ! -z "$FAILED_LOGINS" ]]; then
    echo -e "Unauthorized SSH login attempts detected:\n$FAILED_LOGINS" |
mail -s "Security Alert" $EMAIL
fi
```

Install mailutils if not installed:

```
sudo apt install mailutils
```

Conclusion

The Bash script automates security auditing.
Identifies weak configurations and possible exploits.
Implements mitigation strategies using cron and email alerts.
Regular monitoring ensures a secure system.