

Privilege Escalation in Linux/Unix

What is SetUID ?

프로그램을 실행할 때 일시적으로 권한이 상승하게 된다.

❖ Account example

- Login allocated by **UID to 500** and **GID to 500**. Identify wishfree account using UID and GID.

```
wishfree : x : 500 : 500 : ydi : /home/wishfree : /bin/bash
```

- ❖ UID, GID identifying who is an account is called **RUID**(Real UID), **RGID**(Real GID)
- ❖ UID, GID is present separately for **do you have any rights**. This is **EUID**(Effective UID), **EGID**(Effective GID).
- ❖ When first logged in, RUID and EUID, RGID and EGID each have the same value.
- ❖ Above condition doesn't match when you run the program with a **SetUID bit**

RUID (Real UID) : 실제 UID

RGID (Real GID) : 실제 G(Group)ID

EUID (Effective UID) : 순간적으로 유효한 다른 권한 (ID)

→ 일시적인 명령어 사용이나 파일접근 때문에 root 권한이 필요할 때가 존재함. 이때 잠깐 root 권한을 갖게 되는 계정이 EUID라고 함.

EGID (Effective GID) :

이렇게 권한에 관해 기능을 사용할 수 있는 것이 SetUID

SetUID 를 사용할 수 있는 것또한 파일 권한에 나타나 있는데, 이 경우 x (Execute) 가 아닌 s (SetUID) 로 나타나 있다.

```
wishfree@localhost:~
File Edit View Search Terminal Help
[wishfree@localhost ~]$ ls -al /usr/bin/passwd
-rwsr-xr-x. 1 root root 28416 Jul 16 06:46 /usr/bin/passwd
[wishfree@localhost ~]$
```

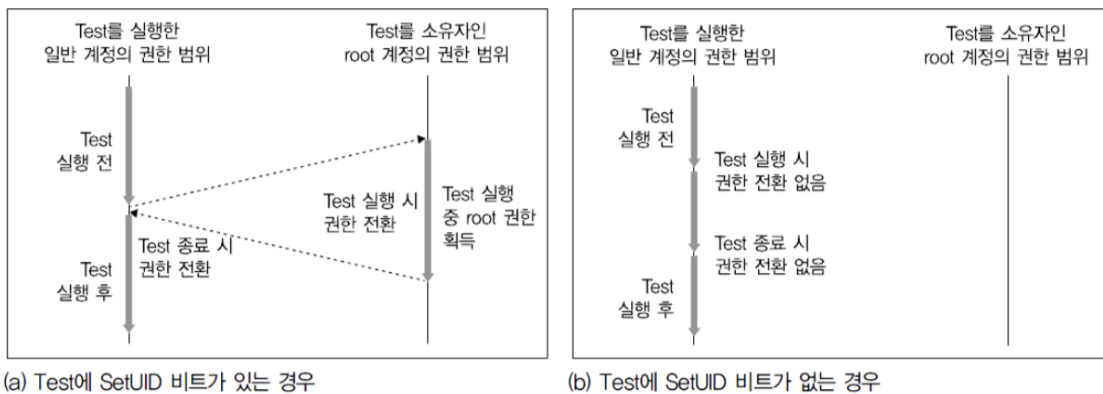
[Fig. 3-11] Checks the permission of /usr/bin/passwd

```
ls -al /usr/bin/passwd
```

- Identify the privilege escalation by **rws** r-x r-x. **s** among permissions indicates **SetUID**.
- SetUID, SetGID is represented 4000, 2000.
- If the file having **4755**, it is represented to rwsr-xr-x.
- Owner x location is **s**(Group x location is changed s in case of SetGID)
- The permission of /usr/bin/passwd is 4755(rws r-x r-x)

이 s 는 4000 을 의미한다. SetUID 의 경우는 4000 그리고 SetGID 는 2000을 의미한다

- SetUID permission is given to **passwd** file. Because file owner is root, the process running the file is executed in **root privilege** during execution



[그림 3-12] SetUID 설정에 따른 프로세스 권한 변경

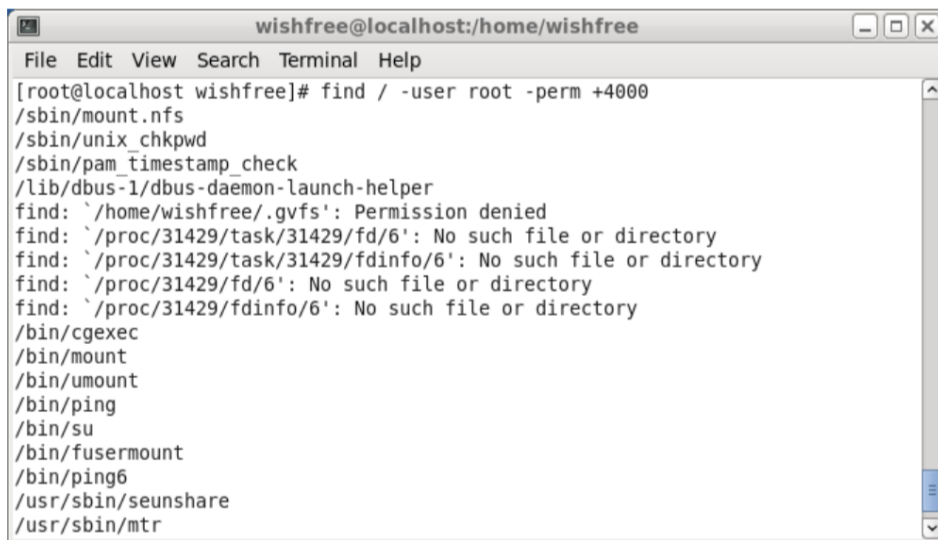
앞서 말했던 것 처럼, 잠깐 권한을 얻게 되는 것.

해커들은 이러한 S 비트 , SetUID 에 대해서 관심이 많다.

→ 권한상승에 대한 취약점이 발생할 수 있는 부분이기 때문에.

- Following *find* command searches the files having a SetUID bit and root owner

```
find / -user root -perm +4000
```



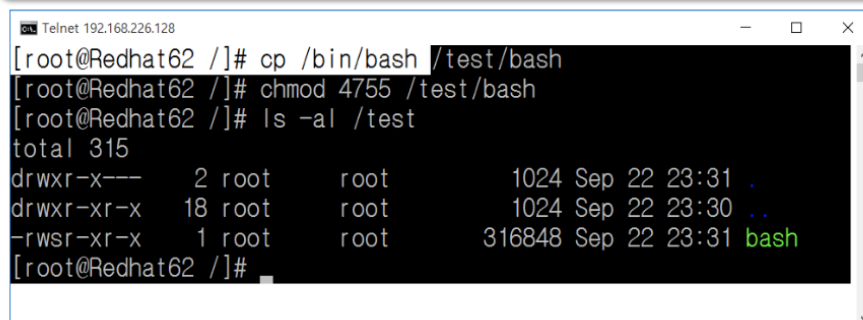
```
wishfree@localhost:/home/wishfree
File Edit View Search Terminal Help
[root@localhost wishfree]# find / -user root -perm +4000
/sbin/mount.nfs
/sbin/unix_chkpwd
/sbin/pam_timestamp_check
/lib/dbus-1/dbus-daemon-launch-helper
find: `/home/wishfree/.gvfs': Permission denied
find: `/proc/31429/task/31429/fd/6': No such file or directory
find: `/proc/31429/task/31429/fdinfo/6': No such file or directory
find: `/proc/31429/fd/6': No such file or directory
find: `/proc/31429/fdinfo/6': No such file or directory
/bin/cgexec
/bin/mount
/bin/umount
/bin/ping
/bin/su
/bin/fusermount
/bin/ping6
/usr/sbin/seunshare
/usr/sbin/mtr
```

[Fig. 3-13] Searching files having a SetUID bit in the system

그런 결과로 명령어 중에 permission 이 s 비트가 존재하는 파일을 볼 수 있다.

Example

```
cp /bin/bash /test/bash
chmod 4755 /test/bash
```



```
Telnet 192.168.226.128
[root@Redhat62 /]# cp /bin/bash /test/bash
[root@Redhat62 /]# chmod 4755 /test/bash
[root@Redhat62 /]# ls -al /test
total 315
drwxr-x---  2 root    root      1024 Sep 22 23:31 .
drwxr-xr-x 18 root    root      1024 Sep 22 23:30 ..
-rwsr-xr-x  1 root    root    316848 Sep 22 23:31 bash
[root@Redhat62 /]#
```

[Figure] Setting a SetUID bit of bash shell

root 권한으로 bash 를 다른 곳에 copy 하고 , 4755 로 바꾸면 setUID 가 설정된다. 따라서 이것을 실행한다면 이 쉘 (bash) 를 사용할 때 root 권한으로 사용할 수 있게 된다.