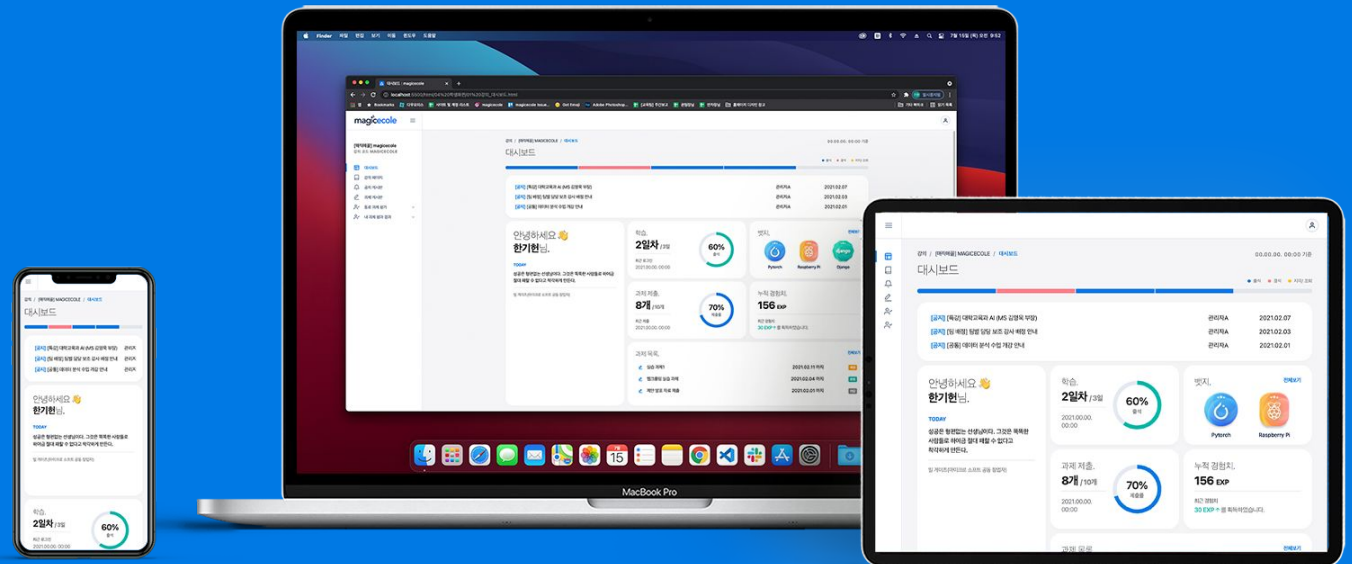


DevOps - Linux

# 사용자 관리와 파일 권한



# 다룰 내용

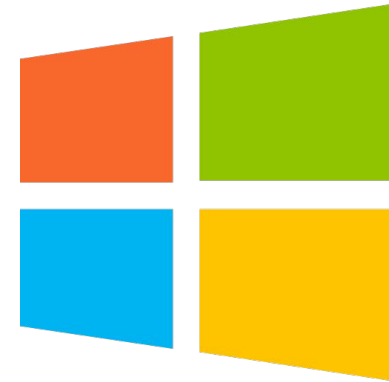
- 리눅스의 사용자와 그룹
- 파일 소유 및 접근 권한

# 리눅스의 사용자와 그룹

# 리눅스 vs 윈도우



VS



CLI

전문가

오픈소스

게임...ㅠ ㅠ

GUI

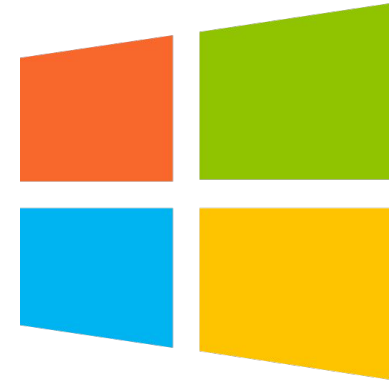
일반인

상용

게임!!



VS



For

**Multiple Users**



**Server**

For

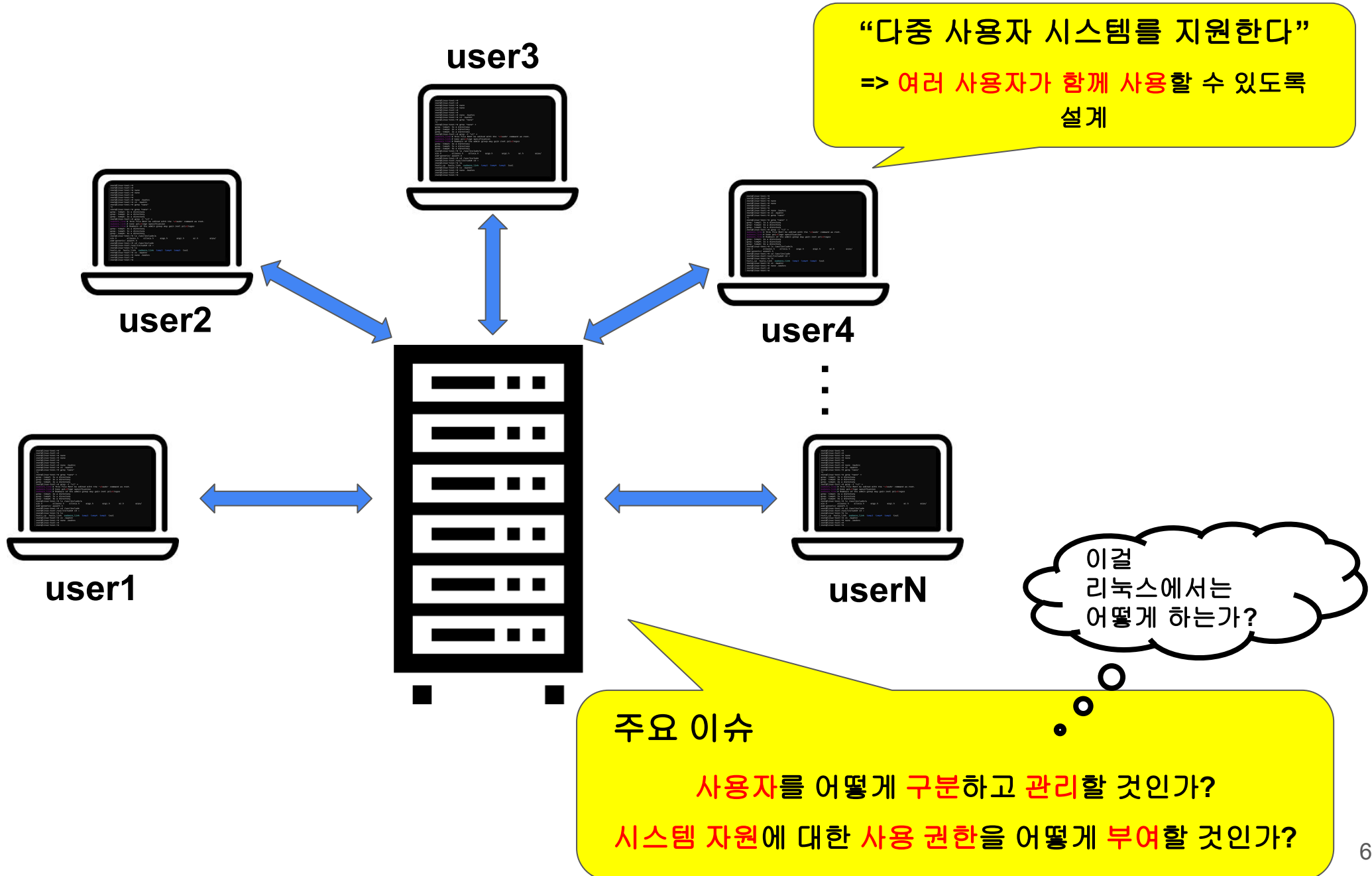
(Mostly) **Single User**



**P**ersonal **C**omputer

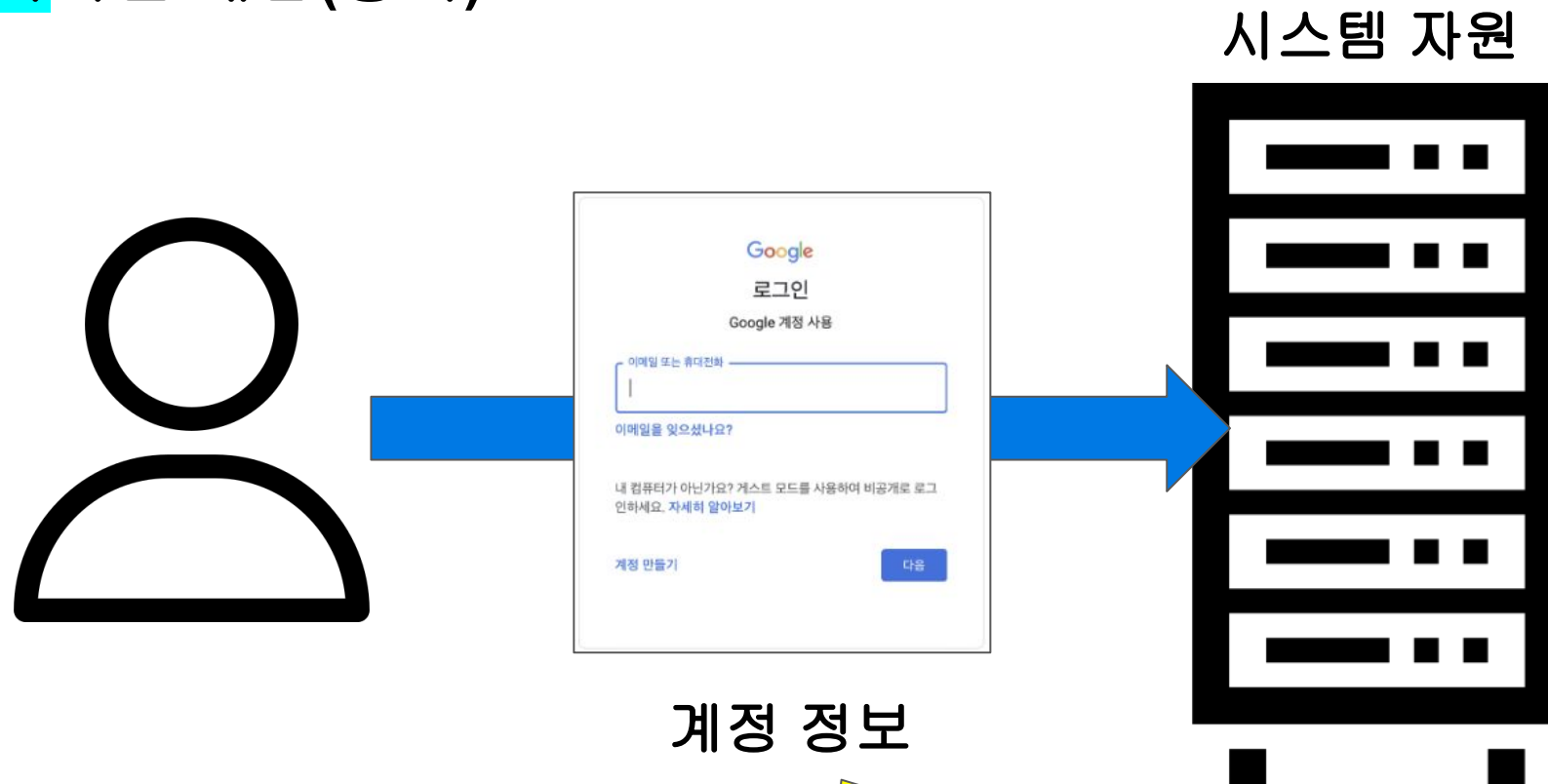
“윈도우 PC”와 “리눅스 서버”라는  
용어가 친숙한 이유

# 다중 사용자 시스템 (Multi-User System)



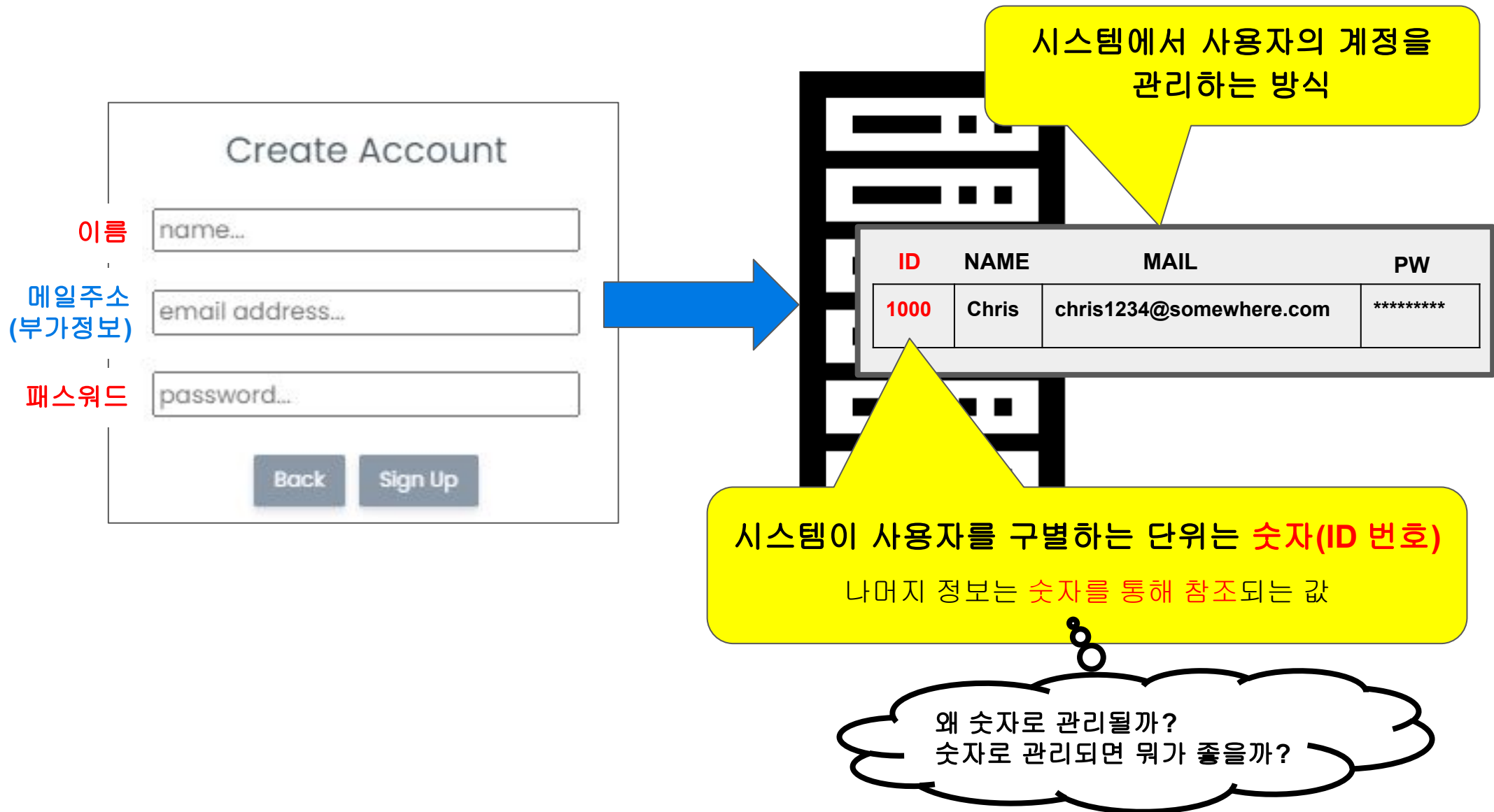
# 시스템의 사용자(User)

## 사용자라는 개념(정의)



어떤 시스템에 대한 **계정** 정보를 갖고 있으며,  
이를 이용해 해당 **시스템의 자원**에 접근 / 활용할 수 있는 **존재**

# 시스템의 사용자(User) - 계정





# 리눅스의 사용자(User)

리눅스 시스템에 대한 **계정** 정보를 갖고 있으며,  
이를 이용해 해당 **시스템의 자원**에 접근 / 활용할 수 있는 **존재**

숫자(ID 번호)를 통해 구별  
**존재**를 사람으로만 제한하지 않음

# 리눅스의 사용자(User)

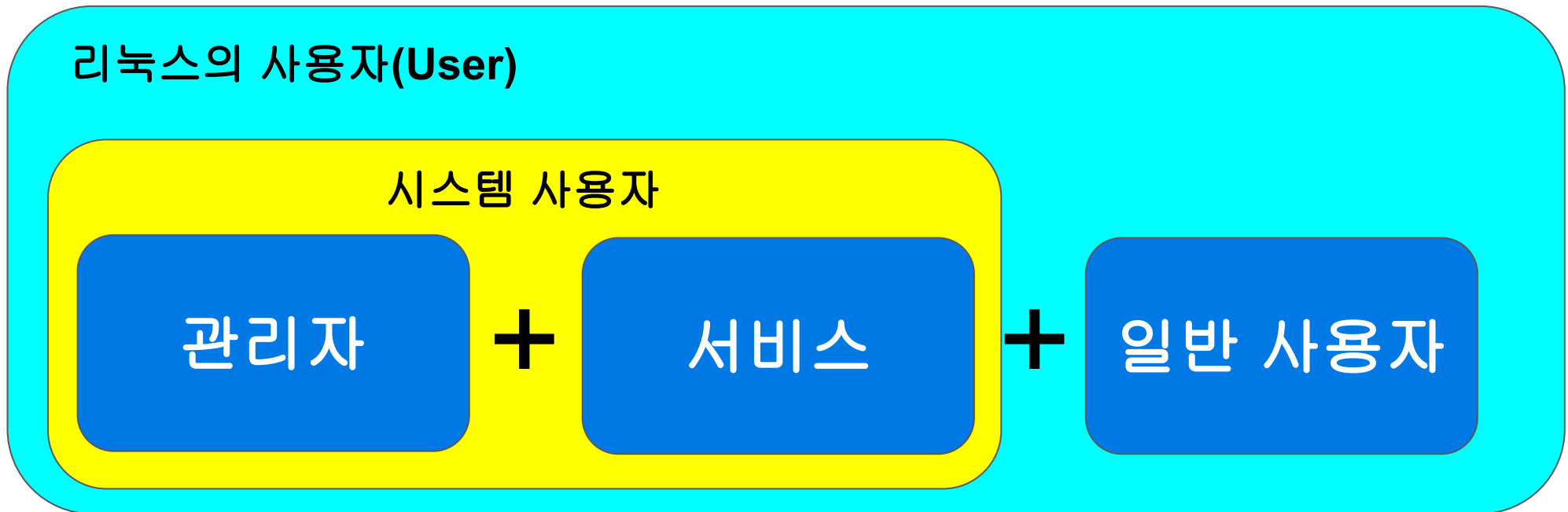
사용자 계정 정보 확인 : `cat /etc/passwd`

파일명만 보면 패스워드를 저장하고 있을 것 같다  
실제 패스워드는 `/etc/shadow`에 암호화되어 저장

```
root@linux-test:~# cat /etc/passwd
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
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd:/bin/false
uidd:x:106:110::/run/uidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1::/var/cache/pollinate:/bin/false
_chrony:x:110:115:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
sshd:x:111:65534::/run/sshd:/usr/sbin/nologin
nbpmon:x:1000:1000:agent-account:/home1/nbpmon:/bin/bash
user:x:1001:1001:,,,:/home/user:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
user3:x:1003:1003:,,,:/home/user3:/bin/bash
user4:x:1004:1004:,,,:/home/user4:/bin/bash
user5:x:1005:1005:,,,:/home/user5:/bin/bash
```

사용자가 생각보다 많다?

# 리눅스의 사용자(User)



# 리눅스의 사용자(User)

사용자 계정 정보 확인 : `cat /etc/passwd`

root 계정 : 리눅스 시스템에서 무엇이든 할 수 있는 계정

root (관리자) 계정

서비스 계정

시스템 사용자 계정

```
root@linux-test:~# cat /etc/passwd
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
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/network:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106:syslog:/home/syslog:/usr/sbin/nologin
messagebus:x:103:107:Message Bus,,,:/nonexistent:/usr/sbin/nologin
_apt:x:104:65534:apt:/nonexistent:/usr/sbin/nologin
lxd:x:105:65534:lxd:/var/lib/lxd:/bin/false
uidd:x:106:110:uid:/run/uidd:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112:landscape:/var/lib/landscape:/usr/sbin/nologin
pollinate:x:109:1:pollinate:/var/cache/pollinate:/bin/false
_chrony:x:110:115:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
sshd:x:111:65534:sshd:/run/sshd:/usr/sbin/nologin
nbpmon:x:1000:1000:agent-account:/home1/nbpmon:/bin/bash
user:x:1001:1001:,,,:/home/user:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
user3:x:1003:1003:,,,:/home/user3:/bin/bash
user4:x:1004:1004:,,,:/home/user4:/bin/bash
user5:x:1005:1005:,,,:/home/user5:/bin/bash
```

리눅스 시스템에서 실행 중인 서비스(백그라운드 프로세스)를 위한 계정

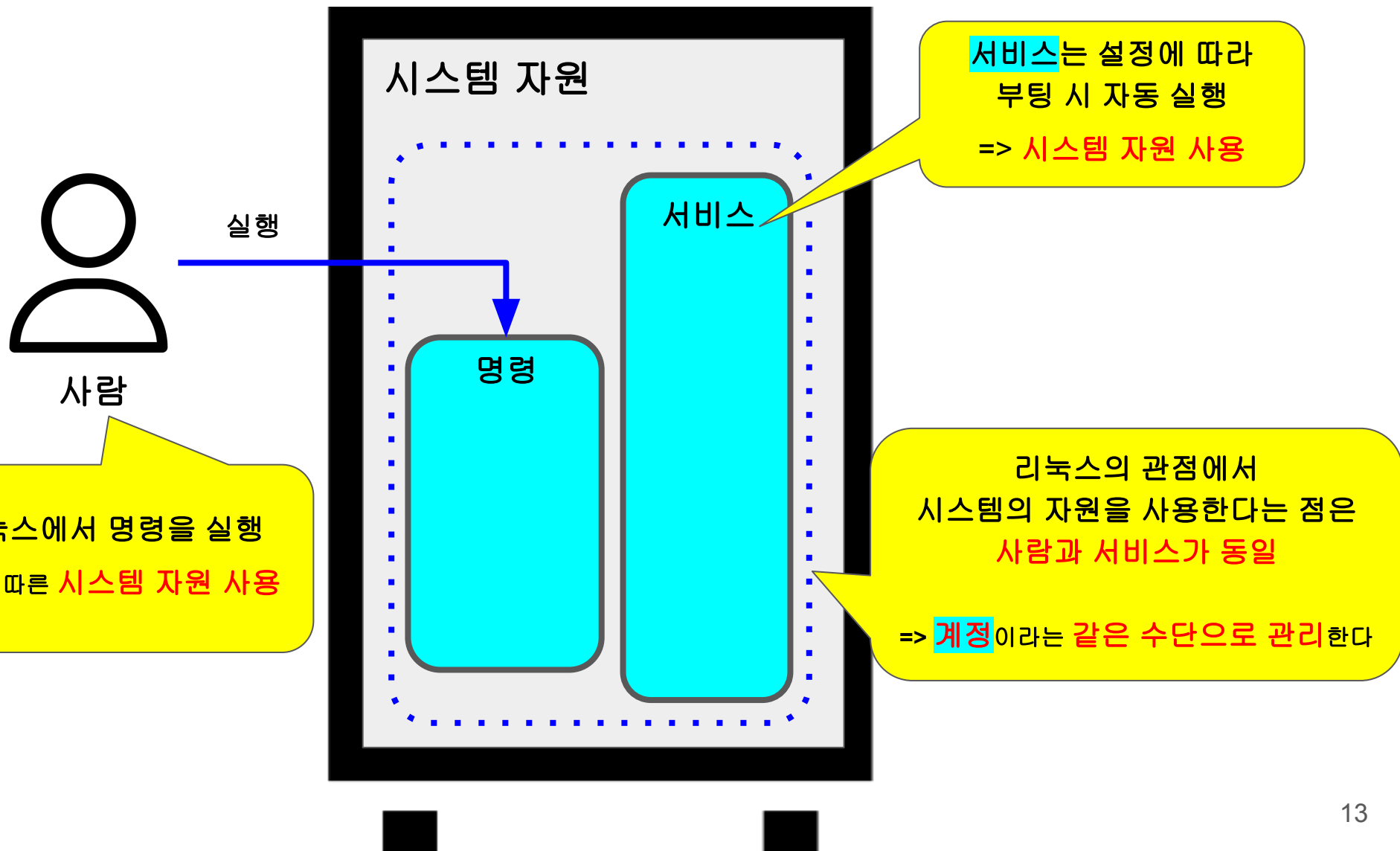
서비스 위한 계정?

일반 사용자 계정

외부에서 리눅스 시스템에 접속하는 어떤 것(사람)을 위한 계정

# 리눅스의 사용자(User)

리눅스가 보는 사람과 서비스



# 리눅스의 사용자(User)

## 사용자 계정 정보( /etc/passwd )

0~999, 65534 : 시스템 사용자  
1000 ~ : 일반 사용자

0 (root)	root 사용자 계정
1 (daemon)	시스템 데몬 계정
2 (bin)	명령어 관리를 위한 계정
65534 (nobody)	의사(pseudo) 사용자 계정

서비스  
계정

로그인 ID (사용자 ID)    사용자 ID 번호 (UID)    설명

```
sshd:x:111:65534:::/run/sshd:/usr/sbin/nologin
```

어디서 많이  
보던 거다?  
(Appendix  
참고)

사용자 암호  
(실제 데이터는 별도 저장)

기본 그룹 ID 번호  
(GID)

홈 디렉토리  
(절대경로)

사용자의 로그인 셸

사용자가 기본으로(1차) 속하는 그룹의 ID

# 리눅스의 사용자(User)

## 일반 사용자 계정

```
nbpmon:x:1000:1000:agent-account:/home1/nbpmon:/bin/bash
user:x:1001:1001:,,,:/home/user:/bin/bash
user2:x:1002:1002:,,,:/home/user2:/bin/bash
```

user, user2랑은 좀 다르다?

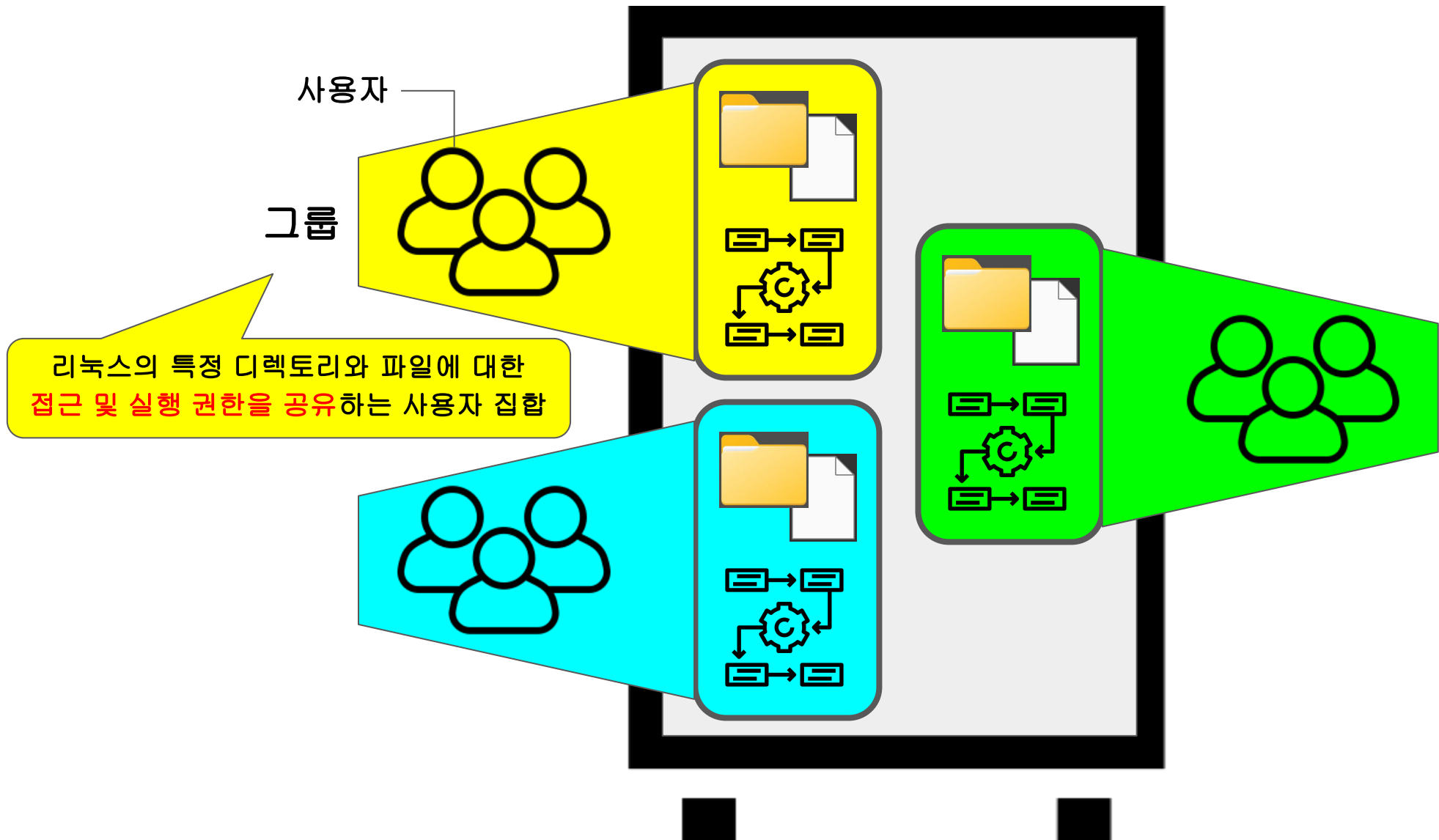
UID가 1000 이상

사용자 홈 디렉토리 경로

bash가 로그인 셸

경로에 패턴이 보인다

# 리눅스의 그룹(Group)





# 리눅스의 그룹(Group)

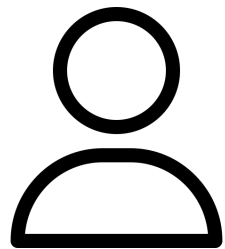
기본 그룹 (1차 그룹)

모든 사용자는 무조건 **1개 또는 그 이상의 그룹**에 소속된다

기본 그룹 : **무조건** 소속되어야 하는 그룹

보조 그룹 : **필요에 따라** 소속되는 그룹

기본 그룹에 속하는 방법



사용자 계정

(1)

존재하는  
그룹에 **소속**

사용자 계정 생성 시  
계정 정보 수정 시

이미 **만들어져** 있는 그룹이어야 한다

(2)

사용자 계정  
**자체 그룹**

사용자 계정 생성 시

소속될 그룹을 명시하지 않을 경우  
**사용자 계정으로 그룹**을 만든다

내가 곧 그룹이다!

# 리눅스의 그룹(Group)

## 기본 그룹 확인

기본 그룹 ID  
(GID)

UID와 GID가 같다  
=> 사용자 계정에 대한 자체  
그룹

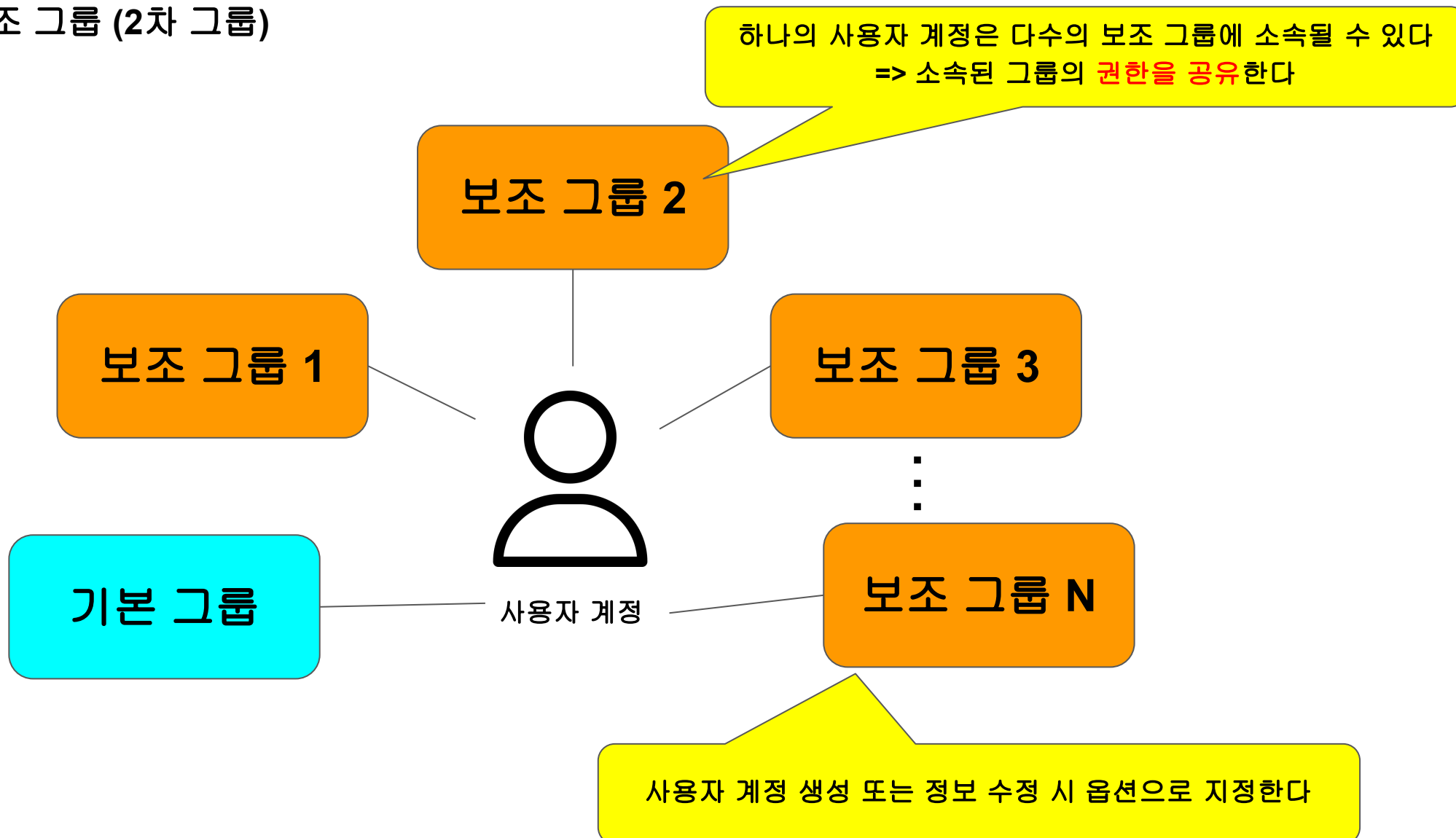
```
nbpmon:x:1000:1000:agent-account:/home1/nbpmon:/bin/bash
```

```
sshd:x:111:65534: :/run/sshd:/usr/sbin/nologin
```

GID가 65534(nobody)인 그룹에  
기본으로 소속된다

# 리눅스의 그룹(Group)

보조 그룹 (2차 그룹)



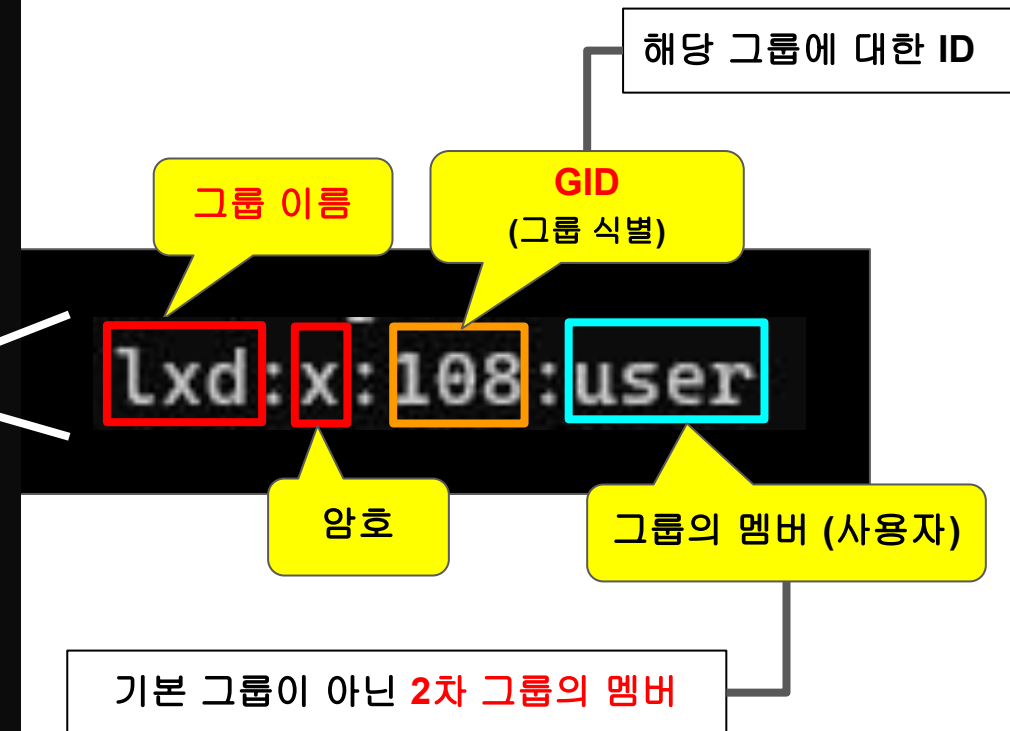
# 리눅스의 그룹(Group)

그룹 정보 확인 : `cat /etc/group`

```

root@linux-test:~# cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog
tty:x:5:
disk:x:6:
mail:x:8:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:
floppy:x:25:
tape:x:26:
sudo:x:27:user
audio:x:29:
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:
sasl:x:45:
plugdev:x:46:
staff:x:50:
users:x:100:
nogroup:x:65534:
systemd-journal:x:101:
systemd-network:x:102:
systemd-resolve:x:103:
input:x:104:
crontab:x:105:
syslog:x:106:
messagebus:x:107:
lxd:x:108:user
mlocate:x:109:
uucidd:x:110:
ssh:x:111:
landscape:x:112:
lpadmin:x:113:
smbshare:x:114:
_chrony:x:115:
netdev:x:116:
nbpmon:x:1000:
user:x:1001:
user2:x:1002:
user3:x:1003:
user4:x:1004:
user5:x:1005:
user6:x:1006:

```



# 사용자 계정 생성

## ● adduser (우분투 권장)

- **기능** : 사용자 계정을 생성
- **형식** : **adduser** [옵션] [사용자 ID]

1. 사용자 계정(user1) 생성 : **adduser user1**

```
root@linux-test:~# adduser user1
Adding user 'user1' ...
Adding new group 'user1' (1001) ...
Adding new user 'user1' (1001) with group 'user1' ...
Creating home directory '/home/user1' ...
Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
    Full Name []: test user
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n]
root@linux-test:~# cat /etc/passwd | grep user1
user1:x:1001:1001:test user,,,:/home/user1:/bin/bash
root@linux-test:~#
```

계정 생성 시 차례대로 진행

사용자 ID, UID, GID 생성

사용자 계정 홈 디렉토리 생성  
사용자 환경 설정 파일 복사

패스워드 입력

부가 정보(설명) 입력

user1 계정 생성

# 사용자 계정 생성

## • adduser (우분투 권장)

2. 옵션을 사용한 사용자 계정(user2) 생성 : **adduser -uid 2001 -gid 27 --home /home1/user2 user2**

UID를 특정 값(2001)으로 지정  
이미 생성된 경우 에러

GID를 특정 값(27)으로 지정  
-gid가 없을 경우 UID와 같은 값으로 지정

```
root@linux-test:~# adduser -uid 2001 -gid 27 --home /home1/user2 user2
```

```
Adding user 'user2' ...
Adding new user 'user2' (2001) with group 'sudo' ...
Creating home directory '/home1/user2' ...
Copying files from '/etc/skel' ...
```

sudo 그룹의 GID가 27  
=> user2의 기본 그룹을 sudo로 지정

```
user2:x:2001:27:::/home1/user2:/bin/bash
```

cat /etc/passwd | grep user2

홈 디렉토리 생성 및 설정 파일 복사 확인

```
root@linux-test:~# ls -al /home1/user2/
total 20
drwxr-xr-x 2 user2 sudo 4096 Mar 13 11:52 .
drwxr-xr-x 4 root  root 4096 Mar 13 11:52 ..
-rw-r--r-- 1 user2 sudo 220 Mar 13 11:52 .bash_logout
-rw-r--r-- 1 user2 sudo 3771 Mar 13 11:52 .bashrc
-rw-r--r-- 1 user2 sudo 807 Mar 13 11:52 .profile
root@linux-test:~#
```

# 사용자 계정 생성

adduser로 계정 생성 시 참조  
수정할 때는 신중하게!!

adduser의 참조 파일 : /etc/adduser.conf

```
root@linux-test:~# cat /etc/adduser.conf
# /etc/adduser.conf: 'adduser' configuration.
# See adduser(8) and adduser.conf(5) for full documentation.

# The DSHELL variable specifies the default login shell on your
# system.
DSHELL=/bin/bash

# The DHOME variable specifies the directory containing users' home
# directories.
DHOME=/home
```

기본 로그인 셸

사용자 홈 디렉토리의 기본 생성 경로

```
# The SKEL variable specifies the directory containing "skeletal" user
# files; in other words, files such as
# copied to the new user's home
SKEL=/etc/skel
```

기본적인 사용자의 환경 설정 파일을 복사해 오는 경로

```
root@linux-test:~# ls -al /etc/skel/
total 20
drwxr-xr-x  2 root root 4096 Feb 21 16:50 .
drwxr-xr-x 94 root root 4096 Mar 13 11:52 ..
-rw-r--r--  1 root root  220 Apr  5 2018 .bash_logout
-rw-r--r--  1 root root 3771 Apr  5 2018 .bashrc
-rw-r--r--  1 root root  807 Apr  5 2018 .profile
```

```
# FIRST_[GU]ID to LAST_[GU]ID inclusive is the range of UIDs of dynamically
# allocated user accounts/groups.
FIRST_UID=1000
LAST_UID=59999

FIRST_GID=1000
LAST_GID=59999
```

일반 사용자의 UID / GID 범위



# 사용자 계정 생성

## ● useradd

CentOS, Fedora 등에서는 기본적으로 useradd를 사용  
adduser 보다 다소 불편함

- **기능** : 사용자 계정을 생성
- **형식** : **useradd** [옵션] [사용자 ID]

1. 사용자 계정(user3) 생성 : **useradd user3**

```
root@linux-test:~# useradd user3
root@linux-test:~#
root@linux-test:~# cat /etc/passwd | grep user3
user3:x:2002:2002::/home/user3:/bin/sh
root@linux-test:~#
root@linux-test:~# ls /home/user3
ls: cannot access '/home/user3': No such file or directory
root@linux-test:~#
```

useradd 명령에서는 패스워드를 지정하지 않음  
=> passwd 명령으로 지정

로그인 셸이 sh (dash)

홈 디렉토리가 생성되지 않음



# 사용자 계정 생성

## • useradd 주요 옵션

옵션	설명	
-u	UID를 지정한다	<code>useradd -u 1001 myuser</code>
-g	기본 그룹의 GID를 지정한다	<code>useradd -g users myuser</code>
-G	보조 그룹을 지정한다	<code>useradd -G wheel,staff myuser</code>
-d	홈 디렉토리의 경로를 지정한다	<code>useradd -d /home/myuser myuser</code>
-m	홈 디렉토리를 생성한다	<code>useradd -m myuser</code>
-s	기본 셸(로그인 셸)을 지정한다	<code>useradd -s /bin/bash myuser</code>
-c	부가적인 설명을 지정한다	<code>useradd -c "My User Account" myuser</code>
-D	기본 값(Default)을 설정하거나 출력한다	
-k	skel 디렉토리(환경 설정 파일이 저장된 경로) 지정	<code>useradd -m -k /etc/skel_custom myuser</code>
-f 일수	패스워드 사용 기한이 지난 후 계정이 잠기기까지 (INACTIVE : 비활성화) 유효기간을 설정한다	
-e 날짜	사용자 계정이 만료되는 (EXPIRE) 날짜를 설정한다	

INACTIVE, EXPIRE에 대한 내용은  
/etc/shadow 관련 내용 참고

# 사용자 계정 생성

## • useradd

2. 옵션을 이용해 계정(user4) 생성하고 패스워드 설정 :

**useradd** **-u 2011** **-g 27** **-m -d /home/user4** **-s /bin/bash** **user4**  
**passwd** **user4**

UID 지정 (2011)

GID 지정 (27)

-m : 홈 디렉토리 생성  
 -d : 홈 디렉토리 경로 설정

로그인 셸 지정

```
root@linux-test:~# useradd -u 2011 -g 27 -m -d /home/user4 -s /bin/bash user4
```

```
root@linux-test:~# cat /etc/passwd | grep user4
```

```
user4:x:2011:27:/home/user4:/bin/bash
```

```
root@linux-test:~#
```

```
root@linux-test:~# ls -al /home/user4
```

```
total 20
drwxr-xr-x 2 user4 sudo 4096 Mar 13 16:52 .
drwxr-xr-x 4 root  root 4096 Mar 13 16:52 ..
-rw-r--r-- 1 user4 sudo  220 Apr  5  2018 .bash_logout
-rw-r--r-- 1 user4 sudo 3771 Apr  5  2018 .bashrc
-rw-r--r-- 1 user4 sudo  807 Apr  5  2018 .profile
```

홈 디렉토리 생성되고  
 기본 사용자 파일 복사 확인  
 ( from /etc/skel )

```
root@linux-test:~#
```

```
root@linux-test:~# passwd user4
```

```
New password:
```

```
Retype new password:
```

```
passwd: password updated successfully
```

```
root@linux-test:~#
```

user4의 패스워드 지정  
 (변경 시에도 사용)

# 사용자 계정 생성

## • useradd

3. 생성하는 계정에 대해 보조 그룹 지정 :

```
useradd -G sudo -m -d /home/user5 -s /bin/bash user5
passwd user5
```

계정 생성 시 UID, GID는  
명시하지 않음

UID는 정해진 기준에 따라 생성  
GID는 UID와 같은 값

```
root@linux-test:~# useradd -G sudo -m -d /home/user5 -s /bin/bash user5
root@linux-test:~# cat /etc/passwd | grep user5
user5:x:2012:2012::/home/user5:/bin/bash
root@linux-test:~# cat /etc/group | grep user5
sudo:x:27:user1,user5
user5:x:2012:
```

user5가 sudo 그룹에 멤버로 소속됨 (2차 그룹)

user5 계정에 대한 자체 그룹 생성

# 사용자 계정 생성

## • useradd

3. useradd의 기본 설정 값 확인하기 : **useradd -D**

```
root@linux-test:~# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/sh
SKEL=/etc/skel
CREATE_MAIL_SPOOL=no
```

useradd로 계정 생성 시 별도의 옵션이 없으면 적용 (GROUP값 제외)

**GROUP** : -N 옵션으로 계정 생성 시 GID에 적용 (100은 users 그룹을 의미)

**CREATE\_MAIL\_SPOOL** : 메일 디렉토리 생성 여부 지정

/etc/default/useradd 파일에도 같은 내용 저장

4. useradd에서 계정 생성 시 적용되는 기본 로그인 셸을 변경하기 : **useradd -D -s /bin/bash**

```
root@linux-test:~# useradd -D -s /bin/bash
root@linux-test:~# useradd -D
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
CREATE_MAIL_SPOOL=no
root@linux-test:~#
```

기본 로그인 셸을 /bin/sh에서 /bin/bash로 변경  
기본 값을 변경할 때는 신중하게!!!

# 사용자 계정 수정

## ● usermod

- **기능** : 사용자 계정 정보를 수정
- **형식** : **usermod** [옵션] [사용자 ID]

옵션	설명	useradd와 유사하다
-u	UID를 수정한다	
-g	기본 그룹을 수정한다	
-G	보조 그룹을 수정한다 (-a : append 옵션과 함께 사용하면 보조 그룹이 추가됨)	<code>usermod -aG groupname username</code>
-d	홈 디렉토리를 수정한다	<code>usermod -d /newhome username</code>
-s	기본 셸(로그인 셸)을 수정한다	<code>usermod -s /bin/bash username</code>
-c	부가적인 설명을 수정한다	
-f 일수	패스워드 사용 기한이 지난 후 계정이 잠기기까지(INACTIVE : 비활성화) 유효기간을 지정한다	
-e 날짜	사용자 계정이 만료되는(EXPIRE) 날짜를 수정한다	
-l	계정 이름을 바꾼다	<code>usermod -l newusername oldusername</code>

# 사용자 계정 수정

## • usermod

1. user1 계정의 UID 수정 : **usermod -u 1100 user1**

```
user1:x:1001:1001:test user,,,:/home/user1:/bin/bash
root@linux-test:~# usermod -u 1100 user1
root@linux-test:~# cat /etc/passwd | grep user1
user1:x:1100:1001:test user,,,:/home/user1:/bin/bash
```

user1의 UID가 1001에서 1100으로 수정

2. user1 계정의 보조 그룹 수정 : **usermod -G users, lpadmin user1**

```
root@linux-test:~# usermod -G users,lpadmin user1
root@linux-test:~#
root@linux-test:~# cat /etc/group | grep -e users -e lpadmin
users:x:100:user1
lpadmin:x:113:user1
root@linux-test:~#
```

user1 계정의 보조 그룹을  
users, lpadmin로 지정

grep의 -e 옵션 : 여러 문자열 동시 검색

lpadmin과 users 그룹에 user1이 소속



# 사용자 계정 삭제

## • userdel

- **기능** : 사용자 계정을 삭제
- **형식** : **userdel** [옵션] [사용자 ID]

1. user5 계정 삭제 : **userdel user5**

사용자 계정만 삭제

```
root@linux-test:~# cat /etc/passwd | grep user5
user5:x:2012:2012::/home/user5:/bin/bash
```

user5 계정 확인

```
root@linux-test:~#
```

```
root@linux-test:~# ls /home
```

```
user1 user4 user5
```

user5 홈 디렉토리 확인

```
root@linux-test:~#
```

```
root@linux-test:~# userdel user5
```

user5 계정 삭제

```
root@linux-test:~#
```

```
root@linux-test:~# cat /etc/passwd | grep user5
```

계정 삭제 확인

```
root@linux-test:~# ls /home
```

```
user1 user4 user5
```

```
root@linux-test:~#
```

계정의 홈 디렉토리는 삭제되지 않음

# 사용자 계정 삭제

- userdel

2. user4 계정을 홈 디렉토리 포함하여 삭제 : **userdel -r user4**

```
root@linux-test:~# cat /etc/passwd | grep user4
user4:x:2011:27:~/home/user4:/bin/bash
```

user4 계정 확인

```
root@linux-test:~#
```

```
root@linux-test:~# ls /home
user1 user4
```

user4 홈 디렉토리 확인

```
root@linux-test:~#
```

```
root@linux-test:~# userdel -r user4
```

user4 계정 삭제

```
userdel: user4 mail spool (/var/mail/user4) not found
```

```
root@linux-test:~#
```

```
root@linux-test:~# cat /etc/passwd | grep user4
```

계정 삭제 확인

```
root@linux-test:~# ls /home
```

```
user1
```

```
root@linux-test:~#
```

user4의 홈 디렉토리도 삭제 확인



# 사용자 계정 삭제

## • deluser

- **기능** : 사용자 계정을 삭제
- **형식** : **deluser** [옵션] [사용자 ID]

userdel과 옵션 및 기능에서 차이  
자세한 내용은 직접 살펴 보세요^^

1. user2 계정을 홈 디렉토리 포함하여 삭제 : **deluser --remove-home user2**

```
root@linux-test:~# cat /etc/passwd | grep user2
user2:x:2001:27:,,,:/home1/user2:/bin/bash
```

user2 계정 확인

```
root@linux-test:~#
root@linux-test:~# ls /home1
nbpmn user2
```

user2 홈 디렉토리 확인

```
root@linux-test:~#
root@linux-test:~# deluser --remove-home user2
```

user2 계정 삭제

```
Looking for files to backup/remove ...
Removing files ...
Removing user 'user2' ...
Warning: group 'sudo' has no more members.
Done.
```

```
root@linux-test:~#
root@linux-test:~# cat /etc/passwd | grep user2
```

계정 삭제 확인

```
root@linux-test:~#
root@linux-test:~# ls /home1
nbpmn
root@linux-test:~#
```

user2의 홈 디렉토리도 삭제 확인

# 사용자 계정 비밀번호 수정

## ● passwd

- **기능** : 사용자 계정의 비밀번호 지정
- **형식** : **passwd** [옵션] [사용자 ID]

1. user1 계정의 신규 비밀번호 지정 : **passwd user1**

```
root@linux-test:~# passwd user1
New password:
Retype new password:
passwd: password updated successfully
root@linux-test:~#
root@linux-test:~# cat /etc/shadow | grep user1
user1:$6$aw4SdKiQ$F0wjACl47Lsa4S.GMB4tH20C40ibTR.abINbFGv1LV1ArrXI/7hIFYwLchnMQ/xSc7n/7Ft8ybZ4otzPm4RSg0:19431:90:120:30:::
```

신규 비밀번호 입력

저장된 비밀번호는 **/etc/shadow**에 암호화되어 저장

### /etc/shadow 파일 정보 구성

로그인 ID : 비밀번호 : 최종 변경일 : MIN : MAX : WARNING : INACTIVE : EXPIRE : Flag

암호화된 비밀번호  
(길다)

비밀번호의 마지막 변경 날짜  
1970/1/1 기준 날짜 지정  
(19431)

변경된 비밀번호를  
사용해야 하는 최소 기간  
(90)

비밀번호를 사용할 수 있는  
최대 기간  
(120)

비밀번호 만료 전  
경고 기간

비밀번호 만료 후  
로그인 가능한 기간  
(Lock 유예 기간)

계정 만료 날짜  
1970/1/1 기준 날짜

비워둠

# 사용자 계정 비밀번호 수정

- passwd

2. user1 계정의 비밀번호 삭제 : **passwd -d user1**

```
root@linux-test:~# passwd -d user1
passwd: password expiry information changed.
root@linux-test:~#
root@linux-test:~# cat /etc/shadow | grep user1
user1::19431:90:120:30:::
root@linux-test:~#
```

저장되어 있던 비밀번호 제거

이후 실습을 위해 **꼭!!** 비밀번호 다시 지정 : **passwd user1**

```
root@linux-test:~# passwd user1
New password:
Retype new password:
passwd: password updated successfully
root@linux-test:~#
```

# 사용자 계정 비밀번호 수정

- passwd

3. user1 계정의 비밀번호 잠금(Lock) : **passwd -l user1**

```
root@linux-test:~# passwd -l user1
passwd: password expiry information changed.
root@linux-test:~#
root@linux-test:~# cat /etc/shadow | grep user1
user1:!$6$aw4SdKiQ$FOWjACl47Lsa4S.GMB4tH20C40ibTR.abINbfG
root@linux-test:~#
```

패스워드 문자열 앞의 !는 잠금 표시

4. user1 계정의 비밀번호 잠금 해제 : **passwd -u user1**

```
root@linux-test:~# passwd -u user1
passwd: password expiry information changed.
root@linux-test:~#
root@linux-test:~# cat /etc/shadow | grep user1
user1:$6$aw4SdKiQ$FOWjACl47Lsa4S.GMB4tH20C40ibTR.
root@linux-test:~#
```

패스워드 문자열 앞의 ! 사라짐 (잠금 해제)

# 그룹 생성

## • groupadd

- **기능** : 그룹을 생성
- **형식** : **groupadd** [옵션] [그룹명]

1. ugroup1 그룹 생성 : **groupadd** ugroup1

```
root@linux-test:~# groupadd ugroup1
root@linux-test:~#
root@linux-test:~# grep ugroup1 /etc/group
ugroup1:x:2003:
```

ugroup1 그룹 생성 확인

cat /etc/group | grep ugroup1 과 같은 명령

2. GID를 3000으로 지정하여 ugroup2 그룹 생성 : **groupadd -g 3000** ugroup2

```
root@linux-test:~# groupadd -g 3000 ugroup2
root@linux-test:~#
root@linux-test:~# grep ugroup2 /etc/group
ugroup2:x:3000:
```

ugroup2 그룹과 GID 확인

# 그룹 생성

## • addgroup

- **기능** : 그룹을 생성
- **형식** : **addgroup** [옵션] [그룹명]

1. ugroup3 그룹 생성 : **addgroup ugroup3**

```
root@linux-test:~# addgroup ugroup3
Adding group 'ugroup3' (GID 1002) ...
Done.
root@linux-test:~# grep ugroup3 /etc/group
ugroup3:x:1002:
```

ugroup3 그룹 생성 확인

GID 값이 명시되지 않으면  
/etc/adduser.conf의 **FIRST\_GID=1000**을 기준

2. GID를 2000으로 지정하여 ugroup4 그룹 생성 : **addgroup --gid 2000 ugroup4**

```
root@linux-test:~# addgroup --gid 2000 ugroup4
Adding group 'ugroup4' (GID 2000) ...
Done.
root@linux-test:~#
root@linux-test:~# grep ugroup4 /etc/group
ugroup4:x:2000:
```

ugroup4 그룹과 GID 확인



# 그룹 정보 수정

## • groupmod

- **기능** : 그룹 정보를 수정
- **형식** : **groupmod** [옵션] [그룹명]

1. ugroup1의 GID를 3100으로 수정 : **groupmod -g 3100 ugroup1**

```
root@linux-test:~# grep ugroup1 /etc/group
ugroup1:x:2003:
root@linux-test:~#
root@linux-test:~# groupmod -g 3100 ugroup1
root@linux-test:~#
root@linux-test:~# grep ugroup1 /etc/group
ugroup1:x:3100:
root@linux-test:~#
```

ugroup1의 GID가 3100으로 수정

2. **ugroup2**의 그룹 이름을 **ugroup002**로 수정 : **groupmod -n ugroup002 ugroup2**

```
root@linux-test:~# grep ugroup2 /etc/group
ugroup2:x:3000:
root@linux-test:~#
root@linux-test:~# groupmod -n ugroup002 ugroup2
root@linux-test:~#
root@linux-test:~# grep ugroup002 /etc/group
ugroup002:x:3000:
root@linux-test:~#
```

그룹 이름만 변경되고 GID는 같다

# 그룹 삭제

## ● groupdel

- **기능** : 그룹을 삭제
- **형식** : **groupdel [그룹명]**

1. ugroup4 그룹 삭제 : **groupdel ugroup4**

```
root@linux-test:~# grep ugroup4 /etc/group
ugroup4:x:2000:
root@linux-test:~#
root@linux-test:~# groupdel ugroup4
root@linux-test:~#
root@linux-test:~# grep ugroup4 /etc/group
root@linux-test:~#
```

ugroup4 그룹 확인

ugroup4 그룹 삭제

그룹 삭제 확인



# 그룹 패스워드 변경 / 멤버 추가 및 제거

## ● gpasswd

- **기능** : 그룹의 암호를 지정하거나 멤버를 추가 또는 삭제
- **형식** : **gpasswd** [옵션] [그룹명]

1. ugroup1 그룹의 패스워드 설정 : **gpasswd ugroup1**

```
root@linux-test:~# gpasswd ugroup1
Changing the password for group ugroup1
New Password:
Re-enter new password:
root@linux-test:~#
root@linux-test:~# cat /etc/gshadow | grep ugroup1
ugroup1:$6$ysO4YM2Dpxg$4xaDm0ErBjH8AAaTYIRVumIco4I0dZS4m2om9PAL4c5VLdPwSOvXeU285XtG6ry64WW9GaPW9lTEZa1GFtc116:::
root@linux-test:~#
```

newgrp 명령으로 사용자 계정의  
소속 그룹을 일시적으로 변경 시 그룹 패스워드  
입력

저장된 패스워드는 **/etc/gshadow**에 암호화되어 저장

/etc/gshadow 파일 정보 구성

그룹 이름 : 그룹 암호 : 관리자 : 그룹 멤버

2. ugroup1 그룹의 패스워드 삭제: **gpasswd -r ugroup1**

```
root@linux-test:~# gpasswd -r ugroup1
root@linux-test:~#
root@linux-test:~# cat /etc/gshadow | grep ugroup1
ugroup1:::
root@linux-test:~#
```

/etc/gshadow에 있던 ugroup1의 패스워드  
제거됨

# 그룹 패스워드 변경 / 멤버 추가 및 제거

- gpasswd

3. user1을 ugroup1 그룹에 추가 : **gpasswd -a user1 ugroup1**

```
root@linux-test:~# gpasswd -a user1 ugroup1
Adding user user1 to group ugroup1
root@linux-test:~#
root@linux-test:~# grep ugroup1 /etc/group
ugroup1:x:3100:user1
root@linux-test:~#
```

user1이 보조 그룹으로 ugroup1에 추가됨

4. user1을 ugroup1 그룹에서 제거 : **gpasswd -d user1 ugroup1**

```
root@linux-test:~# gpasswd -d user1 ugroup1
Removing user user1 from group ugroup1
root@linux-test:~#
root@linux-test:~# grep ugroup1 /etc/group
ugroup1:x:3100:
root@linux-test:~#
```

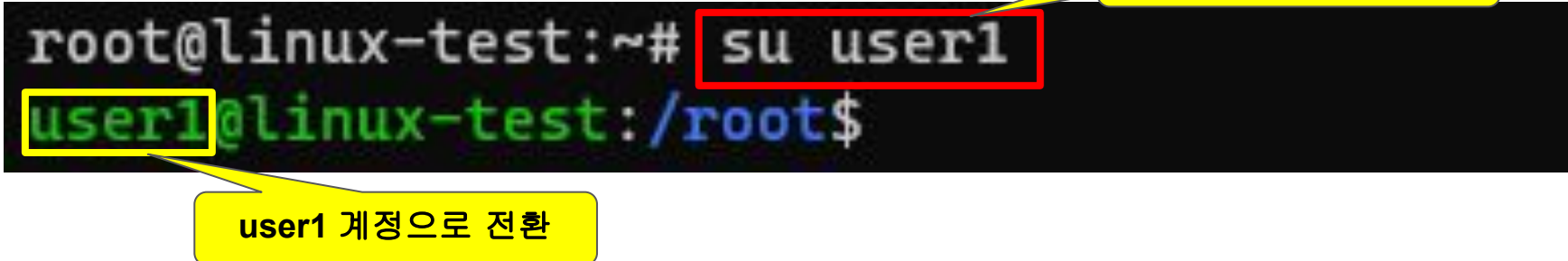
user1이 ugroup1에서 제거됨

# 사용자 전환

- **SU** (**S**witch **U**ser)

- **기능** : (로그아웃 없이) 사용자 계정을 전환
- **형식** : **su** [옵션] [사용자 ID]

1. user1 계정으로 전환 : **su user1**



```
root@linux-test:~# su user1
user1@linux-test:/root$
```

The terminal screenshot shows the command `su user1` being executed. A red box highlights the command, and a yellow callout bubble points to it with the text "user1 계정으로 전환". Below the command, the prompt changes to `user1@linux-test:/root$`, with a yellow box highlighting the new prompt and another yellow callout bubble pointing to it with the text "user1 계정으로 전환".

# 사용자 전환

- **SU** (**S**witch **U**ser)

2. user1의 환경 변수를 적용하여 계정을 전환 : **su - user1**  
**su -l user1**

바꿀 계정의 환경 변수가 적용됨

=> (로그아웃하고) 이 계정으로 직접 로그인한 것처럼 적용

로그인 계정(root)에서 환경 변수 선언 (VAR1)

```
root@linux-test:~# export VAR1=hello
```

```
root@linux-test:~#
```

```
root@linux-test:~# su user1
```

su user1으로 계정 전환하고 환경 변수 확인

=> root 계정에서 선언한 VAR1이 함께 전달

```
user1@linux-test:/root$ env | grep VAR1
```

```
VAR1=hello
```

```
user1@linux-test:/root$ exit
```

```
exit
```

```
root@linux-test:~# su -l user1
```

su -l user1으로 계정 전환하고 확인

=> VAR1이 전달되지 않음

user1의 환경변수 적용

```
To run a command as administrator (root) on a system with less than 1.5MB of swap space, use sudo with the -i option. See "man sudo_root" for details.
```

```
user1@linux-test:~$ env | grep VAR1
```

```
user1@linux-test:~$
```

# 사용자 전환

- **SU** (**S**witch **U**ser)

보안 주의

3. 일반 계정에서 **root** 계정으로 전환 : **su**

뒤에 root는 생략 가능

**su -**

**su -l**

> root 계정의 환경 변수 적용

일반 사용자가 root 권한을 갖는다는 의미는?

기존 세션 로그아웃 후  
user1 계정으로 다시 로그인

```
PS C:\Users\magicnotebook> ssh user1@106.10.32.76 -p 1024
user1@106.10.32.76's password:
```

user1 계정의 패스워드 입력

(중략)

```
user1@linux-test:~$ su -
Password:
root@linux-test:~#
```

user1 계정에서 root 계정으로 전환

root 계정의 패스워드 입력

=> 일반 계정에서 다른 계정으로 전환할 때는  
바꾸고자 하는 계정의 패스워드 필요

# 사용자 전환

- **su (Switch User)**

보안 주의

셸이 변경되지 않음

뒤에 root는 생략 가능

4. 일반 계정에서 **root** 계정 권한으로 명령 실행 : **su -c "apt update"**

root 권한으로 실행해야 하는 명령

일반 계정(user1)에서 **apt update** 명령 실행

user1에는 apt update의 실행 권한이 없으므로 **에러 발생**

```
user1@linux-test:~$ apt update
Reading package lists... Done
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Permission denied)
E: Unable to lock directory /var/lib/apt/lists/
W: Problem unlinking the file /var/cache/apt/pkgcache.bin - RemoveCaches (13: Permission denied)
W: Problem unlinking the file /var/cache/apt/srcpkgcache.bin - RemoveCaches (13: Permission denied)
user1@linux-test:~$ su -c "apt update"
Password:
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [91.5 kB]
Hit:2 http://kr.archive.ubuntu.com/ubuntu bionic InRelease [91.5 kB]
Get:3 http://kr.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://kr.archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Fetched 261 kB in 2s (153 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
29 packages can be upgraded. Run 'apt list --upgradable' to see them.
user1@linux-test:~$
```

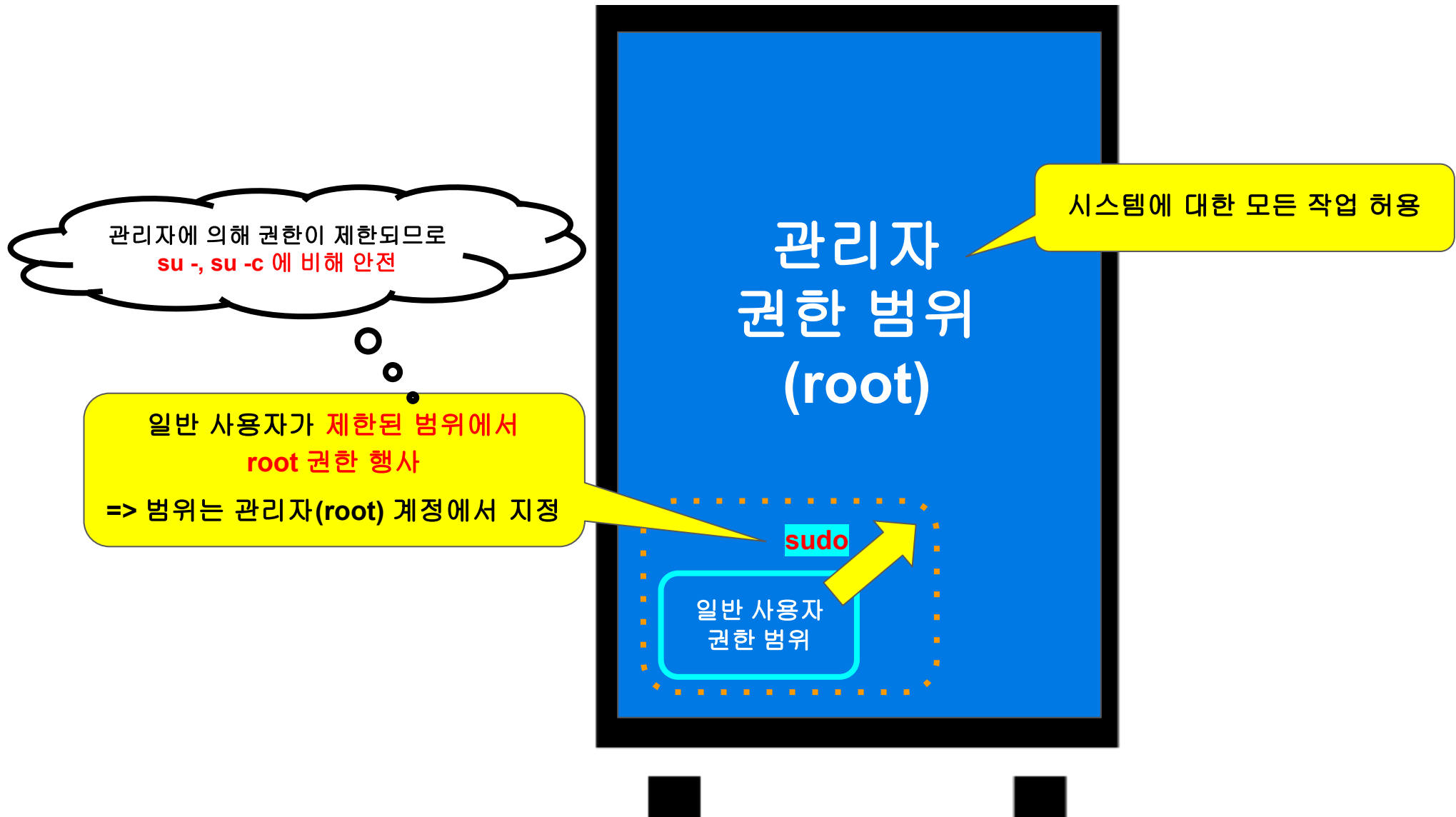
관리자 권한으로 다시 실행  
(root 패스워드 입력 필요)

apt update 명령 실행 확인

셸은 user1으로 유지

# sudo

sudo (**S**Uper-user **D**O)





# sudo 권한 설정

## • visudo

root 권한으로 실행

- **기능** : sudo 권한 지정 파일(/etc/sudoers)을 수정
- **형식** : visudo

### 1. 일반 계정(user1)에 adduser 실행 권한 부여

root 계정에서  
실행

```
root@linux-test:~# visudo
```

```
GNU nano 2.9.3 /etc/sudoers.tmp
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults    env_reset
Defaults    mail_badpass
Defaults    secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin   ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
#include_dir /etc/sudoers.d

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text   ^J Justify    ^C Cur Pos   M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell   ^_ Go To Line M-E Redo
```

다음과 같이 추가

```
# User privilege specification
root    ALL=(ALL:ALL) ALL
user1   ALL=/usr/sbin/adduser
```

Ctrl + o : 저장



# sudo 명령

## • sudo

- **기능** : root 권한으로 명령 실행
- **형식** : **sudo [명령]**

1. 일반 계정(user1)에서 root 권한으로 user001 계정 추가 : **sudo adduser user001**

```
user1@linux-test:~$ adduser user001
adduser: Only root may add a user or group to the system.
user1@linux-test:~$
user1@linux-test:~$ sudo adduser user001
Adding user 'user001' ...
Adding new group 'user001' (1004) ...
Adding new user 'user001' (1002) with group 'user001' ...
Creating home directory '/home/user001' ...
Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for user001
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n]
user1@linux-test:~$
user1@linux-test:~$ cat /etc/passwd | grep user001
user001:x:1002:1004:,,,:/home/user001:/bin/bash
user1@linux-test:~$
```

일반 계정에서 adduser 명령 실행 시  
에러 발생 (root 권한 요구)

sudo를 통해 adduser 명령을 root 권한으로 실행  
=> 앞서 visudo를 통해 user1 계정에 대해 adduser  
실행 권한을 부여했기 때문에 가능

user001 계정 생성 확인

# sudo 권한 설정 - sudo 그룹 지정

사용자 계정을 sudo 그룹으로 지정 (**root 권한 부여**)

별도의 visudo 편집 필요 없음  
편하지만 **보안 주의** 필요

1. 일반 계정(user001)을 sudo 그룹에 추가 : `usermod -aG sudo user001`  
`gpasswd -a user001 sudo`

root 계정에서  
실행

```
root@linux-test:~# usermod -aG sudo user001
root@linux-test:~#
root@linux-test:~# grep sudo /etc/group
sudo:x:27:user001
root@linux-test:~#
```

user001 계정을 sudo 그룹에 추가  
-a : append, -G : secondary group

2. 일반 계정(user001)으로 root 권한 명령 (apt update) 실행 : `sudo apt update`

```
root@linux-test:~# su - user001
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

root 계정에서  
user001 계정으로 전환

```
user001@linux-test:~$ sudo apt update
[sudo] password for user001:
Hit:1 http://kr.archive.ubuntu.com/ubuntu bionic InRelease [88.7 kB]
Hit:2 http://kr.archive.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Hit:3 http://kr.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:4 http://kr.archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Reading package lists... Done
Building dependency tree
Reading state information... Done
29 packages can be upgraded. Run 'apt list --upgradable' to see them.
user001@linux-test:~$
```

sudo를 통해 apt update 명령 실행

root 계정이 아닌  
user001의 비밀번호 입력

사용자  
(Real User)

VS

유효 사용자  
(Effective User)

ssh등으로 로그인한 사용자  
/etc/passwd에 등록된 사용자

실행되는 명령(프로세스) 안에서  
실제로 권한을 행사하는 사용자

이 둘이 다른 경우가 있나?

어떤 사용자의 권한으로 이 명령을 실행하는가

# 사용자와 유효 사용자

다른 경우가 또 있나?

사용자와 유효 사용자가 다른 경우 1. 사용자 전환

```
PS C:\Users\magicnotebook> ssh user1@106.10.32.76 -p 1024
user1@106.10.32.76's password:
Welcome to Ubuntu 18.04.5 LTS (Linux 4.15.0-118-generic x86_64)

Last login: Mon Aug 13 10:23 from 221.146.63.213
user1@linux-test:~$
user1@linux-test:~$ su -
Password:
root@linux-test:~# apt update
Get:1 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Hit:2 http://kr.archive.ubuntu.com/ubuntu bionic InRelease
Get:3 http://kr.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 http://kr.archive.ubuntu.com/ubuntu bionic-backports InRelease [83.3 kB]
Fetched 261 kB in 2s (111 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
29 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@linux-test:~#
```

(중략)

user1으로 로그인

=> 사용자는 user1  
유효 사용자도 user1

root로 사용자 전환

유효 사용자가 root인 영역

root의 권한으로 명령어 실행

# 사용자 확인 명령

## • who am i

RUID (UID) 라고 쓰임

- 기능 : ssh등으로 로그인한 사용자 계정(Real User ID) 출력
- 형식 : who am i

```
PS C:\Users\magicnotebook> ssh user1@106.10.32.76 -p 1024
user1@106.10.32.76's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-46-generic x86_64)
```

user1 계정으로 ssh 접속

(중략)

```
Last login: Thu Mar 16 11:04:41 2023 from 221.146.63.213
user1@linux-test:~$ who am i
user1 pts/0 2023-03-16 11:30 (221.146.63.213)
user1@linux-test:~$
```

사용자 계정 출력

사용자 ID

접속한 단말기

pts : 네트워크를 통한 접속

로그인 시각

접속한 컴퓨터의 IP 주소

# 사용자 확인 명령

## • who

- **기능** : 현재 시스템에 접속한 모든 사용자(Real User ID) 출력
- **형식** : **who** [옵션]

시스템에 접속한 모든 사용자 리스트 출력  
=> root, user001은 별도의 터미널 세션으로 접속

```
user1@linux-test:~$ who
```

user1	pts/0	2023-03-16	11:30	(221.146.63.213)
root	pts/1	2023-03-16	11:31	(221.146.63.213)
user001	pts/2	2023-03-16	11:43	(221.146.63.213)



# 사용자 확인 명령

## • whoami

통상 EUID라고 쓰임

- 기능 : 현재의 유효 사용자(Effective User ID)를 출력
- 형식 : whoami

```
PS C:\Users\magicnotebook> ssh user1@106.10.32.76 -p 1024
user1@106.10.32.76's password:
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-118-generic x86_64)
```

(중략)

```
Last login: Thu Mar 16 11:30:42 2023 from 221.146.63.213
```

```
user1@linux-test:~$ whoami
```

```
user1
```

```
user1@linux-test:~$ su -
```

root로 사용자 전환

```
Password:
```

```
root@linux-test:~# whoami
```

```
root
```

유효 사용자 출력

## who am i 명령과 비교

```
root@linux-test:~# who am i
```

```
user1
```

root로 전환한 상태에서도 처음 로그인한 사용자 출력

# 사용자 확인 명령

## • id

- **기능** : 사용자 또는 유효 사용자의 ID, 그룹 ID 등을 출력
- **형식** : **id** [옵션] [사용자 ID]

1. 현재 유효 사용자에 대한 정보 출력 : **id**

```
user1@linux-test:~$ id
uid=1001(user1) gid=1001(user1) groups=1001(user1),100(users),113(lpadmin)
```

유효 사용자의  
UID 값과 사용자 ID

유효 사용자의  
GID (기본 그룹 ID)

유효 사용자가 속한 그룹 리스트  
(기본 그룹, 보조 그룹)

```
user1@linux-test:~$ su -
Password:
root@linux-test:~# id
uid=0(root) gid=0(root) groups=0(root)
```

root로 사용자 전환

root에 대한 정보 출력

2. user001 사용자에 대한 정보 출력 : **id user001**

```
user1@linux-test:~$ id user001
uid=1002(user001) gid=1004(user001) groups=1004(user001),27(sudo)
```



# 유효 사용자의 소속 그룹 확인

## ● groups

- **기능** : 사용자 또는 유효 사용자의 소속 그룹을 출력
- **형식** : **groups** [사용자 ID]

```

user1@linux-test:~$ groups
user1 users lpadmin
user1@linux-test:~$ su -
Password:
root@linux-test:~# groups
root
root@linux-test:~# groups user001
user001 : user001 sudo
root@linux-test:~#
  
```

유효 사용자의 소속 그룹 출력  
(기본 그룹, 보조 그룹)

지정한 사용자 계정의  
소속 그룹 출력

# 기본 그룹 임시 변경

## • newgrp

- **기능** : 현재 사용자 계정의 기본 그룹을 **임시로 변경** (임시 셸 실행)
- **형식** : **newgrp [그룹명]**

### 1. user1 계정이 소속된 2차 그룹 중 하나를 기본 그룹으로 변경

```

user1@linux-test:~$ id
uid=1001(user1) gid=1001(user1) groups=1001(user1),100(users),113(lpadmin)
user1@linux-test:~$
user1@linux-test:~$ newgrp lpadmin
user1@linux-test:~$ id
uid=1001(user1) gid=113(lpadmin) groups=113(lpadmin),100(users),1001(user1)
user1@linux-test:~$ exit
exit
user1@linux-test:~$ id
uid=1001(user1) gid=1001(user1) groups=1001(user1),100(users),113(lpadmin)
user1@linux-test:~$
  
```

임시 셸 실행하면서  
user1의 기본 그룹을  
lpadmin으로 변경

임시 셸 종료

임시 셸에서 변경된 기본 그룹

기존 user1의 GID

# 기본 그룹 임시 변경

- newgrp

## 2. 소속되지 않은 그룹으로 임시 변경

```
user1@linux-test:~$ newgrp sudo
Password:
newgrp: failed to crypt password with previous salt: Invalid argument
user1@linux-test:~$
```

소속되지 않은 그룹으로 변경하고 할 때는  
해당 그룹에 대한 패스워드 필요

# 파일 소유 및 접근 권한

# 파일의 소유자와 소유 그룹

## /root (root 계정의 홈 디렉토리)

```
root@linux-test:~# ls -al
total 76
drwx----- 9 root root 4096 Mar 16 15:57 .
drwxr-xr-x 25 root root 4096 Oct 13 2020 ..
-rw----- 1 root root 0 Mar 16 15:50 .bash_history
-rw-r--r-- 1 root root 39 Mar 6 03:07 .bash_logout
-rw-r--r-- 1 root root 3108 Mar 6 09:10 .bashrc
drwx----- 2 root root 4096 Feb 9 11:14 .cache
drwx----- 3 root root 4096 Oct 6 2020 .gnupg
drwxr-xr-x 3 root root 4096 Oct 6 2020 .local
-rw----- 1 root root 189 Mar 5 21:47 .node_repl_history
drwxr-xr-x 2 root root 4096 Mar 16 09:57 .nsight
-rw-r--r-- 1 root root 148 Aug 18 2015 .profile
-rw-r--r-- 1 root root 75 Oct 6 2020 .selected_editor
drwx----- 2 root root 4096 Oct 6 2020 .ssh
drwxr-xr-x 2 root root 4096 Mar 6 17:53 .vim
-rw----- 1 root root 13623 Mar 15 10:44 .viminfo
drwxr-xr-x 5 root root 4096 Feb 20 12:16 .vscode-server
-rw-r--r-- 1 root root 183 Feb 20 12:16 .wget-hsts
```

## /home/user1 (user1 계정의 홈 디렉토리)

```
user1@linux-test:~$ ls -al
total 28
drwxr-xr-x 4 user1 user1 4096 Mar 16 15:59 .
drwxr-xr-x 4 root root 4096 Mar 15 14:21 ..
-rw----- 1 user1 user1 0 Mar 16 15:58 .bash_history
-rw-r--r-- 1 user1 user1 220 Mar 15 11:18 .bash_logout
-rw-r--r-- 1 user1 user1 3771 Mar 15 11:18 .bashrc
drwx----- 2 user1 user1 4096 Mar 15 11:23 .cache
drwx----- 3 user1 user1 4096 Mar 15 11:23 .gnupg
-rw-r--r-- 1 user1 user1 807 Mar 15 11:18 .profile
```

리눅스 시스템 상의 모든 파일은 소유자와  
소유 그룹이 지정되어 있음

```
-rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
```

파일 소유자

파일 소유 그룹

이건 어떻게 결정되는 걸까?

# 파일의 소유자와 소유 그룹

chatGPT에게 물어보니...

유효 사용자

리눅스에서 파일의 소유자와 소유 그룹은 어떻게 결정되는거야?

리눅스에서 파일의 소유자와 소유 그룹은 각 파일이 생성될 때 결정됩니다. 파일을 생성하는 사용자는 일반적으로 파일의 소유자가 되며, 그 사용자가 속한 기본 그룹이 파일의 소유 그룹이 됩니다.

하지만 파일의 소유자와 소유 그룹은 다른 사용자나 그룹으로 변경될 수 있습니다. 이를 위해 리눅스는 다양한 명령어와 옵션을 제공합니다.



# 파일의 소유자와 소유 그룹

## 파일 생성에 따른 소유자 / 소유 그룹 결정 - 실습1

```
user1@linux-test:~$ touch test
```

유효 사용자가 **user1**인 상태에서  
**user1**의 홈 디렉토리에 **test** 파일 생성

```
user1@linux-test:~$ ls -al
total 28
drwxr-xr-x 4 user1 user1 4096 Mar 16 16:51 .
drwxr-xr-x 4 root  root  4096 Mar 16 16:50 ..
-rw----- 1 user1 user1    0 Mar 16 16:51 .bash_history
-rw-r--r-- 1 user1 user1  220 Mar 15 11:18 .bash_logout
-rw-r--r-- 1 user1 user1 3771 Mar 15 11:18 .bashrc
drwx----- 2 user1 user1 4096 Mar 15 11:23 .cache
drwx----- 3 user1 user1 4096 Mar 15 11:23 .gnupg
-rw-r--r-- 1 user1 user1  807 Mar 15 11:18 .profile
-rw-rw-r-- 1 user1 user1    0 Mar 16 16:51 test
```

**test** 파일의 소유자와 소유 그룹이 **user1**

```
user1@linux-test:~$ su root
```

```
root@linux-test:/home/user1# touch test2
```

유효 사용자가 **root**인 상태에서  
같은 디렉토리에 **test2** 파일 생성

```
root@linux-test:/home/user1# ls -al
total 28
drwxr-xr-x 4 user1 user1 4096 Mar 16 16:52 .
drwxr-xr-x 4 root  root  4096 Mar 16 16:50 ..
-rw----- 1 user1 user1    0 Mar 16 16:51 .bash_history
-rw-r--r-- 1 user1 user1  220 Mar 15 11:18 .bash_logout
-rw-r--r-- 1 user1 user1 3771 Mar 15 11:18 .bashrc
drwx----- 2 user1 user1 4096 Mar 15 11:23 .cache
drwx----- 3 user1 user1 4096 Mar 15 11:23 .gnupg
-rw-r--r-- 1 user1 user1  807 Mar 15 11:18 .profile
-rw-rw-r-- 1 user1 user1    0 Mar 16 16:51 test
-rw-r--r-- 1 root  root    0 Mar 16 16:52 test2
```

**test2** 파일의 소유자와 소유 그룹이 **root**

# 파일의 소유자와 소유 그룹

## 파일 생성에 따른 소유자 / 소유 그룹 결정 - 실습2

```

user1@linux-test:~$ su - user001
Password:
user001@linux-test:~$ id
uid=1002(user001) gid=1004(user001) groups=1004(user001),27(sudo)
user001@linux-test:~$
user001@linux-test:~$ newgrp sudo
user001@linux-test:~$
user001@linux-test:~$ id
uid=1002(user001) gid=27(sudo) groups=27(sudo),1004(user001)
user001@linux-test:~$
user001@linux-test:~$ touch test3
user001@linux-test:~$
user001@linux-test:~$ ls -al
total 28
drwxr-xr-x 4 user001 user001 4096 Mar 16 17:06 .
drwxr-xr-x 4 root    root    4096 Mar 16 16:50 ..
-rw----- 1 user001 user001    0 Mar 16 17:05 .bash_history
-rw-r--r-- 1 user001 user001  220 Mar 15 14:21 .bash_logout
-rw-r--r-- 1 user001 user001 3771 Mar 15 14:21 .bashrc
drwx----- 2 user001 user001 4096 Mar 16 11:43 .cache
drwx----- 3 user001 user001 4096 Mar 16 11:43 .gnupg
-rw-r--r-- 1 user001 user001  807 Mar 15 14:21 .profile
-rw-r--r-- 1 user001 user001    0 Mar 15 15:55 .sudo_as_admin_successful
-rw-rw-r-- 1 user001 sudo      0 Mar 16 17:06 test3
user001@linux-test:~$

```

유효 사용자를 user001로, 기본 그룹을 sudo로 변경

파일을 생성할 때의 유효 사용자 / 기본 그룹이  
파일 소유자 / 소유 그룹으로 지정



어떤 명령(프로세스)의 **유효 사용자**가  
시스템 내의 **파일 또는 디렉토리**에 행사할 수 있는 **권한**



# 파일 권한

## 권한의 종류

root 계정은 권한에서 자유로움

권한	파일	디렉토리
읽기(Read)	파일의 내용을 볼 수 있는 권한	디렉토리 내부의 내용(파일 또는 하위 디렉토리)을 볼 수 있는 권한 (ls 명령)
쓰기(Write)	파일의 내용을 수정할 수 있는 권한	디렉토리 내부에 파일을 생성 또는 삭제할 수 있는 권한 디렉토리 내부의 파일을 다른 곳으로 이동할 수 있는 권한
실행(eXcute)	실행 가능한 파일을 실행시킬 수 있는 권한	디렉토리 내부로 접근할 수 있는 권한 (cd 명령)

## 권한의 표기 방법

-rwxr-xr-x

User(Owner) Group Other  
- r w x r - x r - x

그 외 사용자가 할 수 있는 권한 (읽기, 실행)

파일의 소유 그룹에 속한 멤버가 할 수 있는 권한 (읽기, 실행)

파일의 소유자가 할 수 있는 권한 (읽기, 쓰기, 실행)

이런 파일이 있다고 하면...

## sample

소유자 : user1

소유 그룹 : ugroup1

User(Owner)

Group

Other

User(Owner)	Group	Other
<b>r w x</b>	<b>r w -</b>	<b>r - -</b>

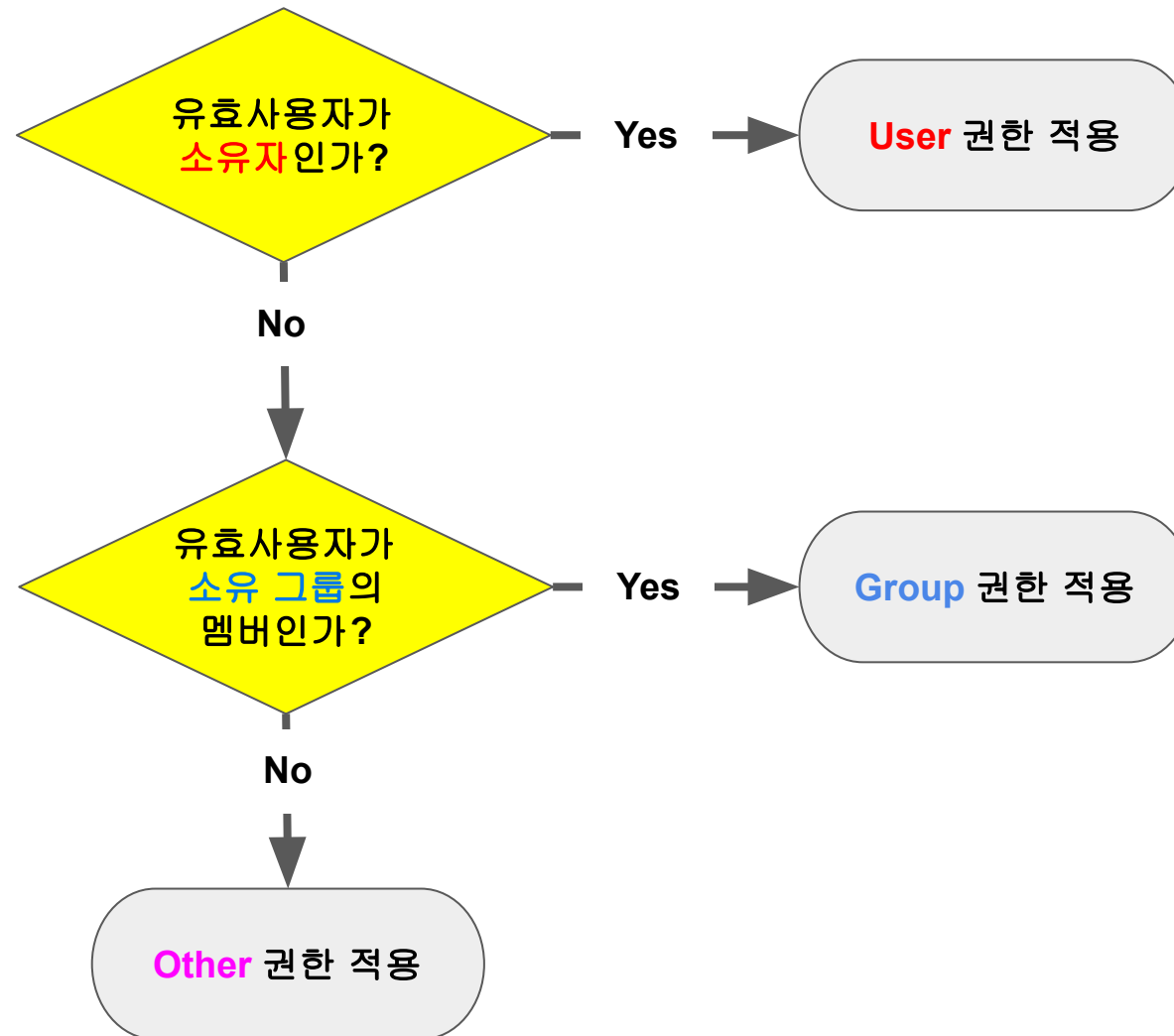
**user1**은 이 파일에 대해  
읽기, 쓰기, 실행 가능

**ugroup1** 소속 멤버들은  
읽기, 쓰기 가능

**그 외 사용자**들은  
읽기만 가능

# 파일 권한 적용

## 파일에 대한 권한 확인 절차



# 파일 권한 적용 - 디렉토리

디렉토리에 대한 권한 적용 확인

유효 사용자는 **user1**

root 디렉토리(/) 밑에 있는  
파일 및 디렉토리 확인

```
user1@linux-test:~$ ls -al /
total 104
drwxr-xr-x 25 root root 4096 Oct 13 2020 .
```

/ 디렉토리에 대해 user1은 **other 권한** 적용 (읽기, 실행)

(중략)

/bin 디렉토리에 대해 user1은 **other 권한** 적용 (읽기, 실행)

```
drwxr-xr-x 2 root root 4096 Feb 21 16:51 bin
```

(중략)

```
drwx--- 9 root root 4096 Mar 17 15:37 root
```

(하략)

/root 디렉토리에 대해 user1은 **other 권한** 적용  
(권한 없음)

# 파일 권한 적용 - 디렉토리

디렉토리에 대한 권한 적용 확인 - /bin

디렉토리 내의 파일 목록 출력  
=> /bin에 대한 **읽기 권한** 확인

```
user1@linux-test:~$ ls -l /bin | head -5
total 15328
-rwxr-xr-x 1 root root 1113504 Apr 19 2022 bash
-rwxr-xr-x 1 root root 716464 Mar 13 2018 btrfs
lrwxrwxrwx 1 root root 5 Mar 13 2018 btrfsck -> btrfs
-rwxr-xr-x 1 root root 375952 Mar 13 2018 btrfs-debug-tree
user1@linux-test:~$
```

디렉토리 내의 쓰기(write) 권한이 필요한 명령 실행이 모두 거부됨  
=> /bin 에 대한 user1의 **쓰기 권한 없음** 확인

```
user1@linux-test:~$ touch /bin/test
touch: cannot touch '/bin/test': Permission denied
user1@linux-test:~$
user1@linux-test:~$ rm /bin/false
rm: remove write-protected regular file '/bin/false'? y
rm: cannot remove '/bin/false': Permission denied
user1@linux-test:~$
user1@linux-test:~$ mv /bin/false ~
mv: cannot move '/bin/false' to '/home/user1/false': Permission denied
user1@linux-test:~$
```

/bin 디렉토리로 이동

=> /bin에 대한 **실행 권한** 확인

```
user1@linux-test:~$ cd /bin
user1@linux-test:/bin$
```

# 파일 권한 적용 - 디렉토리

디렉토리에 대한 권한 적용 확인 - /root

/root 디렉토리에 대해 user1은  
아무 권한이 없음

```
user1@linux-test:~$ ls -al /root
ls: cannot open directory '/root': Permission denied
user1@linux-test:~$
user1@linux-test:~$ touch /root/test
touch: cannot touch '/root/test': Permission denied
user1@linux-test:~$
user1@linux-test:~$ cd /root
-bash: cd: /root: Permission denied
user1@linux-test:~$
```

# 파일 권한 적용 - 파일

```

user1@linux-test:~$ id
uid=1001(user1) gid=1001(user1) groups=1001(user1),100(users),113(lpadmin)
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 12
-rw-r--r-- 1 user1 user1  12 Mar 17 17:21 HelloWorld
-rw-r--r-- 1 root  root   17 Mar 17 16:43 HowAreYou
-rw-rw-r-- 1 root  lpadmin 19 Mar 17 18:07 ImFine
user1@linux-test:~$

```

설명을 위해 이런 파일을 생성해 보았다

어떻게 파일의 소유자, 소유  
그룹을 이렇게 생성할 수  
있을까?

유효 사용자가 **user1**일 때 각 파일에는 어떤 권한이 적용될까?



# 파일 권한 적용 - 파일

현재 user1의 홈  
디렉토리에 위치

user1에 대한  
UID, GID, 소속 그룹 출력

```
user1@linux-test:~$ id
uid=1001(user1) gid=1001(user1) groups=1001(user1),100(users),113(lpadmin)
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 12
-rw-r--r-- 1 user1 user1 12 Mar 17 17:21 HelloWorld
-rw-r--r-- 1 root  root  17 Mar 17 16:43 HowAreYou
-rw-rw-r-- 1 root  lpadmin 19 Mar 17 18:07 ImFine
```

각 파일에 대해 user1이 적용되는 권한

- HelloWorld : user1이 소유자이므로 **rw-** 권한
- HowAreYou : user1이 소유자, 소유 그룹 모두 아니므로 **r--** 권한
- ImFine : user1이 소유자는 아니지만, 소유 그룹에 속해 있으므로 **rw-** 권한

# 파일 권한 적용 - 파일

```

user1@linux-test:~$ ls -l
total 12
-rw-r--r-- 1 user1 user1  12 Mar 17 17:21 HelloWorld
-rw-r--r-- 1 root  root   17 Mar 17 16:43 HowAreYou
-rw-rw-r-- 1 root  lpadmin 19 Mar 17 18:07 ImFine
user1@linux-test:~$
user1@linux-test:~$
user1@linux-test:~$ cat HelloWorld
Hello World
user1@linux-test:~$ echo HaHaHa >> HelloWorld
user1@linux-test:~$ cat HelloWorld
Hello World
HaHaHa
user1@linux-test:~$
user1@linux-test:~$ cat HowAreYou
Hi, How are you?
user1@linux-test:~$ echo HoHoHo >> HowAreYou
-bash: HowAreYou: Permission denied
user1@linux-test:~$
user1@linux-test:~$
user1@linux-test:~$ cat ImFine
I'm Fine Thank You
user1@linux-test:~$ echo KKKKKK >> ImFine
user1@linux-test:~$ cat ImFine
I'm Fine Thank You
KKKKKK
user1@linux-test:~$

```

HelloWorld 파일에 대해 user1은  
읽기 가능, 쓰기 가능

HowAreYou 파일에 대해서는  
읽기 가능, 쓰기 불가

ImFine 파일에 대해서는  
읽기 가능, 쓰기 가능

user1이 cat 명령으로 /etc/passwd 파일을 조회하기 위한 조건은?  
=> 이를 위해 필요한 권한은?

```
user1@linux-test:~$ cat /etc/passwd  
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
```

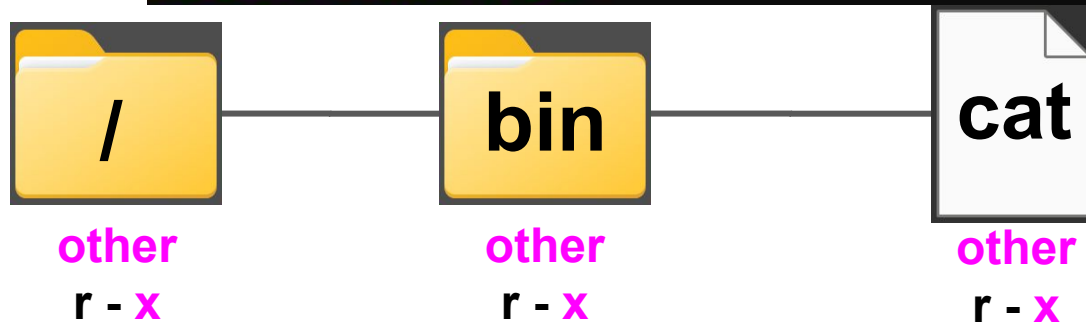
(하락)

# 파일 접근 권한

user10이 cat 명령을 실행하기 위해 필요한 권한

cat 실행 파일의  
경로

```
user1@linux-test:~$ which cat
/bin/cat
user1@linux-test:~$ ls -al / | head -2
total 104
drwxr-xr-x 25 root root 4096 Oct 13 2020 .
user1@linux-test:~$
user1@linux-test:~$ ls -al /bin | head -2
total 15336
drwxr-xr-x 2 root root 4096 Feb 21 16:51 .
user1@linux-test:~$
user1@linux-test:~$ ls -al /bin/cat | head -2
-rwxr-xr-x 1 root root 35064 Jan 18 2018 /bin/cat
user1@linux-test:~$
```



cat 파일에 대한 실행 권한 뿐 아니라  
경로 상의 각 디렉토리에 대한 접근  
권한도 필요

경로 상의 각 디렉토리에 대한  
접근(eXcute) 권한

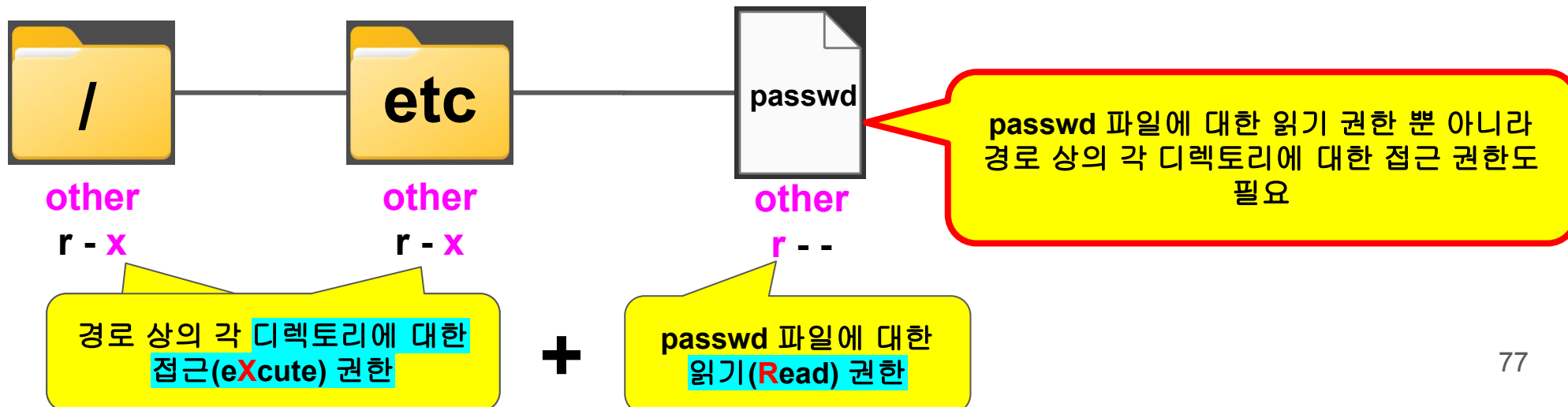
+

cat 파일에 대한  
실행(eXcute) 권한

# 파일 접근 권한

user10이 /etc/passwd 파일을 읽기 위해 필요한 권한

```
user1@linux-test:~$ ls -al / | head -2
total 104
drwxr-xr-x 25 root root 4096 Oct 13 2020 .
user1@linux-test:~$
user1@linux-test:~$ ls -al /etc | head -2
total 864
drwxr-xr-x 94 root root 4096 Mar 17 14:42 .
user1@linux-test:~$
user1@linux-test:~$ ls -al /etc/passwd | head -2
-rw-r--r-- 1 root root 1587 Mar 16 17:25 /etc/passwd
user1@linux-test:~$
```





# 파일 소유자 / 소유 그룹 변경

## ● chown

- **기능** : 파일과 디렉토리의 소유자와 소유 그룹 변경
- **형식** : **chown** [옵션] [사용자 ID (: 그룹명)] [파일명 또는 디렉토리명]

### 1. chown 실습 준비

```

root@linux-test:~# su - user001
user001@linux-test:~$
user001@linux-test:~$ mkdir temp
user001@linux-test:~$
user001@linux-test:~$ cp /etc/hosts .
user001@linux-test:~$ cp /etc/services temp
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 user001 user001 189 Mar 19 21:36 hosts
drwxrwxr-x 2 user001 user001 4096 Mar 19 21:36 temp
user001@linux-test:~$
user001@linux-test:~$ ls -l temp/
total 20
-rw-r--r-- 1 user001 user001 19183 Mar 19 21:36 services
user001@linux-test:~$

```

user001로 사용자 전환

각 명령이 무슨 의미인지  
이해하며 따라 합니다

복사한 파일의 소유자, 소유 그룹에 주목  
(user001, user001)

# 파일 소유자 / 소유 그룹 변경

## • chown

2. 복사한 hosts 파일의 소유자를 user1 변경 : chown **user1** hosts

```
user001@linux-test:~$ chown user1 hosts
chown: changing ownership of 'hosts': Operation not permitted
user001@linux-test:~$
user001@linux-test:~$ sudo chown user1 hosts
[sudo] password for user001:
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 user1 user001 189 Mar 19 21:36 hosts
drwxrwxr-x 2 user001 user001 4096 Mar 19 21:36 temp
user001@linux-test:~$
```

chown 명령은 root 권한 필요

sudo를 이용해 같은 명령을 관리자 권한으로 실행  
=> 이전에 user001를 sudo 그룹 멤버로 지정하였음

hosts 파일의 소유자가 user1으로 변경



# 파일 소유자 / 소유 그룹 변경

- chown

3. hosts 파일의 소유자를 root, 소유 그룹을 ugroup1으로 동시 변경

: chown **root:ugroup1** hosts

```
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 user1 user001 189 Mar 19 21:36 hosts
drwxrwxr-x 2 user001 user001 4096 Mar 19 21:36 temp
user001@linux-test:~$
user001@linux-test:~$ sudo chown root:ugroup1 hosts
[sudo] password for user001:
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 root ugroup1 189 Mar 19 21:36 hosts
drwxrwxr-x 2 user001 user001 4096 Mar 19 21:36 temp
user001@linux-test:~$
```

hosts 파일의 기존 권한 확인

소유자를 root, 소유 그룹을 ugroup1으로 동시 변경

변경된 소유자와 소유 그룹 확인

# 파일 소유자 / 소유 그룹 변경

- chown

## 4. temp 디렉토리 및 하위 파일에 대한 소유자, 소유 그룹을 변경

: chown -R root:ugroup1 temp

```

user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 root    ugroup1 189 Mar 19 21:36 hosts
drwxrwxr-x 2 user001 user001 4096 Mar 19 21:36 temp
user001@linux-test:~$
user001@linux-test:~$ ls -l temp/
total 20
-rw-r--r-- 1 user001 user001 19183 Mar 19 21:36 services
user001@linux-test:~$
user001@linux-test:~$ sudo chown -R root:ugroup1 temp
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 root    ugroup1 189 Mar 19 21:36 hosts
drwxrwxr-x 2 root    ugroup1 4096 Mar 19 21:36 temp
user001@linux-test:~$
user001@linux-test:~$ ls -l temp/
total 20
-rw-r--r-- 1 root    ugroup1 19183 Mar 19 21:36 services
user001@linux-test:~$

```

temp 디렉토리의 기존 권한 확인

temp 디렉토리 아래 services  
파일 기존 권한 확인

소유자를 root, 소유 그룹을 ugroup1으로  
동시 변경

변경된 소유자와 소유 그룹 확인

# 파일 소유자 / 소유 그룹 변경

## • chgrp

- **기능** : 파일과 디렉토리의 소유 그룹 변경
- **형식** : `chown [옵션] [그룹명] [파일명 또는 디렉토리명]`

1. hosts 파일의 소유 그룹을 root로 변경 : `chgrp root hosts`

```
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 root ugroup1 189 Mar 19 21:36 hosts
drwxrwxr-x 2 root ugroup1 4096 Mar 19 21:36 temp
user001@linux-test:~$
user001@linux-test:~$ chgrp root hosts
chgrp: changing group of 'hosts': Operation not permitted
user001@linux-test:~$
user001@linux-test:~$ sudo chgrp root hosts
[sudo] password for user001:
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 8
-rw-r--r-- 1 root root 189 Mar 19 21:36 hosts
drwxrwxr-x 2 root ugroup1 4096 Mar 19 21:36 temp
user001@linux-test:~$
```

chgrp 명령은 root 권한 필요

hosts 파일의 소유 그룹이 ugroup1에서 root로 변경

# 파일 권한 변경

## ● chmod

- **기능** : 파일이나 디렉토리의 접근 권한 변경
- **형식** : **chmod** **[옵션]** **[권한]** **[파일명 또는 디렉토리명]**

대표적인 옵션으로 **-R**이 있음

권한은 **문자** 또는 **수치**로 표시 가능

## 문자를 통한 권한 지정

chmod **O+W** sample.txt

권한 종류  
**read, write, excute**

### 사용자 범주

**u** : User(Owner) - 파일 소유자  
**g** : Group - 파일 소유 그룹  
**o** : Other - 그 외 사용자  
**a** : 모든 카테고리

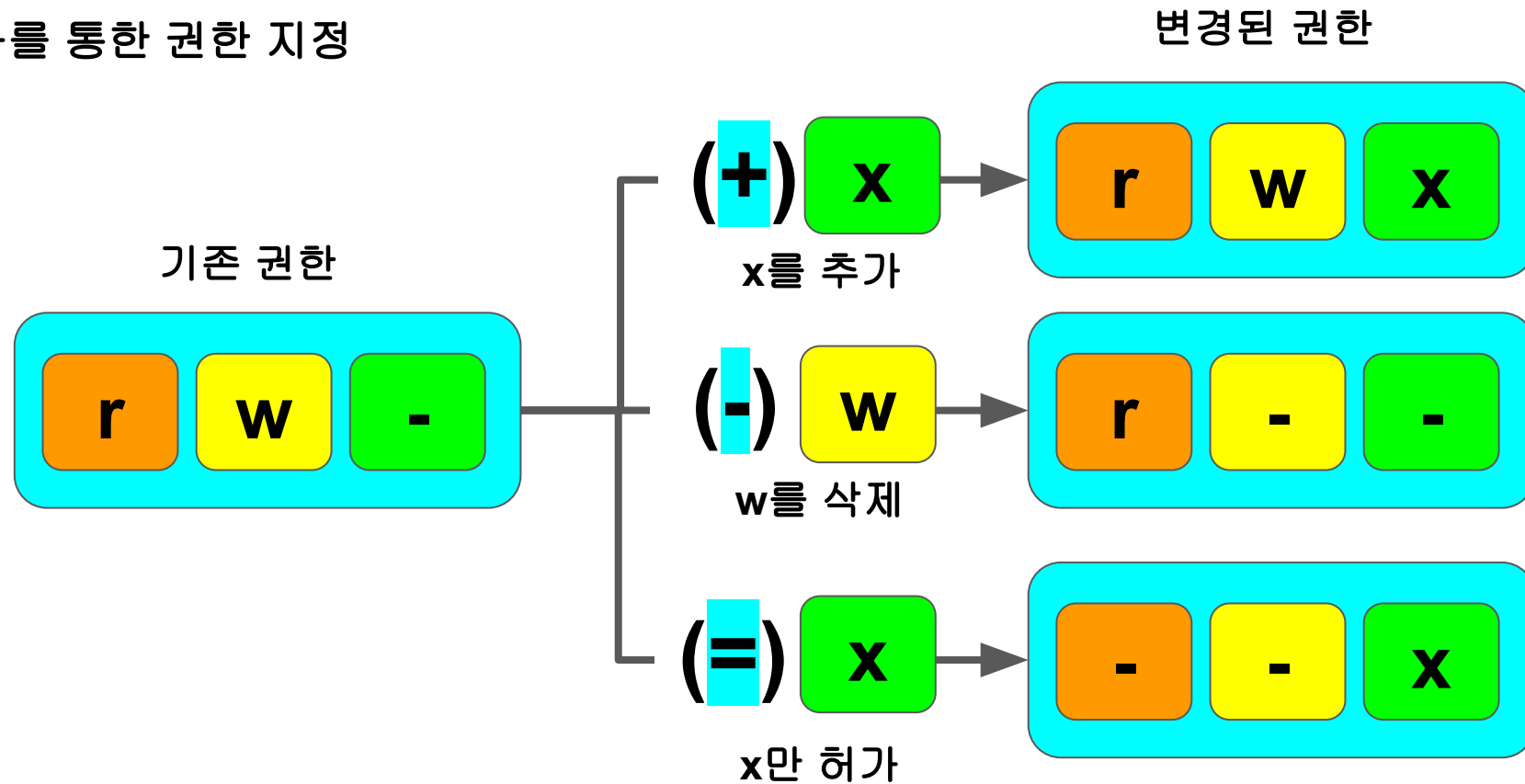
### 설정의 종류

**+** : 설정 추가  
**-** : 설정 삭제  
**=** : 설정 일괄 변경

# 파일 권한 변경

- chmod

문자를 통한 권한 지정



# 파일 권한 변경

- **chmod**

수치를 통한 권한 지정

권한	수치
읽기 (read)	4
쓰기 (write)	2
실행 (execute)	1
허가하지 않음	0

권한에 대한 수치가 4,2,1인 이유는?  
찾아보세요 ^^

**chmod** **7****5****5** sample.txt

**User**에 대한 설정

**7** = 4 + 2 + 1 (모든 권한)

**Group**에 대한 설정

**5** = 4 + 1 (읽기, 실행 권한)

**Other**에 대한 설정

**5** = 4 + 1 (읽기, 실행 권한)

# 파일 권한 변경

- chmod

## 1. chmod 실습 준비

/etc/hosts 파일을 user1의 홈 디렉토리에 복사

```
user1@linux-test:~$ cp /etc/hosts .
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 4
-rw-r--r-- 1 user1 user1 189 Mar 20 00:30 hosts
user1@linux-test:~$
```

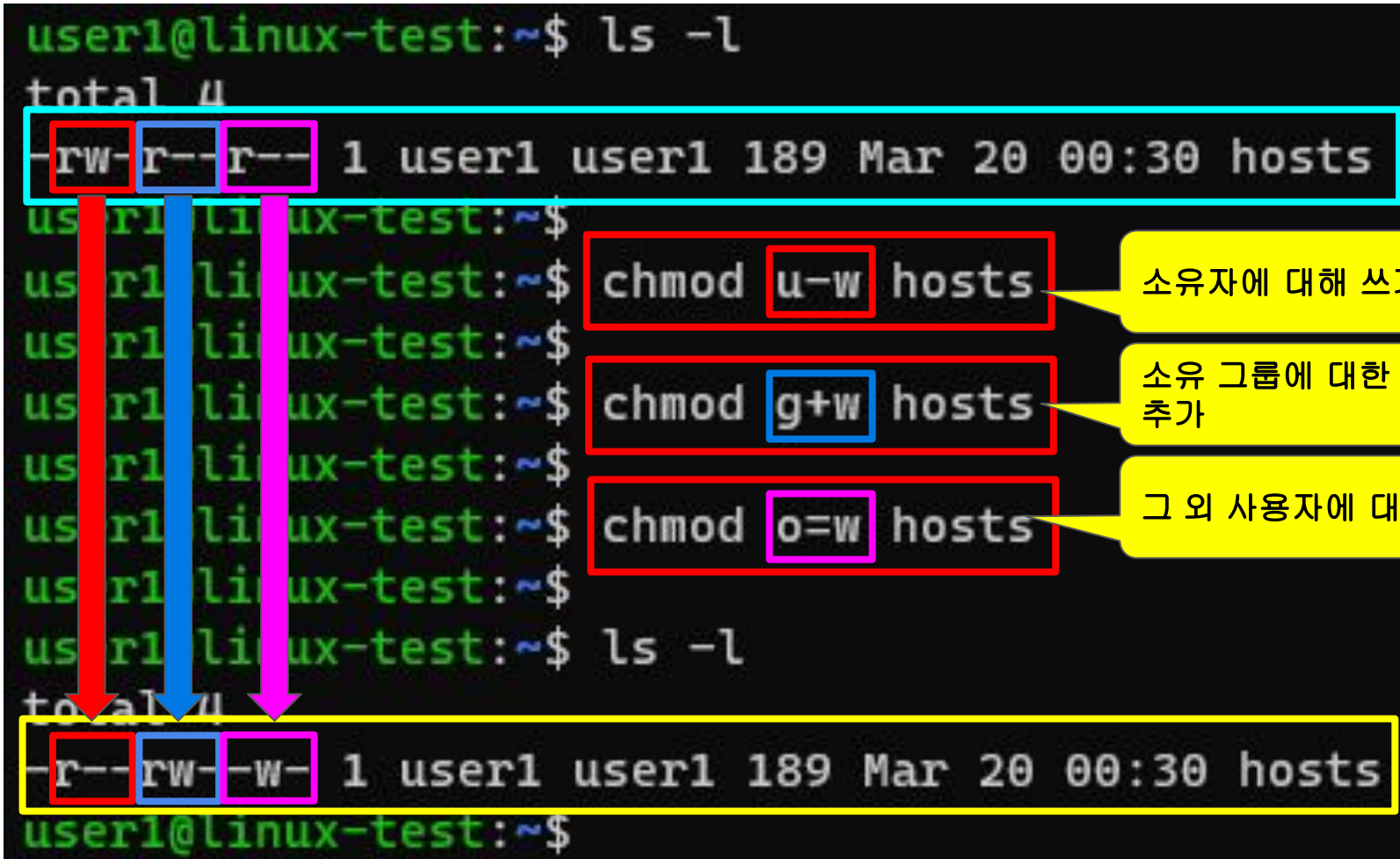
복사한 파일의 권한 확인



# 파일 권한 변경

- chmod

## 2. 문자를 통한 권한 부여 실습



```

user1@linux-test:~$ ls -l
total 4
-rw-r--r-- 1 user1 user1 189 Mar 20 00:30 hosts
user1@linux-test:~$
user1@linux-test:~$ chmod u-w hosts
user1@linux-test:~$
user1@linux-test:~$ chmod g+w hosts
user1@linux-test:~$
user1@linux-test:~$ chmod o=w hosts
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 4
-r--rw--w- 1 user1 user1 189 Mar 20 00:30 hosts
user1@linux-test:~$
  
```

Diagram illustrating the chmod command and its effects on file permissions:

- Initial State:** `-rw-r--r--` (User: read/write, Group: read, Others: read)
- Command 1:** `chmod u-w hosts` (Red box around `u-w`)
  - Effect: Remove write permission for the owner (User).
- Command 2:** `chmod g+w hosts` (Blue box around `g+w`)
  - Effect: Add write permission for the group (Group).
- Command 3:** `chmod o=w hosts` (Pink box around `o=w`)
  - Effect: Add write permission for others (Others).
- Final State:** `-r--rw--w-` (User: read, Group: read/write, Others: read/write)

# 파일 권한 변경

## 3. 수치를 통한 권한 부여 실습

```

user1@linux-test:~$ ls -l
total 4
-r--rw--w- 1 user1 user1 189 Mar 20 00:30 hosts
user1@linux-test:~$
user1@linux-test:~$
user1@linux-test:~$ chmod 754 hosts
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 4
-rwxr-xr-- 1 user1 user1 189 Mar 20 00:30 hosts
user1@linux-test:~$
  
```

소유자에게 모든 권한 부여  
 소유 그룹에 읽기, 실행 권한 부여  
 그 외 사용자에게 읽기 권한 부여

# 파일 기본 권한 설정

## • umask

- **기능** : 파일이나 디렉토리에 대한 기본 접근 권한을 출력하거나 변경
- **형식** : **umask** [옵션] [마스크 값]

마스크 값의 의미

파일과 디렉토리의 기본 권한

파일/디렉토리의 기본 권한

마스크 값

—  
(빼기)

생성 시 부여하지 않을 권한

==  
(적용)

생성 시 적용되는 권한

파일

디렉토리

u g o  
0666  
rw- rw- rw-

u g o  
0777  
rwx rwx rwx

0002  
... -w-

0002  
... -w-

0664  
rw- rw- r--

0775  
rwx rwx r-x

# 파일 기본 권한 설정

- umask

## 1. 현재 설정된 마스크 값 출력

```
user1@linux-test:~$ umask
0002
user1@linux-test:~$ umask -S
u=rwx,g=rwx,o=rx
user1@linux-test:~$
```

현재 설정된 기본 마스크 값  
출력

파일, 디렉토리 생성 시 적용되는  
권한을 문자로 출력

참고 : 계정에 따라 값이 다를 수 있음

```
root@linux-test:~# umask
0022
root@linux-test:~#
```

# 파일 기본 권한 설정

- umask

## 2. 기본 마스크 값 변경

```

user1@linux-test:~$ umask 0077
user1@linux-test:~$ umask
0077
user1@linux-test:~$
user1@linux-test:~$ touch sample.txt
user1@linux-test:~$ mkdir sample_dir
user1@linux-test:~$
user1@linux-test:~$ ls -l
total 8
-rwxr-xr-- 1 user1 user1 189 Mar 20 00:30 hosts
drwx----- 2 user1 user1 4096 Mar 20 02:15 sample_dir
-rw----- 1 user1 user1  0 Mar 20 02:15 sample.txt
user1@linux-test:~$

```

기본 마스크 값을 0077로 변경

=> 이후 생성되는 파일, 디렉토리에  
group, other에 대한 모든 권한 삭제

마스크 값 변경 후 파일, 디렉토리 생성

생성된 파일, 디렉토리에 대한 권한 확인

# 파일 특수 권한 - SetUID, SetGID

SetUID : **Set EUID** with the **Owner**

(외부) 명령이 실행되는 동안  
실행 파일의 소유자를 유효 사용자로 설정한다  
=> 해당 명령은 **파일 소유자의 권한으로 실행**된다

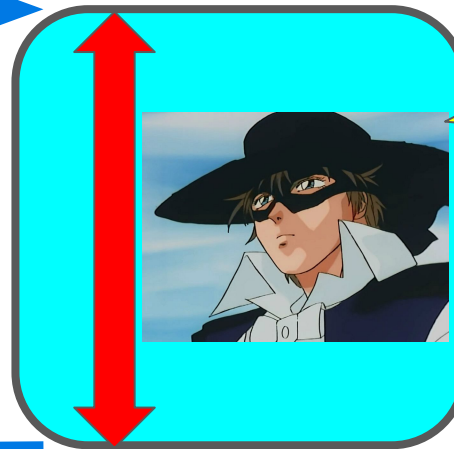
SetGID : **Set EGID** with the **Group**

(외부) 명령이 실행되는 동안  
실행 파일의 소유 그룹을 유효 사용자의 그룹으로 설정한다  
=> 해당 명령은 **파일 소유 그룹의 권한으로 실행**된다



사용자

SetUID 또는 SetGID가 설정된  
(외부) 명령 실행



명령이 실행 되는 구간에서는  
**다른 사용자/그룹의 권한**을  
가진다

...

왜 필요한 걸까?

명령 종료



사용자

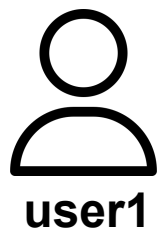
# 파일 특수 권한 - SetUID, SetGID

SetGID도 기본 개념은 SetUID와 거의 같음

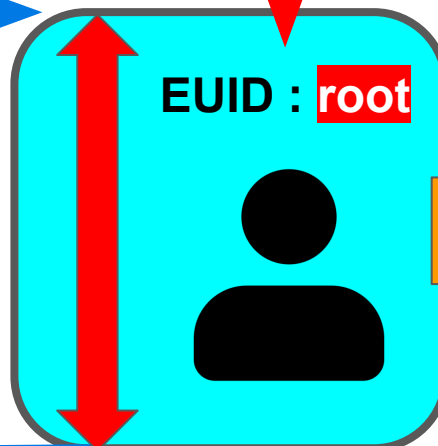
SetUID가 필요한 경우 예시

```
user1@linux-test:~$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 59640 Nov 29 21:25 /usr/bin/passwd
```

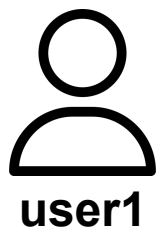
```
user1@linux-test:~$ ls -l /etc/shadow
-r----- 1 root shadow 1331 Mar 16 17:25 /etc/shadow
```



passwd 명령 실행



명령 종료



/etc/shadow

root : shadow

User(Owner)	Group	Other
r - -	- - -	- - -

root 권한을 통해 /etc/shadow 파일에 대한 사용 권한 획득  
=> root가 아닌 사용자가 자신의 패스워드를 수정할 수 있는 이유



# 파일 특수 권한 - Sticky Bit

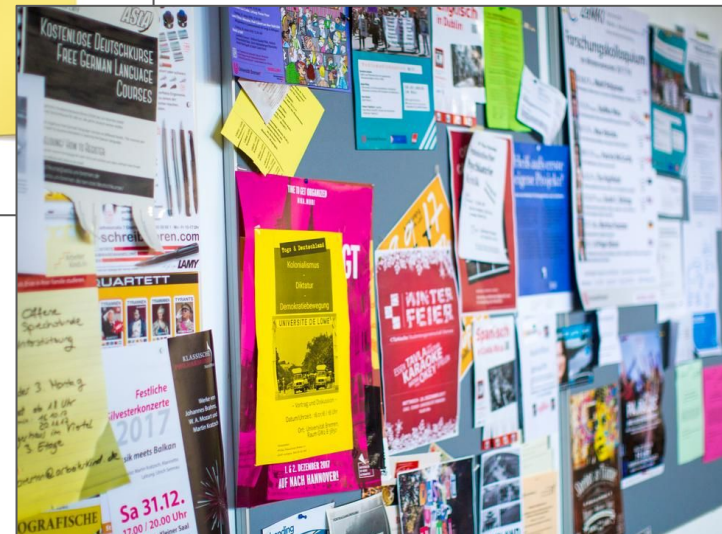


Sticky~~~

Sticky Note



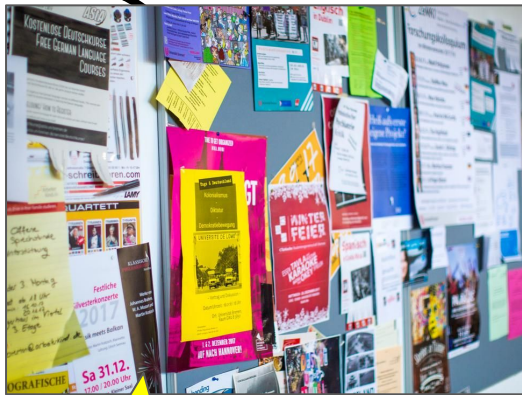
파일에 Sticky한 Bit를 설정한다?



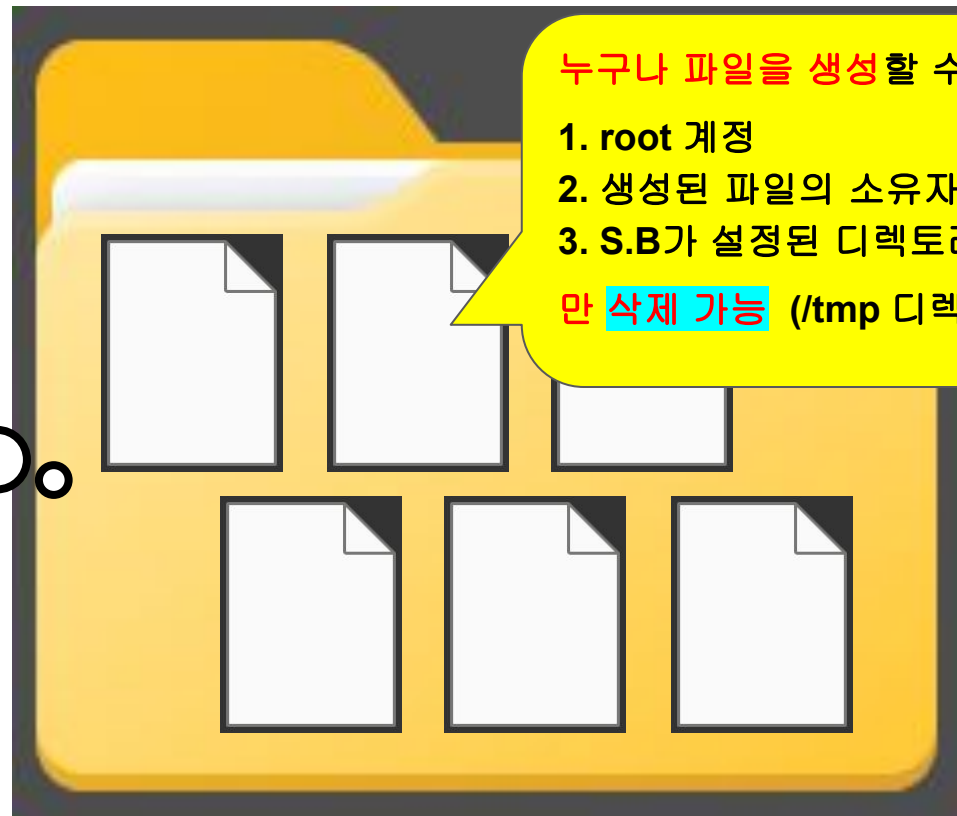
# 파일 특수 권한 - Sticky Bit

Sticky Bit는 디렉토리에 설정

Sticky Bit가 설정된 디렉토리



누구나 붙일 수 있지만  
아무나 못 떼는 게시판



누구나 파일을 생성할 수 있지만

1. root 계정
  2. 생성된 파일의 소유자
  3. S.B가 설정된 디렉토리의 소유자
- 만 삭제 가능 (/tmp 디렉토리)

# 파일 특수 권한 설정

SetUID 설정 :

SetUID : 맨 앞자리가 4

chmod 4755 sample

파일에 대한 실행(eXcute) 권한 필요

```
-rwsr-xr-x 1 root user001 0 Mar 20 14:28 sample
```

SetUID 설정 확인

SetGID 설정 :

SetGID : 맨 앞자리가 2

chmod 2755 sample

파일에 대한 실행(eXcute) 권한 필요

```
-rwxr-sr-x 1 user001 ugroup1 0 Mar 20 14:28 sample
```

SetGID 설정 확인

# 파일 특수 권한 설정

## SetUID 설정 및 확인 실습

### 1. 실습 준비

```
user1@linux-test:~$ su - user001
Password:
user001@linux-test:~$
user001@linux-test:~$ nano get_euid.c
```

1) root 권한을 사용할 수 있는 user001로 사용자 전환

2) 실습에 사용할 실행 파일 코드(C언어) 작성

```
GNU nano 2.9.3 get_euid.c

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main() {

    int euid = 0;

    euid = geteuid();

    if (euid != 0) {
        printf("This program running as %d\n", euid);
        exit(1);
    }

    printf("Running as root\n");

    return 0;
}
```

이 파일을 실행하는  
유효 사용자(EUID) 확인

3) 소스 코드를 통해 실행 파일 생성(컴파일)

```
user001@linux-test:~$ gcc -o get_euid get_euid.c
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 16
-rwxrwxr-x 1 user001 user001 8432 Mar 20 12:42 get_euid
-rw-rw-r-- 1 user001 user001 766 Mar 20 12:36 get_euid.c
user001@linux-test:~$
```

4) 실행 파일 생성 확인

# 파일 특수 권한 설정

## SetUID 설정 및 확인 실습

### 2. 실행 파일의 동작 비교

```

user001@linux-test:~$ ls -l
total 16
-rwxrwxr-x 1 user001 user001 8432 Mar 20 12:42 get_euid
-rw-rw-r-- 1 user001 user001 266 Mar 20 12:36 get_euid.c
user001@linux-test:~$ id
uid=1002(user001) gid=1004(user001) groups=1004(user001),27(sudo),100(users)
user001@linux-test:~$ ./get_euid
This program running as 1002
user001@linux-test:~$ sudo chown root get_euid
user001@linux-test:~$ sudo chmod 4775 get_euid
user001@linux-test:~$ ./get_euid
Running as root
user001@linux-test:~$

```

실행 파일이 **user001 (1002)** 권한으로 실행

실행 파일의 **소유자를 root로 변경**

실행 파일에 **SetUID 설정**

실행 파일이 **root** 권한으로 실행



# 파일 특수 권한 설정

## SetGID 설정 및 확인 실습

### 1. 실습 준비

```
user001@linux-test:~$ nano get_egid.c
```

1) 실습에 사용할 실행 파일 코드(C언어) 작성

```
GNU nano 2.9.3 get_egid.c Modified

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main() {
    gid_t egid = getegid();
    printf("Effective group ID: %d\n", egid);
    return 0;
}
```

이 파일을 실행하는  
유효 사용자의 그룹 ID(EGID) 확인

2) 소스 코드를 통해 실행 파일 생성(컴파일)

```
user001@linux-test:~$ gcc -o get_egid get_egid.c
user001@linux-test:~$
user001@linux-test:~$ ls -l
total 32
-rwxrwxr-x 1 user001 user001 8352 Mar 20 14:07 get_egid
-rw-rw-r-- 1 user001 user001 169 Mar 20 14:04 get_egid
.c
-rwsrwxr-x 1 root user001 843 Mar 20 12:42 get_euid
-rw-rw-r-- 1 user001 user001 169 Mar 20 12:36 get_euid
.c
user001@linux-test:~$
```

3) 실행 파일 생성 확인

# 파일 특수 권한 설정

## SetGID 설정 및 확인 실습

### 2. 실행 파일의 동작 비교

```

user001@linux-test:~$ ls -l
total 32
-rwxrwxr-x 1 user001 user001 8352 Mar 20 14:07 get_egid
-rw-rw-r-- 1 user001 user001 169 Mar 20 14:04 get_euid

user001@linux-test:~$ id
uid=1002(user001) gid=1004(user001) groups=1004(user001),27(sudo),100(users)

user001@linux-test:~$ ./get_egid
Effective group ID: 1004

user001@linux-test:~$ sudo chgrp ugroup1 get_egid
[sudo] password for user001:
user001@linux-test:~$ sudo chmod 2775 get_egid
user001@linux-test:~$ ./get_egid
Effective group ID: 3100

user001@linux-test:~$ cat /etc/group | grep 3100
ugroup1:x:3100:
  
```

소유 그룹 **user001 (1004)** 권한으로 실행

실행 파일의 **소유 그룹을 ugroup1로 변경**

실행 파일에 **SetUID 설정**

소유 그룹 **ugroup1** 권한으로 실행



# 파일 특수 권한 설정

Sticky Bit 설정 (디렉토리) :

chmod **1****777** sticky\_dir

SetGID : 맨 앞자리가 **1**

디렉토리에 대한 실행(eXcute) 권한 필요  
파일을 생성하기 위해서는 디렉토리에 쓰기(Write) 권한도 필요  
=> 삭제 권한만 제한됨

```
drwxrwxrwt 2 user001 user001 4096 Mar 20 15:15 sticky_dir
```

Sticky Bit 설정 확인

# 파일 특수 권한 설정

## Sticky Bit 설정 실습

### 1. 디렉토리 생성 및 Sticky Bit 설정

```

user001@linux-test:~$ mkdir sticky_dir
user001@linux-test:~$
user001@linux-test:~$ ls -al sticky_dir/
total 8
drwxrwxr-x 2 user001 user001 4096 Mar 20 15:15 .
drwxr-xr-x 6 user001 user001 4096 Mar 20 15:15 ..
user001@linux-test:~$
user001@linux-test:~$ chmod 1777 sticky_dir/
user001@linux-test:~$
user001@linux-test:~$ ls -al sticky_dir/
total 8
drwxrwxrwt 2 user001 user001 4096 Mar 20 15:15 .
drwxr-xr-x 6 user001 user001 4096 Mar 20 15:15 ..
user001@linux-test:~$

```

Sticky Bit 설정

Sticky Bit 확인

# 파일 특수 권한 설정

## Sticky Bit 설정 실습

### 2. Sticky Bit 동작 확인

```
user001@linux-test:~$ cd sticky_dir/
user001@linux-test:~/sticky_dir$
user001@linux-test:~/sticky_dir$ touch file1
user001@linux-test:~/sticky_dir$
user001@linux-test:~/sticky_dir$ ls -al
total 8
drwxrwxrwt 2 user001 user001 4096 Mar 20 15:28 .
drwxr-xr-x 6 user001 user001 4096 Mar 20 15:15 ..
-rw-rw-r-- 1 user001 user001    0 Mar 20 15:28 file1
```

user001 계정으로 sticky\_dir에  
파일 생성

sticky\_dir은 모든 사용자에게  
쓰기 권한 부여

```
user001@linux-test:~/sticky_dir$ su user1
```

user1 계정으로 전환

```
user1@linux-test:/home/user001/sticky_dir$ ls -al
total 8
drwxrwxrwt 2 user001 user001 4096 Mar 20 15:28 .
drwxr-xr-x 6 user001 user001 4096 Mar 20 15:15 ..
-rw-rw-r-- 1 user001 user001    0 Mar 20 15:28 file1
user1@linux-test:/home/user001/sticky_dir$
user1@linux-test:/home/user001/sticky_dir$ rm file1
rm: remove write-protected regular empty file 'file1'? y
rm: cannot remove 'file1': Operation not permitted
user1@linux-test:/home/user001/sticky_dir$
```

user1도 쓰기 권한이  
있어서 삭제가 가능할 것  
같지만,  
**Sticky Bit에 걸려서  
삭제되지 않음**

# Appendix – sshd



리눅스의 sshd에 대해 설명해줘



sshd는 Secure Shell Daemon의 약자로, 리눅스와 유닉스 기반 시스템에서 안전하고 암호화된 원격 접속을 가능하게 하는 서비스입니다. sshd는 ssh 프로토콜을 사용하여 클라이언트와 서버 간에 보안된 터널링을 설정하고, 암호화된 세션을 제공하여 불법적인 액세스와 스니핑을 방지합니다.

sshd는 기본적으로 리눅스 시스템에서 설치되어 있으며 SSH 클라이언트가 사용자의 로그인 정보와 함께 인증하여 원격으로 시스템에 액세스할 수 있도록 합니다. sshd는 기본적으로 22번 포트에서 수신 대기하며, 이 포트를 통해 접속하는 클라이언트는 인증되어야 합니다.

sshd는 다양한 보안 기능을 제공합니다. 예를 들어, 사용자 인증 시스템을 통해 사용자가 암호나 키를 제공하여 인증을 받을 수 있도록 합니다. 또한, 기본적으로 sshd는 패스워드 인증 대신에 키 기반 인증을 사용하는 것이 좋습니다. 또한, sshd는 IP 주소 기반의 액세스 제한 및 접속 시도에 대한 로그를 기록하는 기능도 제공합니다.

따라서, sshd는 리눅스 시스템에서 원격 접속을 안전하고 보안적으로 처리할 수 있도록 해주는 중요한 서비스 중 하나입니다.

# Thank You