redhat CentOS

ENTERPRISE LINUX ADMIN GUIDE

파일의 속성 관리

단원목표

- 파일의 속성 정보 확인
- 파일의 속성 정보 변경
- 파일의 소유권한 / 그룹권한 변경
- 파일의 퍼미션 변경
- 파일의 특수퍼미션 변경

파일의 속성 정보 변경 명령어

```
# ls -l file1
-rw-r--r-- 1 root root 0  5월 12 12:24 file1
```

- File Type :
- Permisstion Mode : chmod
- Link Count : ln
- Owner : chown
- Group : chgrp
- File Size :
- Modify Time : touch -t
- Fine Name : mv

파일의 속성 정보 변경 명령어에는 chown, chgrp, chmod, umask와 같은 명령어 등이 있다.

파일에 대한 정보로 소유권에 대한 기록을 담고 있다. 단, 소유권에 대한 권한 설정은 속성 정보가 아닌 사용자와 그룹에게 권한을 부여해 주는 것이므로 파일의 속성정보는 권한을 뺀 나머지의 정보가 모두 파일 속성에 해당 하는 것이다. (권한은 사용자가 가지고 있다!)

(파일의 속성 정보 변경 명령어)

```
# ls -l file1
-rw-r--r-- 1 root root 1945  6월 11 14:13 file1
```

File Type :
Permission Mode : chmod
Link Count : ln
Owner : chown
Group : chgrp
File Size :
Mtime : touch -t
File Name : mv

파일의 소유권한 / 그룹권한 변경

- **chown 명령어**

```
# chown user01 file1
# chown user01.other file1
# chown user01:other file1
# chown .other file1
# chown -R user01 dir1
```

- **chgrp 명령어**

```
# chgrp other file1
# chgrp -R other dir1
```

1 chown CMD

```
NAME
       chown - 파일의 소유주와 그룹을 바꾼다.

SYNOPSIS
       chown [-Rcfv] [--recursive] [--changes] [--help] [--version] [--silent]
       [--quiet] [--verbose] [user][:.][group] file...

DESCRIPTION
       이 문서는 더이상 최신 정보를 담고 있지않다. 그래서, 몇몇 틀릴 경우도 있
       고,  부족한 경우도 있을 것이다.  완전한 매뉴얼을 원하면, Texinfo 문서를
       참조하기 바란다.

       이 매뉴얼 페이지는 chown 명령의 GNU 버전에 대한 것이다.  chown 명령은
       주워진 file의 소유주와 그룹을 user group으로 바꾼다.  명령행에서 지정하
       는 순서에 따라 바뀐다.  만약 user만 지정했다면, 소유주만 바뀌고 그룹 은
       바 뀌지 않는다.  만약 점(.) 또는 콜론(:)으로 시작하는 group만 지정하면,
       소유주는 바뀌지 않고, 그룹만 바뀐다. 이것은 chgrp 명령과 같은 기능을 한
       다.    한편 user와 group을 점(.) 또는 콜론(:)으로 구분지어모두 사용하게
       되면, 소유주와 그룹은 모두 바뀐다.  이때 user, group은 그 이름이 올  수
       도 있고, ID가 올 수도 있다.

   OPTIONS
       -R, --recursive
              경로와 그 하위 파일들 모두를 바꾼다.
```

chown 명령어는 Unix 계통 시스템에서 파일의 소유권을 바꾸기 위해서(change the owner of a file)사용된다. 대부분의 경우, 이것은 오직 슈퍼 사용자에 의해서만 실행될 수 있다. 그들이 소유하고 있는 파일의 그룹을 바꾸고 싶어하는 비특권화된 (일반적인) 사용자들은 chgrp을 사용해야 한다.

```
# ls -l file1
-rw-r--r-- 1 root root 0 Jan 10 12:43 file1
```

[명령어 형식]

```
# chown user01 file1
# chown user01.other file1  (# chown user01:other file1)
# chown .other file1
# chown -R user01 dir1
# chown -R user01:other dir1
```

[명령어 옵션]

옵션	설명
-c	바뀌어지는 파일들에 대해서만 자세하게 보여준다.
-f	바뀌어 지지 않는 파일들에 대해서 오류 메시지를 보여주지 않는다.
-v	작업 상태를 자세히 보여준다.
-R	경로와 그 하위 파일들 모두를 바꾼다.

[EX1] chown 명령어 실습

```
# cd /test ; rm -rf /test/*

# touch file1
# ls -l file1
-rw-r--r-- 1 root root 0 Feb 14 10:41 file1

# chown fedora file1
# ls -l file1
-rw-r--r-- 1 fedora root 0 Feb 14 10:41 file1

# chown .user01 file1
# ls -l file1
-rw-r--r-- 1 fedora user01 0 Feb 14 10:41 file1

# chown root:root file1  (# chown root.root file1)
# ls -l file1
-rw-r--r-- 1 root root 0 Feb 14 10:41 file1
```

[참고] 그룹 생성 여부 확인

```
# cat /etc/group | grep user01          /* user01라는 그룹이 있는지 파일 안에서 검색 */
user01:x:502:

# cat /etc/group | grep root            /* root 라는 그룹 확인 */
root:x:0:root                           /* root 그룹 */
bin:x:1:root,bin,daemon                 /* root 사용자가 소속된 그룹 */
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
disk:x:6:root
wheel:x:10:root
pkcs11:x:103:root
```

[참고] useradd 명령어의 수행

```
# useradd user02
# passwd user02

# grep user02 /etc/passwd
# grep user02 /etc/group
```

```
# cat /etc/passwd          UID
# cat /etc/group           GID
```

[EX2] chown -R 옵션 실습

(실습용 구조)

```
/test -- dir1 --+-- dir2 -- dir3
                |
                +-- file1
                |
                +-- file2
```

```
# cd /test ; rm -rf /test/*

# mkdir -p dir1/dir2/dir3
# touch dir1/file1
# touch dir1/file2

# ls -lR dir1      (# find . -ls)
```

```
dir1:
total 4.0K
drwxr-xr-x 3 root root 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 root root    0 Feb 14 10:47 file1
-rw-r--r-- 1 root root    0 Feb 14 10:47 file2

dir1/dir2:
total 4.0K
drwxr-xr-x 2 root root 4.0K Feb 14 10:47 dir3/

dir1/dir2/dir3:
total 0
```

```
# chown -R fedora:fedora dir1
# ls -lR dir1    (# find . -ls)
```

```
dir1:
total 4.0K
drwxr-xr-x 3 fedora fedora 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 fedora fedora    0 Feb 14 10:47 file1
-rw-r--r-- 1 fedora fedora    0 Feb 14 10:47 file2

dir1/dir2:
total 4.0K
drwxr-xr-x 2 fedora fedora 4.0K Feb 14 10:47 dir3/

dir1/dir2/dir3:
total 0
```

```
# chown -R .user01 dir1
# ls -lR dir1    (# find . -ls)
```

```
dir1:
total 4.0K
drwxr-xr-x 3 fedora user01 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 fedora user01    0 Feb 14 10:47 file1
-rw-r--r-- 1 fedora user01    0 Feb 14 10:47 file2

dir1/dir2:
total 4.0K
drwxr-xr-x 2 fedora user01 4.0K Feb 14 10:47 dir3/

dir1/dir2/dir3:
total 0
```

```
# chown -R root:root dir1
# ls -lR dir1
```

```
dir1:
total 4.0K
drwxr-xr-x 3 root root 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 root root    0 Feb 14 10:47 file1
-rw-r--r-- 1 root root    0 Feb 14 10:47 file2

dir1/dir2:
total 4.0K
drwxr-xr-x 2 root root 4.0K Feb 14 10:47 dir3/

dir1/dir2/dir3:
total 0
```

2 chgrp CMD

NAME
chgrp - 파일의 사용자 그룹을 바꾼다.

SYNOPSIS
chgrp [-Rcfv] [--recursive] [--changes] [--silent] [--quiet] [--verbose] [--help] [--version] group file...

DESCRIPTION
이 문서는 더이상 최신 정보를 담고 있지않다. 그래서, 몇몇 틀릴 경우도 있고, 부족한 경우도 있을 것이다. 완전한 매뉴얼을 원하면, Texinfo 문서를 참조하기 바란다.

이 매뉴얼 페이지는 chgrp 명령의 GNU 버전에 대한 것이다. chgrp 명령은 주 워진 file의 그룹을 지정한 group으로 바꾼다. 여기서 group으로 사용될 수 있는 것은 그 그룹의 이름이나, 그 그룹의 ID이다.

OPTIONS
-R, --recursive
주로 file 이름으로 경로를 사용해서, 그 안에 있는 모든파일도 함께 gourp으로 바꾼다.

파일의 속성정보 중 그룹명을 변경하는 명령어이다.

```
# ls -l file1
-rw-r--r-- 1 root root 0 Jan 10 12:43 file1
```

[명령어 형식]
chgrp user01 file1

[명령어 옵션]

옵션	설명
-c	작업 상태를 자세히 보여주나, 바뀌어 지는 것만 보여준다.
-f	그룹이 바뀌어 지지 않는 파일들에 대한 오류 메시지를 보여주지 않는다.
-v	작업 상태를 자세히 보여준다.
-R	주로 file 이름으로 경로를 사용해서, 그 안에 있는 모든파일도 함께 gourp으로 바꾼다.

[EX1] chgrp 명령어 실습
```
# cd /test
# touch file1
# ls -l file1
-rw-r--r-- 1 root root 0 Feb 14 10:52 file1
```

```
# chgrp fedora file1
# ls -l file1
-rw-r--r-- 1 root fedora 0 Feb 14 10:52 file1
```

[EX2] chgrp -R 옵션 실습

```
# mkdir -p dir1/dir2/dir3
# ls -lR
```

```
.:
total 4.0K
drwxr-xr-x 3 root root   4.0K Feb 14 10:47 dir1/
-rw-r--r-- 1 root fedora    0 Feb 14 10:52 file1

./dir1:
total 4.0K
drwxr-xr-x 3 root root 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 root root    0 Feb 14 10:47 file1
-rw-r--r-- 1 root root    0 Feb 14 10:47 file2

./dir1/dir2:
total 4.0K
drwxr-xr-x 2 root root 4.0K Feb 14 10:47 dir3/

./dir1/dir2/dir3:
total 0
```

```
# chgrp fedora dir1
# ls -lR
```

```
.:
total 4.0K
drwxr-xr-x 3 root fedora 4.0K Feb 14 10:47 dir1/
-rw-r--r-- 1 root fedora    0 Feb 14 10:52 file1

./dir1:
total 4.0K
drwxr-xr-x 3 root root 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 root root    0 Feb 14 10:47 file1
-rw-r--r-- 1 root root    0 Feb 14 10:47 file2

./dir1/dir2:
total 4.0K
drwxr-xr-x 2 root root 4.0K Feb 14 10:47 dir3/

./dir1/dir2/dir3:
total 0
```

```
# chgrp -R user01 dir1    (# chown -R root:user01 dir1)
# ls -lR
```

```
.:
total 4.0K
drwxr-xr-x 3 root user01 4.0K Feb 14 10:47 dir1/
-rw-r--r-- 1 root fedora    0 Feb 14 10:52 file1

./dir1:
total 4.0K
drwxr-xr-x 3 root user01 4.0K Feb 14 10:47 dir2/
-rw-r--r-- 1 root user01    0 Feb 14 10:47 file1
-rw-r--r-- 1 root user01    0 Feb 14 10:47 file2

./dir1/dir2:
total 4.0K
drwxr-xr-x 2 root user01 4.0K Feb 14 10:47 dir3/

./dir1/dir2/dir3:
total 0
```

파일의 퍼미션 변경

- **퍼미션 변경 방법**
  - **심볼릭 모드(Symbolic Mode)**
  - **옥탈 모드(Octal Mode)**

- **심볼릭 모드(Symbolic Mode)**

```
# chmod u + r file1
        g - w
        o = x

# chmod u+x file1
# chmod g+x,o+x file1
# chmod u-r file1
# chmod u-wx file1
# chmod a=rwx file1
```

3 chmod CMD

NAME
chmod - 파일 접근 권한을 바꾼다.

SYNOPSIS
chmod [-Rcfv] [--recursive] [--changes] [--silent] [--quiet] [--verbose] [--help] [--version] mode file...

DESCRIPTION
이 문서는 더이상 최신 정보를 담고 있지않다. 그래서, 몇몇 틀릴 경우도 있고, 부족한 경우도 있을 것이다. 완전한 매뉴얼을 원하면, Texinfo 문서를 참조하기 바란다.

이 매뉴얼 페이지는 chmod 명령의 GNU 버전에 대한 것이다. chmod 풀그림은 지 정한 mode로 지정한 파일의 권한을 바꾼다. mode로 사용될 수 있는 것은 일군의 기호들이나(symbolic mode), 그 기호들과 상응하는 8진수 숫자 들 이다.

심 블릭 모드의 표현 방식은 '[ugoa...][[+-=][rwxXstugo...]...][,...]' 이렇고, 또한 쉼표(,)로 구분하여, 여러개의 기호군들을 사용할 수 있다.

처음에 나오는 'ugoa'는 소유자(u), 그룹(g), 다른 사용자(o), 모든 사 용자(a) 를 뜻하며, 이것을 생략하면, 모든 사용자로 간주한다. 하지만 umask로 지정된 bit는 영향받지 않는다(?).

'+'는 권한 부여, '-'는 권한 박탈, '=' 원래 권한.

'rwxXstugo'는 새롭게 부여할 권한. 읽기(r), 쓰기(w), 실행(디렉토리일 경우 는 접근허용)(x), 파일이 디렉토리이거나, 이미 다른 사용자에게는 실행 권한이 있는파일의 실행(X), 소유주와 그룹만 실행(s), 스왑 장치에서 풀 그림 텍스트저장(?)(t), 소유주 권한(u), 그룹 권한(g), 다른 사용자 권한(o)

예) chmod a+w foo : foo 파일을 모든 사용자가 쓸 수 있게 한다.

8 진수를 사용하는 방법은 4,2,1 숫자를 더한 값을 100단위에는소유주, 10단위에는 그룹, 1단위에는 다른 사용자로 지정해서 사용한다. 4는 읽기, 2 는 쓰기, 1은 실행.

예) chmod 666 foo : foo 파일을 모든 사용자가 쓸 수 있게 한다.

chmod 풀그림은 심블릭 링크 파일에 대해서는 아무런 작업도 하지 않는다. 즉, 심블릭 링크의 권한은 그 심블릭 대상이 된 파일의 권한을 따른다.

OPTIONS
-R, --recursive
파일과 그 디렉토리의 아래까지 모두 바꾼다.

파일이나 디렉토리를 새로운 권한으로 변경하는 명령이다. 파일의 소유자나 관리자만이 chmod를 사용할 수 있으며 파일의 소유자, 파일의 그룹, 다른 사용자로 나누어 각각의 권한을 설정 할 수 있다.

```
# ls -l file1
-rw-r--r-- 1 root root 0 Jan 10 12:43 file1
```

[명령어 형식]

```
# chmod u+x file1
# chmod 755 file1
```

■ 퍼미션(Permission)을 변경하는 방법

- 심볼릭 모드(Symbolic Mode) : **# chmod u+x file1**
- 옥탈 모드(Octal Mode) : **# chmod 744 file1**

(1) 심볼모드(symbolic mode)를 이용한 권한 변경

[사용자 기호]

기호		설명
u	user	파일/디렉토리의 소유자
g	group	파일/디렉토리의 그룹
o	other	다른 사용자
a	all	소유자, 그룹, 다른 사용자 모두(아무 표시 안할 경우 기본적으로 설정됨)

[설정 기호]

기호		설명
+	퍼미션 허가	지정한 퍼미션을 허가한다.
-	퍼미션 금지	지정된 퍼미션을 금지시킨다.
=	퍼미션 지정	지정한 퍼미션만 허가하고 나머지는 금지 시킨다. 설명 : 이전에 권한을 어떻게 줬던 지금 설정해주는 권한으로 대체시켜서 사용 가능 (+,-는 내가 이전에 어떠한 권한을 설정해 줬는지 확인해야 하지만 =는 이전에 어떻게 설정하였던 상관없이 설정이 가능한 것이다.)

[권한 기호]

r	w	x
read	write	excute

설명 : 읽기 권한이 없으면 파일 안에 있는 내용을 볼 수 없다. 따라서 파일을 수정하기 위해서는 반드시 read권한을 부여해야지 파일을 열어서 보고 수정 할 수가 있는 것이다. 만약에 write권한만 있다면 파일을 열어 볼 수 없으므로 수정이 불가능 한 것이다. 따라서 파일을 수정하려면 read&write권한이 모두 있어야 한다.

```
# ls -l file1
-rw-r--r-- 1 root    fedora     0 Aug 17 16:06 file1

# chmod  u + r  file1
         g - w
         o = x
         a

         ■■심볼의 정의
         u(User)
         g(Group)
         o(Other)
         a(all)

         +(Add)
         -(Deny)
         =(equal)

         r(Read)
         w(Write)
         x(Excute)
```

[EX1] 심볼 모드를 사용한 퍼미션 변경

```
# cd /test
# rm -rf /test/*

# touch file1
# ls -l file1
-rw-r--r-- 1 root root 0 Feb 14 11:12 file1

# chmod u+x file1
# ls -l file1
-rwxr--r-- 1 root root 0 Feb 14 11:12 file1*

# chmod g-r file1
# ls -l file1
-rwx---r-- 1 root root 0 Feb 14 11:12 file1*

# chmod u-x,g+x file1
# ls -l file1
-rw---xr-- 1 root root 0 Feb 14 11:12 file1*

# chmod a=rwx file1
# ls -l file1
-rwxrwxrwx 1 root root 0 Feb 14 11:12 file1*
```

(2) 수치모드(octal mode)를 이용한 권한 변경

소유자권한비트			그룹권한비트			기타권한비트		
r	w	x	r	w	x	r	w	x
1	1	1	1	1	1	1	1	1
4	2	1	4	2	1	4	2	1

파일 소유자 권한 : 400 = 읽기 권한, 200 = 쓰기 권한, 100 = 실행 권한
그룹 사용자 권한 : 40 = 읽기 권한, 20 = 쓰기 권한, 10 = 실행 권한
기타 사용자 권한 : 4 = 읽기 권한, 2 = 쓰기 권한, 1 = 실행 권한

```
--- : 권한 없음 0
--x : 실행 권한 1
-w- : 쓰기 권한 2
-wx : 쓰기 실행 3
r-- : 읽기 권한 4
r-x : 읽기 실행 5
rw- : 읽기 쓰기 6
rwx : 읽기 쓰기 실행 7
```

 # chmod 744 file1 (rwxr--r--)

[EX1] Octal Mode 실습

```
# cd /test ; rm -rf /test/*
# touch file1
# ls -l file1
-rw-r--r-- 1 root root 0 Feb 14 11:20 file1
```

```
# chmod 744 file1
# ls -l file1
-rwxr--r-- 1 root root 0 Feb 14 11:20 file1*
```

```
# chmod 754 file1
# ls -l file1
-rwxr-xr-- 1 root root 0 Feb 14 11:20 file1*
```

파일과 디렉토리 퍼미션의 정확한 의미

- 파일에 대한 퍼미션
  - r (read) : 파일을 읽을 수 있는 권한
  - w (write) : 파일을 수정할 수 있는 권한
  - x (excute) : 파일을 실행 할 수 있는 권한

- 디렉토리에 대한 퍼미션
  - r (read) : **디렉토리에서** ls CMD 수행권한
  - w (write) : 디렉토리안의 파일들의 생성과 삭제를 할 수 있는 권한
    touch CMD , rm CMD
  - x (excute) : 디렉토리내부로의 cd CMD 수행할 수 있는 권한

(파일과 디렉토리의 퍼미션의 정확한 의미)

파일	디렉토리
r	r(ls CMD)
w	w(생성 & 삭제)
x	x(cd CMD)

[TERM1] fedora 사용자 터미널
ssh fedora@localhost
fedora 사용자로 로그인

$ ls -ld /home/fedora

```
drwx------ 7 fedora fedora 4096  1월 27 02:45 /home/fedora
```

$ mkdir dirtest
$ touch dirtest/test2.txt
$ ls -lR

```
drwxrwxr-x 2 fedora fedora 4096  1월 27 02:45 dirtest
./dirtest:
합계 0
-rw-rw-r-- 1 fedora fedora 0  1월 27 02:45 test2.txt
```

[TERM2] user01 사용자 터미널
ssh user01@localhost
user01 사용자로 로그인

$ cd ~fedora ($ cd /home/fedora)

```
-bash: cd: ../fedora: Permission denied
```

```
drwx------ 7 fedora fedora ..... /home/fedora
             A      A
             |      |
             V      V
           user01 user01
```

$ id fedora
$ id user01

$ cd ~fedora/dirtest ($ cd /home/fedora/dirtest)

```
-bash: cd: ../fedora/dirtest: Permission denied
```

-> 하위디렉토리에 접근 권한이 있더라도 상위 디렉토리에 접근 불가하므로 하위디렉토리인 dirtest 디렉토리도 접근 불가

$ rm -f ~fedora/dirtest/test2.txt ($ rm -f /home/feodra/dirtest/test2.txt)

```
rm: cannot remove `../fedora/dirtest/test2.txt': Permission denied
```

[TERM1] fedora 사용자 터미널
$ id

```
uid=500(fedora) gid=500(fedora) groups=500(fedora)
```

$ chmod 757 /home/fedora
$ ls -ld /home/fedora

```
drwxr-xrwx 7 fedora fedora 4096  1월 27 02:45 /home/fedora
```

$ chmod 755 dirtest
$ touch dirtest/test3
$ chmod 646 dirtest/test3
$ ls -lR

```
drwxr-xr-x 2 fedora fedora 4096  1월 27 02:56 dirtest
./dirtest:
합계 0
-rw-r--rw- 1 fedora fedora 0  1월 27 02:56 test3
-rw-rw-r-- 1 fedora fedora 0  1월 27 02:45 test2.txt
```

```
/home/fedora (rwxr-xrwx fedora fedora)
      |
      +----- dirtest (rwxr-xr-x fedora fedora)
                  |
                  +----- test3 (rw-r--rw- fedora fedora)
```

[TERM2] user01 사용자 터미널
$ id

```
uid=501(user01) gid=501(user01) groups=501(user01)
```

$ cd ~fedora ($ cd /home/fedora)
$ cd dirtest
$ rm -f test3

```
rm: cannot remove `test3': Permission denied
```

$ mkdir dirtest1

```
mkdir: cannot create directory `dirtest1': Permission denied
```

[TERM1] fedora 사용자 터미널
$ id

```
uid=500(fedora) gid=500(fedora) groups=500(fedora)
```

$ chmod 757 dirtest
$ ls -lR

```
.:
합계 4
drwxr-xrwx 2 fedora fedora 4096  3월 16 15:20 dirtest

./dirtest:
합계 0
-rw-rw-r-- 1 fedora fedora 0  3월 16 15:15 test3.txt
```

```
/home/fedora (rwxr-xrwx fedora fedora)
      |
      +----- dirtest (rwxr-xrwx fedora fedora)
                  |
                  +----- test3 (rw-r--rw- fedora fedora)
```

[TERM2] user01 사용자 터미널

```
$ id
```

```
uid=501(user01) gid=501(user01) groups=501(user01)
```

```
$ rm -f test3
$ touch test4.txt
$ ls -l
```

```
-rw-r--rw- 1 fedora fedora 0  3월 16 15:20 test2.txt
-rw-rw-r-- 1 user01 user01 0  3월 16 15:23 test4.txt
```

umask 명령어

- **umask CMD**
  파일과 디렉토리가 생성될 때 기본 퍼미션을 조정할 수 있는 명령어

- **umask CMD**

  관리자 : **/etc/bashrc**
  사용자 : **$HOME/.bashrc**

4 umask CMD

```
NAME
       umask - get or set the file mode creation mask

SYNOPSIS
       umask [-S][mask]

DESCRIPTION
       The umask utility shall set the file mode creation mask of the  current
       shell  execution  environment (see Shell Execution Environment ) to the
       value specified by the mask operand. This mask shall affect the initial
       value  of  the  file  permission bits of subsequently created files. If
       umask is called in a subshell or separate  utility  execution  environ-
       ment, such as one of the following:

              (umask 002)
              nohup umask ...
              find . -exec umask ... W;

       it  shall  not affect the file mode creation mask of the caller's envi-
       ronment.

       If the mask operand is not specified, the umask utility shall write  to
       standard  output  the value of the invoking process' file mode creation
       mask.
```

파일이나 디렉토리 생성시에 파일과 디렉토리에는 기본적으로 적용되는 퍼미션이 있다. 기본적으로 설정되는 퍼미션의 경우 umask에 의해 결정이 된다. umask는 디렉토리와 파일의 기본 퍼미션을 결정해주는 명령어이다.

[기본 퍼미션(Default Permission) 변경]

	파일	디렉토리
Default Permission	666	777
umask	022	022
생성 기본퍼미션	644	755

많이 쓰이는 umask 값은 002, 022, 027를 사용한다.

[명령어 형식]
umask
umask 027
umask 022

[EX1] umask 간단한 실습
cd /test
rm -rf /test/*

umask
```
0022
```

touch file1
mkdir dir1
ls -l
```
-rw-r--r-- 1 root root    0 Feb 11 10:45 file1
drwxr-xr-x 2 root root 4096 Feb 11 10:47 dir1
```

(666 - 022 = 644)
(777 - 022 = 755)

■ Default Permission

File	Directory		
666	777	666	777
022	022	027	027
644	755	640	750

umask 002
umask
```
0002
```

touch file2
mkdir dir2
ls -ld *2
```
drwxrwxr-x 2 root root 4.0K Feb 14 12:04 dir2/
-rw-rw-r-- 1 root root    0 Feb 14 12:04 file2
```

■ Default Permission

File	Directory
666	777
002	002
664	775

umask 027
umask
```
0027
```

touch file3 (666 - 027 = 640)
mkdir dir3 (777 - 027 = 750)
ls -ld *3
```
drwxr-x--- 2 root root 4.0K Feb 14 12:05 dir3/
-rw-r----- 1 root root    0 Feb 14 12:05 file3
```

■ Default Permission

File	Directory
666	777
027	027
640	750

[EX2] /etc/bashrc 파일에 등록된 umask 확인

(관리자) /etc/bashrc
(사용자) $HOME/.bashrc

 # cat /etc/bashrc

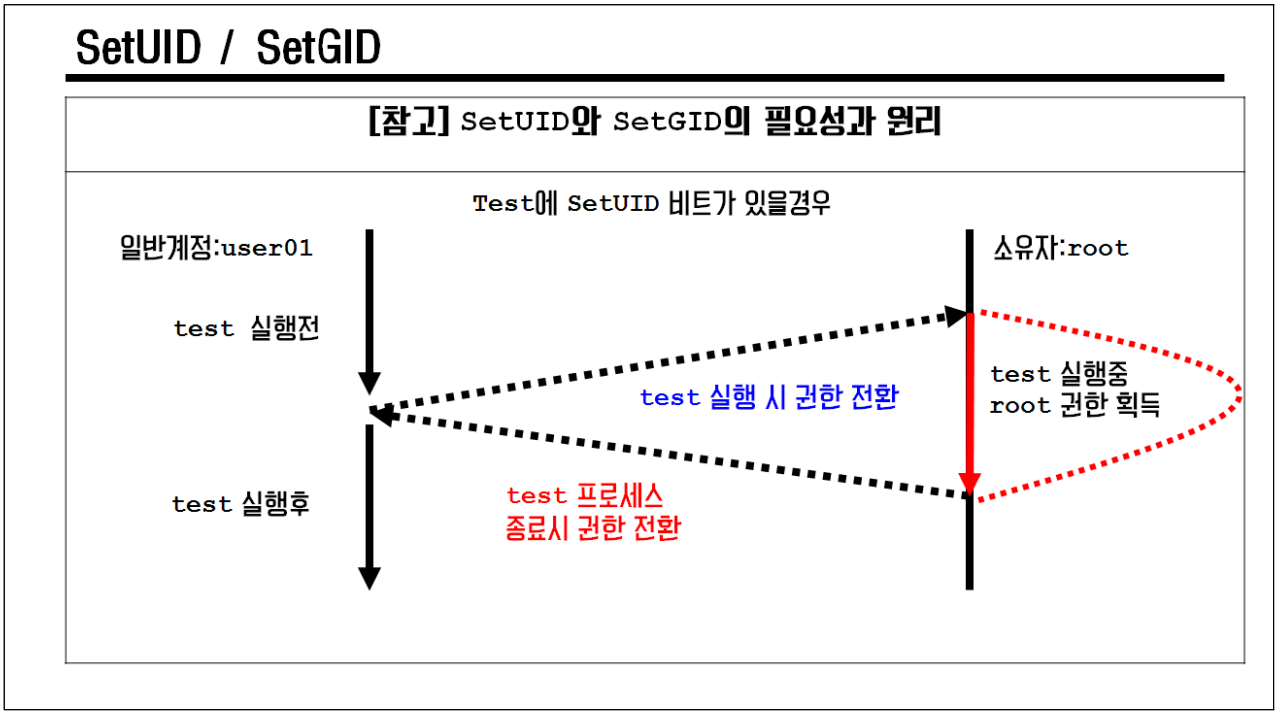
```
..... (중략) .....
    # By default, we want umask to get set. This sets it for non-login shell.
    # You could check uidgid reservation validity in
    # /usr/share/doc/setup-*/uidgid file
    if [ $UID -gt 99 ] && [ "`id -gn`" = "`id -un`" ]; then
       umask 002
    else
       umask 022
    fi
..... (중략) .....
```

-> umask 확인

5 SetUID/SetGID

파일에 대한 소유권을 잠시 다른 사용자에게 빌려 줌으로 인해 소유권이 없는 사용자가 잠시 동안 파일에 대한 소유권으로 권한을 행사 할 수 있는 것을 말한다.

공유한 디렉토리나 파일에 대한 특별한 퍼미션을 부여하는 것으로서 파일 소유자(owner)나 superuser만이 파일에 대해서는 setuid와 setgid를 설정하고,디렉토리에 대해서는 setgid 퍼미션을 설정 할 수 있다. absolute mode (octal mode)나 symbolic mode를 사용하여 지정하거나 해지할 수 있다.



[참고] SetUID와 SetGID를 실행할 수 있는 프로그램

실행 권한(x)이 주어진 프로그램에 setuid 퍼미션이 주어지면 누구에게나 그 프로그램의 소유자처럼 그 프로그램을 실행할 권한이 주어지고, 또한 어느 누구에게나 그 프로그램의 그룹에 속한 것처럼 할 수 있다. 즉, 실행 권한(x)을 가진 프로그램이 setuid와 setgid 퍼미션이 주어지면 그 프로그램의 owner나 group으로부터 UID와 GID를 얻는다. 이는 해당 프로그램이 시작될 때 프로세스로부터 UID와 GID를 상속받는 것과는 다르다.

[참고] SetUID를 가진 파일의 예시

패스워드를 변경할 수 있는 권한을 root의 권한이다. 그렇기에 일반 사용자는 패스워드 권한을 변경할 수 없다. 일반 사용자도 패스워드를 변경할 때 만큼은 root가 되도록 만들어주어야 한다. 그렇기 위해선 /usr/bin/passwd실행파일의 권한이 root의 SetUID 권한이 부여되어 있으면 된다.

[특수권한 SetUID, SetGID, sticky bit 퍼미션]

특수권한			소유자권한비트			그룹권한비트			기타권한비트		
SetUID	SetGID	sticky bit	r	w	x	r	w	x	r	w	x
1	1	1	1	1	1	1	1	1	1	1	1
4	2	1									
천			백			십			일		

```
# chmod 755 file1
        # chmod 0755 file1

# chmod 22 file1
        # chmod 0022 file1

# chmod 4755 file1                  # chmod 6755 file1
        (0755 : rwxr-xr-x)                (0755 : rwxr-xr-x)
        (4755 : rwsr-xr-x)                (6755 : rwsr-sr-x)

# chmod 2755 file1
        (0755 : rwxr-xr-x)
        (2755 : rwxr-sr-x)

# chmod 1777 dir1
```

(0777 : rwxrwxrwx)
(1777 : rwxrwxrwt)

[EX] SetUID,SetGID 권한 부여

```
# cd /test
# rm -rf /test/*

# touch file1
# ls -l file1
```

```
-rw-r--r-- 1 root root 0 Feb 11 09:37 file1
```

```
# chmod 755 file1
# ls -l file1
```

```
-rwxr-xr-x 1 root root 0 Feb 11 09:37 file1
```

```
# chmod 4755 file1
# ls -l file1
```

```
-rwsr-xr-x 1 root root 0 Feb 11 09:37 file1
```

```
# chmod 2755 file1
# ls -l file1
```

```
-rwxr-sr-x 1 root root 0 Feb 11 09:37 file1
```

```
# chmod 6755 file1
# ls -l file1
```

```
-rwsr-sr-x 1 root root 0 Feb 11 09:37 file1
```

[참고] 권한 설정시 유의사항

```
# chmod 4755 file1 (rwxr-xr-x  ->  rwsr-xr-x)
# chmod 4100 file1 (--x------  ->  --s------)
# chmod 4655 file1 (rw-r-xr-x  ->  rwSr-xr-x)

# chmod 2755 file1 (rwxr-xr-x  ->  rwxr-sr-x)
# chmod 2010 file1 (-----x---  ->  -----s---)
# chmod 2765 file1 (rwxrw-r-x  ->  rwxrwSr-x)
원래는 권한이 없어 - 로 표시되어야 하는 자리지만 SETID가 들어가면서 이전 권한이 보이지 않게되어 대문자로 표시된다.
```

[EX] SetUID 확인

(일반사용자) $ passwd
-> /etc/passwd(user01:x:501:501::/home/user01:/bin/bash)
-> /etc/shadow(user01:1DOa001Ns$Src3NlrAeQH8YlQwJ44bp1:15911:0:99999:7:::)

```
# ls -l /usr/bin/passwd /etc/passwd /etc/shadow
```

```
-rw-r--r-- 1 root root 2936  Feb 11 06:29 /etc/passwd
-r--------- 1 root root 1275  6월 11 15:17 /etc/shadow
-rwsr-xr-x 1 root root 22984 Jan  7 2007  /usr/bin/passwd
```

```
# chmod 755 /usr/bin/passwd
# ls -l /usr/bin/passwd
```

```
-rwxr-xr-x 1 root root 22984  1월  7  2007 /usr/bin/passwd
```

```
# su - fedora
$ passwd
```

```
Changing password for user fedora.
Changing password for fedora
(current) UNIX password: (fedora)
passwd: Authentication token manipulation error
```

```
$ exit
# passwd fedora
```

```
Changing password for user fedora.
New UNIX password: (fedora)
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password: (fedora)
passwd: all authentication tokens updated successfully.
```

(원복) /usr/bin/passwd (rwxr-xr-x -> rwsr-xr-x)

```
# chmod 4755 /usr/bin/passwd
```

[EX] fedora가 user01 사용자의 권한을 빌리는 경우
telnet localhost
user01 사용자로 로그인

$ chmod 775 /home/user01
$ cp /bin/touch /home/user01
$ ls -l

```
-rwxr-xr-x 1 user01 user01 42284  3월 17 01:20 touch
```

/bin/touch   (rwxr-xr-x 1 root root)
/home/user01/touch (rwxr-xr-x 1 user01 user01)

$ ./touch file1
$ ls -l

```
-rw-rw-r-- 1 user01 user01     0  3월 17 01:20 file1
-rwxr-xr-x 1 user01 user01 42284  3월 17 01:20 touch
```

/home/user01 (rwxrwxr-x user01 user01)
        |
        +--- file1((0)생성가능)

$ chmod 4755 touch
$ ls -l

```
-rw-rw-r-- 1 user01 user01     0  3월 17 01:20 file1
-rwsr-xr-x 1 user01 user01 42284  3월 17 01:33 touch
```

$ su - fedora
-> fedora 암호 입력

$ cd /home/user01
$ touch file2

```
touch: cannot touch `file2': 허가 거부됨
```

/home/user01 (rwxrwxr-x user01 user01)
        |
        +--- file2((X)생성 불가능)

/bin/touch   (rwxr-xr-x 1 root root)
/home/user01/touch (rwsr-xr-x 1 user01 user01)

$ ./touch file2
$ ls -l

```
-rw-rw-r-- 1 user01 user01     0  3월 17 01:20 file1
-rw-rw-r-- 1 user01 fedora     0  3월 17 01:20 file2
-rwsr-xr-x 1 user01 user01 42284  3월 17 01:33 touch
```

$ exit
$ id
-> user01 사용자 확인

$ chmod 2755 touch (rwxr-xr-x ->)
$ ls -l touch

```
-rwxr-sr-x  user01 user01  size mtime  touch
```

$ su - fedora
-> fedora 사용자 암호 입력

$ cd /home/user01
$ ./touch file3
$ ls -l

```
-rw-rw-r-- 1 user01 user01     0 Apr 21 15:59 file1
-rw-rw-r-- 1 user01 fedora     0 Apr 21 16:02 file2
-rw-rw-r-- 1 fedora user01     0 Apr 21 16:04 file3
-rwxr-sr-x 1 user01 user01 42284 Apr 21 15:59 touch
```

/home/user01   (rwxrwxr-x user01 user01)
        |
        +--- file3((0)생성가능)

$ exit
$ exit
#

[EX] bash쉘의 SetUID 권한 부여

[TERM1] root 사용자 터미널
```
# ls -l /bin/bash
```
```
-rwxr-xr-x 1 root root 735004 Jan 22  2009 /bin/bash
```

```
# cp /bin/bash /test
# cd /test
# ls -l bash
```
```
-rwxr-xr-x 1 root root 735004 Feb 11 09:06 bash
```

```
        /bin/bash   (-rwxr-xr-x 1 root root 719K Jul 22  2011 /bin/bash*)
        /test/bash  (-rwxr-xr-x 1 root root 719K Jan 13 11:33 bash*)

# bash
# ps
# exit
```
```
exit
```

```
# ./bash
# ps
# exit

# chmod 4755 bash
# ls -l bash
```
```
-rwsr-xr-x 1 root root 735004 Feb 11 09:06 bash
```

[TERM2] root 사용자의 두번째 터미널
```
# su - fedora
$ cd /test
$ ls -l bash
```
```
-rwsr-xr-x 1 root root 735004  Feb 11 09:06 bash
```

```
$ ./bash
$ id        /* 커널 2.6버전부터는 쉘은 SetUID, SetGID권한 부여 막음 */
```
```
uid=500(fedora) gid=500(fedora) groups=500(fedora)
```

```
$ exit
$ exit
#
```

```
        [참고] gcc 프로그램 설치가 되어 있지 않으면
        # which gcc
        # rpm -qa | grep gcc
        # yum install gcc   (# yum -y install gcc)
```

[TERM1] root 사용자의 첫번째 터미널
```
# cd /test
# vi backdoor.c   /* 함수를 이용하여 프로그램 내에서 쉘을 실행 시키는 것 */
```
```
#include <stdio.h>

main()
{
        setuid(0);
        setgid(0);
        system("/bin/bash");
}
```

```
# gcc -o bashshell backdoor.c
# ls -l bashshell
```
```
-rwxr-xr-x 1 root root 4934 Feb 11 09:22 bashshell
```

```
        # file /bin/ls
        # file /test/bashshell
```

[TERM2] fedora 사용자 터미널
ssh fedora@localhost
-> fedora 사용자로 로그인

$ cd /test
$ ls -l bashshell

```
-rwxr-xr-x 1 root root 4934 Feb 11 09:22 bashshell
```

$ ./bashshell
$ ps

```
  PID TTY          TIME CMD
 7195 pts/2    00:00:00 bash
 7226 pts/2    00:00:00 bashshell
 7227 pts/2    00:00:00 bash
 7249 pts/2    00:00:00 ps
```

$ id

```
uid=500(fedora) gid=500(fedora) groups=500(fedora)
```

$ exit
$

[TERM1] root 사용자 터미널
cd /test
chmod 4755 bashshell
ls -l bashshell

```
-rwsr-xr-x 1 root root 4934 Feb 11 09:22 bashshell
```

[TERM2] fedora 사용자의 터미널
$ ./bashshell
# id

```
uid=0(root) gid=0(root) groups=500(fedora)
```

pwd

```
/test
```

chmod 755 bash
ls -l

```
-rw-r--r--  1 root root     77  2??11 09:22 backdoor.c
-rwxr-xr-x  1 root root 735004  2??11 09:06 bash
-rwsr-xr-x  1 root root   4934  2??11 09:22 bashshell
```

./bash
id

```
uid=0(root) gid=0(root) groups=500(fedora)
```

# exit
# exit
$ exit
#

[참고] 특수 퍼미션을 가진 파일 찾기

[TERM1] # find / -perm -4000 -type f
[TERM2] # find / -perm -2000 -type f
[TERM3] # find / \(-perm -4000 -o -perm -2000 \) -type f (-o : OR, -a : AND)

6 Sticky Bits

파일에 쓰기 권한 없어도 디렉토리에 쓰기 권한이 있는 경우 디렉토리 권한에 의해 파일은 삭제 된다. 특정 디렉토리의 경우 공유의 목적으로 사용 할 때 사용자들에 의해 파일이 생성 될 수 있으나 디렉토리에 파일을 마음대로 삭제 할 수 없도록 sticky 권한을 부여해 줄 수 있다. 디렉토리에 쓰기 (파일생성: touch, vi등) 권한이 있어도 파일에 삭제권한(rm)을 제거하려 할 때 사용한다. 파일의 소유자나 그룹의 경우 또는 관리자의 경우 파일에 대한 소유권을 행사 할 수 경우는 제외 된다. 일반적으로 sticky bit는 디렉토리가 777 권한일 때 사용된다.

ex) 게시판의 경우

특수권한			소유자권한비트			그룹권한비트			기타권한비트		
SetUID	SetGID	sticky bit	r	w	x	r	w	x	r	w	x
0	0	1	1	1	1	1	1	1	1	1	1
천			백			십			일		

[참고] sticky bit로 설정된 디렉토리

```
# ls -ld /tmp
drwxrwxrwt 13 root root 4096 Feb 11 09:22 /tmp

        # chmod 1777 dir1 (drwxrwxrwx  ->  drwwrwxrwt)
        # chmod 1001 dir1 (d--------x  ->  d--------t)
        # chmod 1776 dir1 (drwxrwxrw-  ->  drwxrwxrwT)
```

[EX] sticky bit로 파일 삭제 권한 제거

[TERM1] user01 사용자 터미널
```
# ssh user01@localhost
```
user01 사용자로 로그인

```
$ id
uid=501(user01) gid=501(user01) groups=501(user01)

$ cd /tmp
$ mkdir stickybit
$ echo 1111 > file10
$ echo 2222 > stickybit/file2
$ ls -l | grep user01
-rw-rw-r-- 1 user01 user01    5  3월 18 12:44 file1
drwxrwxr-x 2 user01 user01 4096  3월 18 12:44 stickybit
```

[TERM2] fedora 사용자 터미널
```
# ssh fedora@localhost
```
fedora 사용자로 로그인

```
$ id
uid=500(fedora) gid=500(fedora) groups=500(fedora)

$ cd /tmp
$ mkdir linux
$ echo 3333 > file3
$ echo 4444 > linux/file4
$ ls -l | grep fedora
-rw-rw-r-- 1 fedora fedora    5  3월 18 12:45 file3
drwxrwxr-x 2 fedora fedora 4096  3월 18 12:45 linux

$ vi file3                    ($ echo 4444 >> file3 )
3333
4444        <----- '새로운 라인 추가'

$ rm -rf linux
$ rm file3
$

$ ls -l file10
-rw-rw-r-- 1 user01 user01 5  3월 18 12:44 file10
```

$ rm file10

```
rm: remove write-protected 일반 파일 `file10'? y
rm: cannot remove `file1': 명령이 허용되지 않음
```

$ mv file10 file2

```
mv: cannot move `file1' to `file2': 명령이 허용되지 않음
```

$ cp file10 file2
$ ls -l file10 file2

```
-rw-rw-r-- 1 user01 user01 5  4월 14 10:46 file1
-rw-rw-r-- 1 fedora fedora 5  4월 14 10:51 file2
```

[결론] 자신이 소유권을 가지지 않은 파일에 대해서는 원본 파일을 수정할 수 있는 명령어는 허용되지 않는다.