리눅스

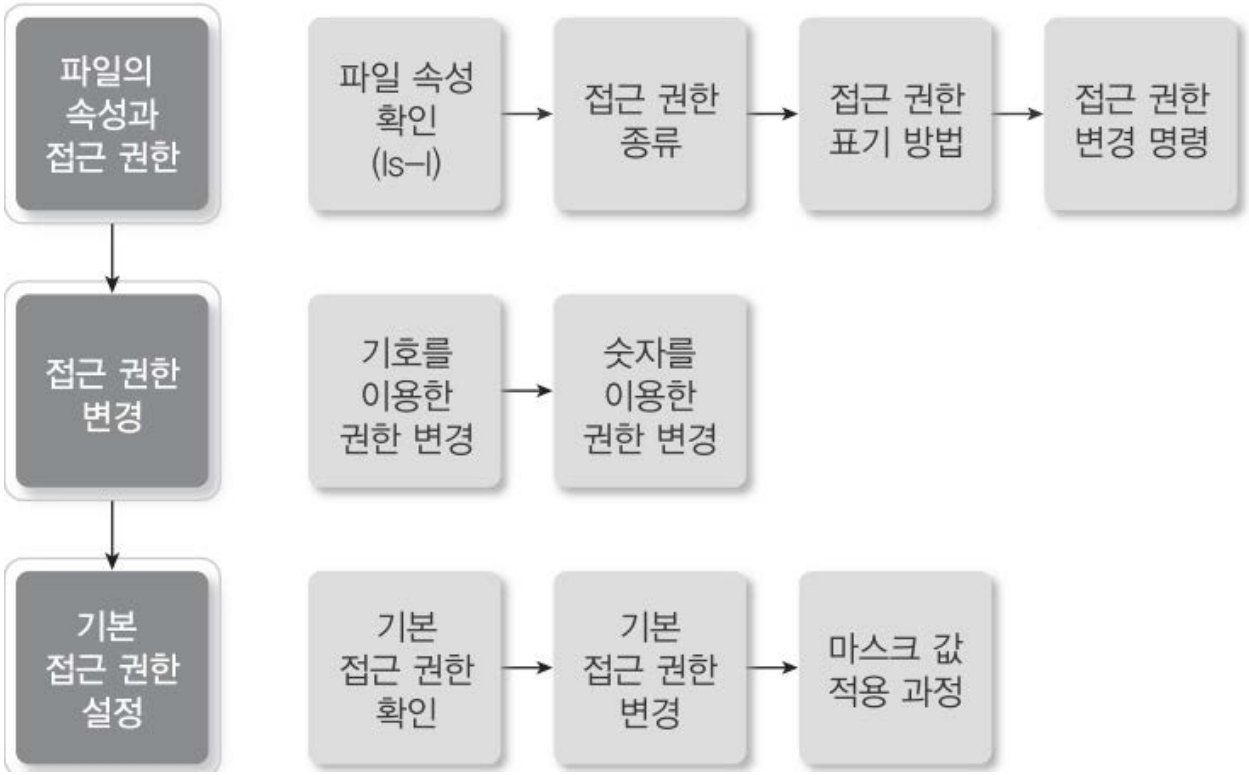
- 순　　서 : 제7장 (File Permission)
- 학　　기 : 2019학년도 1학기
- 학　　과 : 가천대학교 컴퓨터공학과 2학년
- 교　　수 : 박 양 재

목 차

학습목표

- 파일의 속성을 이해하고 설명할 수 있다.
- 접근 권한의 종류와 표기 방법을 이해하고 설명할 수 있다.
- 접근 권한을 바꾸기 위해 기호 모드에서 원하는 권한을 기호로 표기할 수 있다.
- 접근 권한을 바꾸기 위해 숫자 모드에서 원하는 권한을 숫자로 표기할 수 있다.
- 기호로 표기된 접근 권한을 숫자로 바꿀 수 있다.
- 숫자로 표기된 접근 권한을 기호로 바꿀 수 있다.
- 기본 접근 권한을 확인할 수 있다.
- 기본 접근 권한을 원하는 값으로 바꿀 수 있다.

00 개요


파일의 속성과 접근 권한
파일 속성 확인 (ls-l)
접근 권한 종류
접근 권한 표기 방법
접근 권한 변경 명령
접근 권한 변경
기호를 이용한 권한 변경
숫자를 이용한 권한 변경
기본 접근 권한 설정
기본 접근 권한 확인
기본 접근 권한 변경
마스크 값 적용 과정

01 파일의 속성

- 파일 접근 권한 보호
  - 리눅스는 파일에 무단으로 접근하는 것을 방지하고 보호하는 기능을 제공
  - 사용자는 자신의 파일과 디렉터리 중에서 다른 사용자가 접근해도 되는 것과 그렇지 않은 것을 구분하여 접근 권한을 제한
- 파일의 속성

```
[park@localhost ~]$ ls -l /etc/hosts
-rw-r--r--. 1 root root 158  8월  6  2012 /etc/hosts
[park@localhost ~]$
```

번호	속성 값	의미
1	-	파일의 종류(- : 일반 파일, d : 디렉터리)
2	rw-r--r--	파일을 읽고 쓰고 실행할 수 있는 접근 권한 표시
3	1	하드 링크의 개수
4	root	파일 소유자의 로그인 ID
5	root	파일 소유자의 그룹 이름
6	158	파일의 크기(바이트 단위)
7	8월 6 2012	파일이 마지막으로 수정된 날짜
8	/etc/hosts	파일명

01 파일의 속성

파일의 종류

- 파일 속성의 첫 번째 항목은 파일의 종류를 표시
- -는 일반 파일을, d는 디렉터리를 의미
- 파일의 종류를 알려주는 명령

file

기능 지정한 파일의 종류를 알려준다.
형식 file 파일명
사용 예 file /etc/services

```
[park@localhost ~]$ file /etc/hosts temp
/etc/hosts: ASCII text
temp:       directory
[park@localhost ~]$
```

파일의 접근 권한 표시

- 파일의 소유자와 그룹이나 기타 사용자들이 파일에 대해 가지고 있는 접근 권한을 표시

하드 링크의 개수

- 하드 링크는 한 파일에 대해 여러 개의 파일명을 가질 수 있도록 하는 기능

01 파일의 속성

- 파일 소유자의 로그인 ID
  - 리눅스에서 모든 파일은 소유자가 있음
- 파일 소유자의 그룹 이름
  - **ls -l** 명령에서 출력되는 그룹명은 파일이 속한 그룹
  - 사용자가 속한 기본 그룹은 시스템 관리자가 사용자를 등록할 때 결정
  - 사용자가 속한 그룹을 알려주는 명령은 **groups**

groups

기능	사용자가 속한 그룹을 알려준다.
형식	groups [사용자명]

```
[park@localhost ~]$ groups
user1 wheel dialout
[park@localhost ~]$ groups root
root : root
[park@localhost ~]$
```

- 파일의 크기: 바이트 단위
- 파일이 마지막으로 수정된 날짜

02 파일의 접근 권한

접근 권한의 종류

- 읽기 권한, 쓰기 권한, 실행 권한 등 세 가지로 구성

파일과 디렉터리의 접근 권한

권한	파일	디렉터리
읽기	파일을 읽거나 복사할 수 있다.	ls 명령으로 디렉터리 목록을 볼 수 있다(ls 명령의 옵션은 실행 권한이 있어야 사용할 수 있다).
쓰기	파일을 수정, 이동, 삭제할 수 있다(디렉터리에 쓰기 권한이 있어야 한다).	파일을 생성하거나 삭제할 수 있다.
실행	파일을 실행할 수 있다(셸 스크립트나 실행 파일의 경우).	cd 명령을 사용할 수 있다. 파일을 디렉터리로 이동하거나 복사할 수 있다.

접근 권한의 표기 방법

- 사용자 카테고리별로 누가 파일을 읽고 쓰고 실행할 수 있는지를 문자로 표현한 것
- 읽기 권한은 **r**, 쓰기 권한은 **w**, 실행 권한은 **x**로 나타내며, 해당 권한이 없는 경우에는 -로 표기
- 사용자 카테고리별로 세 가지 권한의 부여 여부를 **rwx** 세 문자를
- 묶어서 표기

```
[park@localhost ~]$ ls -l /etc/hosts
-rw-r--r--. 1 root root 158  8월  6  2012 /etc/hosts
[park@localhost ~]$
```

02 파일의 접근 권한

- 접근 권한의 표기 방법

파일의 접근 권한 표기

다양한 접근 권한 조합의 예

접근 권한	의미
rwxr-xr-x	소유자는 읽기, 쓰기, 실행 권한을 모두 가지고 있고 그룹과 기타 사용자는 읽기와 실행 권한만 가지고 있다.
r-xr-xr-x	소유자, 그룹, 기타 사용자 모두 읽기와 실행 권한만 가지고 있다.
rw-------	소유자만 읽기, 쓰기 권한을 가지고 있고 그룹과 기타 사용자는 아무 권한이 없다.
rw-rw-rw-	소유자, 그룹, 기타 사용자 모두 읽기와 쓰기 권한을 가지고 있다.
rwxrwxrwx	소유자, 그룹, 기타 사용자 모두 읽기, 쓰기, 실행 권한을 가지고 있다.
rwx------	소유자만 읽기, 쓰기, 실행 권한을 가지고 있고 그룹과 기타 사용자는 아무 권한이 없다.
r--------	소유자만 읽기 권한을 가지고 있다.

02 파일의 접근 권한

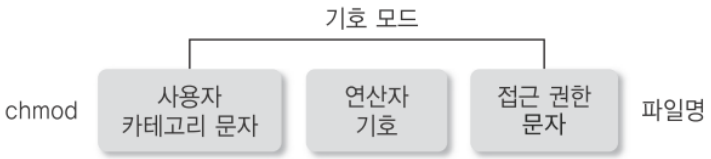
접근 권한의 변경 명령

chmod	
기능	파일이나 디렉터리의 접근 권한을 변경한다.
형식	chmod [옵션] 권한 모드 파일 또는 디렉터리명
옵션	-R : 하위 디렉터리까지 모두 변경할 수 있다.

- 기호 모드 : 접근 권한을 변경하기 위해 문자와 기호를 사용하여 권한을 표시
- 숫자 모드 : 접근 권한을 변경하기 위해 숫자를 사용

03 기호를 이용한 파일 접근 권한 변경

- 기호 모드



기호 모드의 구성 요소

기호 모드에서 사용하는 문자와 기호

구분	문자/기호	의미
사용자 카테고리 문자	u	파일 소유자
	g	소유자가 속한 그룹
	o	소유자와 그룹 이외의 기타 사용자
	a	전체 사용자
연산자 기호	+	권한 부여
	–	권한 제거
	=	접근 권한 설정
접근 권한 문자	r	읽기 권한
	w	쓰기 권한
	x	실행 권한

03 기호를 이용한 파일 접근 권한 변경

- 기호 모드를 사용한 접근 권한 설정의 예

기호 모드를 사용한 접근 권한 설정의 예

권한 표기	의미
u+w	소유자(u)에게 쓰기(w) 권한 부여(+)
u-x	소유자(u)의 실행(x) 권한 제거(-)
g+w	그룹(g)에 쓰기(w) 권한 부여(+)
o-r	기타 사용자(o)의 읽기(r) 권한 제거(-)
g+wx	그룹(g)에 쓰기(w)와 실행(x) 권한 부여(+)
+wx	모든 사용자에게 쓰기(w)와 실행(x) 권한 부여(+)
a+rwx	모든 사용자에게 읽기(r), 쓰기(w), 실행(x) 권한 부여(+)
u=rwx	소유자(u)에게 읽기(r), 쓰기(w), 실행(x) 권한 부여(=)
go+w	그룹(g)과 기타 사용자(o)에게 쓰기(w) 권한 부여(+)
u+x,go+w	소유자(u)에게 실행(x) 권한을 부여하고(+) 그룹(g)과 기타 사용자(o)에게 쓰기(w) 권한 부여(+)

03 기호를 이용한 파일 접근 권한 변경

- 기호를 이용한 접근 권한 변경 예

① 현재 접근 권한 확인: **rw-r--r—**

```
[park@localhost ch7]$ ls -l
합계 4
-rw-r--r--. 1 park park 158  3월 25 18:03 test.txt
[park@localhost ch7]$
```

② 소유자의 쓰기 권한을 제거: **u-w**

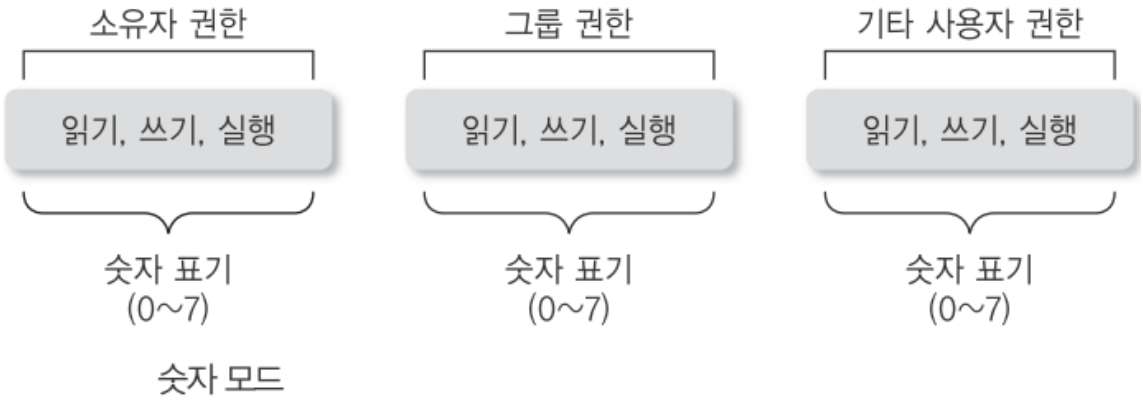
```
[park@localhost ch7]$ chmod u-w test.txt
[park@localhost ch7]$ ls -l
합계 4
-r--r--r--. 1 park park 158  3월 25 18:03 test.txt
[park@localhost ch7]$
```

03 기호를 이용한 파일 접근 권한 변경

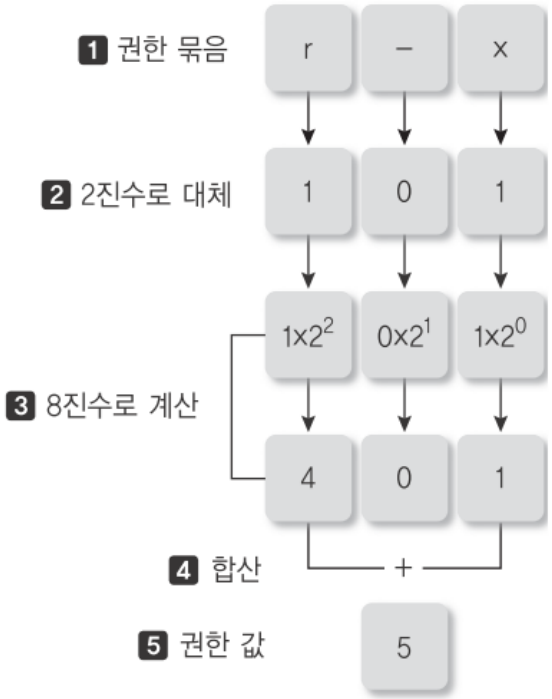
- 실습-변경전 화면과 변경 후 화면을 비교하세요.
  - 그룹에 쓰기와 실행 권한을 부여한다 : **g+wx**
  - 기타 사용자에게 실행 권한을 부여한다 : **o+x**
  - 그룹과 기타 사용자의 실행 권한을 제거한다 : **go-x**
  - 모두에게 실행 권한을 부여한다 : **a+x**
  - 소유자에게 쓰기 권한을 부여하고 그룹의 쓰기 권한은 제거한다 : **u+w ,g-w**

04 숫자를 이용한 파일 접근 권한 변경

- 숫자로 환산하기
  - 숫자 모드에서는 각 권한이 있고 없고를 0과 1로 표기하고 이를 다시 환산하여 숫자로 표현
  - 카테고리별로 권한의 조합에 따라 0부터 7로 나타내는 것



04 숫자를 이용한 파일 접근 권한 변경



권한을 숫자로 환산하는 과정

권한을 숫자로 환산하는 과정

① r-x라고 할 때 권한이 있는 것은 1로, 없는 것은 0으로 변환

② 2진수 1, 0, 1로 변환

③ 2진수를 각 자릿수별로 10진수로 환산하면 4, 0, 1이 된다.

④ 세 숫자를 더하면

⑤ 최종 권한 값은 5가 됨

접근 권한과 숫자의 대응 관계

접근 권한	환산	숫자	의미
rwx	111 → 4+2+1	7	읽기, 쓰기, 실행
rw–	110 → 4+2+0	6	읽기, 쓰기
r–x	101 → 4+0+1	5	읽기, 실행
r––	100 → 4+0+0	4	읽기
–wx	011 → 0+2+1	3	쓰기, 실행
–w–	010 → 0+2+0	2	쓰기
––x	001 → 0+0+1	1	실행
–––	000 → 0+0+0	0	권한이 없음

04 숫자를 이용한 파일 접근 권한 변경

- 전체 권한을 숫자로 표기

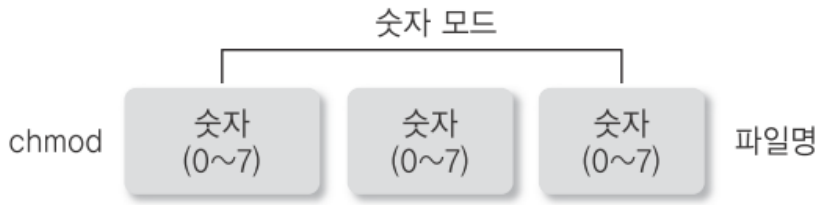
전체 권한을 숫자로 표기한 예

숫자로 표현한 접근 권한의 예

접근 권한	8진수 모드	접근 권한	8진수 모드
rwxrwxrwx	777	rw–r––r––	644
rwxr–xr–x	755	rwx––––––	700
rw–rw–rw	666	rw–r–––––	640
r–xr–xr–x	555	r–––––––––	400

04 숫자를 이용한 파일 접근 권한 변경

- 숫자 모드로 접근 권한 변경하기



숫자 모드를 이용한 접근 권한 변경

- 숫자의 각 위치가 사용자 카테고리를 나타내기 때문에 사용자 카테고리를 따로 지정할 필요가 없다
- 항상 세 자리 수를 사용해야 하므로 변경하려는 사용자 카테고리의 권한뿐만 아니라 그룹과 기타 사용자의 권한도 반드시 같이 명시

04 숫자를 이용한 파일 접근 권한 변경

- 숫자 모드로 접근 권한 변경하기 예

① 현재 접근 권한: 644(rw-r--r--)

```
[park@localhost ch7]$ ls -l
합계 4
-rw-r--r--. 1 park park 158  3월 25 18:03 test.txt
[park@localhost ch7]$
```

② 소유자의 쓰기 권한을 제거: r--r--r--이므로 444

```
[park@localhost ch7]$ chmod 444 test.txt
[park@localhost ch7]$ ls -l
합계 4
-r--r--r--. 1 park park 158  3월 25 18:03 test.txt
[park@localhost ch7]$
```

③ 그룹에 쓰기와 실행 권한을 부여: r--rwxr--이므로 474

```
[park@localhost ch7]$ chmod 474 test.txt
[park@localhost ch7]$ ls -l
합계 4
-r--rwxr--. 1 park park 158  3월 25 18:03 test.txt*
[park@localhost ch7]$
```

04 숫자를 이용한 파일 접근 권한 변경

- 실습-변경전 화면과 변경 후 화면을 비교하세요.
  - 기타 사용자에게 실행 권한을 부여한다 : **o+x , r--rwxr-x → 475**
  - 그룹과 기타 사용자의 실행 권한을 제거한다 : **go-x, r--rw-r-- → 464**
  - 모두에게 실행 권한을 부여한다 : **a+x, r-xrwxr-x → 575**
  - 소유자에게 쓰기 권한을 부여하고 그룹의 쓰기 권한은 제거한다 : **u+w,g-w, rwxr-xr-x → 755**
  - 소유자의 권한만 남기고 나머지 사용자의 권한은 모두 제거: **rwx------ → 700**

05 기본 접근 권한 설정

기본 접근 권한

- 리눅스에서는 파일이나 디렉터리를 생성할 때 기본 접근 권한이 자동적으로 설정
- 일반 파일의 경우 소유자와 그룹은 읽기와 쓰기 권한이 설정되고 기타 사용자는 읽기 권한만 설정(644)
- 디렉터리의 경우 소유자와 그룹은 읽기, 쓰기, 실행 권한이 설정되고 기타 사용자는 읽기, 실행 권한만 설정(775)

```
[park@localhost ch7]$ touch fedora.txt
[park@localhost ch7]$ mkdir temp
[park@localhost ch7]$ ls -l
합계 8
-rw-rw-r--. 1 park park    0  3월 25 22:01 fedora.txt
drwxrwxr-x. 2 park park 4096  3월 25 22:01 temp/
----------. 1 park park  158  3월 25 18:03 test.txt
[park@localhost ch7]$
```

05 기본 접근 권한 설정

기본 접근 권한 확인하고 변경하기

umask

기능	기본 접근 권한을 출력하거나 변경한다.
형식	umask [옵션] [마스크 값]
옵션	-S : 마스크 값을 문자로 출력한다.
사용 예	umask 022 umask

- 아무 인자 없이 **umask** 명령만 사용할 경우 기본 마스크 값 출력

```
[park@localhost ch7]$ umask
0002
[park@localhost ch7]$
```

마스크 값의 의미

- 마스크 값은 파일이나 디렉터리 생성 시 부여하지 않을 권한을 지정해놓는 것
- 마스크 값이 **002**일 경우 이는 -------**w**-이고, 기타 사용자에게 쓰기 권한은 부여하지 않겠다는 뜻

```
[park@localhost ch7]$ umask -S
u=rwx, g=rwx, o=rx
[park@localhost ch7]$
```

05 기본 접근 권한 설정

- 마스크 값 변경하기

```
[park@localhost ch7]$ umask 077
[park@localhost ch7]$ umask
0077
[park@localhost ch7]$
```

- 마스크 값을 바꾸면 파일을 생성할 때 적용되는 기본 접근 권한도 변경(파일은 처음 생성 시 실행권한 없음)

```
[park@localhost ch7]$ touch linux.txt
[park@localhost ch7]$ ls -l linux.txt
-rw-------. 1 park park    0  3월 25 22:22 linux.txt
[park@localhost ch7]$
```

- 마스크 값을 바꾸면 디렉터리를 생성할 때 적용되는 기본 접근 권한도 변경

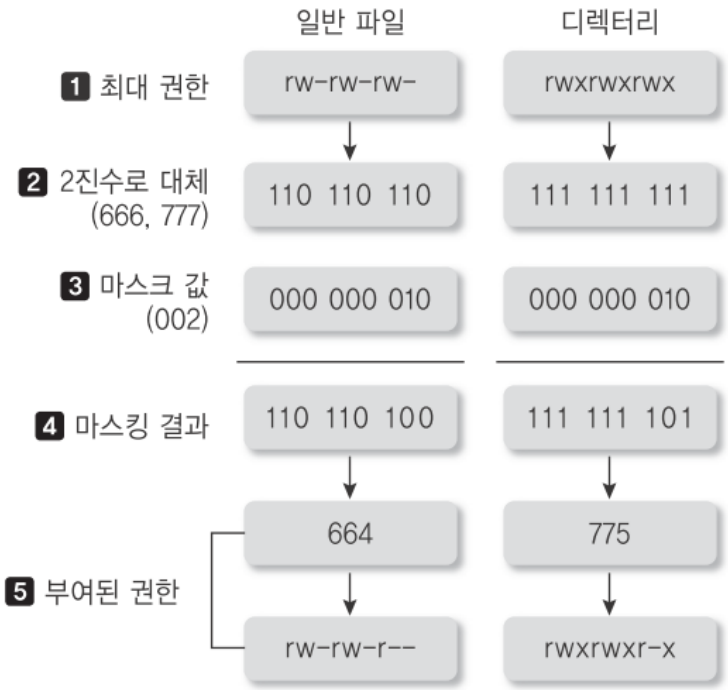
```
[park@localhost ch7]$ mkdir abc
[park@localhost ch7]$ ls -l abc
drwx------. 1 park park 4096  3월 25 22:01 abc
[park@localhost ch7]$
```

05 기본 접근 권한 설정

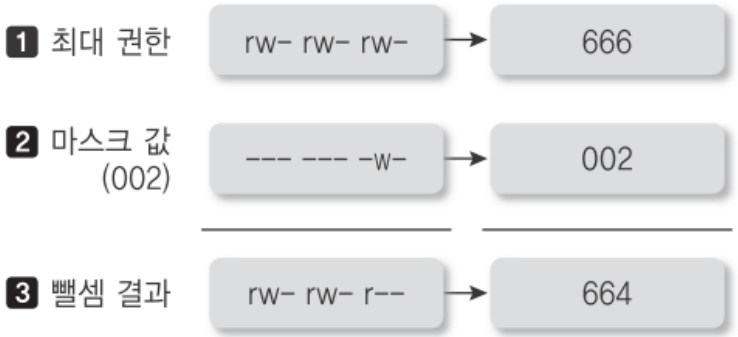
- 마스크 값의 적용 과정

umask 진리표

요청 권한	1	1	0	0
마스크	1	0	1	0
부여된 권한	0	1	0	0



마스크 값을 적용하는 과정



마스크 값에 뺄셈을 적용하는 과정

05 기본 접근 권한 설정

- 여러 가지 마스크 값

마스크 값 목록

마스크 값	일반 파일	디렉터리	의미
022	644	755	그룹과 기타 사용자는 읽기만 가능하다.
077	600	700	그룹과 기타 사용자의 접근 권한을 모두 제거한다.
027	640	750	그룹은 읽기와 실행만 가능하고, 기타 사용자의 접근 권한은 모두 제거한다.

- **umask**로 마스크 값을 바꿀 때 파일과 디렉터리에 모두 적용해봐야 함
- 마스크 값이 파일에는 적합하지만 디렉터리에는 적합하지 않을 수도 있음

06 특수 접근 권한

- 특수 접근 권한
- 리눅스의 권한은 소유자, 그룹 소유자, 나머지 사용자의 세 부류로 나누며 각각 파일에 대한 권한을 읽기, 쓰기, 실행의 세 가지로 나타낸다.
- 관리자의 입장에서 시스템의 모든 상황을 고려해야 하기 때문에 이러한 권한 외에 몇 가지 특별한 권한을 사용하여 시스템 관리의 효율성을 높인다

특수접근 권한	절대모드 표현
SetUID	4000
SetGID	2000
Sticky bit	1000

06 특수 접근 권한

- SetUID(4000)퍼미션
  - -일반적으로 x 권한으로 설정된 파일을 실행하면 실행한 사용자의 권한으로 실행된다.
  - -SetUID가 적용되면 파일이 실행되는 동안 실행한 사용자가 아닌 파일의 소유자의 권한을 부여하게 된다.
  - -SetUID는 실행 파일에만 명시할 수 있다.
  - -적용되면 소유자의 퍼미션 값에 실행권한에 x 가 아닌 s로 표시된다.

- SetUID가 부여된 대표적인 명령어 : passwd
  - - passwd를 변경하는 과정에서 /etc/shadow 파일의 내용을 읽거나 쓸 수 있어야 변경한 패스워드를 기록할 수 있기 때문에 passwd 파일에는 SetUID가 걸려 있고 소유주가 root로 설정되어 있음.
  - - passwd 파일을 보면은 root 권한으로 설정되어 있는 것을 볼 수 있다.
  - -관리자 외의 일반 사용자들이 자신의 비밀번호를 변경하려면 관리자 권한이 필요할 때 SetUID를 부여함으로써 일반 사용자가 패스워드를 변경하고 실행하는 동안만 일반 사용자한테 관리자 관한을 주는 것 이다. 이 후 다시 회수 한다.

06 특수 접근 권한

- SetUID가 부여된 대표적인 명령어 : passwd

```
[root@localhost ~]# which passwd
/usr/bin/passwd
[root@localhost ~]#
```

```
[root@localhost ~]# ls -l /usr/bin/passwd
-rwsr-xr-x. 1 root root 22648 2009-09-14 21:14 /usr/bin/passwd
[root@localhost ~]#
[root@localhost ~]# ls -la /etc/passwd
-rw-r--r--. 1 root root 1991 2016-04-12 23:03 /etc/passwd
```

- /usr/bin/passwd 파일과 /etc/passwd 파일의 차이점은 무엇인가?

06 특수 접근 권한

- SetUID 설정하기

```
[root@localhost ~]# mkdir gachon
[root@localhost ~]#
[root@localhost ~]# cd gachon/
[root@localhost gachon]#
[root@localhost gachon]# ls
[root@localhost gachon]#
[root@localhost gachon]# touch exfile1 exfile2
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 0
-rw-r--r--. 1 root root 0 2016-04-20 10:49 exfile1
-rw-r--r--. 1 root root 0 2016-04-20 10:49 exfile2
[root@localhost gachon]#
[root@localhost gachon]#
[root@localhost gachon]# chmod 4644 exfile1
[root@localhost gachon]# chmod u+s exfile2
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 0
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile2
[root@localhost gachon]#
```

06 특수 접근 권한

- SetGID(2000) 접근권한

  - SetUID 접근권한과 유사하게 파일이 실행되는 동안 그룹 소유자의 권한을 갖게 된다

  -실행파일과 디렉토리에도 설정 가능하다.

  -적용시 그룹 소유자의 퍼미션 값에 실행권한이 x가 아니 s로 표시 된다.

```
[root@localhost gachon]# ls
exfile1  exfile2
[root@localhost gachon]#
[root@localhost gachon]# touch exfile3 exfile4
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 0
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile2
-rw-r--r--. 1 root root 0 2016-04-20 10:59 exfile3
-rw-r--r--. 1 root root 0 2016-04-20 10:59 exfile4
[root@localhost gachon]#
[root@localhost gachon]# chmod 2644 exfile3
[root@localhost gachon]# chmod g+s exfile4
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 0
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root 0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root 0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root 0 2016-04-20 10:59 exfile4
[root@localhost gachon]#
```

06 특수 접근 권한

- SetGID(2000) 접근권한 – 디렉토리 경우(chmod g+s dir1)

※ 하위 디렉토리도 적용된다고 했는데 그림을 보면 상위 디렉토리만 SetGID가 적용되었다.

어떤 경우에 하위 디렉토리도 적용 된다는 것일까?

```
[root@localhost gachon]# ls
exfile1  exfile2  exfile3  exfile4
[root@localhost gachon]#
[root@localhost gachon]#
[root@localhost gachon]# mkdir dir1
[root@localhost gachon]#
[root@localhost gachon]# mkdir dir1/dir1_1
[root@localhost gachon]#
[root@localhost gachon]# touch dir1/testfile1
[root@localhost gachon]#
[root@localhost gachon]# ls -Rl
.:
합계 4
drwxr-xr-x. 3 root root 4096 2016-04-20 11:08 dir1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile4

./dir1:
합계 4
drwxr-xr-x. 2 root root 4096 2016-04-20 11:07 dir1_1
-rw-r--r--. 1 root root    0 2016-04-20 11:08 testfile1
```

```
./dir1/dir1_1:
합계 0
[root@localhost gachon]# chmod g+s dir1
[root@localhost gachon]# ls -Rl
.:
합계 4
drwxr-sr-x. 3 root root 4096 2016-04-20 11:08 dir1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile4

./dir1:
합계 4
drwxr-xr-x. 2 root root 4096 2016-04-20 11:07 dir1_1
-rw-r--r--. 1 root root    0 2016-04-20 11:08 testfile1

./dir1/dir1_1:
합계 0
[root@localhost gachon]#
```

06 특수 접근 권한

- SetGID(2000) 접근권한 – 디렉토리 경우(chmod g+s dir1)

※ 하위 디렉토리도 적용 되려면 어떻게 SetGID를 설정해야 하는가?

```
[root@localhost gachon]# mkdir dir2
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 8
drwxr-sr-x. 3 root root 4096 2016-04-20 11:08 dir1
drwxr-xr-x. 2 root root 4096 2016-04-20 11:19 dir2
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile4
[root@localhost gachon]# chmod g+s dir2
[root@localhost gachon]#
[root@localhost gachon]# ls -l
합계 8
drwxr-sr-x. 3 root root 4096 2016-04-20 11:08 dir1
drwxr-sr-x. 2 root root 4096 2016-04-20 11:19 dir2
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile4
[root@localhost gachon]#
[root@localhost gachon]# mkdir dir2/dir2_1
```

```
[root@localhost gachon]# ls -Rl
.:
합계 8
drwxr-sr-x. 3 root root 4096 2016-04-20 11:08 dir1
drwxr-sr-x. 3 root root 4096 2016-04-20 11:20 dir2
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile1
-rwSr--r--. 1 root root    0 2016-04-20 10:49 exfile2
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile3
-rw-r-Sr--. 1 root root    0 2016-04-20 10:59 exfile4

./dir1:
합계 4
drwxr-xr-x. 2 root root 4096 2016-04-20 11:07 dir1_1
-rw-r--r--. 1 root root    0 2016-04-20 11:08 testfile1

./dir1/dir1_1:
합계 0

./dir2:
합계 4
drwxr-sr-x. 2 root root 4096 2016-04-20 11:20 dir2_1

./dir2/dir2_1:
합계 0
[root@localhost gachon]#
```

*.결론 : 상위 디렉토리에 SetUID가 설정된 후에 하위 디렉토리를 생성하면 SetGID가 적용된다.

06 특수 접근 권한

- Sticky bit(1000) 접근권한
  - 디렉토리에 부여되는 권한으로 적용 시 모든 사용자는 읽고, 쓰기가 가능하나 삭제는 소유자만
  - 접근권한이 적용되면 others의 접근권한에서 실행권한이 x가 아닌 t로 표시된다.
  - 대표적으로 sticky bit가 적용된 디렉토리는 /tmp와 /var/tmp가 있다.

```
[root@localhost gachon]# ls -l / | grep tmp
drwxrwxrwt.  15 root root  4096 2016-04-20 11:19 tmp
[root@localhost gachon]#
[root@localhost gachon]# ls -l /var/ | grep tmp
drwxrwxrwt.   2 root root 4096 2016-04-19 17:47 tmp
[root@localhost gachon]#
```

06 특수 접근 권한

- Sticky bit(1000) 접근권한 –실습
  - 디렉토리에 부여되는 권한으로 적용 시 모든 사용자는 읽고, 쓰기가 가능하나 삭제는 소유자만
  - 접근권한이 적용되면 others의 접근권한에서 실행권한이 x가 아닌 t로 표시된다.
  - 대표적으로 sticky bit가 적용된 디렉토리는 /tmp와 /var/tmp가 있다.

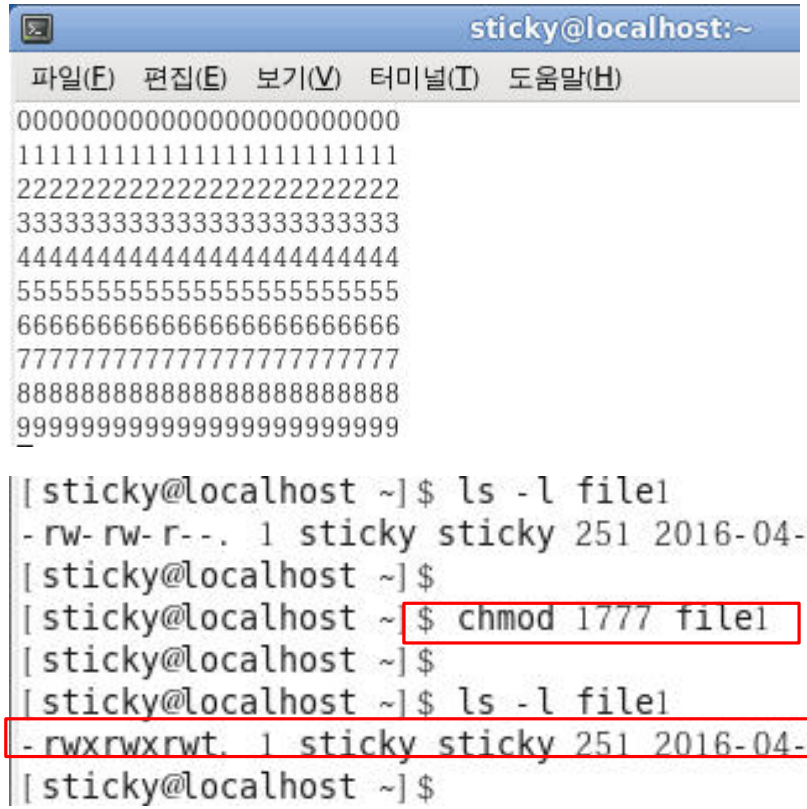
```
[root@localhost gachon]# useradd sticky
[root@localhost gachon]#
[root@localhost gachon]# su sticky
[sticky@localhost gachon]$ cd /home/
[sticky@localhost home]$ ls -l
합계 12
drwx------.  4 kim    kim    4096 2016-04-12 23:03 kim
drwx------. 28 park   park   4096 2016-04-20 11:11 park
drwx------.  4 sticky sticky 4096 2016-04-20 11:36 sticky
[sticky@localhost home]$ chmod 755 sticky
[sticky@localhost home]$ ls -l
합계 12
drwx------.  4 kim    kim    4096 2016-04-12 23:03 kim
drwx------. 28 park   park   4096 2016-04-20 11:11 park
drwxr-xr-x.  4 sticky sticky 4096 2016-04-20 11:36 sticky
[sticky@localhost home]$ cd sticky/
[sticky@localhost ~]$ ls
[sticky@localhost ~]$
[sticky@localhost ~]$ touch file1
```

06 특수 접근 권한

- Sticky bit(1000) 접근권한 –실습

– 디렉토리에 부여되는 권한으로 적용 시 모든 사용자는 읽고, 쓰기가 가능하나 삭제는 소유자만

- File1의 내용



06 특수 접근 권한

- Sticky bit(1000) 접근권한 –실습

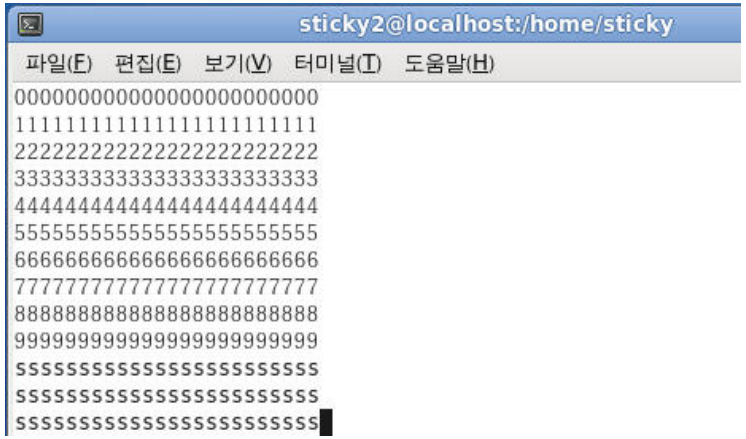
– 디렉토리에 부여되는 권한으로 적용 시 모든 사용자는 읽고, 쓰기가 가능하나 삭제는 소유자만

- 다른 사용자 sticky2가 sticky 소유의 file1의 내용을 읽고, 쓰기 가능여부와 삭제 가능여부?

```
[sticky@localhost ~]$ su - root
암호:
[root@localhost ~]#
[root@localhost ~]# useradd sticky2
[root@localhost ~]#
[root@localhost ~]# su sticky2
[sticky2@localhost root]$
[sticky2@localhost root]$ cd /home/
[sticky2@localhost home]$ ls -l
합계 16
drwx------.  4 kim     kim     4096 2016-04-12 23:03 kim
drwx------. 28 park    park    4096 2016-04-20 11:11 park
drwxr-xr-x.  4 sticky  sticky  4096 2016-04-20 11:47 sticky
drwx------.  4 sticky2 sticky2 4096 2016-04-20 11:47 sticky2
[sticky2@localhost home]$ cd sticky
[sticky2@localhost sticky]$
[sticky2@localhost sticky]$ ls -l
합계 4
-rwxrwxrwt. 1 sticky sticky 251 2016-04-20 11:44 file1
[sticky2@localhost sticky]$
[sticky2@localhost sticky]$ rm file1
rm: cannot remove `file1': 허가 거부
[sticky2@localhost sticky]$
[sticky2@localhost sticky]$ vi file1
```

#. 다른 사용자는 삭제 불가능

#. 다른 사용자도 내용을 읽고, 수정이 가능!



File1에 추가된 내용

4.보고서

다음을 설명하세요.

1. 파일의 속성은 어떤 명령으로 확인 가능한지 설명하세요.
2. 파일의 읽기권한과 디렉토리의 읽기 권한의 차이점을 설명하세요.
3. 디렉토리에 실행 권한이 없다는 것의 의미를 설명하세요.
4. **/etc** 디렉토리에 있는 **group, passwd, shadow** 파일의 소유자, 그룹, 기타 사용자 권한을 찾아보고 설명하세요.
5. 마스크 값이 **007**일 때, 파일과 디렉토리를 생성 할 경우 기본 접근권한은 어떻게 되는지 설명하세요
6. 파일의 접근 권한을 확인하였더니 **'rwSr—r—'**이였다. **S**가 의미하는 것을 설명하세요.
7. 특수접근 권한 **SetUID, SetGID, sticky bit**의 기능을 예를 들어 설명하세요.