

자동 완성과 히스토리

- 자동 완성이란 파일명의 일부만 입력한 후에 Tab키를 눌러 나머지 파일명을 자동으로 완성하는 기능을 말함.

예) cd /etc/sysconfig/network-scripts/ 를 입력하려면
cd /et[Tab키]sysco[Tab키]network[Tab키]

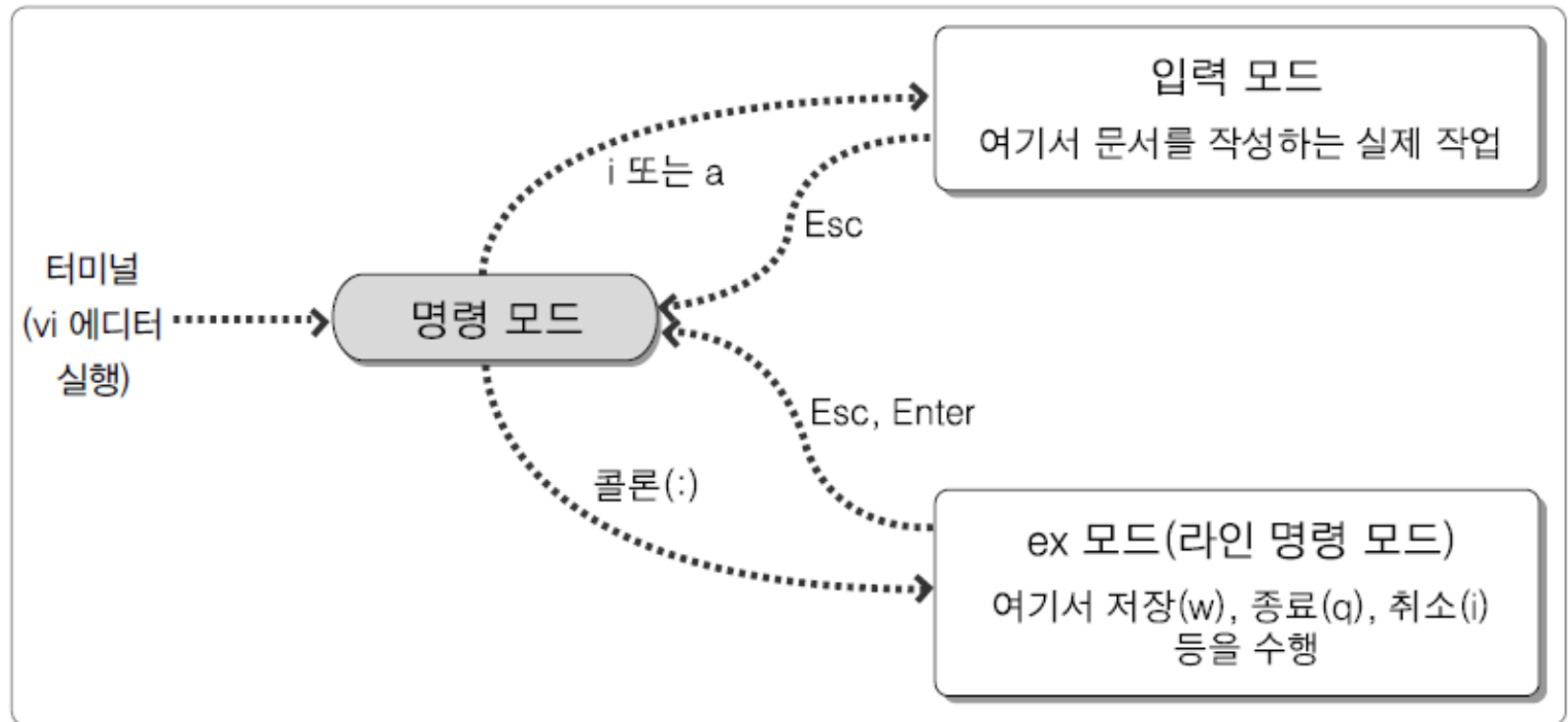
➤ 자동 완성기능은 빠른 입력효과도 있지만, 파일명이나 디렉터리가 틀리지 않고 정확하게 입력되는 효과도 있으므로 자주 활용된다.

- 도스 키란 이전에 입력한 명령어를 상/하 화살표 키를 이용해서 다시 나타내는 기능을 말함.

에디터 사용

gedit 에디터와 vi 에디터

- vi 에디터 사용법 개요도



vi 기능 요약

• 명령모드 → 입력모드

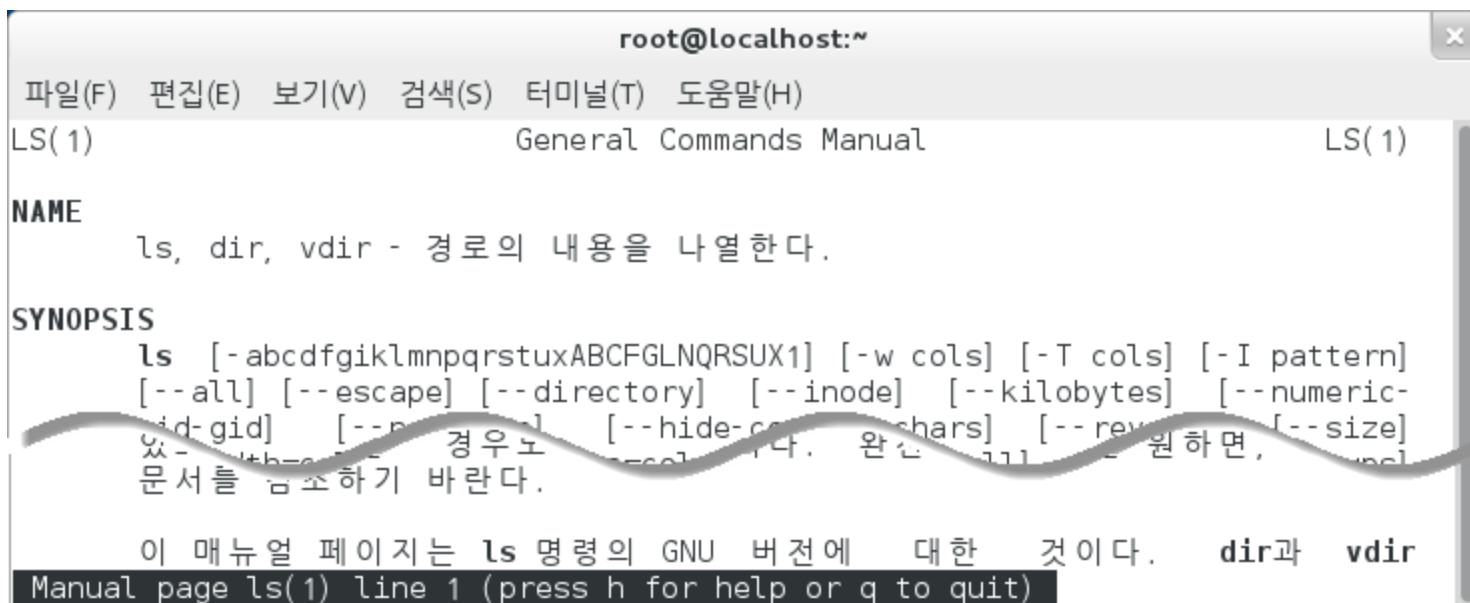
i	현재 커서의 위치부터 입력([I])	I	현재 커서 줄의 맨 앞에서부터 입력([Shift] + [I])
a	현재 커서의 위치 다음 칸부터 입력([A])	A	현재 커서 줄의 맨 마지막부터 입력([Shift] + [A])

h	커서를 왼쪽으로 한 칸 이동([←]와 같은 의미, [H])	j	커서를 아래로 한 칸 이동([↓]와 같은 의미, [J])
k	커서를 위로 한 칸 이동([↑]와 같은 의미, [K])	l	커서를 오른쪽으로 한 칸 이동([→]와 같은 의미, [L])

x	현재 커서가 위치한 글자 삭제([Del]과 같은 의미, [X])	X	현재 커서가 위치한 앞 글자 삭제([BackSpace]와 같은 의미, [Shift] + [X])
dd	현재 커서의 행 삭제([D] 연속 두 번 입력)	숫자 dd	현재 커서부터 숫자만큼의 행 삭제(숫자 다음 [D] 연속 두 번 입력)
yy	현재 커서가 있는 행을 복사([Y] 연속 두 번 입력)	숫자 yy	현재 커서부터 숫자만큼의 행을 복사(숫자 다음 [Y] 연속 두 번 입력)
p	복사한 내용을 현재 행 이후에 붙여 넣기([P])	P	복사한 내용을 현재 행 이전에 붙여 넣기([Shift] + [P])

도움말 사용법

- “man 명령어”를 사용하면 도움말 출력



```
root@localhost:~
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
LS(1)                                General Commands Manual                                LS(1)

NAME
    ls, dir, vdir - 경로의 내용을 나열한다.

SYNOPSIS
    ls [-abcdgiklmnpqrstuxABCFGLNQRSUX1] [-w cols] [-T cols] [-I pattern]
    [--all] [--escape] [--directory] [--inode] [--kilobytes] [--numeric-
    uid-gid] [--no-color] [--hide-control-chars] [--rev] [--size]
    [--sort=TYPE] [--time=TIME] [--width=WIDTH] [--help]
    이 매뉴얼 페이지는 ls 명령의 GNU 버전에 대한 것이다. dir과 vdir
    Manual page ls(1) line 1 (press h for help or q to quit)
```

리눅스 기본 명령어 (1)

- ls

Windows의 "dir"과 같은 역할로, 해당 디렉터리에 있는 파일의 목록을 나열
예) # ls /etc/sysconfig

- cd

디렉터리를 이동

예) # cd ../etc/sysconfig

- '.' (현재 디렉터리)
- '..' (현재의 상위 디렉터리)

- pwd

현재 디렉터리의 전체 경로를 출력

- rm

파일이나 디렉터를 삭제

예) # rm -rf abc

- 리눅스는 별도의 숨김 파일(Hidden File)이라는 속성이 존재하지 않는다. 파일명이나 디렉터리의 제일 앞글자를 "."으로 하면 자동으로 숨김 파일이 된다.

리눅스 기본 명령어 (2)

- **cp**
파일이나 디렉토리를 복사
예) # cp abc.txt cba.txt
- **touch**
크기가 0인 새 파일을 생성, 이미 존재하는 경우 수정 시간을 변경
예) # touch abc.txt
- **mv**
파일과 디렉토리의 이름을 변경하거나 위치 이동 시 사용
예) mv abc.txt www.txt
- **mkdir**
새로운 디렉토리를 생성
예) # mkdir abc

리눅스 기본 명령어 (3)

- **rmdir**
디렉토리를 삭제. (단, 비어 있어야 함)
예) # rmdir abc
- **cat**
텍스트로 작성된 파일을 화면에 출력
예) # cat a.txt b.txt
- **head, tail**
텍스트로 작성된 파일의 앞 10행 또는 마지막 10행만 출력
예) # head anaconda-ks.cfg
- **more**
텍스트로 작성된 파일을 화면에 페이지 단위로 출력
예) # more anaconda-ks.cfg

리눅스 기본 명령어 (4)

- less

more와 용도가 비슷하지만 기능이 더 확장된 명령

예) # less anaconda-ks.cfg

- file

File이 어떤 종류의 파일인지를 표시

예) # file anaconda-ks.cfg

- clear

명령창을 깨끗하게 지워줌

예) # clear

사용자와 그룹(1)

- 리눅스는 다중 사용자 시스템(Multi-User System) 임
- 기본적으로 root라는 이름을 가진 수퍼유저(Superuser)가 있으며, 모든 작업을 할 수 있는 권한이 있음
- 모든 사용자를 하나 이상의 그룹에 소속되어 있음
- 사용자는 /etc/passwd 파일에 정의되어 있음

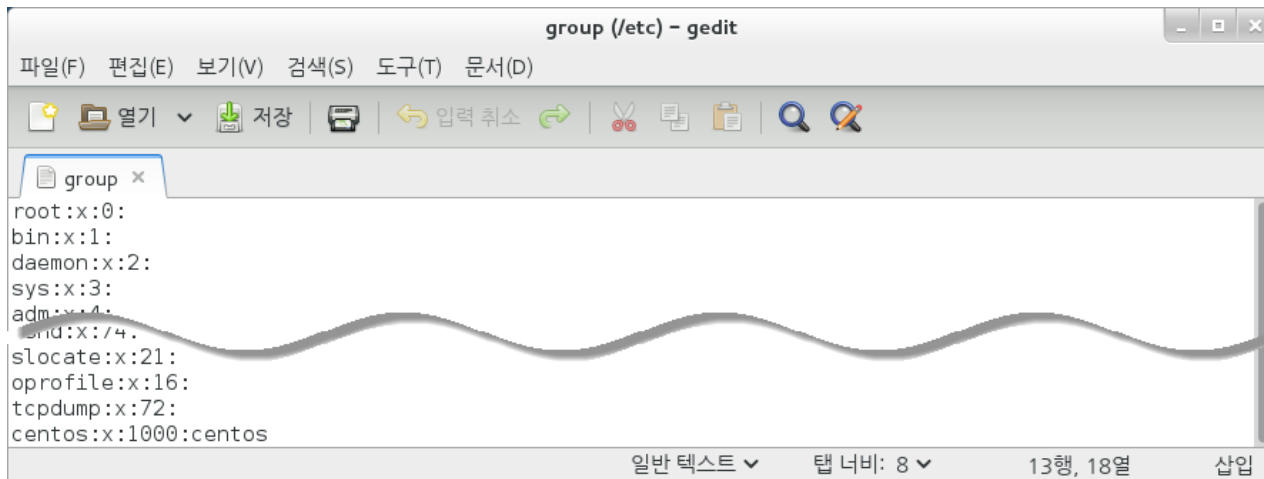
```
passwd (/etc) - gedit
파일(F) 편집(E) 보기(V) 검색(S) 도구(T) 문서(D)
열기 저장 입력 취소
passwd x
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/nologin
postfix:x:89:89:/var/spool/postfix:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/ssh:/sbin/nologin
oprofile:x:16:16:Special user account to be used by OProfile:/var/lib/oprofile:/sbin/nologin
tcpdump:x:72:72::/sbin/nologin
centos:x:1000:1000:centos:/home/centos:/bin/bash
일반 텍스트 탭 너비: 8 1행, 1열 삽입
```

- 각 행의 의미는 다음과 같음

사용자 이름:암호:사용자 ID:사용자가 소속된 그룹 ID:전체 이름:홈 디렉터리:기본 셸

사용자와 그룹(2)

- 사용자의 비밀번호는 /etc/shadow 파일에 정의되어 있음
- 그룹은 /etc/group 파일에 정의되어 있음



```
group x
root:x:0:
bin:x:1:
daemon:x:2:
sys:x:3:
adm:x:4:
dnd:x:/4.
slocate:x:21:
oprofile:x:16:
tcpdump:x:72:
centos:x:1000:centos
```

- 각 행의 의미는 다음과 같음
그룹명:비밀번호:그룹 id:그룹에 속한 사용자명

사용자와 그룹 관련 명령어(1)

- useradd

새로운 사용자를 추가

예) # useradd newuser

➤ 사용자 생성시 옵션

-u : ID 지정

-g : 그룹 지정

-d : 홈 디렉터리 지정

-s : 셸 지정

- passwd

사용자의 비밀번호를 지정하거나 변경

예) # passwd newuser

- usermod

사용자의 속성을 변경

예) # usermod -g root newuser

- userdel

사용자를 삭제

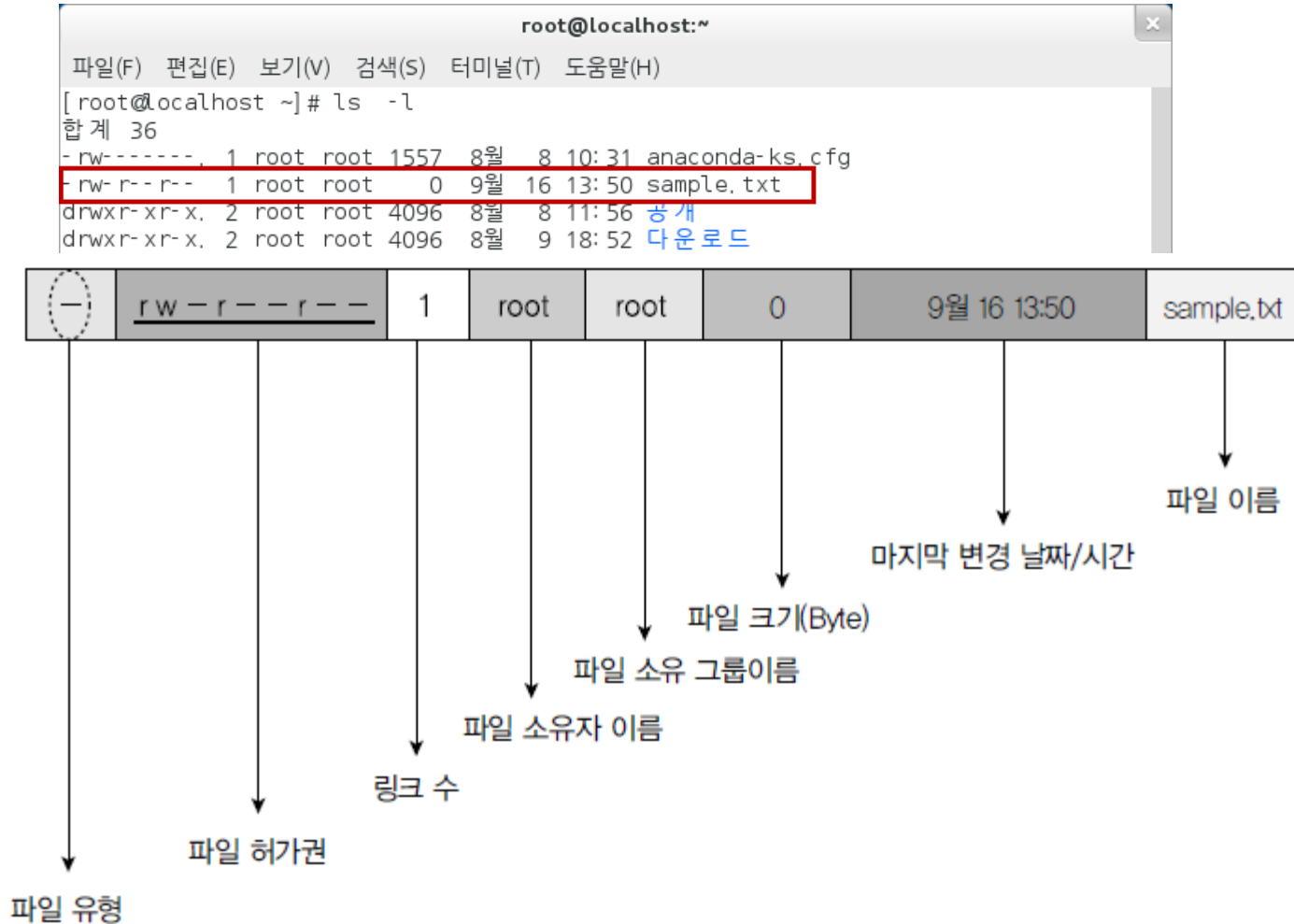
예) # userdel newuser

사용자와 그룹 관련 명령어(2)

- **groups**
현재 사용자가 속한 그룹을 보여줌
예) # groups
- **groupadd**
새로운 그룹을 생성
예) # groupadd newgroup
- **groupdel**
그룹을 삭제
예) # groupdel newgroup
- **groupmod**
그룹의 속성을 변경
예) # groupmod -n newgroup mygroup

파일과 디렉터리의 소유와 허가권 (1)

- 파일의 리스트와 파일 속성



파일과 디렉터리의 소유와 허가권 (2)

- 파일 유형

- 디렉터리일 경우에는 d, 일반적인 파일일 경우에는 -가 표시

- 파일 허가권(Permission)

- "rw-" , " r--" , " r--" 3개씩 끊어서 읽음 (r은 read, w는 write, x는 execute의 약자)
- 첫 번째 "rw-"는 소유자(User)의 파일접근 권한
- 두 번째의 "r--"는 그룹(Group)의 파일접근 권한
- 세 번째의 "r--"는 그 외의 사용자(Other)의 파일접근 권한
- 숫자로도 표시 가능 (오직 스)

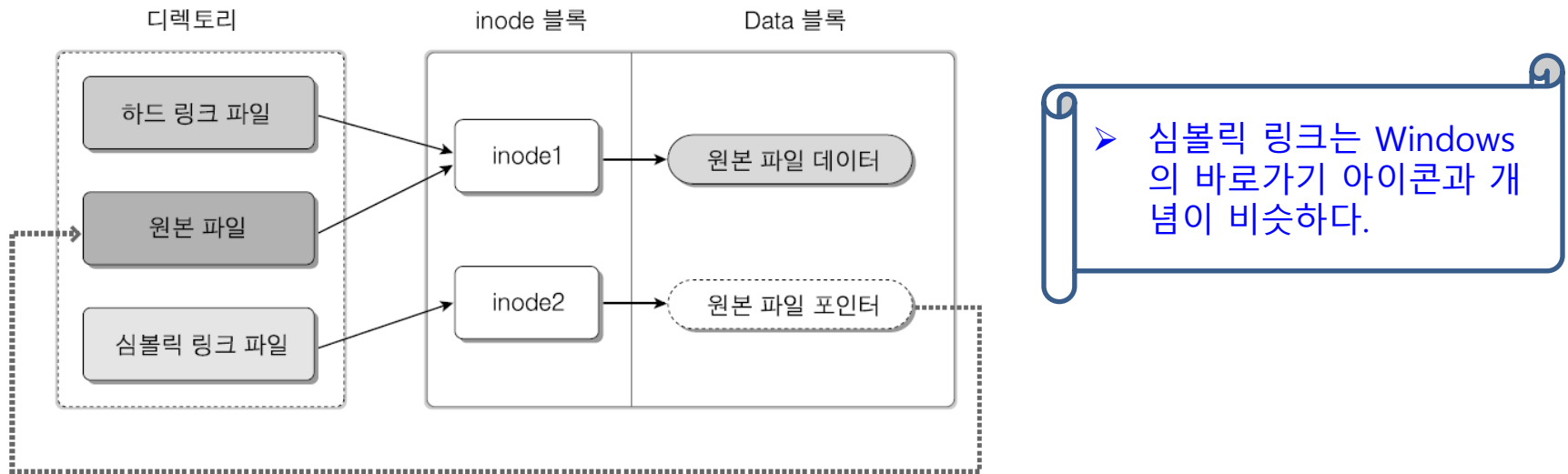
소유자(User)			그룹(Group)			그 외 사용자(Other)		
r	w	-	r	-	-	r	-	-
4	2	0	4	0	0	4	0	0
6			4			4		

파일과 디렉터리의 소유와 허가권 (3)

- **chmod 명령**
 - 파일 허가권 변경 명령어
 - 예) `# chmod 777 sample.txt`
- **파일 소유권(Ownership)**
 - 파일을 소유한 사용자와 그룹을 의미
- **chown/chgrp 명령**
 - 파일의 소유권을 바꾸는 명령어
 - 예) `# chown centos.centos sample.txt` 또는
`# chown centos sample.txt` 및 `# chgrp centos sample.txt`

링크

- 파일의 링크(Link)에는 하드 링크(Hard Link)와 심볼릭 링크(Symbolic Link 또는 Soft Link) 두 가지가 있음



- 하드 링크를 생성하면 "하드링크파일"만 하나 생성되며 같은 inode1을 사용 (명령 : # ln 링크대상파일이름 링크파일이름)
- 심볼릭 링크를 생성하면 새로운 inode2를 만들고, 데이터는 원본 파일을 연결하는 효과 (명령 : # ln -s 링크대상파일이름 링크파일이름)

프로그램 설치를 위한 RPM (1)

- RPM(Redhat Package Manager)
 - Windows의 "setup.exe"와 비슷한 설치 파일
 - 확장명은 *.rpm이며, 이를 '패키지(Package)'라고 부름.



```
root@localhost:/run/media/root/CentOS 7 x86_64/Packages
파일(F) 편집(E) 보기(V) 검색(S) 터미널(T) 도움말(H)
[root@localhost Packages]# pwd
/run/media/root/CentOS 7 x86_64/Packages
[root@localhost Packages]# ls -l gedit-3*
-rw-rw-r-- 2 root root 2706324  7월  4 10:27 gedit-3.8.3-6.el7.x86_64.rpm
[root@localhost Packages]#
```

패키지이름-버전-릴리즈번호.CentOS버전.아키텍처.rpm

- 패키지이름 : gedit → 패키지(프로그램)의 이름
- 버전 : 3.8.3 → 대개 3자리수로 구성. 주버전, 부버전, 패치버전
- 릴리즈번호 : 6 → 문제점을 개선할 때마다 붙여지는 번호
- CentOS 버전 : el7 → CentOS에서 배포할 경우에 붙여짐
- 아키텍처 : x86_64 → 64비트 CPU를 의미

프로그램 설치를 위한 RPM (2)

- 자주 사용하는 RPM 명령어 옵션
 - 설치 : rpm -Uvh 패키지파일이름.rpm
 - U → (대문자) 패키지가 설치/업그레이드
 - v → 설치과정의 확인
 - h → 설치진행과정을 "#"마크로 화면에 출력
 - 삭제 : rpm -e 패키지이름
 - 이미 설치된 패키지 질의
 - rpm -qa 패키지 이름 → 패키지가 설치되었는지 확인
 - rpm -qf 파일의절대경로
→ 파일이 어느 패키지에 포함된 것인지 확인
 - 아직 설치되지 않은 rpm 파일에 대한 질의
 - rpm -qlp 패키지파일이름.rpm
→ 패키지 파일에 어떤 파일들이 포함되었는지 확인
 - rpm -qip 패키지파일이름.rpm → 패키지 파일의 상세정보

프로그램 설치를 위한 RPM (3)

- RPM 단점
 - ‘의존성’ 문제
 - A패키지가 설치되기 위해서 B패키지가 필요할 경우, RPM으로는 해결이 까다로움.
 - 이를 해결하기 위해 YUM이 등장함

편리한 패키지 설치, YUM (1)

- YUM(Yellowdog Updater Modified) 개념
 - “rpm”명령의 패키지 의존성 문제를 완전하게 해결됨.
 - 인터넷을 통하여 필요한 파일을 저장소(Repository)에서 자동으로 모두 다운로드해서 설치하는 방식

➤ 저장소의 URL은 “/etc/yum.repos.d/” 디렉터리

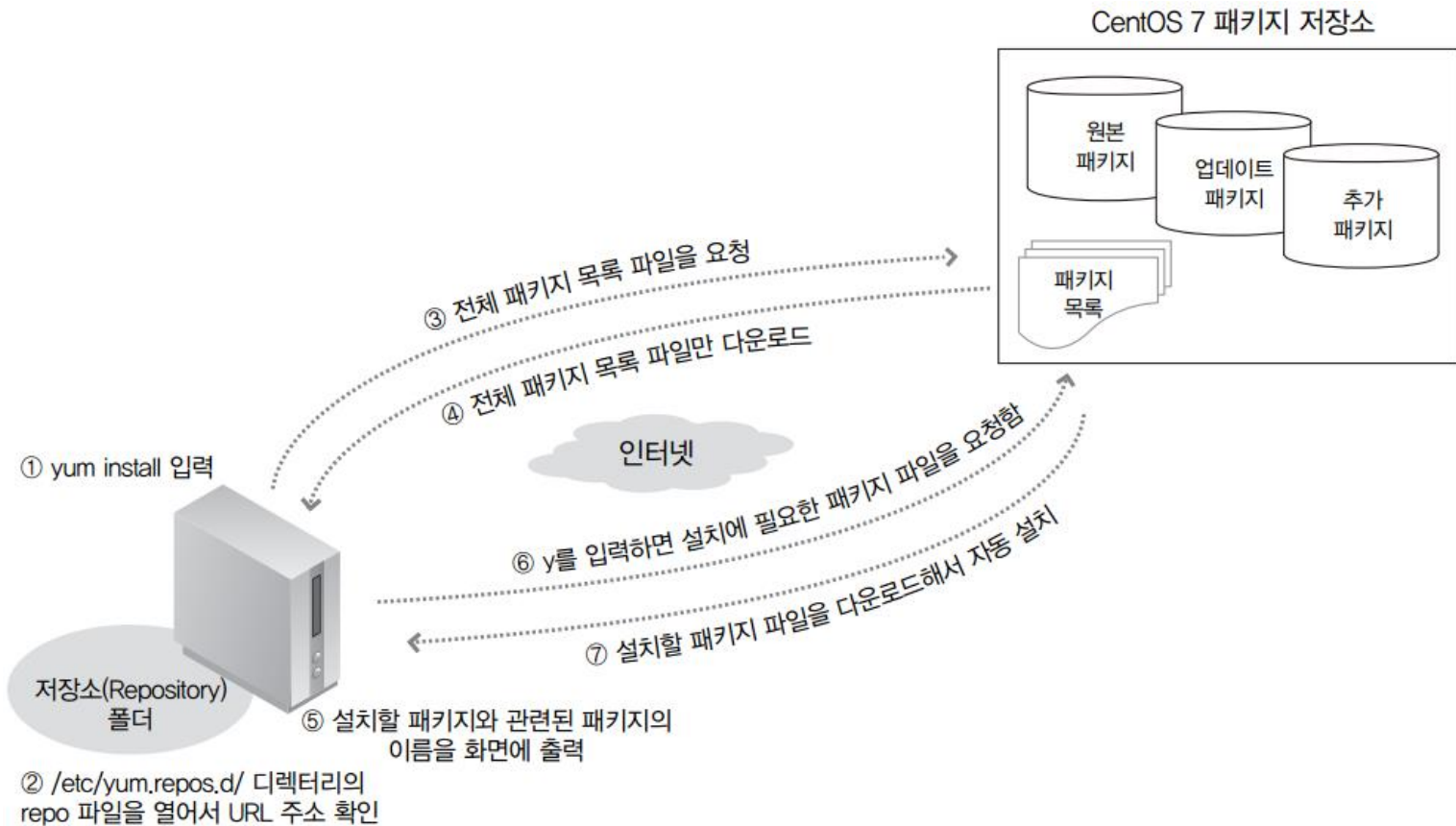
- YUM 기본적인 사용법
 - 기본 설치 : yum install 패키지이름
 - 주로 “yum -y install 패키지이름”으로 사용
 - “-y”는 사용자의 확인을 모두 “yes”로 간주하고 설치를 진행한다는 옵션
 - RPM 파일 설치 : yum localinstall rpm파일이름.rpm
 - 업데이트 가능한 목록 보기 : yum check-update
 - 업데이트 : yum update 패키지이름
 - 삭제 : yum remove 패키지이름
 - 정보 확인 : yum info 패키지이름

편리한 패키지 설치, YUM (2)

- YUM 고급 사용법
 - 패키지 그룹 설치
 - `yum groupinstall "패키지그룹이름"`
 - 패키지 리스트 확인
 - `yum list 패키지이름`
 - 특정 파일이 속한 패키지 이름 확인
 - `yum provides 파일이름`
 - GPG 키 검사 생략
 - `yum install --nogpgcheck rpm파일이름.rpm`
 - CentOS 19에서 검증되지 않은 패키지를 강제로 설치할 때 사용
 - 기존 저장소 목록 지우기
 - `yum clean all`

편리한 패키지 설치, YUM (3)

- YUM 작동 방식 설정 파일
 - 'yum install 패키지이름' 명령이 작동하는 방식

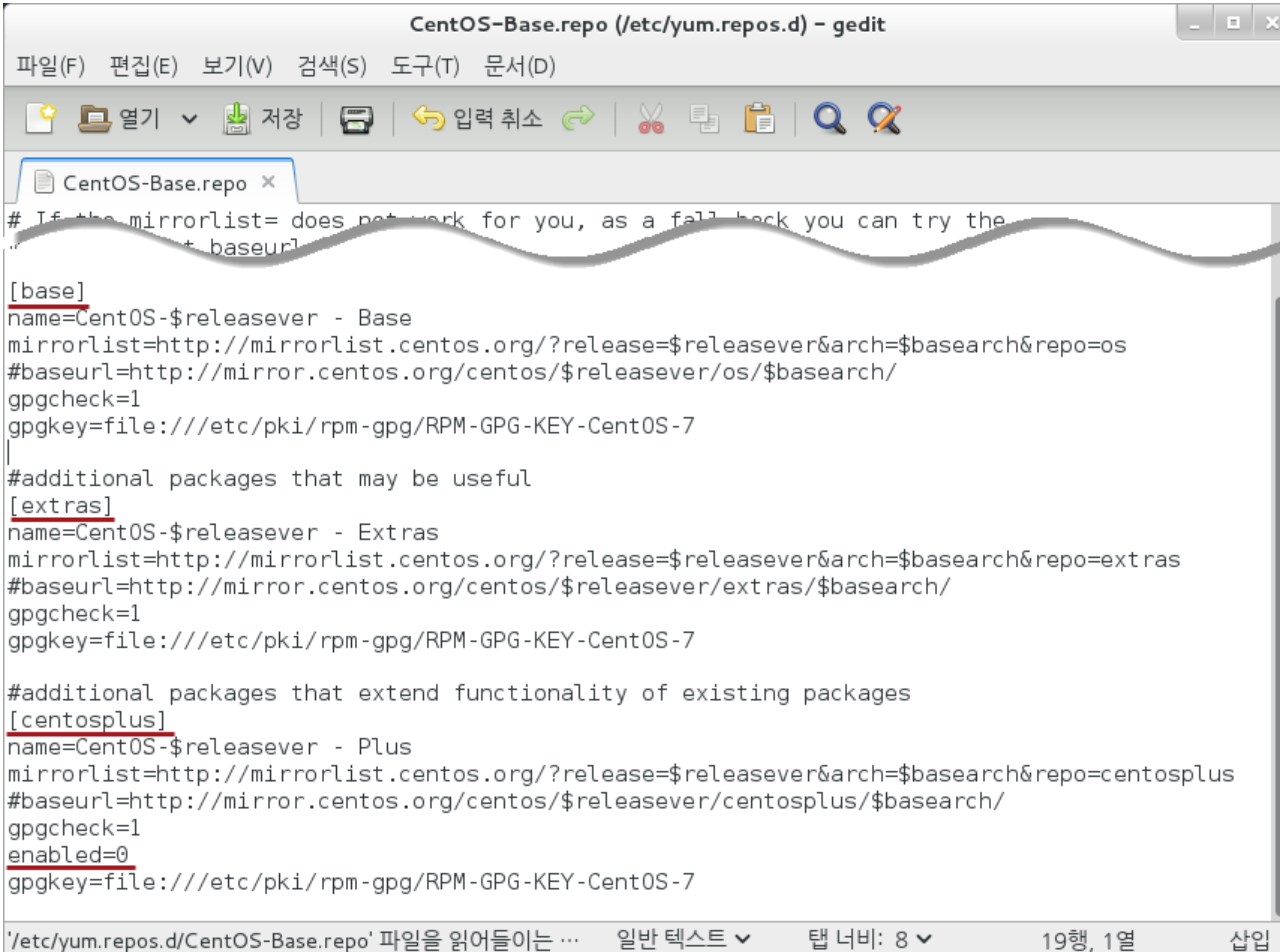


편리한 패키지 설치, YUM (4)

- YUM 작동 방식 설정 파일
 - /etc/yum.conf 파일 : 특별히 변경할 필요 없음
 - /etc/yum.repos.d/ 디렉터리
 - yum 명령을 입력했을 때 검색하게 되는 네트워크의 주소가 들어 있는 여러 개의 파일이 있음
 - /etc/yum.repos.d/ 디렉터리의 *.repo 파일
 - CentOS-Base.repo : [base], [extra]만 남기고 [updates] 부분은 삭제했음. 즉, 출시 시점의 원본 패키지만 설치됨.

편리한 패키지 설치, YUM (5)

- CentOS-Base.repo 파일



```
CentOS-Base.repo (/etc/yum.repos.d) - gedit
파일(F) 편집(E) 보기(V) 검색(S) 도구(T) 문서(D)

CentOS-Base.repo x
# If the mirrorlist= does not work for you, as a fall back you can try the
# baseurl=

[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that may be useful
[extras]
name=CentOS-$releasever - Extras
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=extras
#baseurl=http://mirror.centos.org/centos/$releasever/extras/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

#additional packages that extend functionality of existing packages
[centosplus]
name=CentOS-$releasever - Plus
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=centosplus
#baseurl=http://mirror.centos.org/centos/$releasever/centosplus/$basearch/
gpgcheck=1
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7

'/etc/yum.repos.d/CentOS-Base.repo' 파일을 읽어들이는 ... 일반 텍스트 ▾ 탭 너비: 8 ▾ 19행, 1열 삽입
```


파일의 압축과 묶기 (1)

- 파일 압축

- 압축파일 확장명은 xz, bz2, gz, zip, Z 등
- xz나 bz2 압축률이 더 좋음

- 파일 압축 관련 명령

- xz : 확장명 xz로 압축을 하거나 풀어준다

예) xz 파일명

xz -d 파일명.xz

- bzip2 : 확장명 bz2로 압축을 하거나 풀어준다

예) bzip2 파일명

bzip2 -d 파일명.bz2

- bunzip2 : "bzip2 -d" 옵션과 동일한 명령어

- gzip : 확장명 gz으로 압축을 하거나 풀어준다

예) gzip 파일명

gzip -d 파일명.gz

- gunzip : "gzip -d" 옵션과 동일한 명령어

파일의 압축과 묶기 (2)

- 파일 묶기

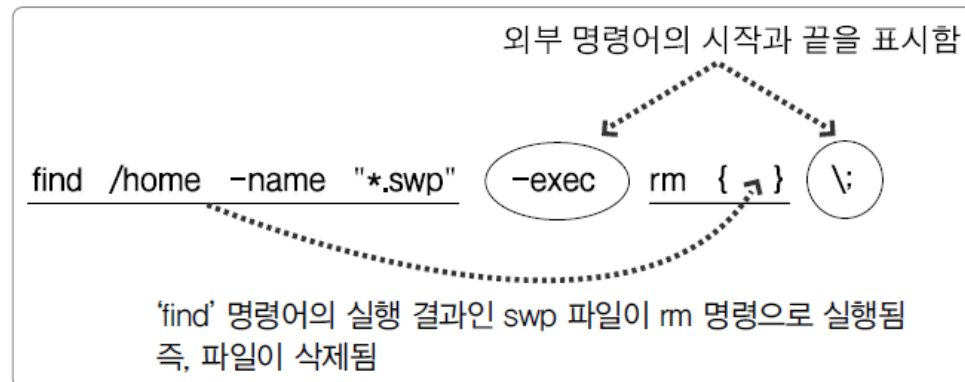
- 리눅스(유닉스)에서는 '파일 압축'과 '파일 묶기'는 원칙적으로 별개의 프로그램으로 수행
- 파일 묶기의 명령어는 'tar'이며, 묶인 파일의 확장명도 'tar'이다

- 파일 묶기 명령(tar)

- tar : 확장명 tar로 묶음 파일을 만들어 주거나 묶음을 풀어 준다
동작 : c(묶기), x(풀기), t(경로확인)
옵션 : f(파일), v(과정보이기), J(tar+xz), z(tar+gzip), j(tar+bzip2)
- 사용 예
tar cvf my.tar /etc/sysconfig/ → 묶기
tar cvfJ my.tar.xz /etc/sysconfig/ /etc/sysconfig/ → 묶기 + xz 압축
tar xvf my.tar → tar 풀기
tar xvfJ my.tar.xz /etc/sysconfig/ → xz 압축 해제 + tar 풀기

파일 위치 검색

- find [경로] [옵션] [조건] [action] : 기본 파일 찾기
 - [옵션] -name, -user(소유자), -newer(전,후), -perm(허가권), -size(크기)
 - [action] -print(디폴트), -exec (외부명령 실행)
 - 사용 예
 - # find /etc -name "*.conf"
 - # find /bin -size +10k -size -100k
 - # find /home -name "*.swp" -exec rm { } \;



- which 실행파일이름 : PATH에 설정된 디렉터리만 검색
- whereis 실행파일이름 : 실행 파일, 소스, man페이지 파일까지 검색
- locate 파일이름 : 파일 목록 데이터베이스에서 검색

네트워크 관련 필수 개념

- TCP/IP
 - 컴퓨터끼리 네트워크 상으로 의사소통을 하는 “프로토콜” 중 가장 널리 사용되는 프로토콜의 한 종류
- 호스트 이름(Hostname)과 도메인 이름(Domain name)
 - 호스트 이름은 각각의 컴퓨터에 지정된 이름
 - 도메인 이름(또는 도메인 주소)는 hanbit.co.kr과 같은 형식
- IP 주소
 - 각 컴퓨터의 랜카드에 부여되는 중복되지 않는 유일한 주소
 - 4바이트로 이루어져 있으며, 각 자리는 0~255까지의 숫자
 - 예) Server의 IP 주소는 192.168.111.100
- 네트워크 주소
 - 같은 네트워크에 속해 있는 공통된 주소 (예 : 192.168.111.0)

중요한 네트워크 관련 명령어 (1)

- nmtui
 - 네트워크와 관련된 대부분의 작업을 이 명령어에서 수행
 - 자동 IP 주소 또는 고정 IP주소 사용 결정
 - IP주소, 서브넷 마스크, 게이트웨이 정보 입력
 - DNS 정보 입력
 - 네트워크 카드 드라이버 설정
 - 네트워크 장치(ens32)의 설정
 - 텍스트 기반으로 작동함
- systemctl <start/stop/restart/status> network
 - 네트워크의 설정을 변경한 후에, 변경된 내용을 시스템에 적용시키는 명령어

중요한 네트워크 관련 명령어 (2)

- ifup <장치이름> 및 ifdown <장치이름>
 - 네트워크 장치를 On 또는 Off 시키는 명령어
- ifconfig <장치이름>
 - 장치의 IP주소 설정 정보를 출력
- nslookup
 - DNS 서버의 작동을 테스트하는 명령어
- ping <IP주소 또는 URL>
 - 해당 컴퓨터가 네트워크상에서 응답하는지를 테스트하는 간편한 명령어

네트워크 설정과 관련된 주요 파일

- `/etc/sysconfig/network`
 - 네트워크의 기본적인 정보가 설정되어 있는 파일
- `/etc/sysconfig/network-scripts/ifcfg-ens32`
 - ens32 장치에 설정된 네트워크 정보가 모두 들어 있는 파일
- `/etc/resolv.conf`
 - DNS 서버의 정보 및 호스트 이름이 들어 있는 파일
- `/etc/hosts`
 - 현 컴퓨터의 호스트 이름 및 FQDN이 들어 있는 파일

프로세스, 데몬 (1)

- 정의
 - 하드디스크에 저장된 실행코드(프로그램)가, 메모리에 로딩되어 활성화된 것
- 포그라운드 프로세스(**Foreground Process**)
 - 실행하면 화면에 나타나서 사용자와 상호작용을 하는 프로세스
 - 대부분의 응용프로그램
- 백그라운드 프로세스(**Background Process**)
 - 실행은 되었지만, 화면에는 나타나지 않고 실행되는 프로세스
 - 백신 프로그램, 서버 데몬 등
- 프로세스 번호
 - 각각의 프로세스에 할당된 고유번호
- 작업 번호
 - 현재 실행되고 있는 백그라운드 프로세스의 순차번호

프로세스, 데몬 (2)

- 부모 프로세스와 자식 프로세스
 - 모든 프로세스는 부모 프로세스를 가지고 있음
 - 부모 프로세스를 kill 하면, 자식 프로세스도 자동으로 kill 됨
- 프로세스 관련 명령
 - **ps**
 - 현재 프로세스의 상태를 확인하는 명령어
 - "ps -ef | grep <프로세스 이름>"을 주로 사용함
 - **kill**
 - 프로세스를 강제로 종료하는 명령어
 - "kill -9 <프로세스 번호>"는 강제 종료