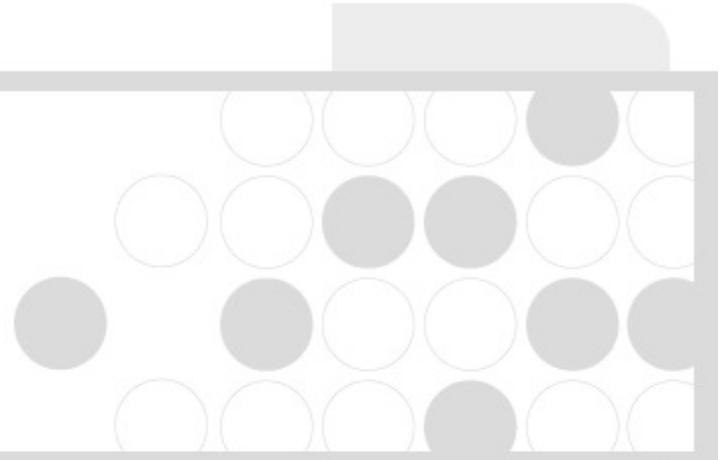


# 1. 리눅스 기초 및 실습1



(주)주스  
박종원

jason.park@juice.co.kr

## 1. 클라이언트 서버 환경

- 1) 개요
- 2) 서버 운영체계
- 3) 유닉스와 리눅스 비교
- 4) 데스크탑 형식/ 서버 형식
- 5) 실제 서버/ 가상서버

# ● 1. 클라이언트 서버 환경

## 1) 개요

- 일상적 정보처리 시스템은 클라이언트/서버 환경으로 구성되어 있음( vs Standalone)
- 클라이언트는 사용자에게 보여지는 부분
- 서버는 클라이언트가 수행할 수 있는 서비스를 지원하는 부분



## ● 1. 클라이언트 서버 환경

### 2) 서버의 운영체계

- 클라이언트는 화면에 보이는 것, 기기의 다양한 기능 사용, 사용자 중심
- 서버는 데이터처리, 통신, 업무 중심, 대용량, 가용성, 확장성, 안정성
- 서버의 운영체계는? Windows보다는 linux/Unix가 유리하며 90%이상을 차지

#### 1) 운영체계(Operating System, OS)

① 운영체계 : 컴퓨터의 하드웨어와 소프트웨어를 제어하여, 사용자가 컴퓨터를 쓸 수 있게 만들어주는 프로그램. <그림 1-1>

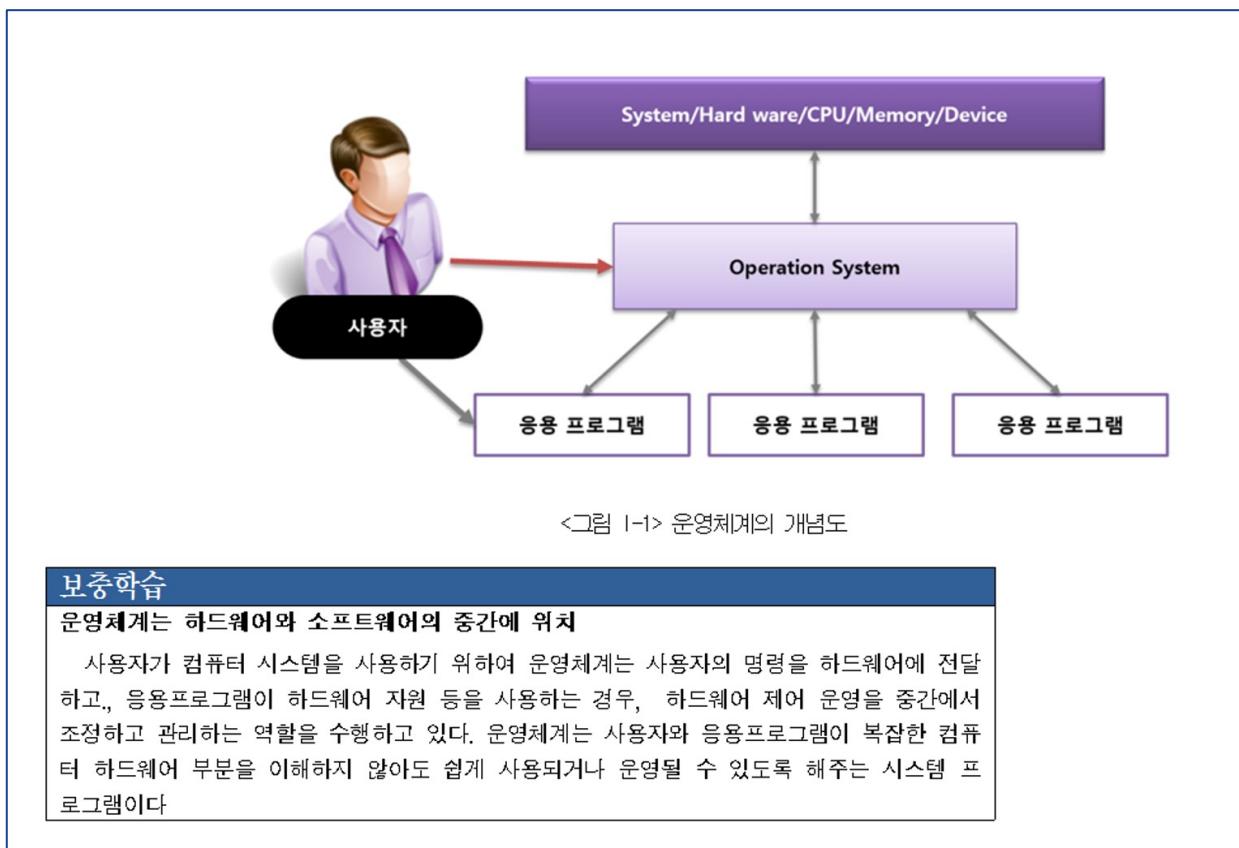
② 운영체계의 기능.

- 하드웨어와 응용프로그램간의 인터페이스 역할
- CPU, 주기억장치, 입출력장치 등의 컴퓨터 자원 관리
- 인간과 컴퓨터간의 상호작용을 제공함과 동시에 컴퓨터의 동작을 구동(Booting)
- 작업의 순서를 정하며 입출력 연산을 제어
- 프로그램의 실행을 제어
- 데이터와 파일의 저장을 관리

## ● 1. 클라이언트 서버 환경

### 2) 서버의 운영체계

- 클라이언트는 화면에 보이는 것, 기기의 다양한 기능 사용, 사용자 중심
- 서버는 데이터처리, 통신, 업무 중심, 대용량, 가용성, 확장성, 안정성
- 서버의 운영체계는? Windows보다는 linux/Unix가 유리하며 90%이상을 차지



## ● 1. 클라이언트 서버 환경

### 3) 유닉스와 리눅스 비교

- 유닉스 : 안정성, 대용량, 기존의 강자, 코어 업무(금융-계정계, 국가, 국방 ...)
- 리눅스 : 오픈소스, 최근에 서버의 OS로 신흥강자, 기존 업무도 많이 변환(U2L)

#### ① 유닉스

- 중 · 대형서버시스템에서 가장 많이 사용되는 OS, 고성능, 고 가용성의 운영 체계
- 예: IBM(AIX), HP(HP-UX), SUN(Solaris)

#### ② 리눅스

- 1991년 리누스 토발즈(Linus Torvalds)가 중 · 대형 기종에서만 작동하던 유닉스 운영체계를 PC에서도 작동할 수 있게 만든 운영 체계

#### ③ 리눅스의 기능

- 프로그램 소스 코드를 무료로 공개하여 사용자는 원하는 대로 특정 기능을 추가함.
- 어느 플랫폼에도 포팅이 가능함.
- 현재는 개인용PC, 기업의 중대형 컴퓨터, 임베디드 기기, 모바일 (Android,iOS)기기에서도 리눅스가 사용됨.

## ● 1. 클라이언트 서버 환경

---

### 4) 데스크탑 형식과 서버형식

- 리눅스
  - 서버의 OS 역할 뿐만 아니라, 현재 윈도우(MacOS)의 PC OS의 역할도 담당할 뻔..
  - 즉 text기반 OS에서 Windows 같은 UI기반의 데스크탑 형식에 대한 노력
- 안드로이드의 탄생 -> 리눅스 데스크탑이 진화한 모바일 버전
- 많은 리눅스 배포판이 존재(명령어들은 대부분 비슷)
  - 시장강자 Ubuntu (70~80%), 또 다른 강자 CentOS (나머지)
  - Ubuntu : 시장에 강자다 보니 참고할 문서, 구글링이 쉽다.
  - CentOS : 돈주고 사는 기존 기업용 리눅스인 red Hat과 호환, 오라클 사용 가능, 큰 기업 고려
- 데스크탑과 달리 서버형식은 딱히 화면을 보여주는 윈도우 UI가 필요하지 않다.
- 경량화, 고성능화를 위하여 굳이 낭비할 필요는 없다

## ● 1. 클라이언트 서버 환경

### 4) 데스크탑 형식과 서버형식

#### 1) 데스크 탑/Desktop) 형식

- ① 일반 사용자 PC의 Windows 운영체제와 유사한 형식임
- ② 윈도우의 상용성의 대응하여 GNU정신의 입각하여 태동함(무료 윈도우).
- ③ 서버의 용도로 사용할 수 있지만, 실제 서버 환경을 운영하는데 윈도우 형태의 GUI가 불필요한 서버 리소스를 사용함.
- ④ 다양한 GUI를 선호하는 경우 Desktop형식을 설치하여 서버로 사용할 수 도 있음



<그림 1-3> 데스크 탑 형식의 리눅스 GUI

## ● 1. 클라이언트 서버 환경

### 4) 데스크탑 형식과 서버형식

#### 2) 서버(Server) 형식

- ① 불필요한 윈도우 GUI를 없애고, 서버용으로 적합하게 구성된 리눅스 형식임
- ② 본 교재에서는 리눅스 명령어 실습, 리눅스 셸 프로그래밍을 익히기에 적절한 형식인 서버 형식으로 실습을 진행함

```
login as: kopoctc
kopoctc@192.168.56.101's password:
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic i686)

 * Documentation:  https://help.ubuntu.com/

 System information as of Mon Jan 18 09:47:26 KST 2016

 System load:  0.0          Processes:      90
 Usage of /:   28.7% of 6.75GB  Users logged in:  0
 Memory usage: 5%
 Swap usage:  0%          IP address for eth0: 10.0.2.15
                           IP address for eth1: 192.168.56.101

 Graph this data and manage this system at:
 https://landscape.canonical.com/

Last login: Wed Jan 13 16:51:05 2016 from 192.168.56.1
kopoctc@kopoctc:~$
```

<그림 1-4> 서버 형식의 리눅스

## ● 1. 클라이언트 서버 환경

---

### 5) 실제 서버, 가상 서버

- 실제서버는 대용량, 고가용성, 다중사용자
- 가상서버 (Virtual Machine/ Cloud기술)기술이 나오기 전에는 작은 PC/서버에 리눅스를 설치하여 개발용으로 사용
- 현재는 PC의 성능이나 용량이 커서 해당 자원을 나누어 사용 (개인 가상머신, 버추얼 클라우드), 하지만 이 가상서버의 용량이 보통 개발자 1인용도이며, 범용 서비스를 하기 위해서는 용량확장을 고민해야 함
- 또는 퍼블릭 클라우드 (AWS,GCP, Azure)에서 작은 용량의 개발서버를 인터넷공간에 두어 개발...-> 서비스 개시때 대용량으로 증설된 클라우드 서버 활용

# ● 1. 클라이언트 서버 환경

## 5) 실제 서버, 가상 서버

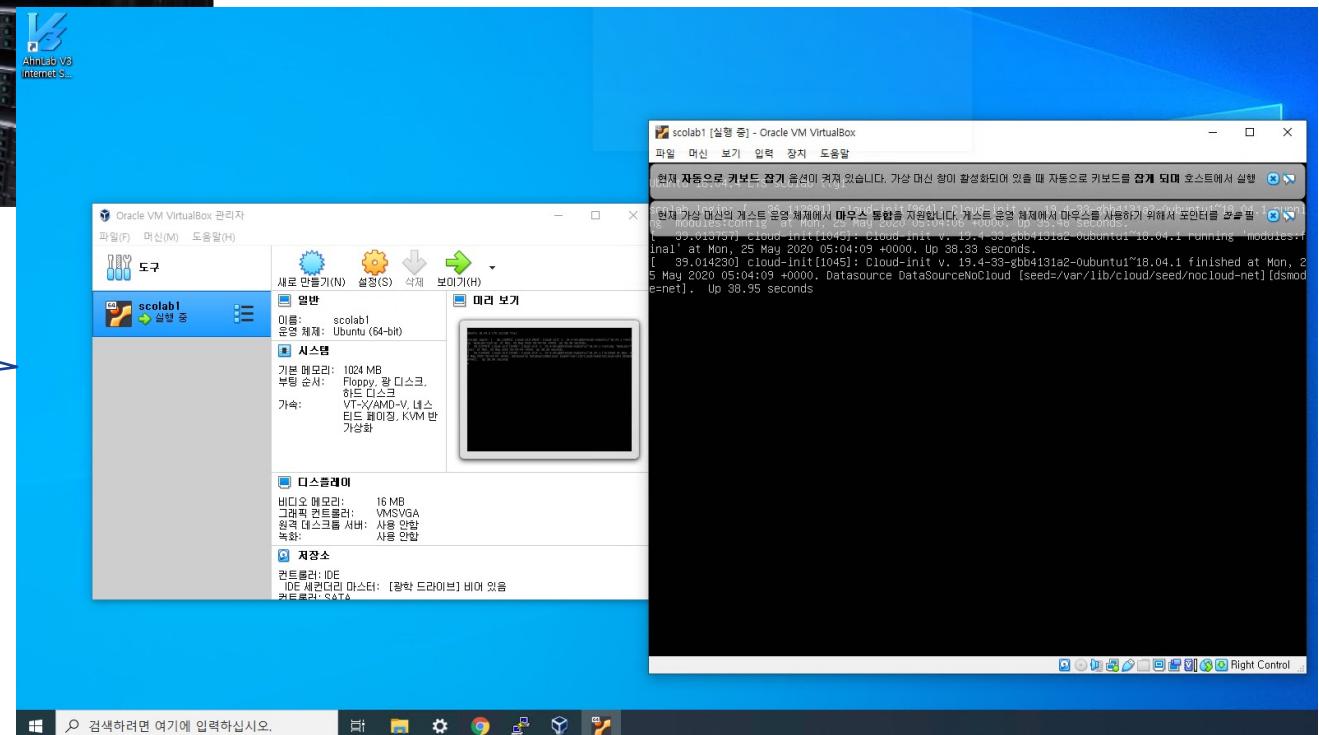


실제 서버는 서버실에서 곱게  
모셔놓고 에어컨에 향온 향습기에  
UPS 등 각종 대우를 받고

자 우리 KB은행, 네이버에 지금  
얼마나 많은 사람이 정보서비스를  
요청할까?

개발 테스트를 위하여 서버(작은  
용량의)를 나의 개발PC에 일부를  
할당하여 리눅스서버를 지원하는  
Virtual Machine 프로그램으로 운영

실제 PC가 대용량  
서버급..(CPU 병렬 연결  
128,256,512개)에서는 이렇게  
분할하여 대용량서버로 가능



### 3. 파일과 디렉토리

- 1) 디렉토리, 절대/ 상대 경로
- 2) 파일 탐색
- 3) 파일 내용 보기
- 4) 파일 다루기
- 5) 명령어 히스토리
- 6) 디렉토리 관리
- 7) 파일의 문자수 세기

### ● 3. 이해하기

---

#### (1) 파일과 디렉토리 다루기

리눅스서버 환경은 윈도우에서 파일 탐색기를 사용하는 것과는 다르게 GUI(Graphic User Interface) 환경이 아니며 하나 하나 명령어 쉘 상태에서 파일과 디렉토리 관련 명령어를 입력하여야 한다. 이러한 명령어를 하나하나 배워보도록 한다.

### ● 3. 이해하기

#### 1) 디렉토리 절대상대 경로

윈도우 OS의 파일을 폴더의 모아두는 개념과 마찬가지로 리눅스/유닉스에서는 디렉토리와 파일의 개념을 사용한다

- ① 디렉토리의 위치는 절대경로와 상대경로로 표시됨
- ② 윈도우에서 최상위 경로는 하드디스크 등 디바이스의 최상위 디렉토리(폴더)로 예를 들어 “C:\W”로 표시되며 이때 최상위 경로는 슬래쉬 기호“\”, “\W”로 표시된다. 유닉스와 리눅스에서는 슬래쉬가 아닌 역 슬래쉬 “/”기호가 최상위 디렉토리이며, 루트(root)디렉토리라고 읽는다.

##### ③ 절대경로

- 루트(root)디렉토리부터 현재 파일이 위치한 디렉토리의 경로를 전체로 표기한 경로

- 예 : /home/kopoetc/test/help.txt

##### ④ 상대경로

- 명령어 월 상태에서 현재 위치로부터 파일이 있는 디렉토리를 표기한 경로

- 현재위치를 점으로 나타내서 표기

- 현재 위치는 점을 한 개“,.”, 하위디렉토리는 점이 두 개“..”로 표시

Tip

예 : 현재 위치가 /home/kopoetc라면

- 1) ./test/help.txt (상대경로 표시) -> /home/kopoetc/test/help.txt
- 2) ../../home/kopoetc/test.help.txt : ..은 상위 디렉토리를 의미하며,  
결국 /home/kopoetc/test/help.txt 과 같은 위치

### ● 3. 이해하기

---

#### 2) 파일 탐색

파일을 탐색하기 위한 명령으로 `pwd`, `cd`, `ls`가 있다. <그림 I -35>  
참고

① `pwd` (print name of current/working directory)

- 현재의 작업 디렉토리가 어디인지 출력

② `cd` (change directory)

- 현재의 작업 디렉토리를 바꿈.

③ `ls` (list directory contents)

- 디렉토리에 파일 목록을 보여줌.

```
root@kopoctc:~# pwd
/root
root@kopoctc:~# cd /home
root@kopoctc:/home# ls
aaa.log  bbb.log  ccc.log  kopoctc  kopoctc1
root@kopoctc:/home#
```

<그림 I -35> `pwd`, `cd`, `ls` 명령어

### ● 3. 이해하기

---

#### 3) 파일 내용 보기

텍스트(TEXT)파일의 내용을 볼 수 있는 명령으로 `cat`, `page`, `head`, `tail`이 있으며, 파일의 내용이 많을 경우 화면 단위로 끊어서 보기위한 명령어로 `|more` 가 있다.

##### ① `cat`

- `cat filename` : `filename`이라는 파일의 내용을 한 번의 출력 <그림 I-36> 참조

##### ② `more`

- | (파이프)를 같이 사용하여 화면 단위로 출력

- 예 : “`cat filename | more`”

##### ③ `page`

- `page filename` : `filename`의 파일을 화면 단위로 출력함.

- 멈춰진 현재 화면에서 아무키나 누르면 다음 페이지가 보여짐.

##### ④ `head`

- 파일의 내용을 맨 앞을 기준으로 보여줌

- `head -n filename` : `filename`의 파일을 처음 n줄을 표시

- 예 : `head -10 abc.txt` : `abc.txt`파일을 처음부터 10줄을 보여줌

##### ⑤ `tail`

- 파일의 내용을 맨 뒤를 기준으로 보여줌

- `tail -n filename` : `filename`의 파일을 뒤부터 n줄을 표시

### ● 3. 이해하기

·**tail -f filename** : 계속 작성중인 파일의 마지막을 계속적으로 표시

·시스템관리 관련 로그저장 파일등을 실시간으로 감시할 때 많이 사용

Tip

모든 명령어는 기본적으로 사용되는 명령어를 소개하는 것을 원칙으로 한다. 하지만 명령어들 대부분이 선택(option)값으로 인자를 “-A”와 같은 형식으로 넘겨서 사용한다. 옵션 값은 다양한 기능들이 있으니 명령어를 사용전 man 명령어로 명령어 사용 옵션을 확인하는 습관을 가지도록 한다

```
root@kopoc tc:/etc/network# cat interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet dhcp
auto eth1
iface eth1 inet static
    address 192.168.56.101
    netmask 255.255.255.0
    network 192.168.56.0
root@kopoc tc:/etc/network#
```

<그림 1-36> cat 명령어 사용

## ● 4. 실습하기(1)

---

### 1) 파일탐색

- ① [pwd]
- ② [cd]
- ③ 상위 디렉토리 , 하위 디렉토리
- ④ 절대경로 , 상대 경로
- ⑤ . / .. \* : 특별한 문자의 사용
- ⑥ [ls –adglsFR]

### 2) 파일 내용보기

- ① [cat filename]
- ② “cat filename | more ”
- ③ [page filename]
- ④ [head –n filename]
- ⑤ [tail –n filename]
- ⑥ [tail –f filename]

### ● 3. 이해하기

---

#### 4) 파일 다루기

파일을 복사하고 파일 이름을 바꾸는 명령으로 `mv`, `cp`가 있다. <그림 I-37>

##### ① mv

· 파일을 이동 (잘라내서 복사)하는 명령

·`mv oldfilename newfilename` : `oldfile`의 파일을 `newfile`로 이동함. 결국 파일명을 바꾸는 명령 (예: `mv a.txt b.txt` : `a.txt`파일명을 `b.txt`로 바꿈)

·`mv filename dirName` : 파일을 해당 디렉토리로 보냄. 뒤의 인자가 파일명이 아니라 디렉토리명인 경우 해당 디렉토리에 파일을 옮기는 결과가 됨 (예: `mv a.txt adir` : `a.txt`파일을 `adir`의 디렉토리로 옮기는 명령이며 결국 `mv a.txt adir/a.txt`와 같은 명령으로 뒤 인자의 `a.txt`가 생략된 것으로 인식된 것)

·`mv oldDirName newDirName` : 두인자의 명칭이 디렉토리인 경우, `oldDirName`의 모든 파일을 `newDirName`이라는 디렉토리를 만든후 옮기는 명령이 됨. 결국 결과는 디렉토리명을 바꾸는 명령임

##### ② cp

· 파일을 복사(기존 파일은 남음)하는 명령

·`cp oldfilename newfilename` : `oldfile`의 파일을 `newfile`로 복사함.

##### Tip

`cp a.txt b.txt` : `a.txt`파일을 `b.txt`로 복사하는 명령으로 실행후 `a.txt`와 `b.txt`파일 두 개가 존재

·`cp filename dirName` : 파일을 해당 디렉토리로 보냄. 뒤의 인자가 파일명이 아니라 디렉토리명인 경우 해당 디렉토리에 파일을 복사하는 결과가 됨

### ● 3. 이해하기

#### Tip

`cp a.txt adir : a.txt파일을 adir의 디렉토리로 복사하는 명령이며 결국 cp a.txt  
adir/a.txt` 같은 명령으로 뒤 인자의 a.txt가 생략된 것으로 인식된 것  
와 같은

· `cp oldDirName newDirName` : 두인자의 명칭이 디렉토리 인 경우, `oldDirName`의 모든 파일을 `newDirName`이라는 디렉토리를 만든 후 복사하는 명령이 됨.

· 실제는 `cp -R oldDirName newDirName` 명령이 많이 사용됨. -R 옵션은 하위디렉토리까지 모두 복사하는 명령

#### ③ rm

· 파일을 지우는 명령

· `rm filename`: 파일을 지움 (예: `rm a.txt` a.txt파일을 지움)

· `rm -i filename`: 파일을 지움, 단 확인 메시지가 나와서 지울것인지 묻고 yes를 선택시만 지움

· `rm -r` : 해당 위치의 파일 및 디렉토리를 지움

· `rm -d` : 빈 디렉토리를 지움

· `rm -f` : 파일을 지울 때 확인 물음을 하지 말고 강제로 지움

· 다음 명령어 사용은 주의하여야 함 [`rm *`] [`rm -F`] [`rm -rf`]

· `rm *` : 모든 파일을 다 지우는 명령

· `rm -F`: 지우는 과정을 묻게 되있더라도 이를 무시하고 묻지않고 지움

· `rm -rf`: 해당 위치의 모든 파일 및 디렉토리를 묻지도 말고 무조건 지움. 만일 최상위 루트(`root`)디렉토리에서 해당 명령을 실행시 서버의 모든 파일이 포맷(format)수준으로 지워짐. 회사 운영자가 이 명령을 잘못 사용하다간 사표쓰는 경우가 생김.

## ● 3. 이해하기

---

```
root@kopoctc:/home/kopoctc# ls
help.txt
root@kopoctc:/home/kopoctc# cp help.txt help2.txt
root@kopoctc:/home/kopoctc# ls
help2.txt  help.txt
root@kopoctc:/home/kopoctc# mv help.txt hep13.txt
root@kopoctc:/home/kopoctc# ls
help2.txt  hep13.txt  root@
kopoctc:/home/kopoctc#
```

<그림 | -37> cp, mv 명령어 사용

## ● 4. 실습하기(2)

---

### 3) 파일 다루기

- ① [mv oldfilename newfilename]
- ② [mv filename dirName]
- ③ [mv oldDirName newDirName]
- ④ [cp filename {path/}filename]
- ⑤ [cp filename path]
- ⑥ [cp DirName newDirName]

### 4) 파일 지우기

- ① [rm filename]
  - ② [rm -i filename]
- 주의 [rm \*] [rm -F] [rm -rF]

### ● 3. 이해하기

---

#### 5) 명령어 히스토리

리눅스 명령은 윈도우 시스템과 달리 매번 명령어를 입력하기 때문에 불편할 수 있다. 하지만 이전에 입력한 명령어를 다시 찾아 명령할 수 있는 기능이 있는데, 이 명령어 히스토리 기능을 이용하면 명령어 사용이 편리해 진다. <그림 I-38>

##### ① [r 또는 !]

- 유닉스 계열(kohn shell)에서는 “r”을 입력하고 리눅스 계열(c shell)에서는 “!” (느낌표)를 입력하면 기존 사용했던 명령어를 찾을 수 있음

- 예를 들어 “!c”이라고 명령하면 기존 사용한 명령어 중 c로 시작하는 명령을 실행한다

##### ② [화살표 위, 아래]

- 지금까지 사용한 명령어를 순차적으로 보여줌.

```
root@kopoctc:/home/kopoctc# ls
help2.txt hep13.txt
root@kopoctc:/home/kopoctc# !c
cp help.txt help2.txt
cp: cannot stat 'help.txt' : No such file or directory
root@kopoctc:/home/kopoctc#
```

<그림 I-38> 명령어 히스토리 사용

##### ③ 명령어 히스토리 저장 파일

- 지금까지 사용한 명령어는 해당 사용자의 기본 디렉토리 내에 “.bash\_history” (리눅스 bash) 또는, “.history”(일반적 유닉스)라는 파일안에 저장되어 있음. <그림 I-39>

### ● 3. 이해하기

---

- 어떤 사용자가 어떤 명령어로 시스템에 접근하였는지 근거파일로 사용됨.
- 기업용 시스템에서는 해당 파일을 실시간으로 중앙통제장치로 전송하여 보관함으로서 불순한 의도의 접근을 감시하는 용도로도 사용됨.

```
kopoetc@kopoetc:~$ pwd  
/home/kopoetc  
kopoetc@kopoetc:~$ ls -a  
. .. .bash_history .bash_logout .bashrc .cache help2.txt hepl3.txt  
kopoetc@kopoetc:~$ tail -n5 .bash_history  
apt update  
su -  
ls  
who  
su -  
kopoetc@kopoetc:~$
```

<그림 1-39> 명령어 히스토리 저장 파일

## ● 3. 이해하기

---

### 6) 디렉토리 관리

디렉토리를 관리하기 위하여 디렉토리를 만들거나, 디렉토리를 지우는 작업을 하는데 이때 `mkdir`, `rmdir` 명령을 사용한다. <그림 I-40>

#### ① `mkdir`

- `[mkdir dirname]` : 새로운 디렉토리를 만듬

#### ② `rmdir`

- `[rmdir dirname]` : 해당 디렉토리를 지움, 단 해당 디렉토리에 파일이 있으면 지울 수 없음

- 해당 디렉토리에 파일이 있는데 모든 파일 및 해당 디렉토리를 지우기 위하여 `rm` 명령어를 응용하여 사용 (예: `abc` 디렉토리 및 디렉토리 하단 파일까지 지우기 위하여 `rm -rf abc`를 사용)

### ● 3. 이해하기

---

```
kopoctc@kopoctc:~$ pwd  
/home/kopoctc  
kopoctc@kopoctc:~$ mkdir mydir  
kopoctc@kopoctc:~$ ls -a  
. .bash_history .bashrc help2.txt mydir .viminfo  
.. .bash_logout .cache hep13.txt .profile  
kopoctc@kopoctc:~$ cd mydir  
kopoctc@kopoctc:~/mydir$ pwd  
/home/kopoctc/mydir  
kopoctc@kopoctc:~/mydir$ touch myfile  
kopoctc@kopoctc:~/mydir$ ls  
myfile  
kopoctc@kopoctc:~/mydir$ cd ../  
kopoctc@kopoctc:~$ rm mydir  
rm: cannot remove ‘mydir’ : Is a directory  
kopoctc@kopoctc:~$
```

<그림 1-40> mkdir, rmdir 명령어 사용

### ● 3. 이해하기

---

#### 7) 파일의 문자수 세기

파일의 글자 수를 세는 명령어로 **wc**가 있는데, 일반적으로 파일의 크기를 글자 수로 알고 싶을 때나, 쉘 프로그래밍에서 응용하여 많이 사용된다. <그림 I-41>

##### ① **wc**

- 파일 내부의 글자 수 및 줄 수를 보여줌
- [**wc**] : 출력되는 순서는 파일의 줄 수[**newline**], 단어 수[**word**], 글자 수[**byte**]로 보여줌

```
kopoctc@kopoctc:~/mydir$ ls -al
total 12
drwxrwxr-x 2 kopoctc kopoctc 4096 Jan 18 10:56 .
drwxr-xr-x 4 kopoctc kopoctc 4096 Jan 18 10:55 ..
-rw-rw-r-- 1 kopoctc kopoctc    27 Jan 18 10:55 myfile
kopoctc@kopoctc:~/mydir$ cat myfile
You are a good guy
Me too.
kopoctc@kopoctc:~/mydir$ wc myfile
      2    7   27 myfile
```

<그림 I-41> **wc** 명령어 사용

## ● 4. 실습하기(2)

---

### 3) 파일 다루기

- ① [mv oldfilename newfilename]
- ② [mv filename dirName]
- ③ [mv oldDirName newDirName]
- ④ [cp filename {path/}filename]
- ⑤ [cp filename path]
- ⑥ [cp DirName newDirName]

### 4) 파일 지우기

- ① [rm filename]
  - ② [rm -i filename]
- 주의 [rm \*] [rm -F] [rm -rF]

## ● 4. 실습하기(3)

---

### 5) 명령어 히스토리

- ① [r , !]
- ②[화살표, “ctrl-[“ 후 “ctrl-k” 과 “ctrl-k”]
- ③[.history .bash\_history]

### 6) 디렉토리 관리, 글자 수 세기

- ① [mkdir dirname]
- ② [rmdir dirname]
- ③ [wc]

앞 강의에서는 리눅스 운영체계에서 파일에 대한 간단한 명령어를 다루었다. 여러분이 리눅스나 유닉스 시스템을 다루는 관리자이거나 프로그램, 데이터베이스를 다루는 업무의 담당자라면 파일을 깊게 다루는 경우가 많다. 각종 서버시스템의 운영 상황들이 실시간으로 기록되는 파일인 로그파일을 분석하기 위하여 파일을 여러 방면으로 다루어야 하며, 프로그램이나 데이터베이스의 입출력으로 일반적인 파일이 많이 사용되기 때문이다. 이번 강의에서는 리눅스에서 파일과 디렉토리 부분을 이해 후 실습하도록 한다.

**Tip**

원도우에서 파일을 다루는 유틸리티 프로그램이나 워드, 아래 아 한글 등에서 기능 등 을 사용하여 두 개의 파일의 내용을 비교하는 방법을 알아보고, 학습 준비도를 높이도록 한다.

## (1) 파일 필터

파일의 내용 중 원하는 부분을 검색하거나 파일에 내용을 쓰거나 출력하고 또는 기존파일에 추가하여 내용을 쓸 수 있는 데, 이는 파일 필터를 활용한 명령어를 이용한다

### 1) 파일(|)과 grep 명령

파일내 지정된 패턴이 있는지 찾아내기 위하여 파일 필터(|)와 grep 명령을 사용 한다

#### ① grep

·grep(Globally find Regular-Expression and Print : 지정된 표현식이 전체에 있는지 찾아서 프린트 함) 명령의 사용형식

·grep pattern filename : 주어진 패턴이 있는지 해당 지정된 파일이나 또는 확장파일형식으로 검색

#### Tip

예1) grep "gh" abc.txt : abc.txt파일안에 gh라는 문자가 있으면 출력

#### Tip

예2) grep "gh" \*.txt : 모든 .txt로 명칭이 끝나는 파일을 대상으로 파일안에 gh라는 문자가 있으면 출력하라

·~ | grep pattern : 필터를 이용하여 검색하는 방식으로 주어진 패턴을 검색하는 방법은 동일함

#### Tip

예) cat abc.txt|grep gh : abc.txt파일을 전부출력하고, 그 출력을 받아서 gh라는 글자 를 찾음. 결국 같은 결과가 나옴

#### ② grep 주요 옵션

- w : 전체 단어가 일치되는 경우 출력
- n : 라인넘버 출력
- v : 단어가 일치하지 않는 경우 출력
- l : 해당되는 파일명을 출력

Tip

예) cat xinetd.conf|grep -nw telnetd : xinetd.conf파일내용 중 “telnetd”라는 단어가 정확히 일치하게 들어간 라인 수를 출력

```
kopoetc@kopoetc:~/mydir$ cat abc.txt
abc
def
ghi
jkl
mno
p

kopoetc@kopoetc:~/mydir$ grep "gh" abc.txt
ghi
kopoetc@kopoetc:~/mydir$ cat abc.txt|grep gh
ghi
kopoetc@kopoetc:~/mydir$ cat abc.txt|grep -w gh
kopoetc@kopoetc:~/mydir$ cat abc.txt|grep -n gh
3:ghi
kopoetc@kopoetc:~/mydir$ cat abc.txt|grep -v gh
abc
def
jkl
mno
p

kopoetc@kopoetc:~/mydir$
```

<그림 1-47> 파일필터 명령어 사용

## 2) 리다이렉션(Redirection)

명령어의 결과를 다른 명령어의 입력으로 사용하는 경우나, 명령어의 결과를 파일에 기록하는 경우에 리다이렉션 기호를 사용한다.

### ① 파이프( | )

- 명령1 | 명령2 : (파이프) 어떤 명령의 결과를 받아 다른 명령을 실행

Tip

예) cat abc.txt | grep gh : 첫 번째 명령 abc.txt파일을 전부 출력하고, 그 출력을 받아 서 두 번째 명령인 gh라는 글자를 찾음.

### ② 리다이렉션, 꺽쇠 (>, >>)

- 명령 > filename : 어떤 명령의 결과를 지정된 명칭의 파일을 새로 생성하여 기록
- 명령 >> filename : 어떤 명령의 결과를 지정된 명칭의 파일 뒤로 계속 붙여서 기록함
- 명령 < filename : 어떤 명령의 입력으로 지정된 명칭의 파일을 사용함
- 예 1) cat xinetd.conf > a.file : xinetd.conf파일을 출력하고 이 결과를 a.file에 기록, 만일 a.file의 명칭으로 파일이 존재 하였다면, 이를 무시하고 새로 생성하고 기록함

Tip

예 1) cat xinetd.conf > a.file : xinetd.conf파일을 출력하고 이 결과를 a.file에 기록, 만일 a.file의 명칭으로 파일이 존재하였다면, 이를 무시하고 새로 생성하고 기록함

Tip

예 2) cat xinetd.conf >> a.file : xinetd.conf파일을 출력하고 이 결과를 a.file에 기록 하는데 기존 a.file의 맨끝부터 추가하여 기록함. 만일 a.file이 없다면 에러가 발생

**Tip**

예 3) cat xinetd.conf | grep telnet > a.file : 여러 가지 명령을 계속해서 연결하여 사용 할 수 있음. xinetd.conf파일을 출력하고, 이 결과중 telnet이라는 문자열이 있으면 그 줄 만 출력하는데, 그 결과를 a.file에 기록함. 이때 telnet이라는 문자열이 있는지 화면에는 출력하지 않고 파일에만 기록됨.

**Tip**

예 4) cat > aa : cat 명령어 뒤에 파일명이 지정되지 않으면 기본 키보드 입력을 받 음. cat > aa이라고 명령후 엔터를 치면 화면이 대기모드가 되며 키보드로 입력되는 값 을 받음. 키보드 입력을 하고 최종 ctrl+c로 입력을 종료하면 지금까지 키보드로 입력한 내용이 aa이라는 파일에 저장되어 있음 <그림 1-xx>

```
kopoctc@kopoctc:~/mydir$ cat aa
aaa
sasa
sasa
kopoctc@kopoctc:~/mydir$ cat > aa
aa
aa
^C
kopoctc@kopoctc:~/mydir$ cat aa
aa
aa
kopoctc@kopoctc:~/mydir$ cat >> aa
sdasa
sasa
^C
kopoctc@kopoctc:~/mydir$ cat aa
aa
aa s
dasa
sasa
```

<그림 1-48> 리다이렉션 명령어 사용

Tip

예 5) grep gh > abc.txt : grep gh 명령을 실행하는데 abc.txt가 입력으로 사용. 결국 cat abc.txt |grep gh 와 동일한 명령이 됨

## ● 4. 실습하기(1)

---

### 1) 파일필터

- ① grep hello abc.txt
- ② grep hello \*
- ③ cat abc.txt | grep hello
- ④ ps -ef |grep http
- ⑤ grep -wn hello abc.txt
- ⑥ grep -wv hello abc.txt
- ⑦ grep -l hello abc.txt
- ⑧ grep -l hello \*
- ⑨ cat aa ; cat >aa; cat aa; cat >>aa;cat aa

## (2) 파일 비교, 정렬, 탐색

두 파일이 동일한 파일인지, 또는 다른 곳이 어디인지 알기 위하여 파일을 비교하거나, 원하는 파일이 어느 디렉토리에 위치해 있는지 파일을 찾기위한 명령어를 다룬다.

### 1) 파일 비교

파일을 비교하는 명령어는 `cmp`, `diff`의 명령이 있다. <그림 I-49>

#### ① cmp

- `cmp file1 file2` : 두 개의(`file1`,`file2`) 파일을 비교하는 명령.
- 쉘 스크립트 프로그래밍에서 사용하기 적합함
- 같으면 `exit code =0`을 반환 (쉘 스크립트에서 사용)
- 다르면 `exit code =1`을 반환 일치하지 않는 첫째 `byte` 출력

#### ② diff

- `diff file1 file2` : 두 개의(`file1`,`file2`) 파일의 차이를 보여 줌
- 두 파일 간 다른 부분을 각각 보여 주기 때문에 프로그램 작성시 프로그램 소스를 비교하는데 편리함

```
kopoctc@kopoctc:~/mydir$ cmp aa bb  
aa bb differ: byte 4, line 2  
kopoctc@kopoctc:~/mydir$ diff aa bb  
2c2  
< aa  
---  
>bb  
4c4  
< sasa  
---  
>cwsasa  
kopoctc@kopoctc:~/mydir$
```

<그림 1-49> cmp, diff 명령어의 차이

## 2) 파일 정렬

파일내용을 정렬하는 명령어는 **sort**의 명령이 있다. <그림 I -50>

### ① sort

- 파일내용을 정렬 조건에 따라 정렬
- 정렬기준인 필드를 선택하여 오름차순이나 내림차순으로 전체 줄을 순서대로 정렬
- 필드 기준은 빈칸을 기준으로 수행함
- **sort abc.txt** : abc.txt파일을 오름차순으로 정렬함
- **sort -r abc.txt** : abc.txt파일을 내림차순으로 정렬함
- **sort -k2 abc.txt >result.txt** : 2번째 필드를 기준으로 정렬

```
kopoctc@kopoctc:~/mydir$ cat aa
0 h      six
3 a      one
4 g      nine
2 i      eight
5 d      three
kopoctc@kopoctc:~/mydir$ sort -kl aa
0 h      six
2 i      eight
3 a      one
4 g      nine
5 d      three
kopoctc@kopoctc:~/mydir$ sort -r aa
5 d      three
4 g      nine
3 a      one
2 i      eight
0 h      six
kopoctc@kopoctc:~/mydir$
```

<그림 1-50> sort 명령어

### 3 파일 검색

파일을 검색하는 명령어는 **find** 명령이 있다. <그림 I-51>

#### ① find

- 원하는 조건의 파일의 위치를 찾아줌

1) 사용법 : **find dirname -option1 ...-optionN**

- **dirname**은 주어진 디렉토리 및 이하 하위 디렉토리를 검색함

- **option**을 지정하여 다양한 조건과 실행법을 지정할 수 있음

2) 주요 조건

- **-name** : 파일 이름이나 패턴을 지정하여 찾음

#### Tip

예1) **find . -name abc.txt**: 현재 디렉토리부터 하위 디렉토리 까지 abc.txt 파일을 찾음

#### Tip

예2) **find . -name \*.txt**: 현재 디렉토리부터 하위 디렉토리까지 맨 끝이 .txt로 끝나는 모든 파일을 찾음

- **-perm** : 파일의 퍼미션(권한)이 일치하는 파일을 찾음

**Tip**

예1) find . -perm 775 찾음( : 현재 디렉토리부터 추후 하위 디렉토리까지 파일권한이 775인 파일을 파일의 권한은 다시 배움)

**-type** : 파일의 파일유형이 일치하는 파일을 찾음

하위옵션) **-type b** : 문자로만 구성된 파일 찾기, **-type d** : 디렉토리 찾기, **-type f** : 일반적인 파일 찾기, **-type l** : 심 볼릭 링크 찾기(심볼릭 링크는 추후 배움)

**Tip**

예) find . -type d -name abc : 현재 디렉토리부터 하위 디렉토리까지 이름이 abc인 디렉토리를 찾음

**-user** : 파일의 사용자(user)가 일치하는 파일을 찾음

**Tip**

예) find / -user kopoetc : 루트(/) 디렉토리부터 하위디렉토리까지 파일의 사용자(만든 자 user)가 kopoetc인 파일을 찾음

**-group** : 파일의 그룹(group)이 일치하는 파일을 찾음

**Tip**

예) find / -group kopoetc : 루트(/) 디렉토리부터 하위디렉토리까지 파일의 그룹이 kopoetc인 파일을 찾음

- **-atime +n/-n/n** : 최근 n일 이전에 액세스된 파일을 찾아줌(**accessed time**)  
+n은 n일 또는 그보다 더 오래 전의 파일을, -n은 오늘 부터 n일 전까지의 파일을, n은 정확히 n일 전에 액세스된 파일을 찾아줌.

**Tip**

예) find / -atime 2 : 루트(/) 디렉토리부터 하위 디렉토리까지 파일중 2일전에 접근한 (access)한 파일을 찾아줌

- **--print** : 찾은 파일명을 출력. 일반적으로 옵션을 주지 않아도 기본적으로 수행
- **-ls** : 찾은 파일에 대한 ls명령을 수행

**Tip**

예) find / -name my\*.cnf -ls : 이 경우 my\*.cnf의 형태의 파일을 찾아서 해당 파일 을 ls명령을 실행함 <그림 | -51>

```
root@kopoc tc:~# find / -name my*.cnf -ls
262426      4 -rw-r--r--    1 root      root          2850 Oct 22
22:06
/usr/share/doc/mysql-server-5.5/examples/my-small.cnf
146136      4 -rw-r--r--    1 root      root         21 Jan 22
2021
/etc/mysql/conf.d/mysqld_safe_syslog.cnf
146131      4 -rw-r--r--    1 root      root        3505 Jan 22
```

<그림 | -51> ls옵션을 실행하여 find명령을 실행

## ● 4. 실습하기(3)

---

### 2) 파일비교 정렬

- ① `cmp abc.txt abc2.txt`
- ② `diff abc.txt abc2.txt`
- ③ `diff -Dflag abc.c abc2.c`
- ④ `sort abc.txt , sort -r abc.txt` : 내림차순
- ⑤ `sort -k2 abc.txt >result.txt` : 2번째 필드를 기준으로 정렬

### 3) 파일 찾기

- ① `find . -name '*.c' -print`
- ② `find . -mtime 14 -print`
- ③ `find . \(-name '*.c' -o -name '*.txt'\) -print` : -o: OR
- ④ `find / |grep abc` 이름에 abc가 들어있는 것은 모두
- ⑤ `find / -name abc` 이름이 abc인 것.

### (3) 파일 보관, 압축

저장공간의 백업이나 자료를 옮기기 위하여 여러 파일을 묶어서 보관하거나 이를 압축하여 보관하고, 또한 파일의 압축을 풀 경우가 있는데, 이러한 파일의 보관과 압축을 위한 명령어를 다룬다.

#### 1) tar (현재는 zip을 많이 사용하나 필요할 때가 있음)

파일을 보관하거나 푸는 명령어는 tar의 명령이 있다. <그림 I -52>

##### ① 일반적 기능

- 파일을 묶고 푸는 기능 (압축 기능 없음)
- 고전적으로 가장 많이 사용
- 파일을 묶어서 테이프로 저장 등에 사용

##### ② 파일을 묶을 때

- 사용법 : `tar -cvf tarFileName fileList`
  - `-c` : fileList에 대한 tar형식의 백업 파일을 생성
  - `-v` : 진행되는 상황을 설명
  - `-f` : tar형식의 백업 파일 이름을 지정(default: `/dev/rmt/0` ... tape drive)

Tip

예) `tar -cvf abc mydir` : 만일 mydir이 디렉토리 일 경우 mydir디렉토리 이하 모든 파일을 abc라는 파일명으로 묶음

**Tip**

예) tar -cvf abc def ghi: 이 경우 def와 ghi파일을 묶어서 abc라는 파일로 저장함. 첫 번째 지정된 명칭이 묶이는 파일명이 되는 것에 유의

### ③ 묶은 파일을 풀 때

- 사용법 : tar -txru tarFileName fileList
- -t : tar형식의 백업 파일 안에 어떤 것들이 들어 있는지 목차만 보임
- -x : 백업 파일로부터 파일을 추출 복귀(extract)
- -r : fileList를 기준의 백업 파일 뒤에 무조건 덧붙임(rear)
- -u : 기존의 백업 파일에 이미 포함되어 있는 fileList 중 수정된 파일들만을 백업 파일의 뒤에 덧붙임.

디렉토리가 있어도 recursive하게 적용

**Tip**

예) tar -xvf abc : abc라는 파일이 tar명령어로 묶음파일일 경우 현재 디렉토리에 묶은 파일을 풀음

**Tip**

예) tar -xvf abc mydir : abc라는 파일이 tar명령어로 묶음파일일 경우 mydir 디렉토리에 묶은 파일을 풀음, 만일 mydir이라는 디렉토리가 없다면 디렉토리를 만들고 풀음

### ④ 기타 tar 이용

- 서버의 테이프 디바이스에 저장또는 테이프 디바이스 읽기
  - tar -cvf tarfile /dev/mmt0 , tar -xvf /dev/mmt0
  - 일반적으로 cd나 테이프 등 외장 저장매체는 /dev/cd0, /dev/mmt0 등으로 생성됨, 가상의 파일형식으로 만들어짐

```
root@kopoc:~/home# tar -cvf a.tar kopoc  
kopoc/  
kopoc/hepl3.txt  
kopoc/help2.txt  
kopoc/.viminfo  
kopoc/mydir/  
kopoc/mydir/myfile  
kopoc/mydir/abc.txt  
kopoc/mydir/aa  
kopoc/mydir/bb  
kopoc/.bashrc  
kopoc/.bash_logout  
kopoc/.bash_history  
kopoc/.profile  
kopoc/.cache/  
kopoc/.cache/motd.legal-displayed  
root@kopoc:~/home# ls  
a.tar  kopoc  
root@kopoc:~/home#
```

<그림 | -52> tar 명령어

## 2) compress / uncompress

파일을 압축과 푸는 명령어는 **compress** / **uncompress**의 명령이 있다.

### ① 압축 및 푸는 기능

- 파일을 압축 (파일 크기를 줄임)하고 푸는 명령어
- 압축하는 경우 **compress abc** 로 명령하면 abc파일이 압축되어 abc.z의 해당 이름의 .z의 확장자가 붙은 압축파일이 생성됨
- 압축을 푸는 경우 **uncompress abc.z** 로 명령하면 abc.z파일이 압축이 풀려서 abc로 복원된 파일이 생성됨
- 파일을 묶어서 압축하는 기능은 **gzip,gunzip** 명령어를 많이 사용하는데 이는 개인용PC에서 많이쓰는 ZIP압축방식을 사용.
- **compress/uncompress**는 유닉스, 리눅스 서버환경에서 전통적으로 많이 사용되는 유틸리티 임.
- **zip**유틸리티보다 일반적으로 압축효율은 떨어짐
- 우분투 리눅스에서 **compress/uncompress**의 명령은 **apt-get install ncompress**로 유틸리티 설치 후 사용가능

**Tip**

예1) **compress \*.txt** : 모든 \*.txt를 확장자로 가진 파일들이 파일 하나하나 별로 압축 이 수행되며 해당 파일명+.Z 의 명칭을 가진 파일이 여러 개 생김

**Tip**

예2) **uncompress \*.Z** : 모든 Z 확장자를 가진 각각의 파일에 대하여 uncompress 풀기를 실행

### 3) gzip / gunzip

파일을 압축과 푸는 명령어로 **gzip** / **gunzip**의 명령이 있다.

#### ① 압축 및 푸는 기능

- 많이 사용되는 파일을 압축 (파일 크기를 줄임)하고 푸는 명령어
- 압축하는 경우 **gzip abc** 로 명령하면 abc파일이 압축되어 abc.gz의 해당 이름의 .gz의 확장자가 붙은 압축파일이 생성됨
- 압축을 푸는 경우 **gunzip abc.z** 로 명령하면 abc.gz 파일이 압축이 풀려서 abc로 복원된 파일이 생성됨
- **gunzip** 명령은 **gzip -d** 명령과 동일

#### ② gzip명령어의 주요 옵션

- **-d** : 압축을 푼다
- **-l** : 현재 압축된 파일의 내용을 보여준다.
- **-r** : 현재 디렉토리부터 하위 디렉토리까지 전부를 압축한다.
- **-t** : 압축된 파일의 완전성을 검사한다.
- **-v** : 압축 진행상황을 보여준다.
- **-9** : 최대한 압축한다.

#### Tip

**gzip \* / gunzip \***로 전체 파일을 대상으로 압축과 풀기를 실행 <그림 1-53>

```
root@kopoctc:/home/kopoctc# ls
help2.txt  hep13.txt  mydir
root@kopoctc:/home/kopoctc# gzip *
gzip: mydir is a directory -- ignored
root@kopoctc:/home/kopoctc# ls
help2.txt.gz  hep13.txt.gz  mydir
root@kopoctc:/home/kopoctc# gunzip *
gzip: mydir is a directory -- ignored
root@kopoctc:/home/kopoctc# ls
help2.txt  hep13.txt  mydir
root@kopoctc:/home/kopoctc#
```

<그림 1-53> gzip/gunzip 명령어

## ● 4. 실습하기(3)

---

### 4) tar 명령어

- ① **tar -cvf tarfile .**
- ② **tar -tvf tarfile**
- ③ **tar -rvf tarfile reverse.c**
- ④ **tar -uvf tarfile reverse.c**
- ⑤ **mkdir ./tmp**
- ⑥ **cd tmp**
- ⑦ **tar -xvf ../tarfile**
- ⑧ **tar -xvf ../tarfile `tar -tf ../tarfile | grep 'lady\*'`**
- ⑨ **tar -xvf ../tarfile ./reverse.c**
- ⑩ **tar -cvf tarfile /dev/rmt0 , tar -xvf /dev/rmt0**

## ● 4. 실습하기(3)

---

### 5) compress, uncompress 명령어

- ① cp abc abc2
- ② compress abc // abc.Z과 abc2와 사이즈비교..
- ③ uncompress abc.z
- ④ compress \*.txt : 수많은 Z파일
- ⑤ uncompress \*.Z : 모든 Z 확장자 파일 uncompress

### 6) gzip, gunzip 명령어

- ① gzip -v9 tarfile
- ② ls tarfile.gz
- ③ gzip -l tarfile.gz
- ④ gzip -vd tarfile.gz
- ⑤ gunzip tarfile.gz
- ⑥ gzip -vd tarfile.gz > tar.log &