redhat CentOS

Unit 5
Directory & File Admin CMDs

http://cafe.daum.net/bscsolaris

단원 목표

- 디렉토리 이동 관련 명령어
- 디렉토리 관리 명령어
- 파일 관리 명령어
- 파일 내용 확인 명령어
- 기타 관리용 명령어

http://cafe.daum.net/bscsolaris

1. 리눅스 파일 시스템 계층 구조

Linux 시스템의 모든 파일은 파일 시스템 계층 구조로 알려진 역전된 단일 디렉토리 트리로 구성된 파일 시스템에 저장된다. 이 구조는 루트(/)가 계층 구조의 최상단에 있고, 디렉토리와 하위 디렉토리 분기는 루트(/) 아래쪽으로 뻗어 가기 때문에 뒤집어진 나무 모양이다.

다음은 tree 명령어를 사용하여 디렉토리 구조를 파악하였다. tree 명령어가 없다면 tree 패키지를 설치한다.
 # tree -d -L 1 /

```
/
|-- bin -> usr/bin
|-- boot
|-- dev
|-- etc
|-- home
|-- lib -> usr/lib
|-- lib64 -> usr/lib64
|-- media
|-- mnt
|-- opt
|-- proc
|-- root
|-- run
|-- sbin -> usr/sbin
|-- srv
|-- sys
|-- tmp
|-- usr
`-- var
```

CentOS 7.X 이후 버전에서는 다음과 같은 심볼릭 링크가 되어 있다.
- /bin -> /usr/bin
- /sbin -> /usr/sbin
- /lib -> /usr/lib
- /lib64 -> /usr/lib64

CentOS 6.X 이하 버전에서는 위의 디렉토리는 각각 서로 다른 파일 세트를 포함하는 별도의 디렉토리였다.

■ 중요한 운영체제 디렉토리

디렉토리	목 적
/usr	설치된 소프트웨어, 공유 라이브러리 포함된 파일 및 읽기 전용 프로그램 데이터, 중요한 하위 디렉토리에는 다음이 포함된다. (EX) **(WIN)** Program Files
/etc	시스템 고유의 구성 파일이 존재한다. OS 부팅시 설정, 보안 설정, 정보 파일들, 각종 서비스 설정 파일들이 존재한다. (EX) **(WIN)** Registry
/var	재부팅 후에도 유지는 시스템 고유의 가변 데이터가 존재한다. 동적으로 변경되는 파일(예: 데이터베이스, 캐시 디렉토리, 로그파일, 프린터로 전송된 문서, 웹 사이트 콘텐츠)은 /var에 존재할 수 있다. (EX) Log Directory : /var/log/*
/run	마지막 부팅 이후 시작된 프로세스의 런타임 데이터이다. 여기에는 프로세스 ID 파일과 잠금 파일 등이 포함된다. 디렉토리 내용은 재부팅하면 다시 생성된다. 이 디렉토리는 이전 버전에 있던 /var/run, /var/lock을 통합한다.
/home	일반 사용자의 홈 디렉토리이다. 일반 사용자의 개인 데이터 및 구성 파일을 저장하는 디렉토리이다. (EX) /home/fedora, **(WIN)** C:₩users₩fedora
/root	root 사용자 홈 디렉토리이다.
/tmp	어디에서나 쓸 수 있는 임시 파일용 공간이다. 10일 동안 엑세스, 변경 또는 수정되지 않은 파일은 이 디렉토리에서 자동으로 삭제된다. 다른 임시 디렉토리는 /var/tmp에 존재한다. 30일 이상 엑세스, 변경 또는 수정되지 않은 파일은 자동으로 삭제된다. **(WIN)** TMP=C:₩Users₩soldesk₩AppData₩Local₩Temp
/boot	부팅 프로세스를 시작하는 데 필요한 파일들이 존재한다. (EX) /boot/vmlinuz-4.18.0-240.el8.x86_64, **(WIN)** C:₩Windows₩Boot
/dev	시스템에서 하드웨어에 엑세스 하는 데 사용되는 특수 장치 파일을 포함한다. (EX) **(WIN)** 장치관리자(devmgmt.msc)

[실습] 파일 시스템에 대한 간단한 정보 확인

■ 사용시스템
- server1

■ 작업 시나리오
- 파일 시스템 구조에 대한 매뉴얼 검색하기

[EX1] 파일 시스템 구조에 대한 매뉴얼 검색하기
- 다음과 같은 매뉴얼 페이지를 확인한다.(5분)

[root@server1 ~]# man 7 file-hierarchy

```
NAME
       file-hierarchy - File system hierarchy overview

DESCRIPTION
       Operating systems using the systemd(1) system and service manager are
       organized based on a file system hierarchy inspired by UNIX, more
       specifically the hierarchy described in the File System Hierarchy[1]
       specification and hier(7), with various extensions, partially
       documented in the XDG Base Directory Specification[2] and XDG User
       Directories[3]. This manual page describes a more generalized, though
       minimal and modernized subset of these specifications that defines more
       strictly the suggestions and restrictions systemd makes on the file
       system hierarchy.

       Many of the paths described here can be queried with the systemd-
       path(1) tool.

GENERAL STRUCTURE
       /
           The file system root. Usually writable, but this is not required.
           Possibly a temporary file system ("tmpfs"). Not shared with other
           hosts (unless read-only).
.... (중략) ....
```

[root@server1 ~]# man 7 hier

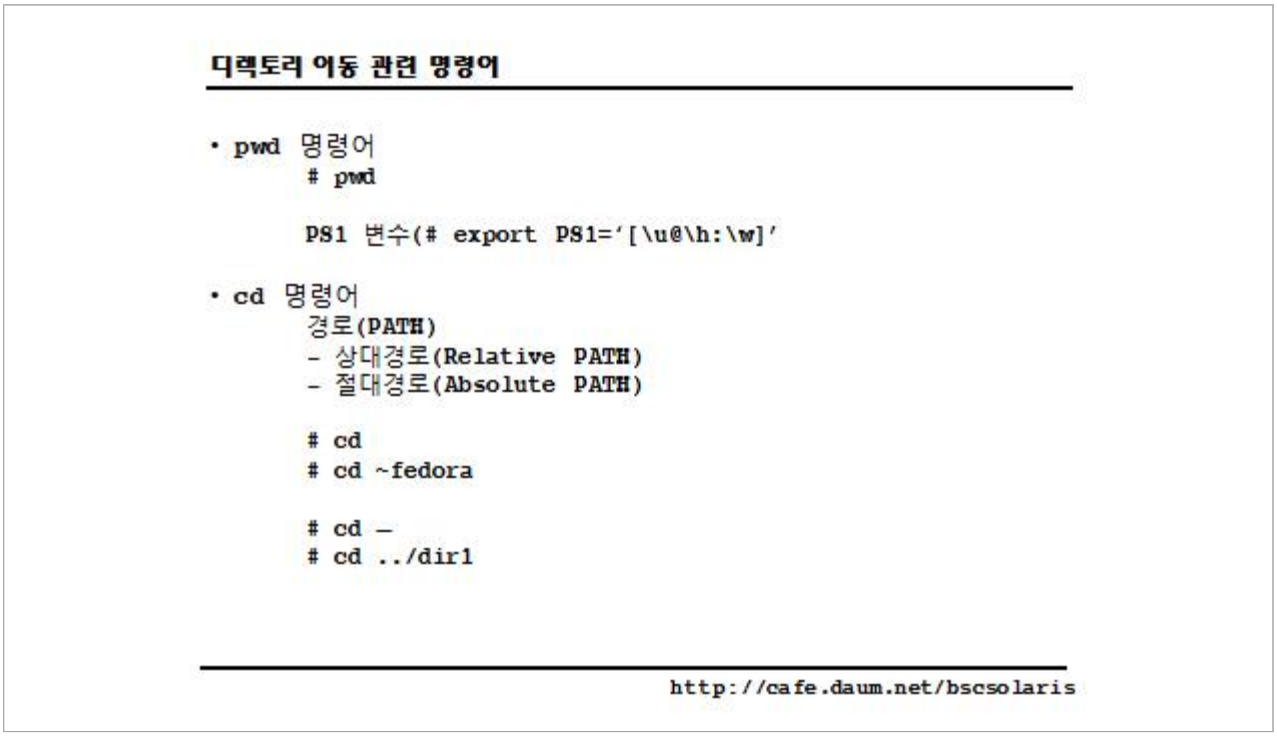
```
NAME
       hier - description of the filesystem hierarchy

DESCRIPTION
       A typical Linux system has, among others, the following directories:

       /      This  is  the  root  directory.  This  is  where the whole tree
              starts.

       /bin   This directory contains executable programs which are needed  in
              single user mode and to bring the system up or repair it.

       /boot  Contains static files for the boot loader.  This directory holds
              only the files which are needed during the  boot  process.  The
              map  installer  and  configuration  files should go to /sbin and
              /etc.  The operating system kernel (initrd for example) must  be
              located in either / or /boot.

       /dev   Special  or  device files, which refer to physical devices.  See
              mknod(1).
.... (중략) ....
```

[EX2] 다음과 같은 디렉토리의 용도를 조사한다.

다음과 같은 명령을 사용하여 표에 나온 디렉토리의 용도를 조사한다.
 # man 7 file-hierarchy
 # man 7 hier

다음 용도를 파악해 주시기 바랍니다.

디렉토리	목 적
/lib /lib64	
/media /mnt	
/opt	
/sys	
/bin /sbin	
/srv	



1 pwd CMD (printing working directory)

```
이름
       pwd - 현재/작업 디렉토리명을 출력한다

개요
       pwd
       pwd {--help,--version}

설명
       이  맨페이지는 GNU 버전의 pwd 를 다룬다.  pwd 는 현재 디렉토리의 완전한
       이름을 출력한다.  즉 출력된 이름의 모든 구성 요소들이 실제 디렉토 리 를
       의미한다.  - 심볼릭 링크가 아니라.

       대 부 분의 유닉스 쉘들은 내장 pwd 명령으로 비슷한 기능을 제공하고 있다.
       따라서 대부분의 pwd 명령은 지금 이것이 아니라 내부에서 수행되는 것일 것
       이다.
```

파일 작업 중이나 자료의 위치로 이동 할 경우 현재 내가 작업하고 있는 디렉토리의 위치를 알고 이동한 디렉토리를 지정해야 한다. 디렉토리 이동 관련 명령어에는 pwd 명령어, cd 명령어가 있다.

현재 내가 작업하고 있는 디렉토리의 위치를 절대경로를 통해 /(root)부터의 전체 경로를 출력해 준다.

[명령어 형식]
pwd

[실습] pwd 명령어 실습

■ 사용시스템
- main
- server1
- server2

■ 실습 시나리오
- pwd 명령어 간단한 실습
- main/server1/server2 - **PS1 변수** 설정

[EX1] pwd 명령어 간단한 실습

① 현재 디렉토리 확인
```
[root@server1 ~]# cd
[root@server1 ~]# pwd
```
```
/root
```

[참고] 홈디렉토리(default)
* 일반사용자 : /home/$USER (EX: /home/fedora)
* root사용자 : /root (EX: /root)

② /etc/sysconfig/network-scripts 디렉토리로 이동
```
[root@server1 ~]# cd /etc/sysconfig/network-scripts
[root@server1 network-scripts]# pwd
```
```
/etc/sysconfig/network-scripts/
```

③ cd 명령어에 인자 없이 수행하기 - 자신의 홈디렉토리로 이동하기
```
[root@server1 network-scripts]# cd
[root@server1 ~]# pwd
```
```
/root
```

[참고] PS1 변수 설정

pwd 명령어를 디렉토리 변경 할 때 마다 치는 것은 불편하다. 따라서 PS1 변수를 환경변수에 선언하여 사용하면 pwd 명령어를 굳이 사용하지 않아도 된다.

- PS1 변수: "**쉘 프롬프트**"를 정의할 때 사용하는 변수

```
# echo $PS1
[Wu@Wh WW]W$      ->     [root@linux249 ~]#

    # man bash
    /PROMPTING
    ------------------------
    Wu : username
    Wh : hostname
    WW : working directory(ex: /etc/sysconfig/network-scripts)
    Ww : working directory(ex: /etc/sysconfig/network-scripts)
    W$ : if      uid == 0 : #
         else    uid != 0 : $
    ------------------------

# PS1='a  '
# PS1='[Wu@Wh Ww]W$ '          /* 쉘프롬프트에 $PWD 변수 넣음 */

# gedit ~/.bashrc
```
```
..... (중략) .....
#
# Sfecific Configuration
#
export PS1='[Wu@Wh Ww]W$ '
```
```
# .  ~/.bashrc     (# source ~/.bashrc)
# echo $PS1
```

[EX2] PS1 변수 설정
- 각 시스템(main, server1, server2)의 PS1 변수 설정하기
- 프롬프트 색깔표와 PS1 프롬프트 설정 예제를 참고하여 설정한다.

```
■ 프롬프트 색깔 표
------------------
31 : 빨간색
32 : 초록색
33 : 갈색
34 : 파랑색
35 : 보라색
36 : 청록색
------------------

■ PS1 프롬프트 설정 예제
export PS1='₩[₩e[32;1m₩][₩u@₩h₩[₩e[33;1m₩] ₩w]₩$ ₩[₩e[m₩] '
```

① (main) main.example.com 서버의 PS1 변수 설정

[참고] main 서버 TUI -> GUI 전환(임시적으로)
 # systemctl isolate graphical.target
root 사용자로 로그인

[root@main ~]# gedit ~/.bashrc

```
.... (중략) ....
#
# Sfecific configuration
#
export PS1='₩[₩e[34;1m₩][₩u@₩h₩[₩e[33;1m₩] ₩w]₩$ ₩[₩e[m₩]'
```

[root@main ~]# source ~/.bashrc
[root@main ~]#

② (server1) server1.example.com 서버의 PS1 변수 설정
[root@server1 ~]# gedit ~/.bashrc

```
.... (중략) ....
#
# Sfecific configuration
#
export PS1='₩[₩e[31;1m₩][₩u@₩h₩[₩e[33;1m₩] ₩w]₩$ ₩[₩e[m₩]'
```

[root@server1 ~]# source ~/.bashrc
[root@server1 ~]#

③ (server2) server2.example.com 서버의 PS1 변수 설정

[참고] server2 서버 TUI -> GUI 전환(임시적으로)
 # systemctl isolate graphical.target
root 사용자로 로그인

[root@server2 ~]# gedit ~/.bashrc

```
.... (중략) ....
#
# Sfecific configuration
#
export PS1='₩[₩e[32;1m₩][₩u@₩h₩[₩e[33;1m₩] ₩w]₩$ ₩[₩e[m₩]'
```

[root@server2 ~]# source ~/.bashrc
[root@server2 ~]#

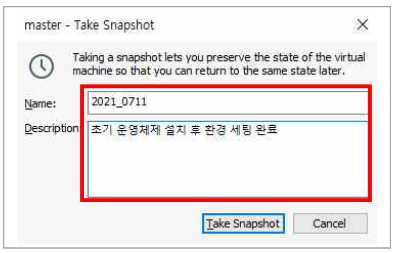
다음 스크립트를 사용하여 자동으로 설정할 수도 있다.

```
# (server1)

# 서버 접속 아이디/패스워드
UNAME=root
PASSWD=centos

# 서버별 색상 지정 - 서버 이름과 색상 코드 매핑
declare -A SERVER_COLOR_MAP=(
  [main]=34
  [server1]=31
  [server2]=32
)

# PS1 템플릿(색상 자리에 %COLOR%를 넣음)
PS1_TEMPLATE='export PS1="W[We[%COLOR%;1mW][Wu@WhW[We[33;1mW] Ww]W$ W[We[mW]"'

# 선수 패키지 설치
dnf install -y sshpass

# 반복 실행
for HOST in "${!SERVER_COLOR_MAP[@]}"
do
  COLOR=${SERVER_COLOR_MAP[$HOST]}
  PS1_STR="${PS1_TEMPLATE//%COLOR%/$COLOR}"
  sshpass -p $PASSWD ssh -o StrictHostKeyChecking=no ${UNAME}@${HOST} W
    "echo '${PS1_STR}' >> $HOME/.bashrc"
done
source ~/.bashrc
```

[실습] 스냅샷(Snapshot) 찍기

모든 시스템 Power OFF
- main/server1/server2 시스템 모두 Power OFF

main/server1/server2 시스템에 대해서 스냅샷을 찍는다.
- VMware > VM > Setting > Snapshot > Take Snapshot > 오늘 날짜 입력(ex: 2021_0329)
- 간단한 설명도 필요하면 달아 놓는다.
  - **"초기 운영체제 설치 후 환경 세팅 완료"**



2 cd CMD (change working directory)

```
NAME
       cd - change the working directory

SYNOPSIS
       cd [-L | -P] [directory]

       cd -

DESCRIPTION
       The cd utility shall change the working directory of the current  shell
       execution  environment  (see Shell Execution Environment ) by executing
       the following steps in sequence. (In the following  steps,  the  symbol
       curpath  represents an intermediate value used to simplify the descrip-
       tion of the algorithm used by cd. There is no requirement that  curpath
       be made visible to the application.)
```

작업 디렉토리를 변경하고자 할 때 사용한다. 인자(Argument)인 디렉토리명은 이동하고자 하는 경로명을 나타낸다. 디렉토리 명을 지정하지 않고 cd 명령어를 단독으로 사용하면 현재 작업디렉토리가 어디에 있든지 사용자의 홈 디렉토리로 이동한다. cd 명령어는 디렉토리를 변경하는 명령어이다. cd 명령어를 사용하여 디렉토리 경로를 변경하는 경우 상대경로(Relative Path)나 절대경로(Absolute Path)를 사용 할 수 있다.

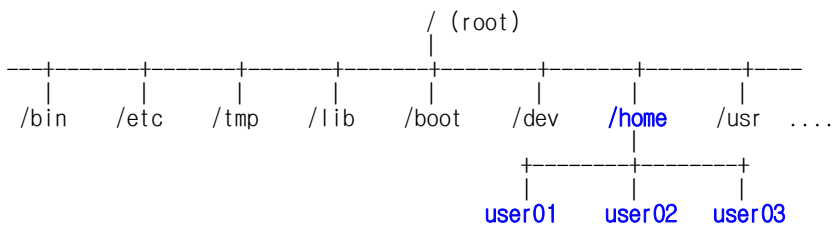
상대경로(Relatvie Path)는 이동하는 기준이 현재 디렉토리이며, 현재 디렉토리를 기준으로 위, 아래로 이동하는 할 때 사용한다. 절대 경로(Absolute Path)는 이동하는 기준이 최상위 디렉토리(/)이며, root(/) 디렉토리를 기준으로 이동할 때 사용한다.

경로(PATH)
- 상대경로(Relative PATH) **# cd dir1**
- 절대경로(Absolute PATH) **# cd /dir1**

작업시에는 상대경로를 사용하여 이동하는 경우, 즉 작업 디렉토리안에 들어가서 직접 파일을 다루는 방법을 권장한다. 이것은 명령어 실수로 인해 불의의 사고를 예방할 수 있거나, 최소화 할 수 있기 때문이다.

[참고] 경로(PATH)에 대해서

경로(PATH)
■ **상대경로** (Relative Path, 현재디렉토리를 기준) (예) **# cd dir1**
■ **절대경로** (Absolute Path, / 디렉토리를 기준) (예) **# cd /dir1**



(1) 상대경로(Relative PATH)를 사용한 디렉토리 이동

상대경로를 이용하여 디렉토리를 이동하는 경우, 디렉토리의 시작 부분에 슬래쉬(/)가 존재하지 않는다. 아래 예제는 상위디렉토리로 이동하는 경우와 하위 디렉토리 dir1으로 이동하는 경우이다.

```
# cd ..      /* 상위 디렉토리로 이동 */
# cd dir1    /* 현재 디렉토리 하위의 dir1 으로 이동 */
```

[참고] 디렉토리 예약어
. : 현재 디렉토리
.. : 상위 디렉토리

[실습] 상대경로를 이용한 디렉토리 이동하기

■ 실습시스템
- server1

■ 작업 시나리오
- 상대경로를 사용하여 디렉토리를 이동하는 연습을 진행한다.

① 현재 디렉토리 확인

현재 디렉토리: /root
이동하고 싶은 디렉토리: /etc/sysconfig/network-scripts

```
[root@server1 ~]# cd /etc
[root@server1 /etc]# pwd
/etc
```

② /etc/sysconfig/network-scripts 디렉토리로 이동

```
[root@server1 /etc]# cd sysconfig
[root@server1 /etc/sysconfig]# cd network-scripts
[root@server1 /etc/sysconfig/network-scripts]# ls -a
.  ..  ifcfg-ens33
```

③ 점(.), 점점(..) 사용해 보기

```
[root@server1 /etc/sysconfig/network-scripts]# cd .
```

[참고] 점(.) 사용되는 예
./script.sh /* 현재 디렉토리의 script.sh 파일을 실행 한다. */

```
[root@server1 /etc/sysconfig/network-scripts]# pwd
/etc/sysconfig/network-scripts/

[root@server1 /etc/sysconfig/network-scripts]# cd ..
[root@server1 /etc/sysconfig]# pwd
/etc/sysconfig/
```

④ 상위디렉토리 이동하기

```
[root@server1 /etc/sysconfig]# cd ../../
[root@server1 /]# pwd
/
```

(2) 절대경로(Asolute PATH)

[참고] 디렉토리 구분자
(WIN) C:₩test₩dir1
(LINUX) /test/dir1

```
/etc/sysconfig/network-scripts
↑  ↑          ↑
❶ ❷         ❸
```

❶는 최상위 디렉토리인 /(root) 디렉토리를 뜻한다.
❷, ❸는 디렉토리의 구분자이다.

다음은 절대경로를 사용하는 다른 예이다.

```
# cd /tmp
# cd /etc/sysconfig
# cd /usr
```

(3) 로그인 된 사용자의 **홈 디렉토리**로 이동

사용자가 로그인하면 기본적으로 자신의 홈디렉토리로 들어 가게 된다. root 사용자로 로그인 하였다면 root의 홈 디렉토리인 /root 디렉토리로 이동 되고, 만약 fedora 사용자로 로그인 하였다면 /home/fedora 사용자로 이동하게 된다.

■ 사용자의 홈디렉토리
* root 사용자 -> /root
* 일반 사용자 -> /home/$USER

■ (root 사용자인 경우)

```
[root@server1 ~]# id
uid=0(root) gid=0(root) groups=0(root)
```

-> root 사용자인 경우

```
[root@server1 ~]# cd /boot
[root@server1 /boot]# cd
[root@server1 ~]# pwd
/root
```

■ (fedora 사용자인 경우)

```
# ssh fedora@localhost     (# ssh fedora@127.0.0.1)
```

fedora 사용자로 로그인

```
[fedora@server1 ~]$ id
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)

[fedora@server1 ~]$ cd /boot
[fedora@server1 boot]$ cd
[fedora@server1 ~]$ pwd
/home/fedora

[fedora@server1 ~]$ exit

[root@server1 ~]# cd ~fedora
[root@server1 /home/fedora]#
```

[참고] ~fedora versus ~/fedora

```
# cd ~fedora      /* fedora 사용자의 홈디렉토리(ex: /home/fedora) */
vs
# cd ~/fedora     /* 나의 홈디렉토리 하위의 fedora 디렉토리(ex: /root/fedora) */
```

(4) 사용한 이전 디렉토리로 이동 하기

■ 작업 시나리오
- 사용자가 /boot 디렉토리에서 작업을 하다가 /etc/sysconfig/network-scripts 디렉토리로 이동하는 경우 상대경로나 절대경로를 사용하는 경우는 불편하게 된다. 상대 경로를 사용하는 경우에는 현재 디렉토리에서 너무 멀리 떨어져 있고, 절대경로를 사용하는 경우에는 적어야 하는 경로의 이름이 길기 때문이다.
- 이런 경우 이전 디렉토리로 바로 이동하기 위해서 "cd - " 명령어를 사용한다.

두 개의 디렉토리를 번갈아 가면서 작업하는 예
/boot <------> /etc/sysconfig/network-scripts

① /boot 디렉토리로 이동
```
[root@server1 ~]# cd /boot
[root@server1 /boot]# pwd
```
```
/boot
```

② /etc/sysconfig/network-scripts 디렉토리로 이동
```
[root@server1 /boot]# cd /etc/sysconfig/network-scripts
[root@server1 /etc/sysconfig/network-scripts]# pwd
```
```
/etc/sysconfig/network-scripts
```

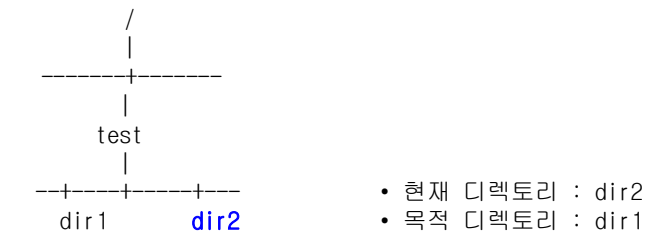
③ 다시 /boot 디렉토리로 이동
```
[root@server1 /etc/sysconfig/network-scripts]# cd /boot
[root@server1 /boot]# pwd
```
```
/boot
```

④ 이전 디렉토리인 /etc/sysconfig/network-scripts 디렉토리로 이동하기
```
[root@server1 /boot]# cd -
```
```
/etc/sysconfig/network-scripts
```
```
[root@server1 /etc/sysconfig/network-scripts]# pwd
```
```
/etc/sysconfig/network-scripts
```

⑤ 이전 디렉토리인 /boot 디렉토리로 이동하기
```
[root@server1 /etc/sysconfig/network-scripts]# cd -
```
```
/boot
```
```
[root@server1 /boot]# pwd
```
```
/boot
```

(5) 옆에 있는 디렉토리로 이동하기

■ 작업 시나리오
- 아래 그림과 같이 /test/dir2 디렉토리에서 /test/dir1 디렉토리로 이동하고 싶은데, 이런 경우 상위 디렉토리로 이동한 후에, 다시 하위 디렉토리인 dir1으로 이동하게 된다.
- 이런경우, 한번에 수행하기 위해서 "cd ../dir1" 형식을 사용할 수 있다.



① 실습 준비 - 작업 디렉토리 생성
```
[root@server1 ~]# mkdir -p /test
[root@server1 ~]# cd /test
```

```
[root@server1 /test]# rm -rf /test/*

[root@server1 /test]# mkdir dir1 dir2
[root@server1 /test]# cd dir2
[root@server1 /test/dir2]# pwd
```

```
/test/dir2
```

② 상위 디렉토리인 /test 디렉토리로 이동

```
[root@server1 /test/dir2]# cd ..
[root@server1 /test]# pwd
```

```
/test
```

③ 하위 디렉토리인 dir1 디렉토리로 이동

```
[root@server1 /test]# cd dir1
[root@server1 /test/dir1]# pwd
```

```
/test/dir1
```

④ 옆에 있는 디렉토리로 이동

```
[root@server1 /test/dir1]# cd ../dir2
[root@server1 /test/dir2]# pwd
```

```
/test/dir2
```

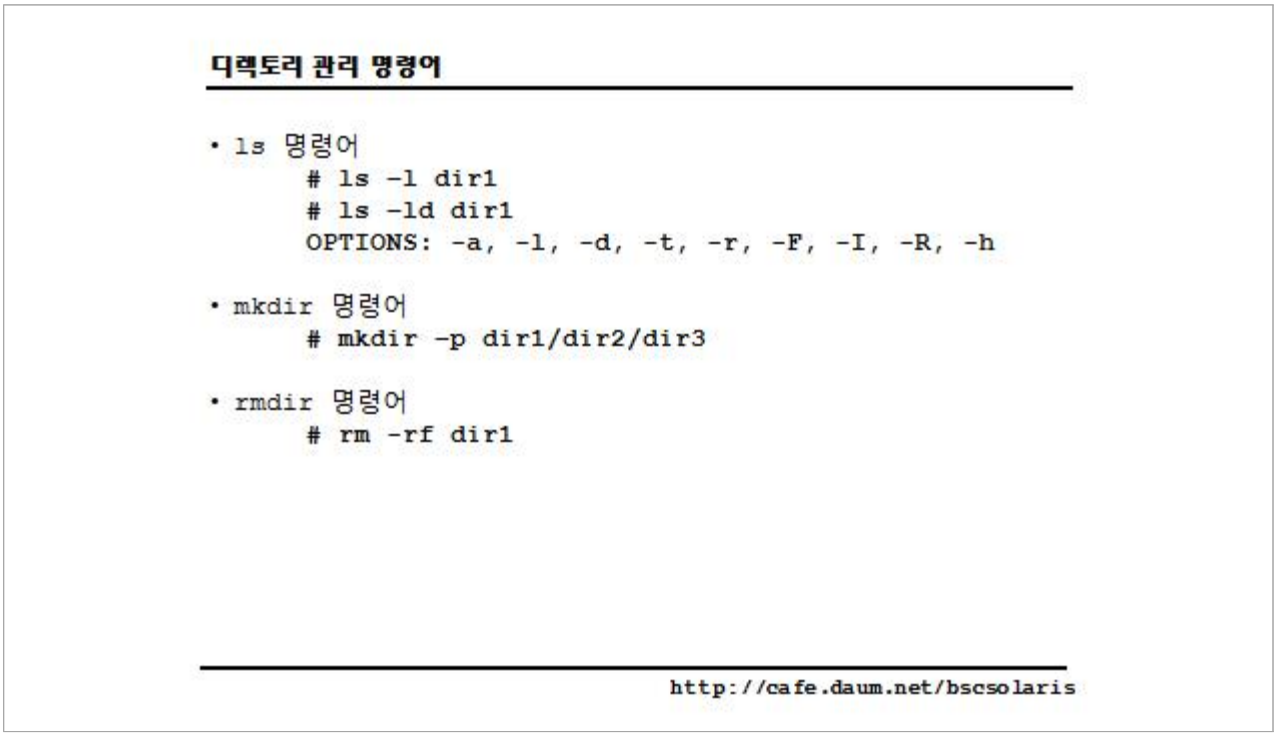
⑤ 다시 옆에 있는 디렉토리로 이동

```
[root@server1 /test/dir2]# cd ../dir1
[root@server1 /test/dir1]# pwd
```

```
/test/dir1
```

[실무 예] Apache 패키지의 디렉토리 구조 이동 방법 - 소스형태로 설치된 경우의 예
소스형태로 설치된(configure/make/make install) Apache2.X 디렉토리 구조가 아래와 같은 경우, 오픈 소스 형태로 설치하면 거의 비슷한 디렉토리가 존재하게 되고, 이름을 유추할 수 있다. 따라서, 옆에 있는 폴더의 이름을 알고 있는 경우에 "cd ../bin" 형식을 사용하여 쉽게 디렉토리를 이동할 수 있다.

```
/usr/local/apache2
        |
        +---- bin   (# cd ../bin)
        |
        +---- conf  (# cd ../conf)
        |
        +---- docs  (# cd ../docs)
        |
        +---- ....
```



1 ls CMD

```
NAME
       ls, dir, vdir - 경로의 내용을 나열한다.

SYNOPSIS
       ls  [-abcdfgiklmnpqrstuxABCFGLNQRSUX1] [-w cols] [-T cols] [-I pattern]
       [--all] [--escape] [--directory]  [--inode]  [--kilobytes]  [--numeric-
       uid-gid]   [--no-group]   [--hide-control-chars]  [--reverse]  [--size]
       [--width=cols]   [--tabsize=cols]   [--almost-all]   [--ignore-backups]
       [--classify] [--file-type] [--full-time] [--ignore=pattern] [--derefer-
       ence]         [--literal]         [--quote-name]          [--recursive]
       [--sort={none,time,size,extension}]        [--format={long,verbose,com-
       mas,across,vertical,single-column}]
       [--time={atime,access,use,ctime,status}]      [--help]      [--version]
       [--color[={yes,no,tty}]] [--colour[={yes,no,tty}]] [name...]

DESCRIPTION
       이 문서는 더이상 최신 정보를 담고 있지않다.  그래서,  몇몇틀릴  경우 도
       있고,  부족한  경우도  있을  것이다.  완전한매뉴얼을 원하면, Texinfo 문
       서를 참조하기 바란다.

       이 매뉴얼 페이지는 ls 명령의 GNU  버전에  대한  것이다.  dir과 vdir  명
       령 은  ls 명령의 심블릭 파일로그 출력 양식을 다르게 보여주는 풀그림들이
       다.  인자로 파일이름이나, 경로 이름이 사용된다.  경로의 내용은 초기값으
       로 알파벳 순으로 나열된다.  ls의 경우는 출력이 표준 출력(터미날 화면)이
       면, 세로로 정열된 것이 가로로 나열된다.  다른 방식의 출력이면 한줄에 하
       나씩 나열된다.  dir의 경우는, 초기값으로 ls와 같으나, 모든 출력에서세로
       로 정열해서 가로로 나열한다.(다른 방식의 출력에서도 항상 같음)  vdir 의
       경우는, 초기값으로 목록을 자세히 나열한다.

   OPTIONS
       -a, --all
              경 로안의 모든 파일을 나열한다.  '.' 으로 시작하는 파일들도 포함된
              다.

       -d, --directory
              경 로안의 내용을 나열하지 않고, 그 경로를 보여준다.(이것은 쉘 스
              크립트에서 유용하게 쓰인다.)

       -i, --inode
              파일 왼쪽에 색인 번호를 보여준다.
```

```
-r, --reverse
       정열 순서를 내림차순으로 한다.

-t, --sort=time
       파일 시간 순으로 정열한다.  최근 파일이 제일 먼저.

-u, --time=atime, --time=access, --time=use
       파일 사용 시간 순으로 정열한다.  자세한 나열이면, 시간 표시는 만
       들어진 날자대신, 사용된 날자를 보여준다.

-F, --classify
       파 일 형식을 알리는 문자를 각 파일 뒤에 추가한다.  일반적으로 실
       행파일은 "*", 경로는 "/", 심블릭링크는 "@", FIFO는 "|", 소 켓 은
       "=", 일반적인파일은 없다.

-R, --recursive
       하위 경로와 그 안에 있는 모든 파일들도 나열한다.

--color, --colour, --color=yes, --colour=yes
       파 일 의  형태에 따라 그 파일의 색깔을 다르게 보여주는 기능한다.
       자세한 이야기는 아래 DISPLAY COLORIZATION 부분을 참조한다.

--color=tty, --colour=tty
       --color 옵션과 같으나, 단지 표준 출력에서만 색깔을 사용한다.  이
       옵 션 은 칼라 제어 코드를 지원하지않는 보기 풀그림을 사용하는 쉘
       스크립트나, 명령행 사용에서아주 유용하게 쓰인다.

--color=no, --colour=no
       색깔 사용하지 않는다. 이것이 초기값이다.  이옵션은 색깔 사 용 을
       이미 하고 있다면, 이 값을 무시한다.
```

디렉토리에 있는 내용을 확인하고자 할 때 (ls 명령에 대해서 확인 : **# man ls**)

[명령어 형식]

```
# ls -l
# ls -ld

# ls -l dir1
# ls -ld dir1

# ls -altr
```

[명령어 옵션]

옵션	설 명
-a	모든 파일 표시, 여기에는 숨김 파일(점(.)으로 시작하는 파일)도 표시한다.
-l	디렉토리가 지정되는 경우 디렉토리의 내용을 자세히 보여준다. 또한 파일의 내용 이 지정되는 경우 파일의 속성 정보를 자세히 보여준다. 파일 종류, 링크 수, 소유자명, 그룹명, 파일 크기, 최종 수정일 및 용량, 파일명 표시한다.
-R	해당 디렉토리와 서브디렉토리의 모든 내용을 표시
-F	디렉토리인 경우에는 디렉토리 “/” 표시를 하고 실행 파일인 경우는 뒤에 “*” 표시를 한다.
-i	해당 파일의 inode 번호를 표시한다.
-n	파일의 소유자와 그룹을 숫자로 표시한다.
-d	찾고자 하는 디렉토리에 관한 정보만을 표시한다.

[실습] ls 명령어에 대한 실습

■ 사용시스템
- server1

■ 작업 시나리오
- ls 명령어의 기본 사용법 테스트
- "ls -a" 옵션 실습
- "ls -a" 옵션 실습
- 'ls -i' 옵션 실습
- 파일 또는 디렉토리만 출력(alias 사용)
- ls -h 옵션 실습
- "ls -altr" 명령어 실습

[EX1] ls 명령어의 기본 사용법 테스트

[참고] ls 명령어의 기본 사용법
```
# ls -l
# ls -ld

# ls -l dir1
# ls -ld dir1
```

(실습용 구조)

```
/test/
|-- dir1/
|   `-- file2
`-- file1
```

① 실습 준비 - 작업용 디렉토리 생성 및 파일 생성
```
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*

[root@server1 /test]# touch file1
[root@server1 /test]# mkdir dir1
[root@server1 /test]# touch dir1/file2
```

② ls -l & ls -ld 차이점 확인
- -l   use a long listing format
- -d, --directory   list directories themselves, not their contents

```
[root@server1 /test]# ls -l
drwxr-xr-x 2 root root 19 Feb  7 02:28 dir1
-rw-r--r-- 1 root root  0 Feb  7 02:28 file1
```

■ "ls -l" 출력 결과 해석
```
-----------------------------------------------------------
-               파일의 종류(File Type), -(일반파일), d(디렉토리파일)
rw-r--r--       퍼미션 모드(Permission Mode)
1               링크 수(Hard Link Count)
root            소유자(Owner)
root            그   룹(Group)
0               파일의 크기(File Size), 기본 단위 : bytes
8월 15 20:37    수정 또는 생성 시간(Mtime: Modify Time)
file1           파일의 이름(File Name)
-----------------------------------------------------------
```

```
[root@server1 /test]# ls -ld          /* 현 디렉토리 정보를 자세히 출력( .디렉토리 출력) */
drwxr-xr-x 3 root root 4096  8월 15 20:37 .
```

③ ls -l dir1 & ls -ld dir1 차이점 확인

```
[root@server1 /test]# ls -l dir1        /* dir1디렉토리의 내용을 출력 */
-rw-r--r-- 1 root root 0  8월 15 20:38 file2

[root@server1 /test]# ls -ld dir1       /* dir1디렉토리의 정보를 출력 */
drwxr-xr-x 2 root root 4096  8월 15 20:38 dir1
```

④ ls -R 옵션 사용

- -R, --recursive list subdirectories recursively

```
[root@server1 /test]# ls -lR            /* -R: Recursive, 하위 디렉토리까지 */
.:
total 0
drwxr-xr-x 2 root root 19 Feb  7 02:28 dir1
-rw-r--r-- 1 root root  0 Feb  7 02:28 file1

./dir1:
total 0
-rw-r--r-- 1 root root 0 Feb  7 02:28 file2
```

- [참고] 현재 디렉토리 하위의 모든 파일 보기

```
# find .        # 단순한 목록 확인하기
# tree          # 디렉토리 트리 목록으로 확인하기
```

[EX2] "ls -a" 옵션 실습

- -a, --all do not ignore entries starting with .

① ls -l & ls -al 차이점 확인

```
[root@server1 /test]# cd
[root@server1 ~]# pwd
/root

[root@server1 ~]# ls -l
drwxr-xr-x 2 root root  4096  4월  6 12:17 Desktop
-rw------- 1 root root  4210  1월 25 22:54 anaconda-ks.cfg
-rw-r--r-- 1 root root     0  4월  7 12:38 file_0407.log
-rw-r--r-- 1 root root 54631  1월 25 22:54 install.log
-rw-r--r-- 1 root root  9641  1월 25 22:50 install.log.syslog

[root@server1 ~]# ls -al
total 52
drwxr-xr-x.  2 root root   80 Feb  6 03:04 바탕화면
dr-xr-x---. 17 root root 4096 Feb  7 02:24 .
dr-xr-xr-x. 18 root root  256 Feb  7 01:44 ..
-rw-------.  1 root root 1363 Feb  6 02:36 anaconda-ks.cfg
-rw-------.  1 root root 3106 Feb  7 02:24 .bash_history
-rw-r--r--.  1 root root   18 May 12  2019 .bash_logout
-rw-r--r--.  1 root root  176 May 12  2019 .bash_profile
-rw-r--r--   1 root root  387 Feb  7 00:33 .bashrc
drwx------. 12 root root  276 Feb  7 01:26 .cache
drwx------. 17 root root 4096 Feb  7 01:26 .config
-rw-r--r--.  1 root root  100 May 12  2019 .cshrc
drwx------.  3 root root   25 Feb  6 02:38 .dbus
drwxr-xr-x   2 root root    6 Feb  6 23:03 Desktop
drwxr-xr-x   2 root root    6 Feb  6 23:03 Documents
drwxr-xr-x   2 root root    6 Feb  6 23:03 Downloads
-rw-------.  1 root root   16 Feb  6 02:39 .esd_auth
-rw-------.  1 root root 2170 Feb  7 01:43 .ICEauthority
-rw-r--r--.  1 root root 1738 Feb  6 02:38 initial-setup-ks.cfg
-rw-------   1 root root   42 Feb  7 02:24 .lesshst
drwx------.  3 root root   19 Feb  6 02:39 .local
drwxr-xr-x   2 root root    6 Feb  6 23:03 Music
drwxr-xr-x   2 root root    6 Feb  6 23:03 Pictures
drwxr-----.  3 root root   19 Feb  6 02:39 .pki
drwxr-xr-x   2 root root    6 Feb  6 23:03 Public
drwx------   2 root root   25 Feb  7 01:59 .ssh
-rw-r--r--.  1 root root  129 May 12  2019 .tcshrc
drwxr-xr-x   2 root root    6 Feb  6 23:03 Templates
drwxr-xr-x   2 root root    6 Feb  6 23:03 Videos
```

[EX3] "ls -F" 옵션 실습
- -F, --classify append indicator (one of */=>@|) to entries

① 실습용 디렉토리 및 파일 생성 - 여러가지 종류의 파일 생성
```
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*
```

다음 내용을 복사해서 사용한다.
```
# (server1)
mkdir -p /test && cd /test && rm -rf /test/*

cp /etc/passwd file1      # 일반 파일
ln -s file1 file2         # 링크 파일
cp /bin/ls file3          # 실행 파일
mkdir dir1                # 디렉토리 파일
```

② ls -F & ls 차이점 확인
```
[root@server1 /test]# ls -F
```
```
dir1/  file1  file2@  file3*
```
* 디렉토리 => /
* 일반파일 => 없음
* 링크파일 => @
* 실행파일 => *

```
[root@server1 /test]# ls
```
```
dir1  file1  file2  file3
```

[EX4] 'ls -i' 옵션 실습
- -i, --inode print the index number of each file

[참고] "inode" vs "inode number"
* inode(index node)? 파일의 속성 정보가 들어가는 공간(ex: stat file1)
* inode number? 파일의 속성 정보를 구분 번호(식별 번호)

① 파일/디렉토리의 inode number 확인
```
[root@server1 /test]# ls -li /test/file1        /* inode(Index Node) 번호까지 확인 */
```
```
100666903 -rw-r--r-- 1 root root 2750 Feb  7 09:14 /test/file1
```

```
[root@server1 /test]# ls -ldi                   /* inode는 각각 고유한 값을 지님 */
```
```
100666900 drwxr-xr-x. 3 root root 57 Feb  7 09:15 .
```

[EX5] 파일 또는 디렉토리만 출력
- 일반적으로 리눅스에서 명령어를 만들어서 사용하기도 한다.

■ 별칭(alias)
(선언) # alias ls='ls -h --color=tty'
(확인) # alias
(해제) # unalias ls

① alias 선언
```
[root@server1 ~]# alias lsf='ls -l | grep "^-"'  /* 파일인 경우 속성 정보에 -로 표시 */
[root@server1 ~]# alias lsd='ls -l | grep "^d"'  /* 디렉토리인 경우 속성 정보에 d로 표시 */
```

② lsf & lsd 엘리어스 실행
```
[root@server1 ~]# cd
[root@server1 ~]# lsf
```
```
-rw-------. 1 root root 1363 Feb  6 02:36 anaconda-ks.cfg
```

```
-rw-r--r--. 1 root root 1738 Feb  6 02:38 initial-setup-ks.cfg
```

[root@server1 ~]# lsd

```
drwxr-xr-x. 2 root root   80 Feb  6 03:04 바탕화면
drwxr-xr-x  2 root root    6 Feb  6 23:03 Desktop
drwxr-xr-x  2 root root    6 Feb  6 23:03 Documents
drwxr-xr-x  2 root root    6 Feb  6 23:03 Downloads
drwxr-xr-x  2 root root    6 Feb  6 23:03 Music
drwxr-xr-x  2 root root    6 Feb  6 23:03 Pictures
drwxr-xr-x  2 root root    6 Feb  6 23:03 Public
drwxr-xr-x  2 root root    6 Feb  6 23:03 Templates
drwxr-xr-x  2 root root    6 Feb  6 23:03 Videos
```

(참고) 선언된 alias 확인
 # alias
 # alias lsf

③ 선언된 alias 확인
[root@server1 ~]# alias lsf /* lsf의 alias의 정보 확인 */

```
alias lsf='ls -l | grep "^-"'
```

[root@server1 ~]# alias lsd /* lsd의 alias의 정보 확인 */

```
alias lsd='ls -l | grep "^d"'
```

(참고) 선언된 alias 확인
alias ls
alias ll

[EX6] ls -h 옵션 설정(-h : human)
- -h, --human-readable with -l and -s, print sizes like 1K 234M 2G etc.

① ls 명령어의 -h 옵션이 없는 경우에 대한 확인
[root@server1 ~]# ls -l /etc/services

```
-rw-r--r--. 1 root root 692252 May 15  2020 /etc/services
```

② 환경 파일에 alias 선언
[root@server1 ~]# gedit ~/.bashrc

```
..... (중략) .....
#
# Sfecific Configuration
#
alias ls='ls --color=auto -h'
```

③ 환경 파일에 변경 사항 적용 및 확인
[root@server1 ~]# . ~/.bashrc (# source ~/.bashrc)
[root@server1 ~]# ls -l /etc/services (# ls -l --color=tty -h /etc/services)

```
-rw-r--r--. 1 root root 677K May 15  2020 /etc/services
```

[EX7] "ls -altr" 명령어 실습
- -t sort by modification time, newest first
- -r, --reverse reverse order while sorting

[root@server1 ~]# ls -altr /tmp /* -t : time sort, -r : reverse sort */

```
.... (중략) ....
drwxrwxrwt.  2 root   root     39 Feb  7 01:43 .X11-unix
drwx------.  2 root   root     20 Feb  7 01:43 .esd-0
dr-xr-xr-x. 18 root   root    256 Feb  7 01:44 ..
drwx------   2 fedora fedora    6 Feb  7 02:01 .esd-1000
drwxrwxrwt. 22 root   root   4.0K Feb  7 09:22 .
```

[참고] 로그 디렉토리의 최근 수정된 로그 파일 확인시
 # cd /Log_dir
 # ls -altr

2 mkdir CMD(Make directory)

```
NAME
       mkdir - 경로 만들기

SYNOPSIS
       mkdir  [-p]  [-m  mode]  [--parents] [--mode=mode] [--help] [--version]
       dir...

DESCRIPTION
       이 문서는 더이상 최신 정보를 담고 있지않다.  그래서,  몇몇틀릴  경우 도
       있고,  부족한  경우도  있을  것이다.  완전한매뉴얼을 원하면, Texinfo 문
       서를 참조하기 바란다.

       이 매뉴얼 페이지는 mkdir 명령의 GNU  버전에  대한  것이다.  mkdir 명 령
       은 주워진 이름으로 경로를 만든다.  초기값으로는 그 만들어지는 경로의 모
       드는 07770이다.

   OPTIONS
       -m, --mode mode
              mode로 사용할 것은 chmod(1)에서 사용하는 기호형식이나, 숫자형 식
              이며, 이 값은 초기값으로 지정되는 모드를 무시한다.

       -p, --parents
              필 요 한   경 우 에   상위 경로까지 만든다.  이 말은 가령  'mkdir
              ~/dest/dir1'  명령을 사용했을 때, 만약  '~/dest'  경로가 없다면  오
              료 메시지를 보인다.  하지만, 이 옵션을 사용하면,  '~/dest'  경로를
              만들고, 그 다음, 그 안에서  'dir1'  경로를 만든다. 만들어 지는  상
              위경로의 모드값은  'u+wx' 이다.
```

새로운 디렉토리를 생성하며, 빈 디렉토리를 생성한다. 옵션을 통하여 여러개의 디렉토리를 한꺼번에 생성 할 수도 있다.

[명령어 형식]

```
# mkdir dir1                    /* 현 디렉토리에 dir1 디렉토리 1개 생성 */
# mkdir dir1 dir2               /* 현 디렉토리에 dir1, dir2 디렉토리 2개 생성 */
# mkdir -p dir3/dir2/dir1       /* dir3 디렉토리 안에 dir2를 생성하고 dir2 안에 dir1을 생성 */
# mkdir -m 755 dir1
```

[명령어 옵션]

옵션	설 명
-m	디렉토리의 퍼미션 권한을 지정 (기본값 : 755)
-p	디렉토리 경로로 생성 (디렉토리를 만들 때 상위 디렉토리가 없을시 상위 디렉토리까지 생성)

[실습] 디렉토리 생성

■ 사용시스템
- server1

■ 실습시나리오
- mkdir -p 옵션에 대해 실습해 본다.

[EX1] mkdir -p 옵션 사용법

① 실습 준비 작업

```
[root@server1 ~]# mkdir -p /test
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*
[root@server1 /test]# pwd
/test
```

② dir4/dir2/dir1 디렉토리 생성 - 실패 메세지 확인

```
[root@server1 /test]# mkdir dir4
[root@server1 /test]# ls -l
drwxr-xr-x 2 root root 6 Feb  7 09:28 dir4

[root@server1 /test]# mkdir dir4/dir2/dir1
mkdir: cannot create directory `dir4/dir2/dir1': No such file or directory
```

- dir4/dir2 디렉토리가 없기 때문에 dir4/dir2/dir1 디렉토리가 생성될수 없다.

③ mkdir -p 옵션 사용

```
[root@server1 /test]# mkdir -p dir4/dir2/dir1
[root@server1 /test]# find .
.
./dir4
./dir4/dir2
./dir4/dir2/dir1
```

3 rmdir CMD(Remove Directory)

```
NAME
       rmdir - 비어 있는 경로를 지운다.

SYNOPSIS
       rmdir [-p] [--parents] [--help] [--version] dir...

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

       이   매뉴얼 페이지는 rmdir 명령의 GNU 버전에 대한 것이다.  rmdir 명령은
       비어있는 경로른 지운다.  아무 옵션 없이 사용되면 오류 번호를 돌려주 고,
       종료된다.

   OPTIONS
       -p, --parents
              상위 경로도 지운다. 이명령은 그 상위 경로 안에 물론 비어있어야지
              가능하다.

       만약 비어 있지 않으나, 그 경로와 그 안에 포함된 모든 파일과 하위 경로들
       을모조리 지우고자 한다면 rm -r 옵션을 사용한다.
```

빈 디렉토리를 삭제하는 명령어로, 여러 개의 디렉토리를 동시에 삭제 할 수 있다. 단, 디렉토리가 비어 있지 않으면 삭제 불가능 하다.

[명령어 형식]

```
# rmdir dir1                  /* dir1 디렉토리 1개 삭제 */
# rmdir dir1 dir2             /* dir1, dir2 디렉토리 2개 삭제 */
# rmdir -p dir4/dir2/dir1     /* 경로에 포함되어 있는 디렉토리 삭제 (비워있어야 함) */
```

[명령어 옵션]

옵션	설명
-p	하위항목을 같이 지움 (조건 : 하위항목도 비워 있어야 함)

[참고] 비어 있지 않는 디렉토리 삭제

```
# rm -rf dir1       /* -r : recursive, -f : force */
```

- (주의) 파일/디렉토리 삭제 주의해야 한다.
- (예) **rm -rf /test/***
- (예) rm -rf /test

파일 관리 명령어

- touch 명령어
  ```
  # touch -t 08301300 file1
  ```
- cp 명령어
  ```
  # cp file1 file2
  # cp file1 dir1
  # cp -r dir1 dir2
  OPTIONS: -a, -p, -r
  ```
- mv 명령어
  ```
  # mv file1 file2
  # mv file1 dir1
  # mv dir1 dir2
  ```
- rm 명령어
  ```
  # rm -rf dir1
  ```

http://cafe.daum.net/bscsolaris

1 touch CMD

```
NAME
       touch - 파일의 시간 정보를 바꾼다.

SYNOPSIS
       touch  [-acfm]  [-r  file]  [-t  MMDDhhmm[[CC]YY][.ss]]  [-d  time]
       [--time={atime,access,use,mtime,modify}]    [--date=time]    [--refer-
       ence=file] [--no-create] [--help] [--version] file...

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

       이  매뉴얼 페이지는 touch 명령의 GNU 버전에 대한 것이다.  touch 명령은
       주어진 파일의 최근 사용시간과 최근 변경 시간(파일 내용이 바뀐 시간)을시
       스 템 의 현재 시간으로 바꾼다.  이때 그런 파일이 없으면 0 바이트 크기의
       이름만 있는 파일을 만든다.  만약, 첫번째 오는 filename에 해서 -t 옵션이
       사 용되고 나머지 filename에 대해서 특별히 -d, -r, -t 옵션이 사용되고 있
       지 않다면, 나머지 모든 파일들은그 첫번째 파일에서 사용할 시간값을 그 대
       로 사용하게 된다.  이런 것을 방지하려면, -- 옵션을 사용해서 각각의 옵션
       들을 분리 시킨다.

       If  changing both the access and  modification  times  to  the  current
       time, touch can change the  timestamps  for  files that  the  user run-
       ning it does not own but has write per  mission  for.  Otherwise,  the
       user must own the files.

   OPTIONS
       -d, --date time
              현재 시간 대신 지정한 time 갑을 사용하는데, 이것은 다양한 형식이
              될 수 있다.  이 시간에는 월 이름, 지역, 'am', 'pm' 등이 포 함 될
              수도 있다.

       -m, --time=mtime, --time=modify
              최근 파일 변경 시간(modify time)만 바꾼다.

       -t MMDDhhmm[[CC]YY][.ss]
              현 재  시간 대신 지정한 시간(MM:달, DD:날, hh:시, mm:분, [CC]YY:
              년, SS:초)으로바꾼다.
```

파일의 이름을 지정하여 기존에 존재하지 않는 파일이름을 지정하였다면 빈 파일을 만들어주고 기존에 존재했다면 지정된 파일이나 디렉토리의 수정시간(mtime, Modify Time)이나 접근시간(atime, Access Time)등을 현재 시간으로 업데이트 시켜준다.

만약 touch 명령어에 -t 옵션을 사용 하여 파일이나 디렉토리의 수정시간을 특정한 시간으로 변경 가능하다.
(해커가 침입하여 파일을 수정하고 아래의 옵션을 통해 수정과 접근 시간을 바꿔놓을 수도 있다.)

[명령어 형식]

```
# touch file2                  /* file2 파일 1개 생성 */
# touch file1 file2            /* file1, file2 파일 2개 생성 */
# touch -t 08081230 file1      /* file1 수정 시간 변경(월,일,시,분) */
```

[명령어 옵션]

옵션	설 명
-a	최근 파일 사용기간 만 변경
-c	파일을 생성하지 않는 명령어
-d [시간]	현재 시간 대신 지정한 시간(시분)으로 변경
-m	최근 파일 변경 시간만 변경 (파일 수정시간)
-r [파일]	현재 시간 대신 지정한 파일의 시간으로 변경
-t MMDDhhmm [[CC]YY][.SS]	현재 시간 대신 지정한 시간(월일시분)으로 변경

[실습] touch 명령어 실습

■ 사용시스템
- server1

■ 실습 시나리오
- 빈 파일 생성 하기
- 파일의 생성을 시간 현재시간으로 변경
- touch -t 옵션 사용

[EX1] 빈 파일 생성 하기
- touch 명령어를 사용하여 빈파일(크기가 0인 파일)을 생성할 수 있다.

```
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*

[root@server1 /test]# touch file1
[root@server1 /test]# ls -l file1
-rw-r--r-- 1 root root 0 Feb  7 09:32 file1
```

-> 파일의 크기 0

[EX2] 파일의 생성 시간을 현재시간으로 변경
- 생성되어진 파일에 대해서 touch 명령어를 수행하면 미리 생성된 파일의 시간(mtime)이 현재 시간으로 변경된다.

```
[root@server1 /test]# cp -p /etc/passwd file2
[root@server1 /test]# ls -l file2
-rw-r--r-- 1 root root 2.7K Feb  7 01:37 file2
```
-> 시간 정보를 확인한다.

```
[root@server1 /test]# touch file2
[root@server1 /test]# ls -l file2
-rw-r--r-- 1 root root 2.7K Feb  7 09:33 file2
```
-> 시간 정보를 확인한다.

```
[root@server1 /test]# date
Sun Feb  7 09:33:46 KST 2021
```
-> 현재 시간을 확인한다.

[EX3] touch -t 옵션 사용
- touch -t 옵션을 사용하여 미리 생성된 파일의 시간(mtime)을 임의의 시간으로 변경할 수 있다.

```
[root@server1 /test]# touch -t 08301300 file2
[root@server1 /test]# ls -l file2
-rw-r--r-- 1 root root 2.7K Aug 30  2021 file2
```

(주의) 파일의 시간(mtime)은 항상 정확한 것은 아니다.

2 cp CMD

```
NAME
       cp - 파일 복사

SYNOPSIS
       cp [options] source dest
       cp [options] source... directory

DESCRIPTION
       이 문서는 더이상 최신 정보를 담고 있지않다.  그래서,  몇몇틀릴  경우 도
       있고,  부족한  경우도  있을  것이다.  완전한매뉴얼을 원하면, Texinfo 문
       서를 참조하기 바란다.

       이 매뉴얼 페이지는 cp 명령의 GNU  버전에  대한  것이다.  마지막 명령 행
       인자로 경로가 지정되면, cp 명령은 지정한 source 파일들을 그 경로로 안으
       로 복사한다.  한편 명령행 인자로 두개의 파일 이름이 사용되면, 첫번째 파
       일 을 두번째 파일로 복사한다.  마지막 명령행 인자가 경로가 아니고, 두개
       이상의 파일이 지정되면, 오류 메시지를 보여준다. 초기값으로 경로는 복 사
       하지 않는다.

   OPTIONS
       -a, --archive
              원 본 파일의 속성, 링크 정보들을 그대로 유지하면서 복사한다.  이
              옵션은 -dpR 옵션과 같은 역활을 한다.

       -i, --interactive
              만약 복사 대상 파일 이미 있으면 사용자에게 어떻게 처리 할 것인지
              물어보는프롬프트를 나타나게 한다.

       -p, --preserve
              원본 파일의 소유주, 그룹, 권한, 시간정보들이 그대로 보존되어  복
              사된다.

       -r     일반 파일이면, 그냥 복사되고, 만약 원본이 경로면, 그 경로와 함께
              경로 안에 있는 모든 하위경로, 파일들이 복사된다.

       -R, --recursive
              경로를 복사할 경우에는 그 안에 포함된 모든 하위경로와 파일들을모
              두 복사한다.
```

파일이나 디렉토리의 내용을 다른 파일 또는 다른 디렉토리에 복사 할 때 사용. 파일을 복사하는 것은 물리적으로 새로운 파일을 하나 생성하며 새로운 파일의 이름과 새로운 inode, 복사된 데이터 블록을 가지게 된다.

[명령어 형식]

```
# cp file1 file2          /* file1 파일내용을 file2로 생성 */
# cp file1 dir1           /* file1 파일내용을 dir1 디렉토리에 file1 생성 */
# cp -r dir1 dir2         /* dir1 디렉토리를 dir2디렉토리로 생성 */
    * dir2 존재하지 않는 경우 => dir2 생성
    * dir2 존재하는 경우 => dir2/dir1 생성
```

[명령어 옵션]

옵션	설 명
-a	원본 파일의 속성, 링크 정보를 유지 하면서 복사
-b	복사할 대상을 덮어쓰거나 지울 때를 대비해서 백업 파일 만듦
-d	심볼릭 파일 자체를 심볼릭 정보와 함께 복사할 때 사용
-f	복사할 파일이 존재할 때 삭제하고 복사
-i	복사할 파일이 존재하는 경우 복사할 것인지 물어봄
-l	디렉토리가 아닌 경우 복사 대신 하드 링크로 만듦
-p	원본 파일의 소유, 그룹, 권한, 허용 시간을 보존한 채로 복사
-r	서브 디렉토리 내에 있는 모든 파일까지 통째로 복사
-s	디렉토리가 아닌 경우 복사 대신 심볼릭 링크로 만듦
-u	대상 파일보다 원본 파일이 새로운 것일 때 복사
-v	복사 상태를 보여줌

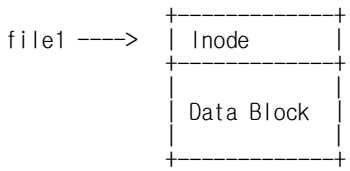
[실습] cp 명령어에 대한 실습

■ 사용시스템
- server1

■ 실습 시나리오
- 파일에 대한 inode를 확인
- cp 명령어의 -r 옵션 사용법 확인
- cp를 이용하여 같은 파일에 덮어 쓰기(Overwrite)하는 경우
- 원본 파일의 소유(X), 그룹(X), 권한(O), 허용 시간(O)을 보존한 채로 복사

[EX1] 파일에 대한 inode를 확인(inode = index node)

■ 파일의 일반적인 구조(EX: 일반 파일)



■ Inode Number ?

① 실습용 구조 만들기
- 실습 할수 있는 구조를 만든다.

```
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*
[root@server1 /test]# touch file1
[root@server1 /test]# ls -li file1
```

```
100666903 -rw-r--r-- 1 root root 0 Feb  7 09:38 file1
```

② "cp file1 file2" 형식 테스트

```
[root@server1 /test]# cp file1 file2
[root@server1 /test]# ls -li file2
```

```
100667008 -rw-r--r-- 1 root root 0 Feb  7 09:39 file2
```

- inode가 틀리다는 것은 곧 다른 파일이라는 뜻이다.

③ "cp file1 dir1" 형식 테스트
```
[root@server1 /test]# mkdir dir1
[root@server1 /test]# cp file1 dir1
[root@server1 /test]# ls -l dir1
```
```
-rw-r--r-- 1 root root 0 Feb  7 09:40 file1
```

[EX2] cp 명령어의 -r 옵션(cp -r dir1 dir2)

① 실습 준비
```
[root@server1 ~]# cd /test
[root@server1 /test]# rm -rf /test/*
[root@server1 /test]# mkdir dir1
[root@server1 /test]# touch dir1/file{1,2}     (# touch dir1/file1 dir1/file2)
```

② cp -r dir1 dir2 형식 테스트
```
[root@server1 /test]# cp -r dir1 dir2
[root@server1 /test]# tree
```
```
.
|-- dir1
|   |-- file1
|   `-- file2
`-- dir2
    |-- file1
    `-- file2

2 directories, 4 files
```

③ dir2 디렉토리가 존재하는 경우의 cp -r dir1 dir2 형식 테스트
```
[root@server1 /test]# cp -r dir1 dir2
[root@server1 /test]# tree
```
```
.
|-- dir1
|   |-- file1
|   `-- file2
`-- dir2
    |-- dir1
    |   |-- file1
    |   `-- file2
    |-- file1
    `-- file2

3 directories, 6 files
```

[참고] 목적지 디렉토리가 존재하는 경우의 디렉토리 복사
```
# cp -r /home/fedora /tmp      (-> /tmp/fedora)
# cp -r /test /tmp             (-> /tmp/test)
```

[EX3] cp를 이용하여 같은 파일에 덮어 쓰기(Overwrite)하는 경우
- 미리 존재하는 파일에 cp 명령어를 사용하여 복사를 진행하면 덮어쓰기 할것인지를 물어보게 된다. RedHat 계열의 OS에서는 미리 몇가지 alias가 $HOME/.bashrc 파일에 설정이 되어 있기 때문이다.

① 실습 준비 작업
```
[root@server1 /test]# cd /test
[root@server1 /test]# rm -rf /test/*
[root@server1 /test]# mkdir dir1
[root@server1 /test]# echo 'linux200' > file1
[root@server1 /test]# cat file1
```
```
linux200
```
```
[root@server1 /test]# touch dir1/file1      /* 동일한 파일명으로 파일 생성 */
[root@server1 /test]# cat dir1/file1        /* 이름만 동일 할 뿐 내용 다름 */
[root@server1 /test]#
```

② dir1/file1 파일이 존재하는 경우 cp file1 dir1 형식 테스트 및 확인

```
[root@server1 /test]# cp file1 dir1          /* # cp file1 dir1/file1, 동일한 파일명에 덮어쓰기 */
cp: overwrite `dir1/file1'? y
```

```
[root@server1 /test]# ls -l file1 dir1/file1
-rw-r--r-- 1 root root 9 Feb  7 09:44 dir1/file1
-rw-r--r-- 1 root root 9 Feb  7 09:43 file1
```

```
[root@server1 /test]# cat dir1/file1         /* copy 내용 확인 가능 */
linux200
```

```
# alias cp
# alias mv
# alias rm
# cat ~/.bashrc
```

[EX4] 원본 파일의 소유(X), 그룹(X), 권한(O), 허용 시간(O)을 보존한 채로 복사(cp -p)

- 원본 파일의 속성정보를 유지(preserve)하면서 복사하기 위해서는 cp -p 명령어 형식을 사용한다.

① 실습 준비 작업

```
[root@server1 /test]# cd /test
[root@server1 /test]# rm -rf /test/*
[root@server1 /test]# touch file1
[root@server1 /test]# chmod 777 file1
[root@server1 /test]# ls -l
-rwxrwxrwx 1 root root 0 Feb  7 09:46 file1
```

② cp & cp -p 형식 차이점 테스트

```
[root@server1 /test]# cp file1 file2
[root@server1 /test]# ls -l
-rwxrwxrwx 1 root root 0 Feb  7 09:46 file1
-rwxr-xr-x 1 root root 0 Feb  7 09:46 file2
```

```
[root@server1 /test]# cp -p file1 file3      /* -p: preserve */
[root@server1 /test]# ls -l                  /* 퍼미션 값과 생성 시간등을 그대로 이어 받아오고 있다 */
-rwxrwxrwx 1 root root 0 Feb  7 09:46 file1
-rwxr-xr-x 1 root root 0 Feb  7 09:46 file2
-rwxrwxrwx 1 root root 0 Feb  7 09:46 file3
```

③ /test 디렉토리 퍼미션 변경

```
[root@server1 /test]# chmod 777 /test        /* 모든 사용자가 파일을 생성 할 수 있도록 퍼미션 변경 */
```

④ fedora 사용자로 전환(Switching)

```
[root@server1 /test]# su - fedora           /* fedora 사용자로 전환 */
[fedora@server1 ~]$ id
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)
```

```
[fedora@server1 ~]$ pwd
/home/fedora
```

⑤ 일반 사용자로 cp & cp -p 차이점 테스트

```
[fedora@server1 ~]$ cd /test
[fedora@server1 test]$ cp file1 file4
[fedora@server1 test]$ ls -l
-rwxrwxrwx 1 root   root   0 Feb  7 09:46 file1
-rwxr-xr-x 1 root   root   0 Feb  7 09:46 file2
-rwxrwxrwx 1 root   root   0 Feb  7 09:46 file3
-rwxrwxr-x 1 fedora fedora 0 Feb  7 09:49 file4
```

```
[fedora@server1 test]$ cp -p file1 file5
[fedora@server1 test]$ ls -l
-rwxrwxrwx 1 root   root   0 Feb  7 09:46 file1
-rwxr-xr-x 1 root   root   0 Feb  7 09:46 file2
-rwxrwxrwx 1 root   root   0 Feb  7 09:46 file3
-rwxrwxr-x 1 fedora fedora 0 Feb  7 09:49 file4
-rwxrwxrwx 1 fedora fedora 0 Feb  7 09:46 file5

[fedora@server1 test]$ exit
```

(실무 예) **설정 파일(예: httpd.conf)**을 백업 받고 설정하는 경우

설정 파일을 백업 받는 경우: cp -p

```
# dnf -y install httpd mod_ssl      /* httpd: (HTTP) Apache Web Server, mod_ssl: (HTTPS) Module */
# cd /etc/httpd/conf
# cp -p httpd.conf httpd.conf.OLD
# /bin/cp -p httpd.conf.OLD httpd.conf
```

웹 소스 디렉토리를 백업 받는 경우: cp -a(-rp) => rsync -a --delete html/ html.OLD

```
# cd /var/www
# cp -a html html.OLD  (# cp -rp dir1 dir2)
```

(실무 예) 로그 파일(EX: file.log) 비우기

로그 파일의 내용을 비우는 여러가지 방법

```
# cp /dev/null file.log
# cat /dev/null > file.log
# > file.log
```

로그 파일을 비우는 간단한 실습

```
# cd /test
# cp -p /var/log/messages file.log
# cp -f /dev/null file.log
# ls -l file.log
```

-> 파일의 크기 '0' 확인

3 mv CMD

```
NAME
       mv - 파일 옮기기(rename)

SYNOPSIS
       mv [options] source dest
       mv [options] source... directory
       Options:
       [-bfiuv]  [-S backup-suffix] [-V {numbered,existing,simple}] [--backup]
       [--force] [--interactive] [--update] [--verbose]  [--suffix=backup-suf-
       fix]  [--version-control={numbered,existing,simple}]  [--help]  [--ver-
       sion]

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 mv 명령의 GNU 버전에 대한 것이다.  마지막 인자로  경
       로   이름이 사용되면, 원본 파일을 그 경로로 이름 똑같게 이동시킨다.  반
       면, 마지막 인자가 파일이름이면, 그 이름으로 바꾼다.  마지막 인자가 경로
       이 름이 아니거나, 원본과 대상이 파일 이름이아닌 경우는 오류 메시지를 보
       인다.  mv 명령은 파일 시스템에 접근이 가능할 때만 사용될 수 있다.  ( 일
       반 사용자는 DOS 파티션의 파일 시스템과 자신의 홈 경로 외에는읽기 접근을
       뺀 나머지는 불가능하다. 이런데, 바로 이런 파일시스템으로 파일을 옮기 고
       자 한다면, mv 명령은 오류 메시지를 낸다.)

       만 약에 파일 모드가 읽기 전용이고, 표준 입력이 tty이고, -f나 --force 옵
       션이 지정되지 않으면, mv 명령은 사용자에게 지정한 파일을 정말 지울것 인
       지 물어본다.  이때,  'y' 나  'Y' 를 입력해 주어야지만 그 파일을 옮긴다.

   OPTIONS
       -f, --force
              대상 파일이 이미 있어도 사용자에게 어떻게 처리할지를 묻지  않 는
              다.

       -i, --interactive
              대 상  파일이 이미 있어, 사용자에게 어떻게 처리할지를 물어 본다.
              이때,  'y' 나  'Y' 를 입력해 주어야지만 그 파일을 옮긴다.  (초기값)
```

파일과 디렉토리의 내용을 다른 파일 또는 다른 디렉토리로 옮길 때 사용하며 파일의 이름이나 디렉토리의 이름을 바꿀 수 있음. 같은 파티션 안에서 파일을 옮긴다는 것은 물리적으로 파일 이름만 변경하며, Inode 정보나 데이터 블록은 그대로 유지가 되고 다른 파티션으로 파일을 옮기는 경우는 새로운 파일 이름과 Inode, 데이터 블록을 할당받게 됨.

[명령어 형식]

```
# mv file1 file2          /* file1 파일이 이름이 file2로 변함 */
# mv file1 dir1           /* file1 파일이 dir1 디렉토리에 하위경로로 이동 */
# mv dir1 dir2            /* dir1 디렉토리가 dir2 디렉토리에 하위경로로 이동 */
    * dir2 존재하지 않는 경우 => dir2
    * dir2 존재하는 경우 => dir2/dir1
```

[명령어 옵션]

옵션	설 명
-b	복사할 대상을 덮어쓰거나 지울 때를 대비해서 백업 파일 만듦
-f	복사할 파일이 존재할 때 삭제하고 복사
-i	복사할 파일이 존재하는 경우 복사할 것인지 물어봄
-u	대상 파일보다 원본 파일이 새로운 것일 때 복사
-v	파일 옮기기 전의 과정을 보여 줌

[실습] mv 명령어 실습

■ 사용시스템
- server1

■ 실습 시나리오
- mv 명령어 사용법
- Inode Number 확인
- 여러 개의 파일을 동시에 이동

[EX1] mv 명령어 사용법
① 실습 준비
```
[root@server1 /test]# cd /test ; rm -rf /test/*
[root@server1 /test]# touch file1
[root@server1 /test]# mkdir dir1
[root@server1 /test]# touch dir1/file2
[root@server1 /test]# tree
```
```
/test
|-- dir1/
|   `-- file2
`-- file1
```

② "mv file1 file2" 형식 테스트

목적: 파일이름 변경: file1 -> file3

```
[root@server1 /test]# mv file1 file3
[root@server1 /test]# tree
```
```
.
|-- dir1
|   `-- file2
`-- fil3
```

③ "mv file1 dir1" 형식 테스트

목적: file3 파일을 dir1 안에 넣고 싶다.(move)

```
[root@server1 /test]# mv file3 dir1
[root@server1 /test]# tree
```
```
.
`-- dir1
    |-- file2
    `-- file3
```

④ "mv dir1 dir2" 형식 테스트

목적: dir1 디렉토리의 이름을 dir2 변경하기

```
[root@server1 /test]# mv dir1 dir2
[root@server1 /test]# tree
```
```
.
`-- dir2
    |-- file2
    `-- file3
```

목적: dir1 디렉토리를 dir2 디렉토리 안으로 넣고 싶다.(move)

```
[root@server1 /test]# cp -r dir2 dir1
[root@server1 /test]# mv dir1 dir2
[root@server1 /test]# tree
```
```
.
`-- dir2
    |-- dir1
    |   |-- file2
    |   `-- file3
    |-- file2
```

```
    `-- file3
```

[EX2] Inode Number 확인

- 같은 파일 시스템내에서 파일을 이동하기 위해서 mv 명령어를 수행하는 경우에는 파일의 이름(경로 포함)만 바꾸는 이다.
- 하지만, 다른 파일 시스템으로 mv 명령어를 수행하는 경우는 새로 파일이 생성되기 때문에 파일의 이름(경로 포함)만 바뀌는 것이 아니라, 새로운 파일이 생성되는 것이다.

① 실습 준비

```
[root@server1 /test]# cd /test ; rm -rf /test/*
[root@server1 /test]# touch file1
[root@server1 /test]# ls -li file1          /* Inode Number 확인 */
```

```
100666903 -rw-r--r-- 1 root root 0 Feb  7 10:01 file1
```

② 같은 파일시스템(File System) 내에서 파일을 이동하는 경우

```
[root@server1 /test]# mv file1 file3
[root@server1 /test]# ls -li file3          /* Inode Number 변하지 않은 걸 알 수 있음 */
```

```
100666903 -rw-r--r-- 1 root root 0 Feb  7 10:01 file3
```

[EX3] 여러 개의 파일을 동시에 이동

① 실습 준비

```
[root@server1 /test]# cd /test
[root@server1 /test]# touch file1 file2 file3 file4
[root@server1 /test]# ls
```

```
file1 file2 file3 file4
```

② 여러개의 파일을 지정하기 위해서 wild card character 사용하기

```
[root@server1 /test]# mkdir dir1
[root@server1 /test]# mv file* dir1     (# mv file1 file2 file3 file4 dir1)
[root@server1 /test]# ls dir1
```

```
file1 file2 file3 file4
```

[참고] 와일드 카드 문자(Wildcard Character)

와일드 카드 문자: 하나의 문자가 여러개의 문자의 의미를 포함하는 특수문자
다음은 와일드 카드 문자의 대표적인 몇가지 예이다.
- * : 0 or more character (except .file) (EX) # cp file* dir1
- ? : one charater (EX) # cp file? dir1
- { } : 여러 문자열(단어) (EX) # cp file{aa,bb,cc} dir1
- [ ] : 선택적인 하나의 문자 (EX) # cp file[123] dir1

```
# cd /test ; rm -rf /test/*
# touch file file1 file2 file22 file3 .file4
# ls -a

■ 와일드문자(* : 0 or more character (except .file))
별표(*): 모든 글자(시작: 0, .(점)으로 시작하는 문자 제외)
# ls file*                -> file file1 file2 file22 file3
# ls *                    -> file file1 file2 file22 file3
=> [참고] rm -rf /test/*
=> [참고] ls /etc/*release

■ 와일드문자(? : one charater)
물음표(?): 모든 한 문자
# ls file?                -> file1 file2 file3
# ls file??               -> file22

■ 와일드문자({} : 여러 문자열(단어))
# ls file{1,2,22,3}   -> file1 file2 file22 file3
# ls file{1..3}       -> file1 file2 file3

=> [참고] touch dir1/file{1,2}

■ 와일드문자([] : 선택적인 하나의 문자)
# ls file[123]        -> file1 file2 file3
# ls file[1-3]
# ls file[0-9A-Za-z]  -> file1 file2 file3

(정리 작업)
# cd
# rm -rf /test
# mkdir -p /test
```

4 rm CMD

```
NAME
       rm - 파일 지우기

SYNOPSIS
       rm  [-dfirvR]  [--directory]  [--force]  [--interactive]  [--recursive]
       [--help] [--version] [--verbose] name...

DESCRIPTION
       이 문서는 더이상 최신 정보를 담고 있지않다.  그래서,  몇몇틀릴  경우 도
       있고,  부족한  경우도  있을  것이다.  완전한매뉴얼을 원하면, Texinfo 문
       서를 참조하기 바란다.

       이 매뉴얼 페이지는 rm 명령의 GNU  버전에  대한  것이다.  rm 명령은 지정
       한 파일을 지운다. 초기값으로는 경로는 지우지 않는다.

       만 약에 파일 모드가 읽기 전용이고, 표준 입력이 tty이고, -f나 --force 옵
       션이 지정되지 않으면, rm 명령은 사용자에게 지정한 파일을 정말 지울것 인
       지 물어본다.  이때,  'y' 나  'Y' 를 입력해 주어야지만 그 파일을 지운다.

       GNU  rm 명령과 같이 getopt(3) 함수를 사용하는 모든 풀그림에서는 -- 옵션
       다음에 오는 것은 옵션이 아닌 것으로 인식한다.  즉 파일 이름이  '-f'  라는
       파일을 지우고자 한다면, 다음 두 방법을 사용한다.
              rm -- -f
       또는
              rm ./-f
       유 닉스 rm 명령의  '-'  문자로 시작하는 옵션들 때문에이런 기능들이 고안된
       것이다.

   OPTIONS
       -f, --force
              지 울 파일이 없을 경우에 아무런 메시지를 보여주지 않고 그냥 넘어
              간다.  이 옵션은 쉘 스크립트 안에서 사용될 때 유용하게 쓰인다.

       -i, --interactive
              각 파일을 하나씩 지울 것인지 사용자에게 일일이 물어본다.   이 때
               'y'  나  'Y' 를 눌러야지만 파일이 지워지다.

       -r, -R, --recursive
              일반 파일이면 그냥 지우고, 경로면, 그 하위 경로와 파일을 모두 지
              운다.
```

파일과 디렉토리를 지우고자 할 때 사용하며 한꺼번에 여러 개를 지울 수도 있으며 지운 파일들은 되살릴 수 없으므로 주위 해서 사용해야 하는데 -i 옵션을 사용하면 한번 더 묻게 되므로 부주의로 인한 파일 삭제를 막을 수 있으며, 옵션 -r를 사용 시 시스템의 모든 파일이 삭제되는 경우도 있으니 신중하게 사용해야 한다.

[명령어 형식]

```
# rm file1            /* file1 파일 1개 삭제 */
# rm file1 file2      /* file1, file2 파일 2개 삭제 */
# rm -r dir1          /* dir1 디렉토리 하위경로까지 삭제 */
```

[명령어 옵션]

옵션	설 명
-f	강제로 파일을 지우고 삭제할 파일이 없을 경우에도 아무런 메시지를 보여주지 않는다.
-i	파일을 삭제할 것인지 사용자에게 물어봄
-r, -R	일반파일이면 그냥 지우고 디렉토리일 경우 그 하위경로와 파일을 모두 지움
-v	삭제되는 파일의 정보를 보여줌

[참고] 비어 있지 않은 디렉토리 삭제
 # rm -rf dir1
* -r : recursive => 비어 있지 않은 디렉토리 삭제할 때 사용하는 옵션
* -f : force => 묻지 않고 삭제하는 옵션

[참고] rm 명령어로 지운 파일 복구(100% 장담할 수 없음)
- (TUI) extundelete CMD 사용하는 방법(ext3, ext4)
- (TUI) xfs_undelete CMD 사용하는 방법(xfs)
- (GUI/TUI) TestDisk 툴을 사용하는 방법(Windows(FAT32/NTFS), Linux(ext3/ext4))

TestDisk 사용법
* https://www.cgsecurity.org/wiki/TestDisk
* https://blog.naver.com/lux-00/222275996490

[실습] rm 명령어 실습

■ 사용시스템
- server1

■ 실습 시나리오
- rm 명령어 -r 옵션 사용에 대해서

[EX1] rm 명령어 -r 옵션 사용에 대해서

① 실습 준비
```
[root@server1 ~]# cd /test; rm -rf /test/*
[root@server1 /test]# mkdir dir1
[root@server1 /test]# touch dir1/file{1,2}
```

② $HOME/.bashrc 환경 파일의 alias 확인
```
[root@server1 /test]# cat ~/.bashrc
```
```
# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# User specific environment
if ! [[ "$PATH" =~ "$HOME/.local/bin:$HOME/bin:" ]]
then
    PATH="$HOME/.local/bin:$HOME/bin:$PATH"
fi
export PATH

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions

alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

[참고] alias 사용법
(선언) **# alias cp='cp -i'**
(확인) **# alias (# alias cp)**
(해제) **# unalias cp**

③ rm -r 옵션을 사용하여 디렉토리 삭제시 메세지 확인
```
[root@server1 /test]# rm -r dir1
```
```
rm: descend into directory `dir1'? <CTRL + C>
```
- rm 명령어에는 기본적으로 -i 옵션이 alias로 설정되어 있으므로, dir1 디렉토리를 지울때 안에 파일의 개수 만큼 물어 보게 된다. 이런 경우 몇번의 y를 선택해야 할지 모르기 때문에 <CTRL + C> 종료한 이후에 rm 명령어에 -rf 옵션을 사용하여 내용이 있는 디렉토리의 내용을 묻지 않고, 모두 지울 수 있다.

④ rm -rf 옵션을 사용하여 디렉토리 삭제 - 묻지 않고 지우기
```
[root@server1 /test]# ls
```
```
dir1
```
-> dir1 존재 확인

```
[root@server1 /test]# rm -rf dir1    /* -f : force */
[root@server1 /test]# ls
[root@server1 /test]#
```

[질문 & 답변]

[선수작업]
다음과 같은 구조를 만든다.
- 디렉토리 생성: **mkdir CMD**
- 파일 생성: **touch CMD**

```
/test
|-- dir1/
|   |-- dir2/
|   |   `-- file3
|   `-- file2
|-- dir3/
|   `-- file4
`-- file1
```

배운 명령어:

* 디렉토리 관리:
  - mkdir, rmdir, ls

* 파일 관리:
  - touch, cp, mv, rm

[질문] /test 디렉토리 밑에 있는 file1 이름의 파일을 file2로 변경하고 싶다.
    * 이름변경: /test/file1 -> /test/file2
[답변]

[질문] /test/dir1 디렉토리 밑에 있는 file2를 /test/dir1/dir2 디렉토리로 복사하고 싶다.
    * 파일복사: /test/dir1/file2 -> /test/dir1/dir2/file2
[답변]

[질문] /test/dir3 디렉토리 이름을 /test/dir4로 변경하고 싶다.
    * 이름변경: /test/dir3 -> /test/dir4
[답변]

[질문] /test/dir1 디렉토리를 삭제하고 싶다.
[답변]

>>>> 작업을 할 때는 작업 디렉토리안에 들아가서 작업한다. <<<<

파일 내용 확인 명령어 (1 of 2)

• cat 명령어

```
# cat /etc/passwd
# cat -n /etc/passwd
# cat file1 file2 > file3

# cat /etc/passwd | grep xinetd
```

• more 명령어

```
# cat /etc/services
# more /etc/services

# CMD | more
# ps -ef | more
# rpm -qa | more
# cat /etc/services | more
```

http://cafe.daum.net/bscsolaris

파일 내용 확인 명령어 (2 of 2)

• head 명령어

```
# head /etc/passwd
# head -5 /etc/passwd

# alias pps='ps -ef | head -1 ; ps -ef | grep $1'
# pps xinetd
```

• tail 명령어

```
# tail /etc/passwd
# tail -5 /etc/passwd

# tail -f /var/log/messages
```

http://cafe.daum.net/bscsolaris

1 cat CMD

```
NAME
       cat - concatenate files and print on the standard output

SYNOPSIS
       cat [OPTION] [FILE]...

DESCRIPTION
       Concatenate FILE(s), or standard input, to standard output.

       -n, --number
              number all output lines
```

파일의 내용을 화면으로 출력. 파일의 내용을 화면에 연속적으로 출력하기 때문에 파이프(Pipe Line)을 사용하여 more 명령어에 연결하여 사용 가능.

[명령어 형식]

```
# cat file1                    /* file1 파일 내용을 출력 */
# cat file1 file2              /* file1, file2 파일 내용을 출력 */
# cat -n file1                 /* file1 파일내용을 줄번호(line number)와 함께 출력 */
# cat file1 file2 > file3      /* file1, file2 출력 결과를 file3에 저장 */
```

[명령어 옵션]

옵션	설명
-e	제어 문자를 ^ 형태로 출력하며 끝에 $를 추가
-n	줄번호를 공백을 포함하여 화면 왼쪽에 나타냄
-s	중복되고 겹치는 빈 행은 하나의 빈 행으로 처리
-v	행바꿈 문자, tab를 제외한 제어문자를 ^ 형태로 출력
-E	각 행 끝에 $ 문자 출력
-T	tab 문자를 출력
-A	-vET 옵션과 동일

[실습] cat 명령어 실습

■ 사용시스템
- server1

■ 실습시나리오
- /etc/passwd 파일 출력하기
- file1, file2 두 개의 파일을 하나의 file3으로 합치기

[EX1] /etc/passwd 파일 출력 하기
- cat 명령 사용시 -n 옵션을 사용하는 방법이나 파이프(|) 기호와 연결하여 사용하는 방법에 대해서 다루어 본다.

[root@server1 /test]# cat /etc/passwd

```
..... (중략) .....
gdm:x:42:42::/var/lib/gdm:/sbin/nologin
clevis:x:977:976:Clevis Decryption Framework unprivileged user:/var/cache/clevis:/sbin/nologin
gnome-initial-setup:x:976:975::/run/gnome-initial-setup/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rngd:x:975:974:Random Number Generator Daemon:/var/lib/rngd:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
fedora:x:1000:1000:fedora:/home/fedora:/bin/bash
```

[root@server1 /test]# cat -n /etc/passwd | more (# nl /etc/passwd | more)

```
     1  root:x:0:0:root:/root:/bin/bash
     2  bin:x:1:1:bin:/bin:/sbin/nologin
     3  daemon:x:2:2:daemon:/sbin:/sbin/nologin
     4  adm:x:3:4:adm:/var/adm:/sbin/nologin
     5  lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
     6  sync:x:5:0:sync:/sbin:/bin/sync
     7  shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
.... (생략) .....
```

[root@server1 /test]# cat /etc/passwd | grep fedora

```
fedora:x:1000:1000:fedora:/home/fedora:/bin/bash
```

* (WIN) C:> type file1.txt | findstr fedora

[EX2] file1, file2 두 개의 파일을 하나의 file3으로 합치기

- cat 명령을 사용하여 2개의 파일을 합치는 작업을 진행할 수 있다.

[root@server1 /test]# echo 1111 > file1
[root@server1 /test]# echo 2222 > file2
[root@server1 /test]# cat file1 file2

```
1111
2222
```

[root@server1 /test]# cat file1 file2 > file3
[root@server1 /test]# cat file3

```
1111
2222
```

- [참고] 파일 쪼개기: **split CMD**

[참고] cat 명령어를 사용하여 바이너리 파일 확인 시

일반적으로 바이너리(binary) 파일에 대해 cat 명령어로 볼려고 하는 경우 정상적으로 보이지 않게 된다. 이런 경우 바이너리 파일에서 볼수 있는 string 들만 뽑아서 보여 주는 명령어가 있는데, 바로 strings 명령어이다.

■ 바이너리 파일 내용 보기
(X) # cat /bin/ls
(O) # strings -f /bin/ls

 # man strings

```
NAME
       strings - print the strings of printable characters in files.

DESCRIPTION
       For  each  file  given,  GNU  strings  prints  the  printable character
       sequences that are at least 4 characters long (or the number given with
       the  options  below)  and are followed by an unprintable character.  By
       default, it only prints the strings from  the  initialized  and  loaded
       sections  of  object  files;  for  other  types of files, it prints the
       strings from the whole file.

       strings is mainly useful  for  determining  the  contents  of  non-text
       files.

OPTIONS
       -f
       --print-file-name
           Print the name of the file before each string.
```

2 more CMD

```
NAME
     more - 문자속성을 살린 파일 보기 풀그림

사용법
     more [-dlfpcsu] [-num] [+/ 표현식] [+ 줄번호] [file ...]

설명
     More 명령은 파일 보기 풀그림이다. 이 풀그림은 유닉스에서 처음부터사용한
     풀그림이다. 하지만, 이 풀그림의 기능을 보다 보강한 less(1) 명령을 사용하
     기를 바란다.

옵션
     다음은 이 풀그림에서 사용할 수 있는 옵션이며, 이 옵션은 MORE 환경변수로도
     지정할 수 있다. 이미 이 환경 변수로 옵션들이 지정되어있는데, 명령행으로
     옵션을 사용하면, 그 환경 변수는 무시된다.

     -c    Do not scroll.  Instead, paint each screen from the top, clearing
           the remainder of each line as it is displayed.
```

큰 파일을 출력할 때 화면 크기 페이지 단위로 출력하며 하단에 "--More--(20%)"는 현재 내용을 20% 보았고 80% 남았다고 표현하며 화면에서 엔터(Enter)키를 누르면 한 개의 라인(line) 단위로 넘어가고 스페이스(space) 키를 누르면 한 페이지 단위로 넘어가는데 less 명령과 함께 사용 하면 더 효율적이다.

[명령어 형식]

```
# more file1          /* file1 파일을 출력 */
```

[명령어 옵션]

옵션	설명
-n(숫자)	출력 행수를 지정
-c	위에서부터 한 행씩 지운 후 한 행씩 출력
-d	스페이스나 q를 누르라는 프롬프트를 출력
-f	보통은 긴 칼럼의 행은 화면에서 행 바꿈을 하여 새로운 행으로 계산되는데 -f 옵션은 새로운 행으로 계산 하지 않으며 화면이 행이 아닌 논리적인 행 수를 계산
-s	여러 개의 빈 공백행은 하나로 취급
-p	스크롤하지 않으며 화면을 지우고 출력
-u	밑줄 치기를 금지

[실습] more 명려어 실습

■ 사용시스템
- server1

■ 실습시나리오
- cat CMD & more CMD 비교
- 주로 사용되는 more 명령어 형식

[EX1] **cat CMD** versus **more CMD**

[참고] cat CMD vs more CMD 차이점
 # cat /etc/services
 # more /etc/services

① CMD | more 형식과 more file 형식 비교
 # more /etc/services

```
# /etc/services:
# $Id: services,v 1.42 2006/02/23 13:09:23 pknirsch Exp $
#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994).  Not all ports
# are included, only the more common ones.
--More--(0%)                          /* 0% 확인 */
```

 # cat /etc/services | more

```
# /etc/services:
# $Id: services,v 1.42 2006/02/23 13:09:23 pknirsch Exp $
#
# Network services, Internet style
#
# Note that it is presently the policy of IANA to assign a single well-known
# port number for both TCP and UDP; hence, most entries here have two entries
# even if the protocol doesn't support UDP operations.
# Updated from RFC 1700, ``Assigned Numbers'' (October 1994).  Not all ports
# are included, only the more common ones.
--More--
```

- 결론적으로 cat /etc/services | more 또는 more /etc/services 형식은 출력 결과가 흡사하다는 것을 알수 있다. 하지만 주로 사용되는 형식은 전자인 CMD | more 형식이 주로 사용된다.

[EX2] 주로 사용되는 more 명령어 형식 - **"CMD | more"** 형식 실습

명령어 형식
 # CMD (EX: **# help**) /* 쉘 내부(내장) 명령어의 목록 확인 */
 # CMD | more (EX: **# help | more**)

자주 사용되는 형식(ex: CMD | more)
 # ps -ef | more
 # cat /etc/services | more
 # netstat -an | more
 # systemctl list-unit-files

[참고] 파이프(|) 기호의 의미에 대해서

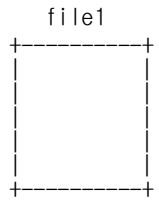
ps -ef | more

ex) cat /etc/passwd | grep fedora

3 head CMD

```
NAME
       head - output the first part of files

SYNOPSIS
       head [OPTION]... [FILE]...

DESCRIPTION
       Print  the  first  10 lines of each FILE to standard output.  With more
       than one FILE, precede each with a header giving the file  name.   With
       no FILE, or when FILE is -, read standard input.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.

       -c, --bytes=[-]N
              print the first N bytes of each  file;  with  the  leading  '-',
              print all but the last N bytes of each file

       -n, --lines=[-]N
              print  the first N lines instead of the first 10; with the lead-
              ing  '-', print all but the last N lines of each file
```

파일의 처음 시작 부분의 몇 줄을 출력 하고 자 할 때 사용. 따라서 긴 파일의 내용의 앞 부분만을 출력 하고자 할 때 유용하게 사용 되며 head 명령어에 아무런 옵션 없이 사용된 경우 문서의 처음 10줄을 보여 준다.



[명령어 형식]

```
# head /etc/passwd          (# head -10 /etc/passwd, # head -n 10 /etc/passwd)
# head -n 5 /etc/passwd     /* 숫자에 해당하는 라인 번호 수 만큼만 출력 (기본은 10줄) */
# head -5 /etc/passwd
```

[명령어 옵션]

옵션	설 명
-n (숫자)	위쪽 행에서부터 출력할 행수를 지정

[실습] head 명령어 실습

■ 사용시스템
- server1

■ 실습 시나리오
- head 명령어를 사용한 파일의 일부 출력
- head 명령어를 이용한 프로세스의 헤더 부분 출력

[EX1] "head -n #" 실습 - head 명령어를 사용한 파일의 일부 출력
- head -n # 명령을 사용하여 파일의 상단에 몇줄을 확인해 본다.

[root@server1 ~]# head -n 5 /etc/passwd (# head -5 /etc/passwd)

```
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
```

[EX2] head 명령어를 이용한 프로세스의 헤더 부분 출력
- ps -ef 명령은 많이 사용되는 명령이며, 뒤에 파이프(|) 기호에 grep 명령을 연결하여 사용하는 경우가 많다. 이런경우 전체 출력결과에서 원하는 패턴이 들어 있는 부분만을 출력하기 때문에 원래 ps 명령어의 출력 결과 중에서 헤드라인정보가 출력 되지 않습니다.
- 따라서, ps 명령어의 헤드라인 정보와 "ps -ef | grep 패턴" 형식의 출력결과를 합쳐서 표시하는 방법을 사용하면 좋다.

자주 사용하는 명령어 형식 => **ps -ef | grep rsyslogd** => 편하게, 조금 더 효과적으로 실행

① ps -ef 명령어 실행
[root@server1 ~]# ps -ef
-> 출력 내용 생략

② ps -ef | more 명령어를 사용하여 헤더 정보 확인
[root@server1 ~]# ps -ef | more

```
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 Feb06 ?        00:00:26 /usr/lib/systemd/systemd --s
witched-root --system --deserialize 18
root           2       0  0 Feb06 ?        00:00:00 [kthreadd]
root           3       2  0 Feb06 ?        00:00:00 [rcu_gp]
root           4       2  0 Feb06 ?        00:00:00 [rcu_par_gp]
.... (생략) ....
```

③ ps + grep 주로 사용되는 형식
[root@server1 ~]# ps -ef | grep rsyslogd

```
root        1391       1  0 Feb06 ?        00:00:03 /usr/sbin/rsyslogd -n
```

- ps + grep 명령어를 조합하여 주로 사용되는 형식을 사용하면 원하는 정보만 보이고, 헤더 정보는 보이지 않기 때문에 출력결과가 직관적이지 못하다.

④ 헤더 정보와 원하는 정보 출력 결과 합치기
[root@server1 ~]# ps -ef | head -1

```
UID        PID  PPID  C STIME TTY          TIME CMD
```

[root@server1 ~]# ps -ef | head -1 ; ps -ef | grep rsyslogd

```
UID          PID    PPID  C STIME TTY          TIME CMD
root        1391       1  0 Feb06 ?        00:00:03 /usr/sbin/rsyslogd -n
```

⑤ 위 ④번의 내용을 alias 선언하기
[root@server1 ~]# alias pps='ps -ef | head -1 ; ps -ef | grep $1'
* $1 : 첫번째 인자(Argument) 변수

[root@server1 ~]# pps rsyslogd

```
UID          PID    PPID  C STIME TTY          TIME CMD
root        1391       1  0 Feb06 ?        00:00:03 /usr/sbin/rsyslogd -n
```

⑤ 환경파일에 등록하고 적용하기
[root@server1 ~]# gedit ~/.bashrc

```
..... (중략) .....
alias pps='ps -ef | head -1 ; ps -ef | grep $1'
```

[root@server1 ~]# . ~/.bashrc (# source ~/.bashrc)
[root@server1 ~]# pps syslogd
-> 정상 출력 내용 확인

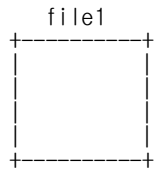
[참고] nstat alias($HOME/.bashrc)
alias nstate='netstat -antup | head -2 ; netstat -antup | grep $1'
nstate :22

>>>> 자주사용되는 명령어 + 유용한 명령어 => alias <<<<

5 tail CMD

```
NAME
       tail - output the last part of files

SYNOPSIS
       tail [OPTION]... [FILE]...

DESCRIPTION
       Print  the  last  10  lines of each FILE to standard output.  With more
       than one FILE, precede each with a header giving the file  name.   With
       no FILE, or when FILE is -, read standard input.

       Mandatory  arguments  to  long  options are mandatory for short options
       too.

       -c, --bytes=N
              output the last N bytes

       -f, --follow[={name|descriptor}]
              output appended data as the file grows; -f, --follow, and --fol-
              low=descriptor are equivalent

       -n, --lines=N
              output the last N lines, instead of the last 10
```

tail은 텍스트파일이나 지정된 데이터의 마지막 몇 줄을 보여주는 데 사용하는 Unix 및 Unix계열 시스템에서의 프로그램이다. 파일의 끝 부분만 출력 하고자 할 때 사용하며, 아무런 옵션 없이 사용된 경우 문서의 마지막 10줄을 보여는데, 예를 들어서 사용자가 추가되면 /etc/passwd 파일에 마지막에 추가 된다. 이때 tail 명령어에 -1 옵션을 사용하여 사용자 추가를 확인 할 수 있다.



[명령어 형식]

```
# tail /etc/passwd     (# tail -10 /etc/passwd, # taill -n 10 /etc/passwd)
# tail -n 5 /etc/passwd
# tail -n +5 /etc/passwd

# tail -f /var/log/messages
```

[명령어 옵션]

옵 션	설 명
-c (숫자)	끝에서부터 지정된 수만큼의 바이트에 해당하는 정보를 보여준다.
-f	파일의 크기가 변할때마다 추가된 정보를 출력한다.
-(숫자) -n (숫자)	끝에서부터 지정된 수만큼의 줄을 보여준다.
-q	출력결과에서 맨 윗줄에 입력파일명을 표시하지 않게 설정한다.
-v	-q와 반대로 출력결과에서 맨 윗줄에 입력파일명을 항상 표시해준다.
--help	도움말을 보여준다.
--version	버젼 정보를 보여준다.

[실습] tail 명령어 실습

■ 사용시스템
- server1

■ 실습 시나리오
- tail 명령어의 기본 사용법
- 로그 파일 모니터링 실습

[EX1] tail 명령어의 기본 사용법
[root@server1 ~]# tail -n 5 /etc/passwd (# tail -5 /etc/passwd)

```
gnome-initial-setup:x:976:975::/run/gnome-initial-setup/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
rngd:x:975:974:Random Number Generator Daemon:/var/lib/rngd:/sbin/nologin
tcpdump:x:72:72::/:/sbin/nologin
fedora:x:1000:1000:fedora:/home/fedora:/bin/bash
```

[참고] tail 명령어의 응용되는 예
ps -ef | grep rsyslogd | tail -n +2 | grep -v grep

[EX2] 로그 파일 모니터링(EX: **tail -f CMD**)
- tail은 실시간으로 파일의 변화를 감지할 수 있게 해주는 -f옵션이라는 특별한 명령행 옵션을 가지고 있다. 마지막 몇줄을 출력하고 끝내는 것에 그치지 않고, tail은 그 줄들을 표시하고 파일을 감독한다. 새 줄들이 다른 프로세스에 의해 그 파일에 추가될 때, tail의 -f 옵션은 그 표시 또한 실시간으로 업데이트한다. 이 옵션은 특히 로그파일(입출력정보파일)들을 감독할때 유용하다.
- 다음의 명령구문은 messages라는 파일의 마지막 열줄을 보여주고 새 줄들이 추가되면 그 줄들을 추가하여 보여준다.

[참고] 원격 접속 서비스
* **telnet CMD** : 평문 통신
* **ssh CMD** : 암호화 통신

(선수작업) telnet 서비스 활성화 방법(일반사용자)
(ㄱ) 패키지 설치
 # yum -y install telnet telnet-server
(ㄴ) telnet 서비스 기동
 # systemctl enable --now telnet.socket
(ㄷ) 서비스 요청
 # telnet localhost
fedora 사용자 로그인
$ id
$ exit
(ㄹ) root 사용자 로그인 테스트
 # telnet localhost
root 사용자로 로그인
 # id
 # tty
 # exit

① (server1) 첫번째 터미널에서 /var/log/messages 로그 파일 모니터링
- **/var/log/messages:** OS 전반적인 기록을 남겨주는 로그파일이다.

[TERM1] 관리자용 윈도우
```
[root@server1 ~]# tail -f /var/log/messages
<ENTER>
<ENTER>
<ENTER>
```

② 두번째 터미널에서 user01 사용자로 로그인
[TERM2] 사용자 윈도우

[참고] 사용자 추가 방법
```
# useradd user01
# passwd user01
```

```
[root@server1 ~]# telnet localhost
```
user01 사용자로 로그인

```
[root@server1 ~]# id
[root@server1 ~]# exit
```

③ 첫번째 터미널의 쌓인 로그 분석
[TERM1] 관리자용 윈도우

■ (로그인 기록)
```
      ❶          ❷       ❸          ❹
Feb  7 10:37:28 server1 systemd[1]: Started Telnet Server ([::1]:49686).
Feb  7 10:37:35 server1 systemd-logind[1088]: New session 12 of user root.
Feb  7 10:37:35 server1 systemd[1]: Started Session 12 of user root.
```

■ (로그아웃 기록)
```
Feb  7 10:37:43 server1 systemd[1]: telnet@2-::1:23-::1:49686.service: Succeeded.
Feb  7 10:37:43 server1 systemd[1]: session-12.scope: Succeeded.
Feb  7 10:37:43 server1 systemd-logind[1088]: Session 12 logged out. Waiting for processes to exit.
Feb  7 10:37:43 server1 systemd-logind[1088]: Removed session 12.
<CTRL + C>
```

로그 기록 해석

❶ Feb 7 10:37:28	로그 생성 시간
❷ server1	로그 생성 서버
❸ systemd[1]:	로그 생성 주체[PID]
❹ Started Telnet Server ([::1]:49686).	로그 메세지

[참고] journalctl CMD 명령어를 사용한 service log 모니터링
```
# systemctl status sshd
[TERM1] # journalctl -f
[TERM2] # systemctl restart sshd
```

[실무 예] 서버를 실시간적으로 모니터링

```
[TERM1] # top                          (# gnome-system-monitor)
[TERM2] # tail -f /var/log/messages    (# gnome-logs)

# tail -f /var/log/messages | grep -i dhcpd
# tail -f /var/log/messages | grep -i named
# tail -f /var/log/messages | grep oracle
# tail -f /var/log/messages | grep wasuser
```

■ 안 좋은 로그 메세지를 모니터링
```
# tail -f /var/log/messages | egrep -i 'warn|fail|error|crit|alert|emerg'
```

■ 연관성이 있는 로그 기록를 한꺼번에 모니터링
```
# tail -f /var/log/messages /var/log/secure
```

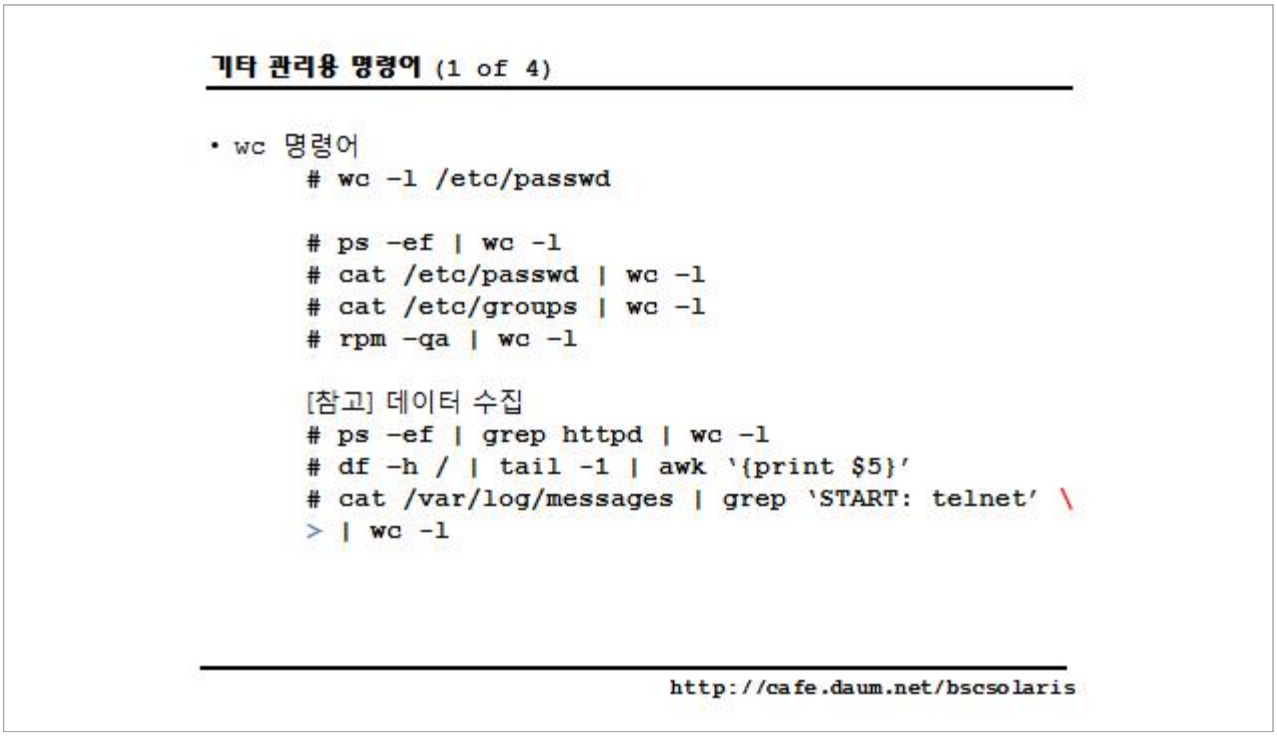
[추가적인 작업] 실습시 필요한 설정(보안상 권장하지는 않는다.)

- main/server1/server2 시스템에 telnet 서비스를 활성화(enable/start) 한다.
- 또한, 관리자(ex: root)로 로그인도 가능해야 한다.

```
# yum -y install telnet telnet-server
# systemctl enable --now telnet.socket
# telnet localhost
root 사용자로 로그인
# exit
#
```

telnet 서비스를 확인하기 위해서 다음 명령을 복사해서 사용할 수도 있다.

```
# (server1) telnet 서비스 기동
# telnet 서비스 활성화
yum -y install telnet telnet-server

# telnet 서비스 확인
CMD() {
  sleep 2; echo 'root'
  sleep 1; echo 'centos'
  sleep 1; echo 'hostname'
  sleep 1; echo 'exit'
}

for i in 10 20 30
do
    CMD | telnet 192.168.10.$i
done
```



1 wc CMD

```
NAME
       wc - print the number of newlines, words, and bytes in files

SYNOPSIS
       wc [OPTION]... [FILE]...

DESCRIPTION
       Print newline, word, and byte counts for each FILE, and a total line if
       more than one FILE is specified.  With no FILE, or when FILE is -, read
       standard input.

       -c, --bytes
              print the byte counts

       -m, --chars
              print the character counts

       -l, --lines
              print the newline counts

       -w, --words
              print the word counts
```

파일 내의 문자수, 단어 수 그리고 라인수를 확인하고자 할 때 사용한다. wc 명령어를 사용하여 프로세스의 수, 시스템에 설치된 패치의 수, 시스템에 설치된 패치의 수 등을 확인 할 때 사용 할 수 있다. wc 명령어에 -l 옵션은 쉘스크립트나 파일에 대한 무결성 체크 등 많은 곳에서 활용이 가능하다.

[명령어 형식]

```
# wc /etc/passwd
# wc -l /etc/passwd
# wc -w /etc/passwd
# wc -c /etc/passwd
```

[명령어 옵션]

옵션	설 명
-c	문자수만 출력
-l	라인수만 출력
-w	단어수만 출력

```
[참고] 데이터 수집(Data Gathering) 예
# cat /etc/passwd | wc -l
# rpm -qa | wc -l
# ps -ef | wc -l
# ps -ef | grep httpd | wc -l
# cat /var/log/messages | grep 'Feb 13' | grep 'Started Telnet Server' | wc -l
# df -h / | tail -1 | awk '{print $5}'
```

[실습] wc 명령어와 데이터 수집

■ 사용시스템
- server1

■ 실습 시나리오
- wc 명령어 기본 사용법
- 시스템 사용자 수 확인
- 실행중인 프로세스의 수 확인
- 설치된 패키지 수 확인
- 시스템성능/사용량 카운트 수집

[EX1] wc 명령어 기본 사용법
wc OPTIONS
- -c, --bytes print the byte counts
- -m, --chars print the character counts
- -l, --lines print the newline counts

[root@server1 ~]# wc /etc/passwd

```
  48  116 2707 /etc/passwd
```

[root@server1 ~]# wc -l /etc/passwd

```
48 /etc/passwd
```

[root@server1 ~]# wc -w /etc/passwd /* 단어수 (word) */

```
116 /etc/passwd
```

[root@server1 ~]# wc -c /etc/passwd /* 문자수 (byte) */

```
2707 /etc/passwd
```

[EX2] 시스템 사용자 수 확인
- 현재 시스템에 존재하는 모든 사용자의 수는?

 # cat /etc/passwd | wc -l

```
59
```

 # wc -l /etc/passwd

```
59 /etc/passwd
```

[EX3] 실행중인 프로세스의 수 확인
- 현재 떠 있는 프로세스의 개수는?

```
# ps -ef | wc -l
```

```
329
```

- ps -ef | tail -n +2 | wc -l

[EX4] 설치된 패키지 수 확인
- 현재 시스템에 설치된 패키지 개수는?
- [참고] RPM(RedHat Package Manager)

```
# rpm -qa | wc -l
```

```
1359
```

-> 시스템에 설치된 패키지 수

[EX5] 시스템성능/사용량 카운트 수집 : 데이터 수집(Data Gathering)
- 아파치 웹서버(process 방식)에서 웹서비스 요청을 클라이언트가 하면 어떻게 되는가?

■ Apache Webserver Example

Web Client ----------> Web Server(www.daum.net)
http://www.daum.net httpd

(선수작업) Apache HTTP 패키지 설치
```
# yum -y install httpd mod_ssl
```

```
[root@server1 ~]# systemctl restart httpd
[root@server1 ~]# ps -ef | grep httpd
```

```
root      24532      1  0 10:52 ?        00:00:00 /usr/sbin/httpd -DFOREGROUND
apache    24540  24532  0 10:52 ?        00:00:00 /usr/sbin/httpd -DFOREGROUND
apache    24541  24532  0 10:52 ?        00:00:00 /usr/sbin/httpd -DFOREGROUND
apache    24542  24532  0 10:52 ?        00:00:00 /usr/sbin/httpd -DFOREGROUND
apache    24543  24532  0 10:52 ?        00:00:00 /usr/sbin/httpd -DFOREGROUND
```

```
[root@server1 ~]#  ps -ef | grep httpd | wc -l >> apache.count
[root@server1 ~]#  cat apache.count
```

```
5
```

[EX6] 디스크 사용량을 모니터링
```
[root@server1 ~]# df -k
```

```
Filesystem           1K-blocks     Used Available Use% Mounted on
devtmpfs               1402100        0   1402100   0% /dev
tmpfs                  1432596        0   1432596   0% /dev/shm
tmpfs                  1432596    26244   1406352   2% /run
tmpfs                  1432596        0   1432596   0% /sys/fs/cgroup
/dev/mapper/cl-root   39425220  5242996  34182224  14% /
/dev/mapper/cl-home   19245056   181140  19063916   1% /home
/dev/sda1              1038336   245320    793016  24% /boot
```

```
[root@server1 ~]# df -k /
```

```
Filesystem           1K-blocks     Used Available Use% Mounted on
/dev/mapper/cl-root   39425220  5242996  34182224  14% /
```

```
[root@server1 ~]# df -k / | tail -1
```

```
/dev/mapper/cl-root   39425220  5242996  34182224  14% /
```

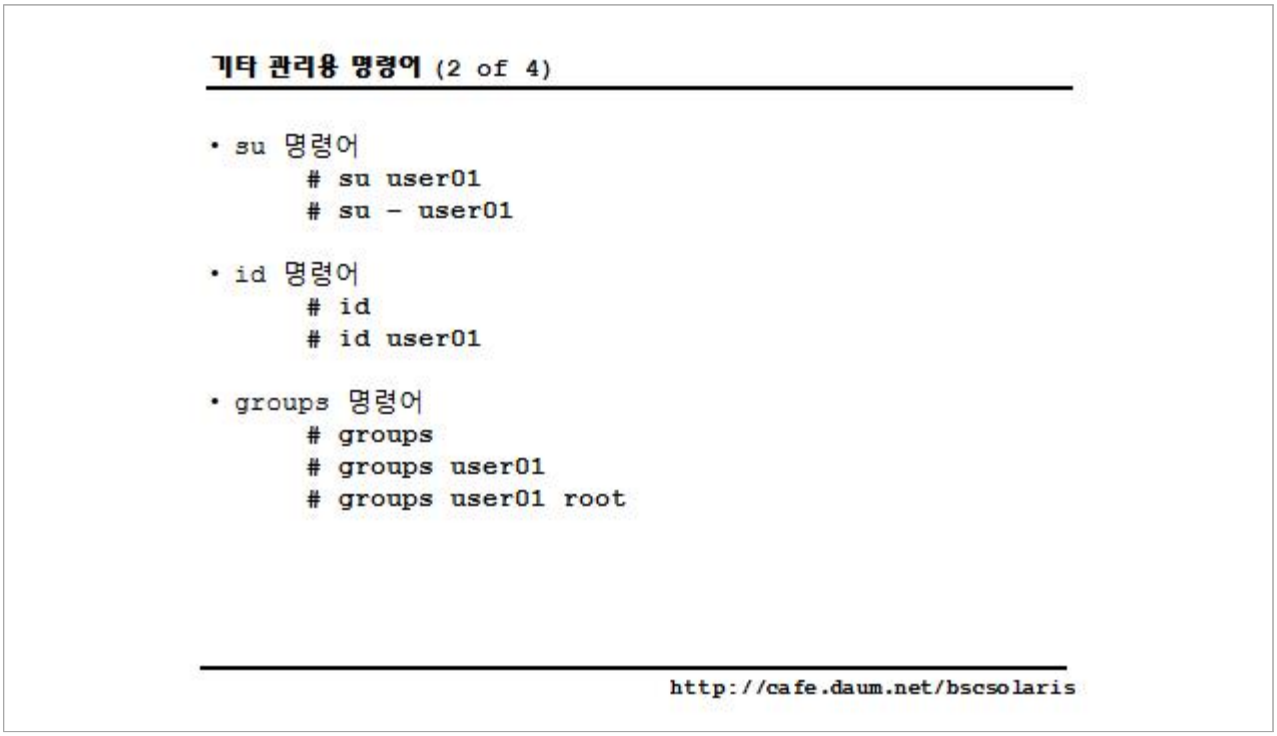
```
[root@server1 ~]# df -k / | tail -1 | awk '{print $5}' > df.count
[root@server1 ~]# cat df.count
```

```
14%
```

```
[root@server1 ~]# df -k / | tail -1 | awk '{print $5}' | awk -F% '{print $1}' > df.count
[root@server1 ~]# cat df.count
```

```
14
```

■ 데이터 수집(Data Gathering)
* 내가 볼수 있는 모든 명령어의 출력 결과
* 내가 볼수 있는 모든 파일에 로그들
    => 데이터 수집



2 su CMD

이름

su - 사용자와 그룹 ID 를 교체하여 쉘을 실행한다.(Switching User)

개요

su [-flmp] [-c 명령] [-s 쉘] [--login] [--fast] [--preserve-environment] [--command=명령] [--shell=쉘] [-] [--help] [--version] [사용자 [인수...]]

설명

이 맨페이지는 GNU 버전의 su 를 설명한다. su 는 한 사용자가 잠시 다른 사용자가 될 수 있도록 해준다. 실제 사용자 ID, 그룹 ID, USER의 보충적인 그 룹 으로 쉘을 실행한다. USER가 주어지지 않으면 기본적으로 수퍼유저인 root 로 설정된다. 실행되는 쉘은 USER의 패스워드 목록에서 찾아오거나 없으 면 /bin/sh 를 수행한다. 만약 USER에 패스워드가 있다면 su 는 실제 사용자 ID 0 (수퍼유저)가 아닌 한 패스워드를 물어온다.

기본적으로, su 는 현재 디렉토리를 변경하지 않는다. USER 의 패스워드 항목 으로부터 'HOME', 'SHELL' 등의 변수를 설정하고 만약 수퍼유저가 아니라면 'USER' 와 'LOGNAME' 을 USER로 설정한다. 기본적으로 이 쉘은 로그인 쉘이 아니다.

만약 한 개 이상의 인수가 주어지면 쉘에 대한 인수로 전달된다.

su는 /bin/sh나 다른 쉘을 특별히 다루지는 않는다. (argv[0]를 "-su"로 하고 -c 를 특정 쉘로 지정하지 않는 한...)

syslog를 가지고 있는 시스템에서는, su 가 실패하는 경우 보고를 하 도 록, 그 리 고 성공의 경우에는 선택적으로 보고하도록컴파일하면 su 가 syslog를 사용한다.

옵션

-c COMMAND, --command=COMMAND
대화형 쉘을 시작하지 않고 -c 옵션을 쉘에 주어서 한 개의 명령만을 수행하도록 한다.

다른 사용자의 권한으로 쉘을 실행한다. 서버에 접속한 상태에서 로그아웃 없이 다른 사용자로 전환할 수 있음 (windows의 사용자 전환 - terminal service (application)를 자동으로 만들어 놓아야 사용자 전환이 실행가능) 원격 접속시에 다른 사용자로 로그인하기 위해 로그아웃하면 접속이 종료가 되는데 접속이 종료 되지 않은 상태에서 다른 사용자로 로그인하고자 할 때 사용한다.

일반사용자가 다른 사용자가 되는 것을 권한이 높아지는 것이기 때문에 전환(Switching) 되는 사용자의 암호를 맞추어야만 전환이 가능하고, root 사용자가 다른사용자로 전환하는 경우에는 권한이 낮아지는 것이기 때문에 암호입력 없이 전환이 가능하다.

■ (사용자 전환의 예)

일반사용자(user01) ----> 다른 일반사용자(user02)
일반사용자(user01) ----> 관리자(root)

관리자(root) ----> 일반사용자(user01)
---

su 명령어 다음에 전환하고 싶은 사용자 이름이 없는 경우 root 사용자로 전환된다. 그리고 su 명령어에 '-' 기호 없이 다른 사용자로 전환하는 경우 지정된 사용자로 전환이 되지만 이전 사용자가 쓰고 있던 export 변수들의 설정이 그대로 따라온다. su 명령어에 '-' 기호를 붙이고 다른 사용자로 전환하는 경우 지정된 사용자로 새로 로그인한 것 처럼 동작을 시켜서 사용자 환경변수로 모두 변경한다.

[명령어 형식]
```
# su    [fedora]    /* fedora 사용자, 이전 사용자 환경변수 */
# su  - [fedora]    /* fedora 사용자, 사용자 환경변수 */
```

[실습] su 명령어 사용법

■ 사용시스템
- server1

■ 실습시나리오
- "su fedora" 형식과 "su - fedora" 형식의 차이점
- 일반사용자(fedora)가 다른 사용자로 전환하는 경우

[EX1] 'su fedora' 형식과 'su - fedora' 형식의 차이점

① (server1) 실습 준비
```
[root@server1 ~]# cd /etc
[root@server1 /etc]# pwd
/etc
```

② (server1) su fedora 형식 테스트 및 확인(root -> fedora)
```
[root@server1 /etc]# su fedora
```
- 사용자 전환이 되었는가?
- 현재 디렉토리 어디인가?
- 환경변수(PATH, PS1) 확인?

```
[fedora@server1 /etc]$ id
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)

[fedora@server1 /etc]$ pwd
/etc

[fedora@server1 /etc]$ echo $PATH
/home/fedora/.local/bin:/home/fedora/bin:/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/root/bin

[fedora@server1 /etc]$ echo $PS1
[\u@\h \W]\$

[fedora@server1 /etc]$ exit
```

③ (server1) 실습 준비
[fedora@server1 /etc]$ cd /etc
[fedora@server1 /etc]$ pwd

```
/etc
```

④ (server1) su - fedora 형식 테스트 및 확인
[root@server1 /etc]# su - fedora
- 사용자 전환이 되었는가?
- 현재 디렉토리 어디인가?
- 환경변수(PATH, PS1) 확인?

[fedora@server1 ~]$ id

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

[fedora@server1 ~]$ pwd

```
/home/fedora
```

[fedora@server1 ~]$ echo $PATH

```
/home/fedora/.local/bin:/home/fedora/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin
```

[fedora@server1 ~]$ echo $PS1

```
[Wu@Wh WW]W$
```

[fedora@server1 ~]$ exit
[root@server1 /etc]#

[비교] (WIN)사용자 전환 vs (*NIX) 사용자 전환

[질문] "ssh fedora@localhost" versus "su - fedora"
[답변] ?

[실무 예] root/oracle 계정 권한을 모두 가지고 있는 경우 root 사용자가 oracle 사용자로 전환할 때
DB(Oralce) -> oracle (X) $ su oracle
---------- (O) $ su - oracle
OS(Linux) -> root

[EX2] 일반 사용자(fedora)가 다른 사용자로 전환하는 경우

■ 원격접속 명령어: telnet, ssh
(telnet 명령어 형식) # telnet 192.168.10.10
(ssh 명령어 형식) # ssh fedora@192.168.10.10

① (server1) ssh 명령어를 사용하여 fedora 사용자로 localhost쪽으로 로그인

[root@server1 ~]# ssh fedora@localhost

```
fedora@localhost's password: (fedora)
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Sun Feb  7 11:00:43 2021
```

② (server1) /etc/shadow 파일 내용 확인 및 에러 메세지 확인
[fedora@server1 ~]$ cat /etc/shadow

```
cat: /etc/shadow: Permission denied
```

- 에러 메세지 확인(원인: 일반사용자에게 /etc/shadow 파일에 대한 퍼미션이 없다.)

[fedora@server1 ~]$ ls -l /etc/shadow

```
----------. 1 root root 1516 Feb  7 10:52 /etc/shadow
```

③ (server1) 권한 상승(root 사용자로 전환)
[fedora@server1 ~]$ su - ($ su - root)

```
Password: (centos)
```

-> root 사용자의 암호 입력

[root@server1 ~]# id

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

④ (server1) 전환된 root 사용자로 /etc/shadow 내용 보기
[root@server1 ~]# cat /etc/shadow

```
.... (중략) ....
fedora:$6$BY.t4HV0fvZ0RnXo$BZU60Bd8W6PYgkNEI7mv01AwCgVfkBYbcMVx/95ikWzPJdnH89GbP6dGP1NTnxI2JRZFr.IcUXcR
6pCuQSvvW/::0:99999:7:::
user01:$6$LrnVPI7WYcLK0YG6$H5t30nhj0Gj0JkuIzMze8acBYTncuM.CzdbYM6wHauXwbiThQj0o9pc.quqqR3Wtjo0K.eb418av
3NuZGIG17.:18665:0:99999:7:::
apache:!!:18665::::::
```

[root@server1 ~]# exit
[fedora@server1 ~]$ exit

3 sudo CMD

```
NAME
     sudo(Super User Do), sudoedit - execute a command as another user

SYNOPSIS
     sudo -h | -K | -k | -V
     sudo -v [-ABknS] [-g group] [-h host] [-p prompt] [-u user]
     sudo -l [-ABknS] [-g group] [-h host] [-p prompt] [-U user] [-u user]
          [command]
     sudo [-ABbEHnPS] [-C num] [-g group] [-h host] [-p prompt] [-r role]
          [-t type] [-T timeout] [-u user] [VAR=value] [-i | -s] [command]
     sudoedit [-ABknS] [-C num] [-g group] [-h host] [-p prompt] [-T timeout]
          [-u user] file ...

DESCRIPTION
     sudo allows a permitted user to execute a command as the superuser or
     another user, as specified by the security policy.  The invoking user's
     real (not effective) user-ID is used to determine the user name with
     which to query the security policy.

     sudo supports a plugin architecture for security policies and input/out-
     put logging.  Third parties can develop and distribute their own policy
     and I/O logging plugins to work seamlessly with the sudo front end.  The
     default security policy is sudoers, which is configured via the file
     /etc/sudoers, or via LDAP.  See the Plugins section for more information.

OPTIONS
     -i, --login
                 Run the shell specified by the target user's password data-
                 base entry as a login shell.  This means that login-specific
                 resource files such as .profile, .bash_profile or .login will
                 be read by the shell.  If a command is specified, it is
                 passed to the shell for execution via the shell's -c option.
                 If no command is specified, an interactive shell is executed.
                 sudo attempts to change to that user's home directory before
                 running the shell.  The command is run with an environment
                 similar to the one a user would receive at log in.  Note that
                 most shells behave differently when a command is specified as
                 compared to an interactive session; consult the shell's man-
                 ual for details.  The Command environment section in the
                 sudoers(5) manual documents how the -i option affects the
                 environment in which a command is run when the sudoers policy
                 is in use.

     -l, --list  If no command is specified, list the allowed (and forbidden)
                 commands for the invoking user (or the user specified by the
                 -U option) on the current host.  A longer list format is used
                 if this option is specified multiple times and the security
                 policy supports a verbose output format.

                 If a command is specified and is permitted by the security
                 policy, the fully-qualified path to the command is displayed
                 along with any command line arguments.  If a command is spec-
                 ified but not allowed by the policy, sudo will exit with a
                 status value of 1.
```

■ sudo CMD(superuser do) => **(WIN)** 관리자 권한으로 실행(관리자 권한으로 실행)

- /etc/sudoers (/etc/sudoers.d/*)
- wheel 그룹(CentOS 6.x 이하, CentOS 7.x 이상)

su CMD versus **sudo CMD**

- **$ su - root**
- **$ sudo CMD**
  (ㄱ) root 암호를 공유할 필요가 없다. -> 사용자 암호 입력
  (ㄴ) 사용자가 관리자권한으로 수행하는 사용할 수 있는 명령어의 제한을 둘수 있다. -> /etc/sudoers
  (ㄷ) 사용자가 sudo CMD 수행할 때 마다 기록에 남는다. => /var/log/secure

[명령어 형식]

```
$ sudo CMD              ($ sudo su - root -c CMD)
$ sudo -l               /* 자신에게 적용되는 sudo 권한 보기 */
$ sudo -l -U fedora     /* 특정 사용자에게 적용되는 sudo 권한 보기 */
```

```
$ sudo -i              ("$ su - root" 비슷/유사)

$ visudo
$ visudo -c
```

[실습] sudo 명령어 사용법

■ 사용시스템
- server1

■ 작업시나리오
- fedora 사용자로 sudo 명령어 수행하기
- sudo 명령어를 사용하여 관리자 되기

[EX] 일반 사용자(fedora)가 sudo 명령어 실행 하기
- 일반사용자가 sudo 명령을 사용하여 관리자가 할 수 있는 작업을 진행해 본다.

① (server1) fedora 사용자로 로그인
```
[root@server1 ~]# ssh fedora@localhost
```
fedora 사용자로 로그인

② (server1) fedora 사용자로 /etc/shadow 파일 확인
```
[fedora@server1 ~]$ cat /etc/shadow
```
```
cat: /etc/shadow: Permission denied
```

③ (server1) sudo 명령어를 사용하여 fedora 사용자가 수행할 수 있는 명령어 확인
```
$ sudo -l
```
```
We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.

[sudo] password for fedora: (fedora)
Matching Defaults entries for fedora on server1:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_reset, env_keep="COLORS DISPLAY HOSTNAME HISTSIZE KDEDIR LS_COLORS",
    env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRESS LC_CTYPE",
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES",
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    env_keep+="LC_TIME LC_ALL LANGUAGE LINGUAS _XKB_CHARSET XAUTHORITY",
    secure_path=/sbinW:/binW:/usr/sbinW:/usr/bin

User fedora may run the following commands on server1:
    (ALL) ALL
```

[참고] sudo -l 명령 수행시 에러가 나는 경우
```
[TERM2] # usermod -aG wheel fedora      /* -a: add, -G: secondary group */
```

④ (server1) sudo 명령어를 사용하여 /etc/shadow 파일 보기
```
[fedora@server1 ~]$ sudo cat /etc/shadow
```
```
..... (중략) .....
fedora:$6$BY.t4HV0fvZ0RnXo$BZU60Bd8W6PYgkNEl7mv01AwCgVfkBYbcMVx/95ikWzPJdnH89GbP6dGP1NTnxl2JRZFr.lcUXcR
6pCuQSvvW/::0:99999:7:::
user01:$6$LrnVPl7WYcLK0YG6$H5t30nhj0Gj0JkulzMze8acBYTncuM.CzdbYM6wHauXwbiThQj0o9pc.quqqR3Wtjo0K.eb418av
3NuZGIG17.:18665:0:99999:7:::
apache:!!:18665::::::
```

⑤ (server1) nmcli 명령어를 사용하여 자신의 connection id를 up 하기
```
[fedora@server1 ~]$ nmcli connection
```
```
NAME    UUID                                  TYPE      DEVICE
eth0    e0bde260-0869-49e2-8297-9449eb4a37c5  ethernet  ens33
virbr0  cbb0119a-e061-4d85-a9b1-a3a1e4b4a865  bridge    virbr0
```
- connection 이름(ex: ens33)은 위의 내용과 틀릴수도 있다. 자신의 출력 내용을 사용한다.

```
[fedora@server1 ~]$ nmcli connection up eth0
```
```
Error: Connection activation failed: Not authorized to control networking.
```

⑥ (server1) sudo 명령어를 사용하여 connection id를 up 하기
[fedora@server1 ~]$ sudo nmcli connection up eth0

```
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/4)
```

⑦ (server1) fedora 사용자 로그아웃
[fedora@server1 ~]$ exit

```
logout
Connection to localhost closed.
```

[root@server1 ~]#

[추가적인 실습] /etc/sudoers 파일에 존재하지 않는 **user01 사용자**로 sudo 명령어 사용
 # su - user01
$ cat /etc/shadow
$ sudo -l
-> Sorry, user user01 may not run sudo on server1.
$ sudo cat /etc/shadow
-> user01 is not in the sudoers file. This incident will be reported.
$ exit

[EX2] sudo 명령어를 사용하여 관리자 되기
- 일반 사용자가 sudo 명령을 사용하여 관리자로 전환하는 작업을 진행한다.

■ fedora 사용자 root 사용자로 전환(sudo 명령어 사용)
$ sudo -i
$ sudo su - root

① (server1) fedora 사용자로 로그인
[root@server1 ~]# ssh fedora@localhost
fedora 사용자로 로그인

② (server1) 관리자로 전환하기
[fedora@server1 ~]$ sudo su -

```
[sudo] password for fedora: (fedora)
```

[root@server1 ~]# id

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

[root@server1 ~]# exit

```
logout
```

③ (server1) 관리자로 전환하기
[fedora@server1 ~]$ sudo -i
[root@server1 ~]# id

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

[root@server1 ~]# exit

```
logout
```

[fedora@server1 ~]$ exit

```
logout
Connection to localhost closed.
```

[root@server1 ~]#

[참고] fedora 사용자를 **[wheel 그룹]**에 속하도록 하기
- OS 설치시에 관리자 권한을 fedora 사용자에게 부여하지 않은 경우에는 다음과 같이 작업을 진행할 수 있다.
- CentOS 7.x 이하 wheel 그룹
  CentOS 8.x 이상 wheel 그룹 -> 관리자(sudo CMD)

(사용자 추가) **# useradd -G wheel fedora**
(사용자 변경) **# usermod -aG wheel fedora**
-a : add
-G : secondary group

```
# id fedora user01
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)
uid=1001(user01) gid=1001(user01) groups=1001(user01)
# usermod -aG wheel user01
# id fedora user01
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)
uid=1001(user01) gid=1001(user01) groups=1001(user01),10(wheel)
# usermod -G "" user01
# id fedora user01
```

[참고] /etc/sudoers(/etc/sudoers.d/*) 파일 설정 실습
 # cat /etc/sudoers

```
# 다음 내용은 /etc/sudoers 파일의 내용 중 중요한 핵심 내용들에 대한 예이다.

# sudo 명령 수행시 PATH 변수 설정이다. 설정된 위치의 명령만 명령 위치를 생략할 수 있다.
Defaults    secure_path = /sbin:/bin:/usr/sbin:/usr/bin

#
# /etc/sudoers 파일 형식
# 형식:     <who>      <hosts>=(<runas_users>:<runas_groups>)  <commands>
# 형식: <사용자/그룹>  <호스트>=(<대상사용자>:<대상그룹>)      <허용 명령어들>
# * <hosts> - 호스트(Host_Alias) 필드
# * <대상사용자>:<대상그룹> - 어떤 사용자:그룹으로 명령을 실행할 수 있는지를 지정하는 필드
# * <허용명령들> - 실행할려고 하는 명령 필드
# user01  myhost=(ansible)   systemctl

# root 사용자는 sudo 명령을 통해 모든 명령 수행이 가능하다. 명령 수행시 암호는 입력해야 한다.
root    ALL=(ALL)    ALL

# wheel 그룹에 속한 모든 사용자는 sudo 명령을 통해 모든 명령 수행이 가능하다.
%wheel   ALL=(ALL)    ALL

# wheel 그룹에 속한 모든 사용자는 sudo 명령을 통해 모든 명령 수행이 가능하며 암호 입력도 하지 않는다.
# %wheel ALL=(ALL)    NOPASSWD: ALL

# /etc/sudoers 파일 외에 /etc/sudoers.d 디렉토리 안의 파일도 설정 파일에 포함한다.
#includedir /etc/sudoers.d
```

```
# 다음 내용은 /etc/sudoers 파일에 들어 갈수 있는 추가적인 예이다.

# user01 사용자는 준 관리자기 되며, 암호 입력 없이 sudo 명령 수행이 가능하다.
user01   ALL=(ALL)   NOPASSWD: ALL

# user02 사용자는 2개의 명령 사용이 가능하다.
user02   ALL=(ALL)   /usr/bin/systemctl restart httpd, /usr/bin/systemctl stop

# user03 사용자는 backup.sh 스크립트 실행이 가능하다.
user03   ALL=(ALL)   /root/bin/backup.sh
```

4 id CMD

```
이름
       id - 실제, 유효 UID와 GID를 출력한다.

개요
       id  [-gnruG]  [--group]  [--name] [--real] [--user] [--groups] [--help]
       [--version] [username]

설명
       이 맨페이지는 GNU 버전의 id 를 다룬다.  id 는 주어진 사용자, 또는 사 용
       자가 주어지지 않는 경우 프로세스의 주인에 대한정보를 출력한다. 기본적으
       로 실제 사용자 ID, 실제 그룹 ID, 만약 실제 사용자 ID와 다르다면 유효 사
       용 자  ID를, 마찬가지로 실제 그룹 ID와 다르다면 유효 그룹 ID를 출력하고
       추가 그룹 ID를 출력한다. 각 항목은 식별 문자 그리고 괄호 안에해당 하 는
       사용자 또는 그룹명으로 표현된다.

       id 에 옵션을 주면 위에서 열거한 정보의 일부만 보여준다.

   옵션
       -u, --user
              effeictive user id만 출력

       -g, --group
              오로지 그룹 ID만 출력한다.

       -G, --groups
              추가 그룹만 출력한다.

       --help 표준출력으로 사용법을 출력하고 정상적으로 종료한다.

       -n, --name
              ID 번호 대신 사용자명, 그룹명을 출력한다.  -u, -g, 또는 -G 를 필
              요로 한다.
```

사용자의 uid, gid, groups을 보여줌 (사용자 이름의 길이가 긴 경우 메모리 기억공간의 낭비가 심함. but 정수로 사용자의 이름을 구분하는 경우 메모리 기억공간을 효율적으로 사용가능. 정수로 사용하여 4byte씩 사용될 경우 0 ~ 4294967295명까지 구분이 가능)

[명령어 형식]

```
# id
# id -un            /* -u : user id, -n : name */
# id -gn            /* -g : group id */
# id -Gn            /* -G : Secondary Group ID */
# id -un fedora
# id -gn fedora
# id -Gn fedora
```

[명령어 옵션]

옵션	설 명
-g	기본 그룹의 gid를 출력 (사용자가 소속되어 있는 기본 그룹)
-G	사용자가 속한 모든 그룹의 gid를 출력
-u	사용자의 uid를 출력
-n	-u과 함께 사용하여 숫자 대신 이름 출력

[실습] id 명령어 사용법

■ 사용시스템
- server1

■ 실습 시나리오
- id 명령어 기본 사용법

[EX] id 명령어 기본 사용법

① (server1) id 명령어 수행하기
[root@server1 ~]# id

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

출력화면 해석

- uid=0(root) : 사용자 UID/이름
- gid=0(root) : 사용자 Primary Group GID/이름
- groups=0(root) : 사용자가 속한 그룹 정보(Primary/Secondary Group)

---

② (server1) UID 표시하기
[root@server1 ~]# id -u

```
0
```

③ (server1) 사용자 이름 표시하기
[root@server1 ~]# id -un

```
root
```

④ (server1) GID 표시하기
[root@server1 ~]# id -g

```
0
```

⑤ (server1) 주 그룹 이름만 표시하기
[root@server1 ~]# id -gn

```
root
```

⑥ (server1) 부 그룹 이름들을 표시하기
[root@server1 ~]# id -Gn

```
root
```

⑦ (server1) fedora 사용자 정보 확인
[root@server1 ~]# id fedora

```
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)
```

⑧ (server1) fedora 사용자 UID 표시하기
[root@server1 ~]# id -u fedora

```
1000
```

⑨ (server1) fedora 사용자 이름 표시하기
[root@server1 ~]# id -un fedora

```
fedora
```

⑩ (server1) fedora 사용자 주 그룹 GID 표시하기
[root@server1 ~]# id -g fedora

```
1000
```

⑪ (server1) fedora 사용자 주 그룹 이름만 표시하기
```
[root@server1 ~]# id -gn fedora
```
```
fedora
```

⑫ (server1) fedora 사용자 부 그룹 이름들을 표시하기
```
[root@server1 ~]# id -Gn fedora
```
```
fedora wheel
```

5 groups CMD

```
이름
       groups - 사용자가 속한 그룹들을 출력한다.

개요
       groups [사용자명...]
       groups {--help,--version}

설명
       이   맨페이지는 GNU 버전의 groups 를 다룬다.  groups 는 주어진 각 user-
       name 또는 프로세스가 속한 추가 그룹의 이름을 출력해준다.  만약 사용자명
       이 주어졌다면 각 사용자명이 소속된 그룹 목록 앞에 표시된다.

       그룹 목록은  'id -Gn' 의 결과와 같다.
```

사용자가 속한 그룹의 정보를 보여준다. 일반적으로 id 명령어로 확인하는 편이고 gruops 명령어는 명령어 라인상에서는 자주 사용되는 명령어는 아니다.

[명령어 형식]

```
# groups                    /* 현재 사용중인 사용자의 그룹을 보여줌 */
# groups user1              /* user1의 그룹을 보여줌 */
# groups user1 user2        /* user1, user2의 그룹을 보여줌 */
```

[실습] groups 명령어 사용법

■ 사용시스템
- server1

■ 실습시나리오
- groups 명령어 간단한 실습

[EX1] groups 명령어 간단한 실습

① (server1) 현재 사용자의 모든 그룹 표시 하기

[root@server1 ~]# groups

```
root
```

② (server1) fedora 사용자의 모든 그룹 표시 하기

[root@server1 ~]# groups fedora

```
fedora : fedora wheel
```

③ (server1) fedora, root 사용자의 모든 그룹 표시하기

[root@server1 ~]# groups fedora root

```
fedora : fedora wheel
root : root
```

④ (server1) /etc/passwd, /etc/group 파일에 대한 정보 확인
- 사용자의 Primary group 정보는 /etc/passwd 파일의 4번째 필드에 존재한다.
- 사용자의 Secondary group 정보는 /etc/group 팡일의 4번째 필드에 존재한다.

[root@server1 ~]# cat /etc/passwd | egrep '^root|^fedora'

```
root:x:0:0:root:/root:/bin/bash
fedora:x:1000:1000:fedora:/home/fedora:/bin/bash
```

[root@server1 ~]# cat /etc/group | egrep wheel

```
wheel:x:10:fedora
```

6 last CMD

```
NAME
       last, lastb - 사용자들의 마지막 로그인했는 기록 목록을 보여준다.

SYNOPSIS
       last [-R] [-num] [ -n num ] [-adx] [ -f file ] [name...]  [tty...]
       lastb [-R] [-num] [ -n num ] [ -f file ] [-adx] [name...]  [tty...]

DESCRIPTION
       Last  명령은 /var/log/wtmp 파일이나, -f 옵션에서 지정한파일을 통해서 사
       용자의 로그인, 로그아웃 시간을 보여준다.  특정 사용자의 이름이나, tty를
       지정할 수 있으며, 이렇게 지정되면, 해당 사용자의 로그인 정보만을 보여준
       다.  tty의 이름은 생략될 수 있어, last 0 명령은 last tty0 명령과  같 은
       기능을 한다.  last가 실행 중에 SIGINT(인터럽트 글쇠, 주로 Ctrl-C에 의해
       서 만들어지는 신호)가 감지되거나, SIGQUIT(마침 글쇠, 주로 Ctrl-W에 의해
       서 만들어지는 신호)가감지되면, 실행을 마친다.

       접속 사용자 이름에 reboot라고 나오는 것은 시스템이 reboot된 것을 나타낸
       다. 즉 last reboot 명령으로기록 파일이 만들어진 후부터 reboot 되었는 기
       록을 모두 볼 수 있다.

       Lastb  명령은 접속 실패를 기록하는 파일인 /var/log/btmp 파일을 대상으로
       한다는 것을 제외하고는 last 명령과 같다.

NOTES
       wtmp  파일이나, btmp 파일이 없을 수도 있다.  시스템은 이런 파일이 있을
       경우에만, 로그인 정보를 그 파일에 기록한다.  만약 없는데, 이제부터 last
       명 령 을 사용하고 싶다면, 단지 간단하게 touch 명령을 사용해서 각 파일을
       만들어 놓으면 된다.  (예를 들면, touch /var/log/wtmp).
```

서버를 이용하는 각 계정사용자들의 로그인 정보를 보여주는 명령어이다. 흔히 관리자는 각 계정별로 서버에 접속한 시간과 IP주소 등을 확인해야 할 경우가 있다. 또한 특정 계정의 서버 접속정보를 확인해야 할 때 에도 마찬가지이다. 다양한 방법으로 사용자들의 로그인정보를 조사한다.

[명령어 형식]

```
# last
# last root
# last -5              /* 5행의 결과만을 확인 */
```

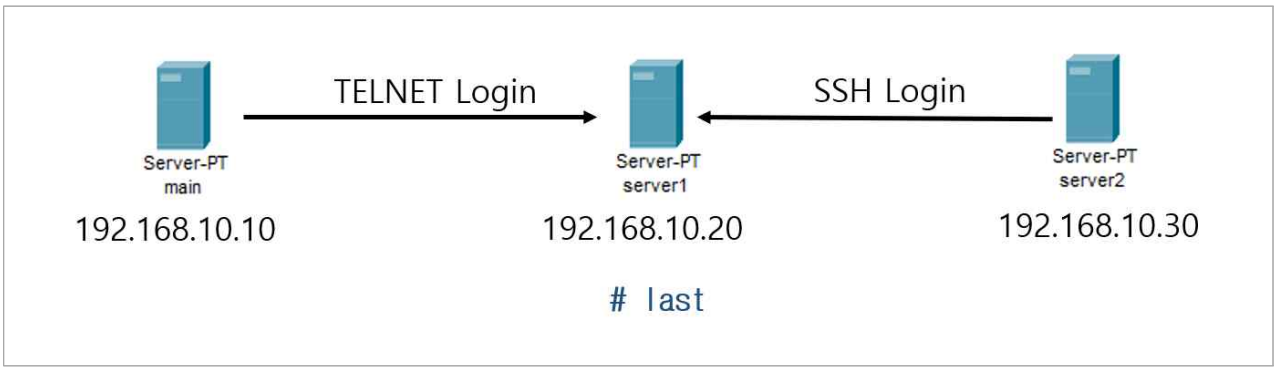
[명령어 옵션]

옵션	설 명
-n	(num) 지정한 num 만큼의 줄만 보여준다.
-f	(file) 지정한 파일에서 정보를 불러온다.
-R	보여주는 목록에서 hostname(IP주소)필드는 보여주지 않는다.
-a	보여주는 목록에서 hostname(IP주소)필드를 마지막 필드에 보여준다.
-d	다른 호스트에서 접속한 것만 보여준다.
-x	shutdown이 일어난 상태나, run level이 바뀐 상태를 보여준다.

[실습] last 명령어 사용법

■ 사용 시스템
- main
- server1
- server2

■ 실습 시나리오
- last 명령어가 사용되는 경우의 예에 대해 실습해 본다. main/server2 시스템에서 server1 시스템으로 원격 접속을 하고, server1에서 last 명령어를 통해 접속된 정보를 확인하고 분석한다.



[EX1] last 명령어 사용

① (main) main 서버에서 server1 서버로 telnet 명령어를 통해 접속(fedora 사용자로 로그인)

```
[root@main ~]# telnet 192.168.10.20
```

fedora 사용자로 로그인

```
$ tty
```

-> 터미널 확인

② (server2) server2서버에서 server1 서버로 ssh 명령어를 통해 접속(root 사용자로 로그인)

```
[root@server2 ~]# ssh 192.168.10.20
```

root 사용자로 로그인

```
[root@server1 ~]# tty
```

-> 터미널 확인

③ (server1) last 명령어를 통해 접속된 사용자 정보 확인
[root@server1 ~]# last -i

- -i, --ip display IP numbers in numbers-and-dots notation

```
root     pts/2        192.168.10.30    Tue Mar 30 14:16   still logged in
fedora   pts/1        192.168.10.10    Tue Mar 30 14:16   still logged in
fedora   pts/1        ::1              Wed Mar 31 14:54 - 14:55  (00:01)
fedora❶ pts/1❷       ::1 ❸            Wed Mar 31 14:49 - 14:53❹ (00:03)❺
user01   pts/4        0.0.0.0          Wed Mar 31 13:04 - 13:04  (00:00)
root     pts/2        0.0.0.0          Wed Mar 31 13:04 - 13:09  (00:05)
..... (중략) .....

wtmp begins Tue Mar 30 14:16:24 2021
```

명령어 출력 결과 해석

```
❶ fedora                        : 로그인 사용자 이름
❷ pts/1                         : 로그인시 할당 받은 터미널 번호
❸ ::1                           : 원격 호스트(IP)
❹ Wed Mar 31 14:49 - 14:53      : 로그인 시간 - 로그아웃 시간
❺ (00:03)                       : 시스템 사용시간(시:분)
```

[root@server1 ~]# last -i -5 (# last | head -5)

```
root     pts/2        192.168.10.30    Tue Mar 30 14:16   still logged in
fedora   pts/1        192.168.10.10    Tue Mar 30 14:16   still logged in
fedora   pts/1        ::1              Wed Mar 31 14:54 - 14:55  (00:01)
fedora   pts/1        ::1              Wed Mar 31 14:49 - 14:53  (00:03)
user01   pts/4        0.0.0.0          Wed Mar 31 13:04 - 13:04  (00:00)

wtmp begins Tue Mar 30 14:16:24 2021
```

(복원) 다른 서버에서 접속한 모든 connection을 해제한다.

[EX2] /var/log/wtmp.* 파일 읽기

① /var/log/wtmp 파일 종류 확인
[root@server1 ~]# file /var/log/wtmp /* data 파일 타입을 가지고 있다 */

```
/var/log/wtmp: data
```

[root@server1 ~]# file /etc/passwd /* text 파일의 경우에 확인 가능 */

```
/etc/passwd: ASCII text
```

② /var/log/wtmp 파일 내용 확인
[root@server1 ~]# cat /var/log/wtmp /* 파일을 열수 있는 형식 아님 */

```
    ptspts/11fedora::ffff:192.168.10.10(b`
                                        

h ptspts/2ts/2root192.168.10.308b`p=
...... (중략) ......
```

③ last -f 옵션을 사용하여 지정된 파일의 내용 읽어들이기
[root@server1 ~]# last -f /var/log/wtmp (# last)

```
root     pts/1        192.168.0.1      Wed Feb 10 15:47   still logged in
root     pts/1        192.168.0.1      Wed Feb 10 12:56 - 14:40  (01:44)
root     :0                            Wed Feb 10 12:44   still logged in
root     :0                            Wed Feb 10 12:44 - 12:44  (00:00)
root     pts/4        linux200         Wed Feb 10 12:43 - 12:56  (00:12)
root     pts/3        linux200         Wed Feb 10 12:32 - 12:56  (00:23)
root     pts/2        192.168.0.1      Wed Feb 10 11:53 - 15:12  (03:18)
fedora   pts/2        192.168.0.1      Wed Feb 10 10:27 - 11:33  (01:05)
root     pts/1        192.168.0.1      Wed Feb 10 10:25 - 12:56  (02:30)
```

- -f, --file <file> use a specific file instead of /var/log/wtmp

[참고] last -f 옵션 사용하는 경우

/var/log/wtmp 파일의 백업 파일이 존재하는 경우, 백업 파일 읽을 때 -f 옵션을 사용한다. wtmp 파일은 사용자들의 로그인/로그아웃 기록들에 대해 최근 한달간의 누적본이 남겨져 있다. 그리고 기본적으로 이전달의 기록 백업(wtmp-날짜)본이 남겨져 있다. 백업본은 스케줄러에의해 자동으로 수행이 된다.

(이번달 기록) last
(지난달 기록) last -f /var/log/wtmp-(백업날짜)

```
# cd /var/log
# ls wtmp*
wtmp  wtmp-20210330

# last       (# last -f /var/log/wtmp)
# last -f /var/log/wtmp-20210330
* (형식) last -f (읽고 싶은 wtmp 파일 이름)
```

[EX3] last 명령어의 사용 예

■ 실습 시스템
- server1
- server2

■ 실습 시나리오

(개발자 요청 내용) 어제 파일(예: file.log)을 삭제한 사용자를 검색해 달라.

(정보1) 어제 파일이 지워졌다.
```
# last | grep 'Jun  8'
```
(정보2) 지워진 파일의 이름 : file.log
```
# cat ~/.bash_history
# cat ~/.bash_history | grep 'file.log' | grep rm
```

① (server1) 작업 준비
```
[root@server1 ~]# cd /test
[root@server1 /test]# chmod 777 /test
[root@server1 /test]# touch file.log
```

② (server2) fedora 사용자로 server1으로 로그인
```
[root@server2 ~]# telnet 192.168.10.20
```
fedora 사용자로 로그인

③ (server2) fedora 사용자로 중요한 파일을 지우기
```
[fedora@server1 ~]$ rm -f /test/file.log
[fedora@server1 ~]$ ls /test
```
-> file.log 파일이 지워졌는지 확인

④ (server2) 로그 아웃
```
[fedora@server1 ~]$ exit
```

>>>> 오늘 fedora 사용자가 중요한 파일(EX: /test/file.log)을 지웠다. <<<<

⑤ (server1) 오늘 로그인/로그아웃 했던 사용자들 파악
```
[root@server1 ~]# last | grep 'Sep  9'
```
```
fedora   pts/4        linux249         Fri Sep  9 10:29 - 10:30  (00:00)
root     pts/3        :0.0             Fri Sep  9 10:11   still logged in
fedora   pts/2        linux249         Fri Sep  9 10:11   still logged in
root     pts/2        linux249         Fri Sep  9 10:10 - 10:10  (00:00)
root     pts/1        :0.0             Fri Sep  9 10:00   still logged in
root     pts/1        :0.0             Fri Sep  9 09:59 - 10:00  (00:00)
root     :0                            Fri Sep  9 09:59   still logged in
root     :0                            Fri Sep  9 09:59 - 09:59  (00:00)
reboot   system boot  2.6.18-164.el5   Fri Sep  9 09:56          (00:35)
```

```
[root@server1 ~]# egrep -l "file.log" /home/*/.bash_history
```
```
/home/fedora/.bash_history
```
* -l, --files-with-matches print only names of FILEs with selected lines

```
[root@server1 ~]# egrep "file.log" /home/fedora/.bash_history
```
```
rm -f /test/file.log
```

⑥ 결론
- 오늘 로그인 한 사용자에 fedora 사용자가 존재하고, fedora 사용자의 홈디렉토안의 히스토리 파일(~/.bash_history)에 rm 명령어를 통해 file.log 파일을 지운 기록이 존재 하였기 때문에, fedora 범인인 가능성이 아주 높다고 판단이 된다.

6 lastlog CMD

```
NAME
       lastlog - reports the most recent login of all users or
       of a given user

SYNOPSIS
       lastlog [options]

DESCRIPTION
       lastlog formats and prints the contents of the last
       login log /var/log/lastlog file. The login-name, port,
       and last login time will be printed. The default (no
       flags) causes lastlog entries to be printed, sorted by
       their order in /etc/passwd.

OPTIONS
       -t, --time DAYS
          Print the lastlog records more recent than DAYS.

       -u, --user LOGIN
          Print the lastlog record for user with specified
          LOGIN only.

       The -t flag overrides the use of -u.

       If the user has never logged in the message ** Never
       logged in** will be displayed instead of the port and
       time.

NOTE
       The lastlog file is a database which contains info on
       the last login of each user. You should not rotate it.
       It is a sparse file, so its size on the disk is usually
       much smaller than the one shown by "ls -l" (which can
       indicate a really big file if you have in passwd users
       with a high UID). You can display its real size with "ls
       -s".

FILES
       /var/log/lastlog
          Database times of previous user logins.
```

사용자의 마지막 로그인 정보만을 출력 해준다. lastlog는 /var/log/lastlog라는 파일의 내용을 출력해 준다. 로그인할 때 마지막 로그인 정보가 출력되는데 이때 출력되는 정보가 /var/log/lastlog의 정보이다. 사용자 계정을 지우게 되면 /var/log/lastlog 파일에도 사용자계정에 해당 정보도 삭제된다.

[명령어 형식]

```
# lastlog
# lastlog -u feodra    /* 지정한 로그인명의 lastlog 정보만을 보여준다. */
# lastlog -t 3         /* 지정한 날짜기간 안에 로그인한 정보만을 보여준다 */
```

[명령어 옵션]

옵션	설 명
-u	지정된 사용자의 lastlog 기록을 보여줌
-t	지정된 날짜기간 안의 로그인 정보만 출력해 준다.

[실습] lastlog 명령어 기본 사용법

■ 사용시스템
- server1

■ 실습시나리오
- lastlog 명령어에 출력 되는 내용을 분석해 본다.

[EX1] lastlog 명령어 사용법

■ 원격 로그인(telnet 사용)시 이전 로그인 기록 확인 화면
```
# telnet localhost (# ssh localhost)
Kernel 4.18.0-240.el8.x86_64 on an x86_64
server1 login: root
Password: (centos)
Last login: Tue Mar 30 15:00:58 on tty2
# exit
```

① lastlog 명령 실행
```
[root@server1 ~]# lastlog
```

```
Username         Port     From             Latest
root             pts/2    linux249         Fri Sep  9 10:41:24 +0900 2011
bin                                        **Never logged in**
daemon                                     **Never logged in**
adm                                        **Never logged in**
lp                                         **Never logged in**
sync                                       **Never logged in**
shutdown                                   **Never logged in**
halt                                       **Never logged in**
mail                                       **Never logged in**

..... (중략) .....

rpcuser                                    **Never logged in**
named                                      **Never logged in**
hsqldb                                     **Never logged in**
sshd                                       **Never logged in**
haldaemon                                  **Never logged in**
avahi-autoipd                              **Never logged in**
xfs                                        **Never logged in**
gdm                                        **Never logged in**
sabayon                                    **Never logged in**
fedora           pts/4    linux249         Fri Sep  9 10:29:54 +0900 2011
user01           pts/3    linux249         Thu Sep  8 10:26:39 +0900 2011
```

```
[root@server1 ~]# lastlog -u fedora
```

```
Username         Port     From             Latest
fedora           pts/2    192.168.0.1      Wed Feb 10 10:27:47 +0900 2010
```

② lastlog 명령에 대한 사용 사례

lastlog 명령 사용 예
- 휴면 계정을 찾을 때 사용할 수 있다.(오랫 동안 로그인 하지 않은 사용자)
- 로그인하면 안되는 사용자의 로그 기록을 확인할 수 있다.(해킹당한 계정 확인)

7 lastb CMD

```
NAME
       last, lastb - 사용자들의 마지막 로그인했는 기록 목록을 보여준다.

SYNOPSIS
       last [-R] [-num] [ -n num ] [-adx] [ -f file ] [name...] [tty...]
       lastb [-R] [-num] [ -n num ] [ -f file ] [-adx] [name...] [tty...]

DESCRIPTION
       Last  명령은 /var/log/wtmp 파일이나, -f 옵션에서 지정한파일을 통해서 사
       용자의 로그인, 로그아웃 시간을 보여준다.  특정 사용자의 이름이나, tty를
       지정할 수 있으며, 이렇게 지정되면, 해당 사용자의 로그인 정보만을 보여준
       다.  tty의 이름은 생략될 수 있어, last 0 명령은 last tty0 명령과  같 은
       기능을 한다.  last가 실행 중에 SIGINT(인터럽트 글쇠, 주로 Ctrl-C에 의해
       서 만들어지는 신호)가 감지되거나, SIGQUIT(마침 글쇠, 주로 Ctrl-W에 의해
       서 만들어지는 신호)가감지되면, 실행을 마친다.

       접속 사용자 이름에 reboot라고 나오는 것은 시스템이 reboot된 것을 나타낸
       다. 즉 last reboot 명령으로기록 파일이 만들어진 후부터 reboot 되었는 기
       록을 모두 볼 수 있다.

       lastb  명령은 접속 실패를 기록하는 파일인 /var/log/btmp 파일을 대상으로
       한다는 것을 제외하고는 last 명령과 같다.
```

접속 실패 로그를 출력해준다. /var/log/btmp파일에 로그가 저장된다. cat이라는 명령어를 통해서 안의 내용을 열어보면 파일의 내용이 제대로 나오지 않는다. 이 파일의 내용을 보기 위해서 우리가 사용하는 명령어가 바로 lastb라는 명령어다. 보통 블루투스 공격(임의의 사용자 계정과 임의의 패스워드로 내 시스템에 접속하려는 것)을 당하는지, 당했는지와 어떤 사용자가 잘못 로그인을 했는지와 같은 정보를 확인 할 수 있다.

[명령어 형식]

```
# lastb
```

[명령어 옵션]

옵션	설 명
-n	(num) 지정한 num 만큼의 줄만 보여준다.
-f	(file) 지정한 파일에서 정보를 불러온다.
-R	보여주는 목록에서 hostname(IP주소)필드는 보여주지 않는다.
-x	shutdown이 일어난 상태나, run level이 바뀐 상태를 보여준다.

[실습] lastb 명령어 실습

■ 실습시스템
- server1
- server2

■ 실습시나리오
- lastb 명령어에 출력 결과를 분석한다.

[EX1] lastb 명령어 실습

(전제 조건) user01 사용자가 존재해야 한다.

```
# cat /etc/passwd | grep user01
# useradd user01
# passwd user01
```

① (server2) server2 시스템에서 server1 시스템으로 ssh 명령어를 사용하여 user01 사용자로 로그인
- server2 시스템에서 server1 시스템으로 ssh 명령어를 사용하여 user01 사용자로 로그인시 여러번의 로그인 실패를 하도록 유도한다.

```
[root@server2 ~]# ssh user01@192.168.10.20
```

```
user01@192.168.10.20's password: (1)     <--- 잘못된 암호입력
Permission denied, please try again.
user01@192.168.10.20's password: (2)
Permission denied, please try again.
user01@192.168.10.20's password: (3)
user01@192.168.10.20: Permission denied (publickey,gssapi-keyex,gssapi-with-mic,password).
```

② (server1) lastb 명령어를 수행하여 failed login 대한 정보 확인

```
[root@server1 ~]# lastb
```

```
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)

btmp begins Tue Mar 30 19:18:37 2021
```

③ (server1) lastb 명령어의 다양한 사용방법 테스트

```
[root@server1 ~]# lastb -5              /* 지정된 만큼만 기록을 보여줌 */
```

```
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)

btmp begins Tue Mar 30 19:18:37 2021
```

```
[root@server1 ~]# lastb | grep user01
```

```
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
user01   ssh:notty    192.168.10.30    Tue Mar 30 19:18 - 19:18  (00:00)
```

```
[root@server1 ~]# lastb | grep user01 | wc -l
```

```
4
```

(중요) lastb 출력 결과 분석
비슷한 시간대에 반복적인 사용자 실패 기록이 발견되면 반드시 처리(?) 해야 한다.

(복원) 다른 서버에서 접속한 모든 connection을 해제한다.

8 who CMD

```
이름
       who - 로그인한 사람들을 보여준다.

개요
       who  [-imqsuwHT]  [--count]  [--idle]  [--heading] [--help] [--message]
       [--mesg] [--version] [--writable] [file] [am i]

설명
       이 맨페이지는 GNU 버전의 who 를 다룬다.  옵션 아닌 인수가 없을  때  who
       는 현재 로그인한 각 사용자에 대하여 다음 정보를 출력한다:

              로그인명
              터미널 라인
              로그인 시간
              원격 호스트 또는 X 디스플레이

       옵션 아닌 인수가 하나 주어지면, who 는 /etc/utmp 를 사용하지 않고 그 인
       수를 로그인한 사용자 이름을 담고 있는화일로 간주하여 사용한 다.   보 통
       /etc/wtmp 를 주면 who 는이전에 로그인했던 사람들의 목록을 보여준다.

       옵션 아닌 인수가 2 개 주어지면, who 는 실행한 사용자에 대한 정보만 출력
       한다.( 표준입력으로부터 결정 ) 전통적으로 두 개의 인수는  'am  i'  로 서
        'who am i'  가 된다.

   옵션
       -H, --heading
              칼럼 헤더 라인을 출력한다.
```

who 명령을 통해 누가 로그인해 있는지, 어떤 장치를 이용하고 있는지, 언제 로그인했는지, 어디에서 로그인했는지 등의 정보를 알 수 있다. who 명령은 /var/run/utmp 파일의 내용을 참고하여 출력한다. /var/run/utmp 파일은 텍스트 파일 형식이 아니고 데이터 형식의 파일이므로 cat 명령을 통해서 볼수는 없다.

[명령어 형식]

```
# who            /* 현재 시스템에 접속 중인 모든 사용자 */
# who -r         /* 현재 사용자의 Runlevel 확인 */
# who am i       /* 로그인한 사용자 정보 확인 */
# who -H         /* 헤드라인과 같이 출력 */
# whoami         /* 현재 사용자명 확인 */
```

[참고] loginctl CMD

$ loginctl

[명령어 옵션]

옵션	기능
-i	idle time 과 함께 사용자 출력
-m	who 명령을 실행한 사용자 표시
-q	사용자 이름과 사용자수 출력
-w, -T	각 사용자의 메시지 설정 상태 출력
-H	헤드라인 정보 표시
-r	run-level 확인

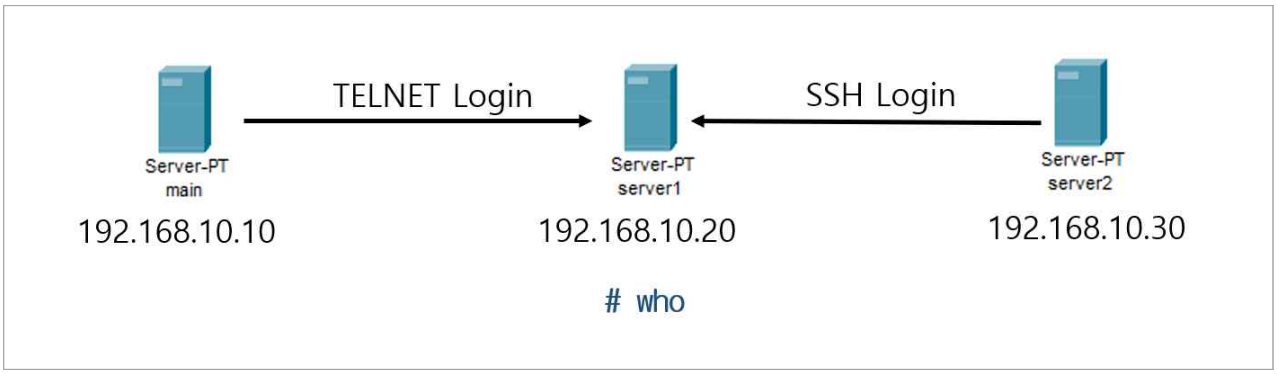
[실습] who 명령어 실습

■ 실습 시스템

- main
- server1
- server2

■ 실습 시나리오

- main/server2 서버에서 telnet/ssh 사용하여 server1 시스템으로 원격 접속을 한다. 그리고 server1에서 원격에서 접속된 사용자를 확인하기 위해서 who 명령을 사용한다.



[EX1] who 명령어 실습

IP 설정(IPv4 Class 사설 IP 대역: 192.168.10.0/24)
* main : 192.168.10.10/24
* server1 : 192.168.10.20/24
* server2 : 192.168.10.30/24

① (main) telnet 명령어를 이용하여 server1 시스템에 fedora 사용자로 로그인

[root@main ~]# telnet 192.168.10.20

fedora 사용자로 로그인

[fedora@server1 ~]$ id

```
uid=1000(fedora) gid=1000(fedora) groups=1000(fedora),10(wheel)
```

[fedora@server1 ~]$ pwd

```
/home/feodora
```

[fedora@server1 ~]$ tty

```
/dev/pts/1
```

② (server2) ssh 명령어를 이용하여 server1 시스템에 root 사용자로 로그인
[root@server2 ~]# ssh 192.168.10.20
root 사용자로 로그인

[root@server1 ~]# id

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

[root@server1 ~]# pwd

```
/root
```

[root@server1 ~]# tty

```
/dev/pts/2
```

③ (server1) who 명령어를 실행하고 내용 분석
[root@server1 ~]# who

```
root     tty2         2021-03-30 15:00  (tty2)
fedora❶ pts/1❷       2021-03-30 19:29❸(::ffff:192.168.10.10)❹
root     pts/2        2021-03-30 19:30  (192.168.10.30)
```

(명령어 출력 결과 해석)

- ❶ fedora 사용자 정보
- ❷ pts/1 제어 터미널
- ❸ 2021-03-30 19:29 로그인 시간
- ❹ 192.168.10.10 원격호스트

[root@server1 ~]# who -a

```
          system boot  2021-03-30 14:54
          run-level 5  2021-03-30 14:55
root     + tty2        2021-03-30 15:00 04:41        2160 (tty2)
fedora   + pts/1       2021-03-30 19:29 00:07       10367 (::ffff:192.168.10.10)
LOGIN      tty3        2021-03-30 19:15             10008 id=tty3
LOGIN      tty4        2021-03-30 19:15             10041 id=tty4
LOGIN      tty5        2021-03-30 19:15             10042 id=tty5
LOGIN      tty6        2021-03-30 19:15             10043 id=tty6
root     + pts/2       2021-03-30 19:30 00:05       10478 (192.168.10.30)
```

[참고] loginctl CMD
[root@server1 ~]# loginctl

(복원) 원격에서 접속한 모든 connection 종료한다.
* (main) # exit
* (server2) # exit

9 w CMD

```
NAME
       w - Show who is logged on and what they are doing.

SYNOPSIS
       w - [husfV] [user]

DESCRIPTION
       w  displays  information  about the users currently on the machine, and
       their processes.  The header shows, in this order,  the  current  time,
       how  long  the  system  has  been running, how many users are currently
       logged on, and the system load averages for the past 1, 5, and 15  min-
       utes.

       The  following entries are displayed for each user: login name, the tty
       name, the remote host, login time, idle time, JCPU, PCPU, and the  com-
       mand line of their current process.

       The  JCPU  time  is the time used by all processes attached to the tty.
       It does not include past background jobs, but  does  include  currently
       running background jobs.

       The  PCPU  time  is  the time used by the current process, named in the
       "what" field.
```

시스템에 login한 사용자가 어떤 명령어를 실행하고 있는지 알아보는 명령어이며, /proc 디렉토리로 부터 사용자에 대한 정보와 실행중인 명령어에 대한 정보를 추출해 낸다.

[명령어 형식]

```
# w -i
# w -i user01
```

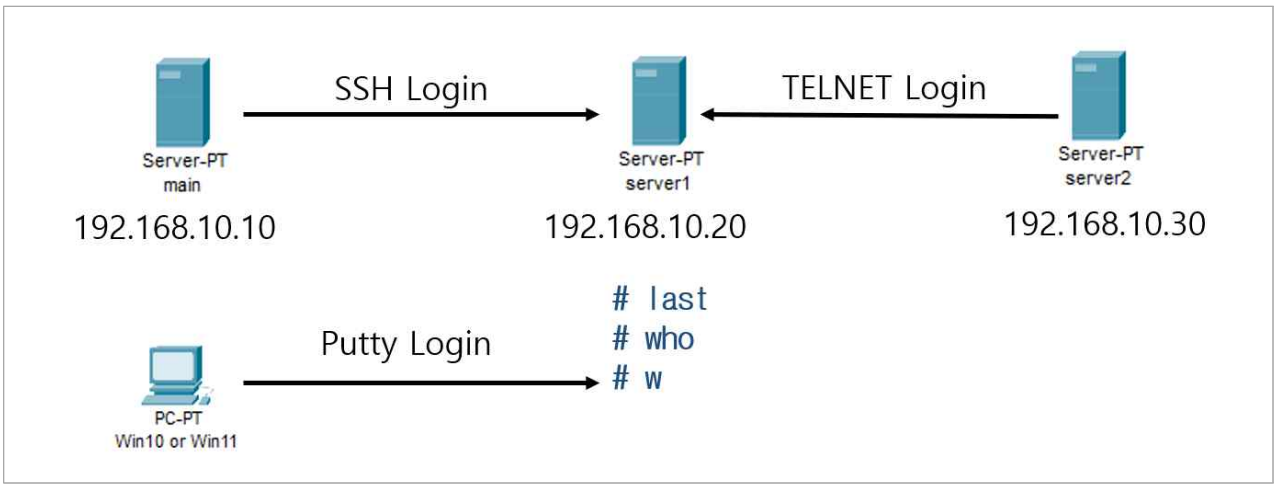
[실습] w 명령어 실습

■ 실습 시스템

- (Optional) HostOS
- main
- server1
- server2

■ 실습 시나리오

- HostOS(Windows 10)에서 server1 시스템으로 putty를 사용하여 접속하고, main/server2에서 원격 접속 명령어로 telnet/ssh 사용하여 접속한다. 그리고 server2 시스템에서 last/who/w 명령어등을 사용하여 접속된 내용들을 분석한다.
- 원격에서 접속하는 악의적인 사용자를 종료하는 작업을 진행해 본다.



[EX1] w 명령어 실습

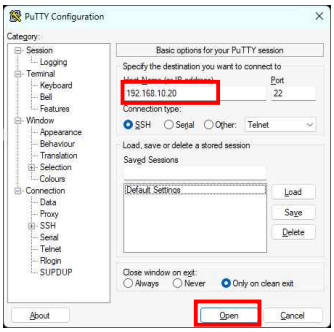
■ 터미널 에뮬레이터: putty, secureCRT, xshell(Xmanager), MovaXterm, BitviseSSH, ...

① (HostOS) HostOS에서 putty 사용하여 server1 시스템에 root 사용자로 로그인
- 만약 HostOS에 putty 프로그램이 없다면 www.putty.org 웹에서 다운로드하고 설치한다.
- putty 프로그램을 통해 리눅스 서버에 접속(EX: 원본 운영체제(Window10) -- putty --> Linux 서버)
- 접속시 SSH 프로토콜을 사용하도록 설정한다.
- (주의) HostOS의 방화벽은 내려간 상태여야 한다.(firewall.cpl)
  <CTRL + ESC> => firewall.cpl

WinPC(HostOS) IP 확인:
- **<CTRL + ESC>** => **cmd** => **ipconfig** => (ex: 172.16.6.X)
  : 자신의 Host OS PC IP를 확인한다.

```
C:WUsersWsoldesk>ipconfig

Windows IP 구성


이더넷 어댑터 이더넷2:

   연결별 DNS 접미사. . . . :
   링크-로컬 IPv6 주소 . . . . : fe80::596:ae60:c3aa:566a%12
   IPv4 주소 . . . . . . . . . . : 172.16.6.1
   서브넷 마스크 . . . . . . . . : 255.255.255.0
   기본 게이트웨이 . . . . . . . : 172.16.6.254
```

Putty 프로그램을 사용하여 server1 시스템에 root 사용자로 로그인



puTTY Configuration 창에서 다음과 같은 내용을 입력한다.
- Host Name (or IP address) : 192.168.10.20
- Port: 22
- Connection type: SSH
- root 사용자로 로그인

[참고] 직접 접속이 되지 않는 경우 포트 포워딩(Port Forwarding)을 구성하는 경우의 작업 내용

HostOS에서 직접 VM의 Linux OS로 putty를 사용하여 접속이 가능하지 않는 경우라면 다음과 같이 VMware Workstation의 포트 포워딩(Port Forwarding) 기능을 사용하여 우회하여 접속하면 된다.

① VMWare에서 포트 포워딩 설정
VMware VMnet8 Port Forwarding
- VMware > Edit > Virtual Network Editor > VMnet8 > NAT Settings > Port Forwarding
  * Host port: 22
  * Type: [v] TCP
  * Virtual machine IP address: 192.168.10.20
  * Virtual machine port: 22
  * Description: server1 SSH

② HostOS IP를 통해 Linux 서버에 putty로 접속하기
Putty 프로그램을 사용하여 server1 시스템에 root 사용자로 로그인
- Host Name (or IP address) : 172.16.6.X (172.16.6.X : WinPC's IP)
- Port: 22
- Connection type: SSH
- root 사용자로 로그인

② (main) ssh 명령어를 사용하여 server1 시스템에 fedora 사용자로 로그인
- (main) --- ssh ---> (server1)

```
[root@main ~]# ssh fedora@192.168.10.20
```
fedora 사용자로 로그인

추가적으로 다음과 같은 명령어를 수행한다.
```
$ hostname
$ id
$ pwd
$ tty
```

③ (server2) telnet 명령어를 사용하여 server1 시스템에 user01 사용자로 로그인
- (server2) --- TELNET ---> (server1)

```
[root@server2 ~]# telnet 192.168.10.20
```
user01 사용자로 로그인

추가적으로 다음과 같은 명령어를 수행한다.
```
$ hostname
$ id
$ pwd
$ tty
```

④ (server2) user01 사용자가 /etc/passwd 파일 열기
```
[user01@server1 ~]$ vi /etc/passwd
```
-> 출력 내용 생략
-> vi: 편집기

⑤ (server1) w 명령어를 수행하고 내용 분석
```
[root@server1 ~]# w -i
```
```
 19:57:50 up  5:03,  4 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty2     tty2             15:00    5:02m  6:44   0.39s /usr/libexec/tracker-miner-fs
fedora   pts/2    192.168.10.10    19:51    6:28   0.03s  0.03s -bash
root     pts/1    192.168.10.1     19:51    6:48   0.02s  0.02s -bash
user01   pts/3    192.168.10.30    19:52    1:13   0.06s  0.03s vim /etc/passwd
```
- -i, --ip-addr display IP address instead of hostname (if possible)

출력 내용 분석
- user01 : 접속한 사용자 이름
- pts/3 : 접속한 사용자에게 할당된 터미널
- 192.168.10.30 : 접속 호스트
- 19:52 : 로그인 시간
- 1:13 : idle time
- 0.06s : 현재 터미널에서 수행된 전체 CPU 실행시간
- 0.03s : 현재 접속한 프로세스가 실행한 CPU 실행시간(ex: vim /etc/passwd)
- vim /etc/passwd : 수행 명령어

[EX2] 악의적인 사용자 로그아웃 시키기

① (server1) 악의적인 사용자일 가능성이 높은 사용자 선택
```
[root@server1 ~]# w -i
```
```
 20:03:00 up  5:08,  4 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty2     tty2             15:00    5:07m  6:51   0.39s /usr/libexec/tracker-miner-fs
fedora   pts/2    192.168.10.10    19:51   11:38   0.03s  0.03s -bash
root     pts/1    192.168.10.1     19:51   11:58   0.02s  0.02s -bash
user01   pts/3    192.168.10.30    19:52    6:23   0.06s  0.03s vim /etc/passwd
```

```
[root@server1 ~]# w -i user01
```
```
 20:03:29 up  5:09,  4 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
user01   pts/3    192.168.10.30    19:52    6:52   0.06s  0.03s vim /etc/passwd
```

② (server1) 사용자 수행 명령어 모니터링 하기

```
[root@server1 ~]# while true
> do
> w -i user01
> sleep 2
> done
```

- 일정한 시간동안 user01 사용자의 수행 명령어를 모니터링 한다.

다음 명령을 복사해서 사용해도 된다.

```
while true
do
    echo ; echo "---$(date)---"
    w -i user01
    sleep 2
done
```

③ (server1) user01 사용자 강제 종료 시키기

```
[root@server1 ~]# ps -f -u user01
UID          PID    PPID  C STIME TTY          TIME CMD
user01     11370       1  0 19:52 ?        00:00:00 /usr/lib/systemd/systemd --user
user01     11373   11370  0 19:52 ?        00:00:00 (sd-pam)
user01     11390   11370  0 19:52 ?        00:00:00 /usr/bin/pulseaudio --daemonize=no
user01     11391   11303  0 19:52 pts/3    00:00:00 -bash
user01     11464   11370  0 19:52 ?        00:00:00 /usr/bin/dbus-daemon --session --address=syst
user01     11544   11391  0 19:56 pts/3    00:00:00 vim /etc/passwd
```

- -u, U, --user <UID> effective user id or name
- -f full-format, including command lines

```
[root@server1 ~]# kill -9 11391
[root@server1 ~]# ps -f -u user01
UID          PID    PPID  C STIME TTY          TIME CMD
```

- user01 사용자가 강제적으로 로그아웃되어서, 떠 있는 프로세스는 없는 것으로 판단이 된다.

[참고] 원격서버에 접속시

- 원격서버에 접속하여 접속한 서버의 이름, 사용자, 작업 디렉토리 확인

```
# ssh 192.168.10.20
```

root 사용자로 로그인

```
# hostname
# id
# pwd
# tty
```

[참고] 관리 서버에 접속시

- 자신이 관리하는 서버에 접속하여 오늘 로그인/로그아웃 사용자 정보 확인
- 현재 접속되어져 있는 사용자 확인
- 현재 접속된 사용자가 수행하는 명령어 확인

```
# last
# who
# w

[TERM1] # top
[TERM2] # tail -f /var/log/messages
```

(복원) 원격에서 접속한 모든 connection 종료한다.

[참고] 모니터링 명령어 - **watch** CMD

■ watch CMD 사용법
watch 'CMD'
watch 'CMD; CMD'

■ 직접 생성하는 경우

```
while true
do
     echo "----`date`----"
     CMD
     sleep 2
done
```

```
while true
do
     echo "----$(date)----"
     CMD
     sleep 2
done
```

```
CMD='df -hT -t xfs'
while true
do
     clear
     echo "----`date`----"
     $CMD
     sleep 2
done
```

```
# while true ; do CMD ; sleep 2 ; done
(예) # while true ; do sha1sum /dev/null ; done
```

10 exit CMD

```
NAME
       exit - cause the shell to exit

SYNOPSIS
       exit [n]

DESCRIPTION
       The exit utility shall cause the shell to exit  with  the  exit  status
       specified  by  the  unsigned decimal integer n.  If n is specified, but
       its value is not between 0 and 255  inclusively,  the  exit  status  is
       undefined.

       A  trap  on  EXIT shall be executed before the shell terminates, except
       when the exit utility is invoked in that trap itself, in which case the
       shell shall exit immediately.
```

exit 명령은 현재의 프로세스(현재쉘)를 종료하는 역할을 가진다. 일반적으로 exit 명령은 하위 쉘을 실행하고 종료하는 경우, 원격 서버에 접속했다가 로그 아웃 하는 경우, 다른 사용자로 전환 한 후 원래 사용자로 돌아 올때 사용할 수 있다.

[명령어 형식]

```
# exit [Number]            /* 값을 지정해준다면 0은 정상종료, 1~255는 비정상 종료 */
```

[실습] exit 명령어 실습

■ 실습 시스템
- server1
- server2

■ 실습 시나리오
- 하위 쉘을 종료하는 경우
- 서버에 원격 접속 후 로그아웃
- 다른 사용자로 전환한 후 원래 사용자로 전환

[EX1] 서브 쉘 종료하는 경우

① (server1) 현재 쉘 확인

```
[root@server1 ~]# ps
  PID TTY          TIME CMD
21224 pts/11   00:00:00 bash
```

② (server1) 하위 쉘(ex: bash) 실행

```
bash
---------+
         |
         |   bash
         +------------>
```

```
[root@server1 ~]# bash
[root@server1 ~]# ps
  PID TTY          TIME CMD
21224 pts/11   00:00:00 bash
21316 pts/11   00:00:00 bash
```

③ (server1) 하위 쉘 종료

```
[root@server1 ~]# exit
exit
```

```
[root@server1 ~]# ps
  PID TTY          TIME CMD
21224 pts/11   00:00:00 bash
```

[EX2] 서버에 접속 후 로그 아웃

① (server1) root 사용자로 원격 접속
[root@server1 ~]# ssh 192.168.10.10
root 사용자로 접속

② (server1) 정상적으로 원격 접속이 되었는지 확인
[root@main ~]# hostname
[root@main ~]# id
[root@main ~]# pwd

③ (server1) main에서 로그 아웃
[root@main ~]# ps
-> bash 쉘 확인
[root@main ~]# exit
[root@server1 ~]#

[EX3] 사용자 전환후에 원래 사용자로 전환

① (server1) user01 사용자로 전환
[root@server1 ~]# su - user01
[user01@server1 ~]$

② (server1) 원래 사용자로 다시 전환
[user01@server1 ~]$ ps
-> bash 쉘 확인
[user01@server1 ~]$ exit
[root@server1 ~]#