

리눅스 시스템 관리

01. 리눅스 소개

02. 리눅스 서버 구축

03. 리눅스 기본 관리

04. vi 에디터

05. 리눅스 네트워크 관리

06. 리눅스 내부구조

(프로세스,메모리,파일시스템)

3.1 셸과 프롬프트

콘솔 또는 원격 터미널을 통해 터미널 화면을 시작하면 다음과 같은 커맨드 프롬프트가 화면에 등장한다.

```
[kickscar@localhost ~] $ _
```

커맨드 프롬프트(prompt) 또는 셸 프롬프트 그냥 줄여서 프롬프트라고 한다.

@ 앞 : 접속 계정

@ 뒤 : 접속한 시스템의 호스트 이름이다.

: 뒤 : 현재 위치(경로이다) 틸드 문자(~)는 접속 계정의 홈 디렉토리(/home/kickscar)의 일종의 shortcut(약칭)이다.

\$: 프롬프트 이다. 뒤에 커서가 깜빡 거리면서 명령어를 입력과 실행(엔터)를 기다리고 있다.

[실습]

```
[kickscar@localhost ~]$ pwd
/home/kickscar
[kickscar@localhost ~]$ cd /
[kickscar@localhost /]$ pwd
/
[kickscar@localhost /]$ cd ~
[kickscar@localhost ~]$ pwd
/home/kickscar
[kickscar@localhost ~]$
```

pwd(Print Working Directory) : 현재 디렉토리 경로를 출력한다.

cd(Change Directory) : 디렉토리를 이동한다. (현재 디렉토리 경로를 변경한다.)

3.2 사용자 관리

- 다중 사용자 시스템
여러 사용자가 동시에 접속해서 사용할 수 있는 시스템
- root 계정
시스템 관리를 책임지는 계정
시스템 변경의 무한한 권한을 가지고 있지만 시스템을 망가뜨릴 수 있는 명령어를 아무런 제재 없이 실행 할 수 있다.
보안상 이유로 root로 로그인 하는 것을 피하는 것이 좋다.
- 모든 사용자는 하나 이상의 그룹에 소속된다.

3.2.1 사용자 추가

useradd

리눅스에서 계정을 추가할 때 사용한다.

사용법

useradd [옵션] 계정이름

옵션

- d : 홈 디렉터리를 지정할 때 사용한다.
- g : 그룹을 지정할 때 사용한다.
- G : 기본 그룹 이외에 추가로 지정할 그룹이 있는 경우 사용한다.
- c : 계정 추가 시 계정에 대한 설명을 설정한다.

3.2 사용자 관리

옵션

-s : 계정 추가 시, 이 계정으로 로그인 한 사용자가 사용할 쉘을 지정한다.

-D : /etc/default/useradd 파일에 설정되어 있는 useradd 명령의 기본 설정 내용을 보여준다.

실습1

user1 계정 생성하기

```
[root@localhost ~]# useradd user1
```

user1이라는 계정이 만들어지고 user1이라는 그룹이 동시에 생성되면서 user1의 그룹으로 설정한다.
계정의 홈 디렉터리는 /home/user1 디렉터리로 생성된다.

실습2

users 그룹에 속하는 user2 계정 추가하기

```
[root@localhost ~]# useradd -g users user2
```

users 라는 그룹은 생성되어 있어야 한다.

3.2 사용자 관리

현재 시스템의 사용자를 확인하려면 `/etc/passwd` 파일을 확인한다.

```
[root@localhost home]# cat /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
...
...
...

sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
kickscar:x:1000:10::/home/kickscar:/bin/bash
user1:x:1001:1001::/home/user1:/bin/bash
user2:x:1002:100::/home/user2:/bin/bash
[root@localhost home]#
```

각 라인은 하나의 계정에 대한 정보를 다음과 같은 형식으로 담고 있다.

```
ID:비밀번호:UID:GID:설명:홈디렉터리:로그인 셸
```

비밀번호는 보안상 이유로 `x` 로 표시되어 있고 따로, `/etc/shadow` 파일에 암호화 되어 관리된다.

3.2 사용자 관리

계정 추가와 함께 생성된 홈 디렉터리의 내용

```
[root@localhost home]# ls -la /home/user1
합계 12
drwx-----. 2 user1 user1  59  2월 13 13:59 .
drwxr-xr-x. 5 root  root   45  2월 13 14:11 ..
-rw-r--r--. 1 user1 user1   18 11월 20 14:02 .bash_logout
-rw-r--r--. 1 user1 user1  193 11월 20 14:02 .bash_profile
-rw-r--r--. 1 user1 user1  231 11월 20 14:02 .bashrc

[root@localhost home]#
```

/etc/skel 디렉터리의 내용을 복사하게 된다.

실습3

홈 디렉터리 생성 없이 user3 계정 추가하기

```
[root@localhost ~]# useradd -M user3
```

3.2 사용자 관리

3.2.2 사용자 비밀번호 설정

passwd

사용자의 비밀번호 설정 및 수정할 때 사용한다.

사용법

passwd 계정

비밀번호는 /etc/shadow 파일에 암호화 되어 있다.

```
[root@localhost mail]# cat /etc/shadow
```

```
root:$6$Vmxj9y1oEZzlg3DZ$Ew7ehXEubX7HyHzojld9dVIBkyuk5M7m6x0gSsskOu.iUr4.vB56mcIoCJY  
/2ZkFD5kL83SYma4LJmU9fixn/::0:99999:7::  
bin:!:16659:0:99999:7::  
...  
...  
...  
kickscar:$6$9T6P43zD$wXfg9F2AUk9hejQV9aMlghm1PH2BOZAYAD8pIPa8a/jrOQ3ZVdUpUEM3VzxOW  
kj9GuucX9CGEBJNvD1G3I4XE1:16842:0:99999:7::  
user1:!:16844:0:99999:7::  
user2:!:16844:0:99999:7::
```

비밀번호가 설정되어 있지 않으면 !!로 표시되어 있고 실제로 로그인 되지도 않는다.

3.2 사용자 관리

3.2.2 사용자 비밀번호 설정

실습1

user1 계정에 비밀번호를 설정해 보자.

```
[root@localhost mail]# passwd user1
user1 사용자의 비밀번호 변경 중
새 암호:
새 암호 재입력:
passwd: 모든 인증 토큰이 성공적으로 업데이트되었습니다.
[root@localhost mail]#
```

비밀번호가 설정된 user1과 설정이 되지 않은 user2로 로그인 테스트를 한다.

3.2 사용자 관리

3.2.3 사용자 삭제

userdel

사용자를 삭제한다.

사용법

userdel [옵션] 계정

옵션

-r : 사용자의 홈 디렉터리를 제거한다.

실습1

userdel 명령을 이용하여 user1을 삭제해 보자.

```
[root@localhost ~]# userdel user1
[root@localhost ~]# ls -l /home
합계 4
drwx-----. 2 kickscar wheel 4096 2월 12 22:48 kickscar
drwx-----. 2 1001 1001 59 2월 13 13:59 user1
drwx-----. 2 user2 users 59 2월 13 14:11 user2
[root@localhost ~]#
```

계정과 그룹이 삭제 되었음을 알 수 있지만, 홈 디렉터리는 삭제되지 않았다.

3.2 사용자 관리

실습2

-r 옵션을 사용해서 계정의 홈 디렉터리 까지 함께 삭제해 보자.

```
[root@localhost ~]# userdel -r user2
[root@localhost ~]# ls -l /home
합계 4
drwx-----. 2 kickscar wheel 4096 2월 12 22:48 kickscar
drwx-----. 2 1001 1001 59 2월 13 13:59 user1
[root@localhost ~]#
```

3.2 사용자 관리

3.2.4 그룹 추가 삭제

groupadd

새로운 그룹을 생성한다.

사용법

groupadd [옵션] 그룹명

옵션

-g GID: 특정 GID번호로 설정한다.

-r : 0~1000 번대 사이로 GID를 자동으로 설정한다.

실습1

group1이라는 그룹을 생성해 보자.

```
[root@localhost ~]# groupadd group1
[root@localhost ~]# cat /etc/group | grep group
group1:x:1004:
[root@localhost ~]#
```

group에 대한 정보는 /etc/group 파일에 있다.

3.2 사용자 관리

실습2

-r 옵션을 사용해서 1000미만 중 가장 가장 큰 수를 지정하도록 그룹 group2를 생성해 보자.

```
[root@localhost ~]# groupadd -r group2
[root@localhost ~]# cat /etc/group | grep group
group1:x:1004:
group2:x:994:
[root@localhost ~]#
```

실습3

-g 옵션을 이용하여 GID를 지정하여 그룹 group3를 생성해 보자

```
[root@localhost ~]# groupadd -g 1100 group3
[root@localhost ~]# cat /etc/group | grep group
group1:x:1004:
group2:x:994:
group3:x:1100:
[root@localhost ~]#
```

3.2 사용자 관리

groupdel

그룹을 삭제하는 명령어이다. 만약, 삭제하려는 그룹에 속한 계정이 있다면 삭제되지 않는다.

사용법

groupdel 그룹명

실습1

group3 그룹을 삭제해 보자.

```
[root@localhost ~]# groupdel group3
[root@localhost ~]# cat /etc/group | grep group
group1:x:1004:
group2:x:994:
[root@localhost ~]#
```

group2, group1 그룹도 삭제해 보자.

3.2 사용자 관리

groups

사용자가 속한 그룹을 보여준다.

사용법

groups 사용자명

실습1

root 사용자가 속한 그룹을 알아보자

```
[kickscar@localhost ~]$ groups root
root : root
[kickscar@localhost ~]$
```

3.2 사용자 관리

3.2.5 사용자 전환

SU

다른 사용자 권한으로 셸을 실행한다. 로그아웃 없이 다른 사용자로 전환할 수 있다.

사용법

su [옵션] 계정

옵션

- : 사용자의 환경변수를 읽는다.

실습1

root 계정으로 전환해 보자.

```
[kickscar@localhost ~]$ su -  
암호:  
마지막 로그인: 토 6월 13 15:39:46 KST 2016 일시 pts/0  
[root@localhost ~]#
```

root 계정에서 다른 계정으로 전환할 때는 비밀번호는 생략된다.

3.3 디렉터리 관리

디렉터리의 목록을 수정, 삭제, 이동하는 등의 디렉터리와 관련된 명령어

3.3.1 pwd

현재 위치한 디렉터리의 절대경로를 출력한다.

```
[root@localhost ~]# pwd
/root
[root@localhost ~]#
```

3.3.2 ls

디렉터리에 있는 파일의 목록을 표시해 준다.

사용법

ls [옵션] [디렉터리]

옵션

- l : 파일에 대해서 권한이나 생성 시간처럼 보다 자세한 내용을 출력한다. (long)
- a : 숨긴 파일이나 디렉터리 등의 현재 디렉터리의 모든 내용을 출력한다. (all)
- h : 파일 크기를 k, m, g 와 같이 사람이 읽기 편한 단위로 출력한다. (human readable)
- F : 실행 파일이나 디렉터리 등이 쉽게 구분 될 수 있도록 출력한다.
- R : 하위 디렉터리의 내용들도 함께 출력한다.

3.3 디렉터리 관리

실습1

옵션 없이 간단히 현재 디렉터리 내용을 살펴 보자.

```
[root@localhost ~]# pwd
/root
[root@localhost ~]# ls
anaconda-ks.cfg  mongodb-linux-x86_64-rhel70-3.2.1.tgz
[root@localhost ~]#
```

실습2

-al 옵션을 사용해서 숨겨진 파일들을 포함해서 모든 파일의 자세한 정보를 나열해 보자

```
[root@localhost ~]# ls -al
합계 62120
dr-xr-x---. 2 root root  4096 2월 12 22:28 .
dr-xr-xr-x. 18 root root  4096 2월 12 22:29 ..
-rw-----. 1 root root  8867 2월 13 17:51 .bash_history
-rw-r--r--. 1 root root   18 12월 29 2013 .bash_logout
-rw-r--r--. 1 root root  176 12월 29 2013 .bash_profile
-rw-r--r--. 1 root root  176 12월 29 2013 .bashrc
-rw-r--r--. 1 root root  100 12월 29 2013 .cshrc
-rw-r--r--. 1 root root  129 12월 29 2013 .tcshrc
-rw-----. 1 root root  1066 2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root 63563953 1월 12 04:07 mongodb-linux-x86_64-rhel70-3.2.1.tgz
[root@localhost ~]#
```

3.3 디렉터리 관리

실습3

ll 명령어를 실습한다. ll은 명령어가 아니다 ls -l 명령어를 지정한 alias 이다.

```
[root@localhost ~]# alias
alias cp='cp -i'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias rm='rm -i'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
[root@localhost ~]#
```

```
[root@localhost ~]# ll
합계 62080
-rw-----. 1 root root 1066 2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root 63563953 1월 12 04:07 mongodb-linux-x86_64-rhel70-3.2.1.tgz
[root@localhost ~]#
```

실습 4

/var/log 디렉터리 내용을 확인해 보자

3.3 디렉터리 관리

3.3.3 mkdir

디렉터리를 생성한다.

사용법

mkdir [옵션] 디렉터리

옵션

- m : 디렉터리의 권한을 지정할 수 있다. 기본값은 755이다.
- p : 상위 디렉터리가 존재하지 않으면 상위 디렉터리도 만든다.

실습1

현재 경로에서 dir1 디렉터를 생성해 보자.

```
[kickscar@localhost ~]$ pwd
/home/kickscar

[kickscar@localhost ~]$ mkdir dir1

[kickscar@localhost ~]$ ls -l
할계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 13 18:10 dir1

[kickscar@localhost ~]$
```

3.3 디렉터리 관리

실습2

현재 경로에서 dir2/subdir1 디렉터를 -p 옵션을 사용해서 한 번에 생성해 보자.

```
[kickscar@localhost ~]$ pwd  
/home/kickscar
```

```
[kickscar@localhost ~]$ mkdir -p dir2/subdir1
```

```
[kickscar@localhost ~]$ ls -l
```

```
합계 0
```

```
drwxr-xr-x. 2 kickscar wheel  6  2월 13 18:10 dir1
```

```
drwxr-xr-x. 3 kickscar wheel 20  2월 13 18:28 dir2
```

```
[kickscar@localhost ~]$ cd dir2
```

```
[kickscar@localhost dir2]$ ls -l
```

```
합계 0
```

```
drwxr-xr-x. 2 kickscar wheel  6  2월 13 18:28 subdir1
```

```
[kickscar@localhost dir2]$
```

3.3 디렉터리 관리

3.3.4 rmdir

비어 있는 디렉터를 삭제한다.

사용방법

rmdir [옵션] 디렉터리

옵션

-p : 상위 디렉터리도 지운다. 상위 디렉터리도 비어 있어야 한다.

실습1

dir1 디렉터를 지워보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel  6  2월 13 18:10 dir1
drwxr-xr-x. 3 kickscar wheel 20  2월 13 18:28 dir2

[kickscar@localhost ~]$ rmdir dir1

[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 3 kickscar wheel 20  2월 13 18:28 dir2
[kickscar@localhost ~]$
```

3.3 디렉터리 관리

실습2

dir2 디렉터를 지워보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 3 kickscar wheel 20  2월 13 18:28 dir2

[kickscar@localhost ~]$ rmdir dir2
rmdir: failed to remove `dir2': 디렉터리가 비어있지 않음

[kickscar@localhost ~]$
```

dir2 아래에는 subdir1 디렉터리가 존재하기 때문에 삭제 되지 않는다.

이럴 경우에는 -p 옵션을 사용해서 subdir1를 삭제하게 되면 dir2 함께 삭제 할 수 있다.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 3 kickscar wheel 20  2월 13 18:28 dir2

[kickscar@localhost ~]$ rmdir -p dir2/subdir1

[kickscar@localhost ~]$ ls -l
합계 0
[kickscar@localhost ~]$
```

3.3 디렉터리 관리

3.3.5 cd

디렉터리를 이동할 때에 사용한다.

사용방법

cd [디렉터리]

디렉터리 명이 생략되면 접속 계정의 홈 디렉토리도 이동한다. (\$ cd ~)

디렉토리 경로는 상대경로와 절대경로로 나타낼 수 있다.

- 절대경로 : /(루트 디렉터리) 부터 모든 경로를 표시 하는 방법
- 상대경로 : 현재 디렉터리 기반으로 특정 디렉터리의 경로를 표시하는 방법이다.
유닉스에서는 상대경로를 표시 할 수 있도록 . (현재 디렉터리) 와 .. (부모 디렉터리)의 심볼을 제공한다.

실습1

/ 루트 디렉터리로 이동해 보자

```
[root@localhost ~]# pwd
/root

[root@localhost ~]# cd /

[root@localhost /]# pwd
/

[root@localhost /]#
```

3.3 디렉터리 관리

실습2

특정 계정으로 이동하기

```
[root@localhost /]# pwd
/  
  
[root@localhost /]# cd ~kickscar  
  
[root@localhost kickscar]# pwd  
/home/kickscar
```

실습3

로그인 계정의 홈 디렉터리로 이동하기

실습4

상위 디렉터리로 이동

```
[root@localhost ~]# pwd  
/root  
[root@localhost ~]# cd ..  
[root@localhost /]# pwd  
/
```


3.3 디렉터리 관리

실습5

키보드의 tab키를 이용하면 디렉터리 이름을 전부 입력하지 않아도 자동으로 완성 된다.
cd /use/lo 까지만 입력하고 tab키를 눌러보자.

```
[root@localhost /]# cd /usr/lo
```

입력까지 해당되는 이름이 여러 개가 존재하면 목록이 표시된다.

```
[root@localhost local]# clear
[root@localhost local]# cd /var/l
lib/  local/ lock/ log/
[root@localhost local]# cd /var/l
```

3.3 디렉터리 관리

[실습과제]

1. 현재 접속 계정의 홈 디렉토리로 이동 한다.
2. 현재 경로를 알아본다.
3. 현재 디렉토리의 내용을 확인한다.
4. test01 디렉토리를 생성한다.
5. 디렉토리 생성을 확인한다.
6. test01 디렉토리로 이동한다.
7. 현재 경로를 알아본다.
8. test011 디렉토리를 생성한다.
9. 디렉토리 생성을 확인한다.
10. test012 디렉토리를 생성한다.
11. 디렉토리 생성을 확인한다.
12. 심볼을 사용해서 계정 홈 디렉토리로 이동한다.
13. 한 번에 test02/test021 디렉토리를 생성한다.
14. test02 디렉토리 생성을 확인해본다
15. 한 번에 test02/test022 디렉토리를 생성한다.
16. 상대 경로로 test02/test022 디렉토리로 이동 한다.
17. 현재 경로를 확인해 본다.
18. 절대 경로로 접속계정의 홈 디렉토리로 이동한다.
19. 실습용으로 만든 모든 디렉토리를 삭제 한다.

3.3 디렉터리 관리

[실습과제]

```
[kickscar@localhost ~]$ cd
[kickscar@localhost ~]$ pwd
/home/kickscar
[kickscar@localhost ~]$ ls -la
합계 24
drwx----- 2 kickscar wheel 4096  2월 13 18:41 .
drwxr-xr-x  4 root    root   33  2월 13 16:07 ..
-rw-----  1 kickscar wheel 2172  2월 13 18:10 .bash_history
-rw-r--r--  1 kickscar wheel  18 11월 20 14:02 .bash_logout
-rw-r--r--  1 kickscar wheel 193 11월 20 14:02 .bash_profile
-rw-r--r--  1 kickscar wheel 231 11월 20 14:02 .bashrc
-rw-r--r--  1 kickscar wheel  44  2월 12 22:48 .dbshell
-rw-----  1 kickscar wheel   0  2월 12 22:35 .mongorc.js
[kickscar@localhost ~]$ mkdir test01
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x  2 kickscar wheel 6  2월 13 19:02 test01
[kickscar@localhost ~]$ cd test01
[kickscar@localhost test01]$ pwd
/home/kickscar/test01
[kickscar@localhost test01]$ mkdir test011
[kickscar@localhost test01]$ ls -l
합계 0
drwxr-xr-x  2 kickscar wheel 6  2월 13 19:03 test011
[kickscar@localhost test01]$ mkdir test012
[kickscar@localhost test01]$ ls -l
합계 0
drwxr-xr-x  2 kickscar wheel 6  2월 13 19:03 test011
drwxr-xr-x  2 kickscar wheel 6  2월 13 19:04 test012
[kickscar@localhost test01]$ cd ~
[kickscar@localhost ~]$ mkdir -p test02/test021
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x  4 kickscar wheel 34  2월 13 19:04 test01
drwxr-xr-x  3 kickscar wheel 20  2월 13 19:04 test02
[kickscar@localhost ~]$ mkdir -p test02/test022
[kickscar@localhost ~]$ cd test02/test022
[kickscar@localhost test022]$ pwd
/home/kickscar/test02/test022
[kickscar@localhost test022]$ cd /home/kickscar
[kickscar@localhost ~]$ rmdir -p test01/test011
rmdir: failed to remove directory `test01': 디렉터리가 비어있지 않음
```

```
[kickscar@localhost ~]$ rmdir -p test01/test012
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x  4 kickscar wheel 34  2월 13 19:05 test02
[kickscar@localhost ~]$ rmdir -p test02/test021
rmdir: failed to remove directory `test02': 디렉터리가 비어있지 않음
[kickscar@localhost ~]$ rmdir -p test02/test022
[kickscar@localhost ~]$ ls -l
합계 0
[kickscar@localhost ~]$
```

3.4 파일 관리

3.3.1 파일 정보 이해하기

파일의 종류

- 유닉스에서 프로그램들은 주변의 장치(device)를 파일로 인식하기 때문에 여러 종류의 파일이 존재 한다.
- ls 명령을 사용한 결과를 보면, 각 각의 파일들을 타입 별로 구분해 놓은 정보가 있다.

파일의 구분

- : 일반(보통) 파일
- b : 블록 디바이스 파일
- c : 문자열 디바이스 파일
- d : 디렉토리
- l : 심볼릭 링크
- p 또는 = : 명명된 파이프(named pipe) / FIFO
- s : 소켓(socket)

3.4 파일 관리

실습1

ls 명령을 사용해서 파일의 타입을 살펴 보자

```
[kickscar@localhost ~]$ ls -la
하계 28
d wx-----, 2 kickscar wheel 4096  2월 14 09:26 .
d wxr-xr-x, 4 root    root   33  2월 13 16:07 ..
-rw-----, 1 kickscar wheel 5871  2월 13 19:40 .bash_history
-rw-r--r--, 1 kickscar wheel  18 11월 20 14:02 .bash_logout
-rw-r--r--, 1 kickscar wheel 193 11월 20 14:02 .bash_profile
-rw-r--r--, 1 kickscar wheel 231 11월 20 14:02 .bashrc
-rw-r--r--, 1 kickscar wheel  44  2월 12 22:48 .dbshell
-rw-----, 1 kickscar wheel   0  2월 12 22:35 .mongorc.js
-rw-r--r--, 1 kickscar wheel   0  2월 14 09:27 test
[kickscar@localhost ~]$ ls -l /dev
하계 0
crw-----, 1 root root   10, 235  2월 11 21:37 autofs
d wxr-xr-x, 2 root root   140  2월 11 21:37 block
d wxr-xr-x, 2 root root    80  2월 11 21:37 bsg
crw-----, 1 root root  10, 234  2월 11 21:37 btrfs-control
d wxr-xr-x, 3 root root    60  2월 12 00:04 bus
lrwxrwxrwx, 1 root root    3  2월 12 03:47 cdrom -> sr0
d wxr-xr-x, 2 root root  2640  2월 11 21:37 char
crw-----, 1 root root    5,  1  2월 11 21:37 console
lrwxrwxrwx, 1 root root   11  2월 11 21:36 core -> /proc/kcore
d wxr-xr-x, 3 root root    80  2월 12 00:04 cpu
crw-----, 1 root root  10, 61  2월 11 21:37 cpu_dma_latency
crw-----, 1 root root  10, 62  2월 11 21:37 crash
d wxr-xr-x, 6 root root   120  2월 12 03:47 disk
lrwxrwxrwx, 1 root root   13  2월 11 21:36 fd -> /proc/self/fd
crw-rw-rw-, 1 root root    1,  7  2월 11 21:37 full
crw-rw-rw-, 1 root root  10, 229  2월 11 21:37 fuse
crw-----, 1 root root  10, 228  2월 11 21:37 hpet
d wxr-xr-x, 2 root root    0  2월 11 21:37 hugepages
lrwxrwxrwx, 1 root root   25  2월 11 21:37 initctl -> /run/systemd/initctl/fifo
d wxr-xr-x, 3 root root   220  2월 11 21:37 input
crw-r--r--, 1 root root    1, 11  2월 11 21:37 kmsg
srw-rw-rw-, 1 root root    0  2월 11 21:36 log
```

3.4 파일 관리

파일의 소유자와 그룹

- 유닉스의 모든 파일(디렉터리 포함)에는 소유자와 그룹이 있다.
- 대부분 파일을 처음 생성한 계정과 그 계정이 속한 그룹이 그 파일의 소유자와 그룹이 되지만 **chown 명령**을 이용하면 소유 계정과 그룹을 바꿀 수 있다.

실습1

ls 명령을 사용해서 파일의 소유 계정과 그룹을 확인해 보자

```
[kickscar@localhost ~]$ ls -la
합계 28
drwx-----, 2 kickscar wheel 4096  2월 14 09:26 .
drwxr-xr-x, 4 root      root   33   2월 13 16:07 ..
-rw-----, 1 kickscar wheel 5871  2월 13 19:40 .bash_history
-rw-r--r--, 1 kickscar wheel  18   11월 20 14:02 .bash_logout
-rw-r--r--, 1 kickscar wheel 193   11월 20 14:02 .bash_profile
-rw-r--r--, 1 kickscar wheel 231   11월 20 14:02 .bashrc
-rw-r--r--, 1 kickscar wheel  44   2월 12 22:48 .dbshell
-rw-----, 1 kickscar wheel   0   2월 12 22:35 .mongorc.js
-rw-r--r--, 1 kickscar wheel   0   2월 14 09:27 test
[kickscar@localhost ~]$
```

3.4 파일 관리

파일 권한

누가 파일에 접근해도 되는지, 접근해서 무엇을 할 수 있는 지를 결정하는 것이 파일 권한(permission, 퍼미션)이라 한다. 다음 ls 명령의 결과를 보면 각각의 파일의 퍼미션을 알 수가 있다.

```
[kickscar@localhost ~]$ ls -la
drwx-----. 2 kickscar wheel 4096 2월 14 09:26 .
drwxr-xr-x. 4 root    root   33 2월 13 16:07 ..
-rw-----. 1 kickscar wheel 5871 2월 13 19:40 .bash_history
-rw-r--r--. 1 kickscar wheel  18 11월 20 14:02 .bash_logout
-rw-r--r--. 1 kickscar wheel 193 11월 20 14:02 .bash_profile
-rw-r--r--. 1 kickscar wheel 231 11월 20 14:02 .bashrc
-rw-r--r--. 1 kickscar wheel  44 2월 12 22:48 .dbshell
-rw-----. 1 kickscar wheel   0 2월 12 22:35 .mongorc.js
-rw-r--r--. 1 kickscar wheel   0 2월 14 09:27 test
[kickscar@localhost ~]$
```

파일 타입 뒤의 3개의 “rwx”가 각 각 파일에 대한 소유 계정의 권한, 소유 그룹에 대한 권한, 모든 사용자에게 대한 권한을 의미한다.(r : 읽기권한, w : 쓰기 권한, x : 실행 권한)

-	rwx	rwx	rwx
파일타입	user 권한	group 권한	other 권한

3.4 파일 관리

실습1

다음 ls 명령 결과를 가지고 권한을 해석해 보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 14 10:24 dowork
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
[kickscar@localhost ~]$
```

dowork 디렉토리는 rwx r-x r-x 권한을 가지고 있다.

1. 파일 소유 계정 kickscar는 디렉터리에 읽기, 쓰기, 실행 권한을 가지고 있다.
2. 파일 소유 그룹 wheel 디렉터리에 읽기, 실행 권한을 가지고 있다.
3. 다른 사용자는 디렉터리에 읽기, 실행 권한을 가지고 있다.

실습2

test 파일에 대한 권한도 해석해 보도록 하자

권한은 숫자로 표시할 수 있다. (r -> 4, w -> 2, x -> 1)

따라서 dowork 디렉터리는 4+2+1, 4+1, 4+1 즉 755로 표현된다(숫자 표기 방식을 많이 쓴다.)

실습3

test 파일의 권한을 숫자로 표시해 보자.

3.4 파일 관리

3.4.2 touch

파일의 시간정보를 변경하는 명령어이다.

크기가 0인 파일을 생성하는 용도로 자주 사용된다.

사용법

touch [옵션] 파일이름

옵션

-c : 현재 시간으로 파일 시간을 변경한다. (파일이 존재하지 않으면 생성하지 않는다.)

-d 시간 : 현재 시간 대신 지정한 시간으로 파일 시간을 변경한다. (\$ touch '2016-02-05 12:00:30' test)

-r 파일 : 현재 시간 대신 지정한 파일 시간으로 파일 시간을 변경한다.

-t MMDDHHMM[[CC]YY][.SS]: 현재 시간대신 지정한 시간으로 변경한다.

실습1

0 byte 크기의 파일 test를 생성해보자.

```
[kickscar@localhost ~]$ touch test
[kickscar@localhost ~]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 13 23:55 test
[kickscar@localhost ~]$
```

3.4 파일 관리

실습2

-d 옵션을 사용해서 파일 시간을 변경해 보자.

```
[kickscar@localhost ~]$ touch -d '2016-01-01 10:00:30' test
[kickscar@localhost ~]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  1월  1 10:00 test
[kickscar@localhost ~]$
```

실습3

-t 옵션을 사용해서 파일 시간을 변경해 보자.

```
[kickscar@localhost ~]$ touch -t 201602011000.30 test
[kickscar@localhost ~]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월  1 10:00 test
[kickscar@localhost ~]$
```

실습4

현재 시간으로 변경해 보자

```
[kickscar@localhost ~]$ date
2016. 02. 14. (일) 09:26:57 KST
[kickscar@localhost ~]$ touch test
[kickscar@localhost ~]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
[kickscar@localhost ~]$
```

3.4 파일 관리

3.4.3 cp

파일을 복사하는 명령어이다.

사용법

cp [옵션] 원본 사본

옵션

- a : 원본 파일의 속성, 링크 정보들을 그대로 유지하면서 복사한다.
- i : 만약 복사 대상에 같은 이름의 파일이 존재하면 사용자에게 물어 본다.
- f : 만약 복사 대상에 같은 이름의 파일이 존재하면 강제로 지우고 복사한다.
- R : 디렉토리를 복사할 경우 그 안에 포함된 모든 하위 디렉터리와 파일들을 모두 복사한다.

실습1

test 파일을 test.bak로 복사하기

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 14 10:24 dowork
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
[kickscar@localhost ~]$ cp test test.bak
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 14 10:24 dowork
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:43 test.bak
[kickscar@localhost ~]$
```

3.4 파일 관리

실습2

test 파일을 dowork 디렉터리로 복사 하기

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 14 10:24 dowork
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:43 test.bak
[kickscar@localhost ~]$ cp test dowork/
[kickscar@localhost ~]$ ls -l dowork
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:45 test
[kickscar@localhost ~]$
```

실습3

dowork 디렉터리를 dowork.bak 디렉터리로 -R 옵션을 사용해서 복사하기

실습4

-i 옵션을 사용해서 같은 이름으로 복사하기

```
[[kickscar@localhost ~]$ cp -i test test.bak
cp: overwrite `test.bak'? n
[kickscar@localhost ~]$ cp -i test test.bak
cp: overwrite `test.bak'? y
[kickscar@localhost ~]$
```

안전을 위해 alias 로 cp='cp -i' 로 하는 것이 좋다.

3.4 파일 관리

실습5

보통 안전을 위해 cp 명령을 cp -i 로 alias 로 설정해서 사용한다.

로그인 시 실행되는 스크립트 파일 중에 보통 alias 관련된 설정은 /etc/bashrc 파일에 하게 된다.

```
[kickscar@localhost ~]$ su -
암호:
[root@localhost ~]# vi /etc/bashrc
fi
# vim:ts=4:sw=4

alias cp='cp -i'

:wq

[root@localhost ~]# exit
logout
[kickscar@localhost ~]$ source /etc/bashrc
[kickscar@localhost ~]$ cp test test.bak
cp: overwrite `test.bak`?
```

실습6

alias 로 설정해 놓은 -i 옵션이 불편 하거나 의도적인 덮어쓰기를 할 경우 다음과 같이 cp 명령을 사용하면 된다.

```
[kickscar@localhost ~]$ Wcp -f test test.bak
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:45 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 09:27 test
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test.bak
[kickscar@localhost ~]$
```

3.4 파일 관리

3.4.4 mv

cp 명령어는 파일을 복사 하지만 mv 명령어는 파일을 이동하는 명령어이다.

unix에는 rename 명령어가 따로 없다. 즉, 파일 이름을 바꿀 때도 사용할 수 있는 명령어이다.

사용법

mv [옵션] 원본 목적지

옵션

-b : 덮어 쓰게 되는 경우 백업 파일을 만들고 파일을 만든다.

-i : 덮어 쓸 때 사용자에게 물어 본다.

-f : 덮어 쓸 때 물어 보지 않는다.

실습1

test 파일을 test2로 바꾸어 보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:45 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 09:27 test
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test.bak
[kickscar@localhost ~]$ mv test test2
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:45 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 09:27 test2
[kickscar@localhost ~]$
```

3.4 파일 관리

실습2

test2 파일을 dowork/test로 바꾸어 보자.

dowork/test 파일은 이미 존재하므로 -b 옵션을 사용해서 백업 파일을 남기고 이동시켜보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:45 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 09:27 test2
[kickscar@localhost ~]$ ls -l dowork
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:45 test
[kickscar@localhost ~]$ mv -b test2 dowork/test
[kickscar@localhost ~]$ ls -l dowork
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 09:27 test
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:45 test~
```

실습3

덮어 쓸 경우 사용자에게 물어보는 옵션 -i 를 사용해 보자.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 29  2월 14 18:12 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test
[kickscar@localhost ~]$ mv -i test dowork
mv: overwrite `dowork/test'? n
[kickscar@localhost ~]$
```

3.4 파일 관리

실습4

mv 명령도 보통 안전을 위해 mv -i 로 alias 만들어 사용하게 된다. alias에 추가 하고 안전한 mv를 사용해 보자

```
[kickscar@localhost ~]$ alias
alias cp='cp -i'
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mv='mv -i'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
[kickscar@localhost ~]$ mv test dowork
mv: overwrite `dowork/test'?
```

실습5

의도적으로 덮어쓰기(overwrite)을 하는 경우 옵션 -f 를 사용하여 확인 메시지가 안 나오게 할 수 있다.

```
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 29  2월 14 18:12 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
-rw-r--r--. 1 kickscar wheel  0  2월 14 16:45 test
[kickscar@localhost ~]$ mv -f test dowork
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 29  2월 14 18:27 dowork
drwxr-xr-x. 2 kickscar wheel 17  2월 14 10:47 dowork.bak
[kickscar@localhost ~]$
```


3.4 파일 관리

3.4.5 rm

파일을 삭제 하는 명령이다.

특히, 파일을 삭제하면 복구가 불가능하기 때문에 파일 삭제에는 항상 주의가 필요하다.

사용법

mv [옵션] 파일이름

옵션

-r, -R : 일반 파일이 아닌 디렉터리인 경우에는 그 하위 디렉터리와 파일 까지 모두 삭제 한다.

-i : 파일을 삭제 할 것인지 사용자에게 물어본다.

-f : -i 옵션을 무시 하고 강제로 삭제한다.

실습1

dowork 디렉터리 안의 test~ 파일을 삭제해 보자.

```
[kickscar@localhost ~]$ cd dowork
[kickscar@localhost dowork]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 16:45 test
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:45 test~
[kickscar@localhost dowork]$ rm test~
[kickscar@localhost dowork]$ ls -l
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 16:45 test
[kickscar@localhost dowork]$
```

3.4 파일 관리

실습2

안전한 rm 을 alias 사용하여 설정하는 것이 좋다.

/etc/bashrc 에 rm 명령을 'rm -i' 으로 alias 설정을 하고 삭제한다.

```
[kickscar@localhost dowork]$ su -  
암호:  
[root@localhost ~]#  
[root@localhost ~]# vi /etc/bashrc  
[root@localhost ~]# exit  
logout  
[kickscar@localhost dowork]$ source /etc/bashrc  
[kickscar@localhost dowork]$ ls -l  
합계 0  
-rw-r--r--. 1 kickscar wheel 0  2월 14 16:45 test  
[kickscar@localhost dowork]$ rm test  
rm: remove 일반 빈 파일 `test`?
```

실습3

많은 파일을 지울 때 매번 확인 메시지가 나타나는 것이 번거로울 수 있고 의도적으로 확인 메시지를 피하는 방법은 -f 옵션을 사용하는 것이다.

```
[kickscar@localhost dowork]$ ls -l  
합계 0  
-rw-r--r--. 1 kickscar wheel 0  2월 14 16:45 test  
[kickscar@localhost dowork]$ rm -f test  
[kickscar@localhost dowork]$ ls -l  
합계 0  
[kickscar@localhost dowork]$
```

3.4 파일 관리

실습4

디렉토리를 삭제 할 때는 `rmdir` 보다 `rm -rf` 를 사용하는 것이 일반적이다.

하지만, `-f` 옵션은 디렉토리 안의 많은 파일들의 확인 메시지를 회피하기 위해 사용하는 것이기 때문에 부 주의하게 삭제하는 것에 대한 책임을 지지 않는다. 따라서 주의가 필요하다.

```
[kickscar@localhost ~]$ ls -l dowork.bak/
합계 0
-rw-r--r--. 1 kickscar wheel 0  2월 14 10:47 test
[kickscar@localhost ~]$ rm -rf dowork.bak
[kickscar@localhost ~]$ ls -l
합계 0
drwxr-xr-x. 2 kickscar wheel 6  2월 14 19:34 dowork
[kickscar@localhost ~]$
```

3.4 파일 관리

3.4.6 cat

파일의 내용을 화면에 출력한다.

사용법

cat [옵션] 파일이름

옵션

- n : 줄 번호를 표시한다.
- b : 빈 행은 제외하고 줄 번호를 표시한다.
- E : 각 행마다 끝에 \$ 문자를 출력한다.

실습1

보통 별다른 옵션 없이 cat 명령이 쓰인다. .bashrc의 내용을 확인해 보자.

```
[kickscar@localhost ~]$ cat .bashrc
# .bashrc

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

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

# User specific aliases and functions
[kickscar@localhost ~]$
```

3.4 파일 관리

실습2

.bashrc의 내용에 -n 옵션을 사용해서 줄 번호를 표시 해 보자.

```
[kickscar@localhost ~]$ cat -n .bashrc
1      # .bashrc
2
3      # Source global definitions
4      if [ -f /etc/bashrc ]; then
5          . /etc/bashrc
6      fi
7
8      # Uncomment the following line if you don't like systemctl's auto-paging feature:
9      # export SYSTEMD_PAGER=
10
11     # User specific aliases and functions
[kickscar@localhost ~]$
```

실습3

옵션 -b, -E 를 사용해서 .bashrc의 내용을 확인해 보자.

실습4

/etc/profile 내용을 cat 명령어를 사용해서 화면에 출력해 보고 문제점을 생각해 보자

3.4 파일 관리

3.4.7 more

화면 단위로 분할해서 파일 내용을 출력 한다.

사용법

more [옵션] 파일이름

옵션

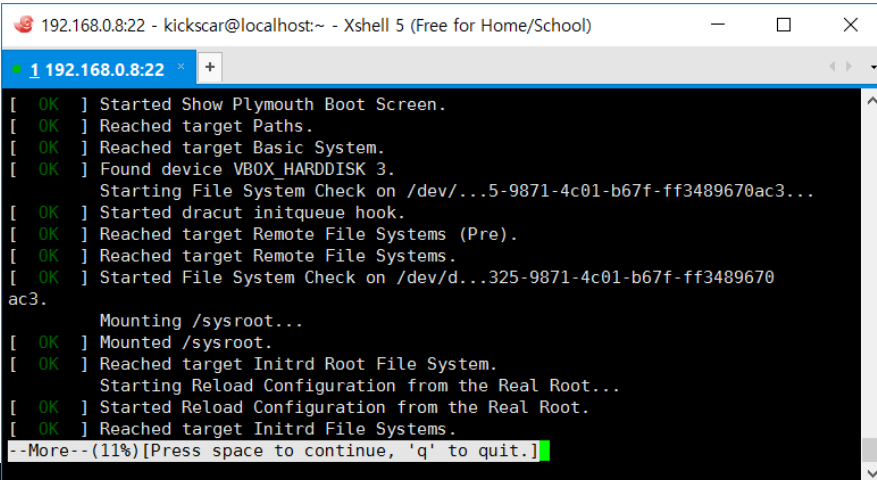
-d : 스페이스나 q키를 누르라는 안내 메시지를 화면에 나타낸다.

-s : 연속되는 빈 공백 행을 하나의 행으로 출력한다.

실습1

보통 별다른 옵션 없이 more 명령을 쓴다. /var/log/boot.log 내용을 more 명령으로 확인해 보자.

엔터와 스페이스 키를 눌러가며 내용을 스크롤 해보자. 그리고 단점이나 불편한 점은 무엇인지 찾아 보자.



```
192.168.0.8:22 - kickscar@localhost:~ - Xshell 5 (Free for Home/School)
1 192.168.0.8:22 * +
[ OK ] Started Show Plymouth Boot Screen.
[ OK ] Reached target Paths.
[ OK ] Reached target Basic System.
[ OK ] Found device VBOX_HARDDISK 3.
Starting File System Check on /dev/...5-9871-4c01-b67f-ff3489670ac3...
[ OK ] Started dracut initqueue hook.
[ OK ] Reached target Remote File Systems (Pre).
[ OK ] Reached target Remote File Systems.
[ OK ] Started File System Check on /dev/d...325-9871-4c01-b67f-ff3489670ac3.
Mounting /sysroot...
[ OK ] Mounted /sysroot.
[ OK ] Reached target Initrd Root File System.
Starting Reload Configuration from the Real Root...
[ OK ] Started Reload Configuration from the Real Root.
[ OK ] Reached target Initrd File Systems.
--More--(11%)[Press space to continue, 'q' to quit.]
```

3.4 파일 관리

3.4.8 find

다양한 조건으로 특정 파일을 찾는다.

사용법

find [시작 디렉터리] [조건]

[시작 디렉터리]부터 시작해서 하위 디렉터리의 모든 파일을 [조건]에 맞는 파일을 검색한다.

[시작 디렉터리]를 / 로 지정하면 시스템 내의 모든 파일을 검색하게 된다.

[조건]은 다양하게 줄 수 있다.

1. -name “문자열”

파일 이름이 문자열과 일치하는 파일을 찾는다. 문자열에는 다음과 같은 형식으로 지정할 수 있다.

“log” : 파일이름이 log인 파일을 찾는다.

“*log” : 파일 이름이 log로 끝나는 모든 파일을 찾는다.

“log*” : 파일 이름이 log로 시작하는 모든 파일을 찾는다.

“*log*” : 파일 이름 중간에 log가 들어가 있는 파일을 찾는다.

“?log” : 파일 이름에서 첫 글자는 어떤 문자라도 상관없고 log로 끝나는 4개 문자 이름의 파일

“log??” : 파일 이름 시작이 log로 시작하고 끝의 두 문자가 어떤 것이어도 상관없는 5개 문자 이름의 파일

실습 1

위의 조건으로 전체 시스템에서 파일을 찾아보자.

3.4 파일 관리

2. -user “유저이름”

특정 유저가 소유한 파일들을 모두 찾는다.

실습 2

현재 로그인 계정이 소유하고 있는 모든 파일을 찾아보자.

```
[kickscar@localhost log]$ find / -user "kickscar"
find: '/boot/grub2': 허가 거부
/dev/pts/0
find: '/proc/tty/driver': 허가 거부
find: '/proc/1/task/1/fd': 허가 거부
...
...

/proc/6921/task/6921/fd/4
/proc/6921/task/6921/fd/5
find: '/proc/6921/task/6921/fd/6': 그런 파일이나 디렉터리가 없습니다

...
...

/home/kickscar/.bash_history
/home/kickscar/.mongorc.js
/home/kickscar/.dbshell
/home/kickscar/dowork
/home/kickscar/.lessht
```

에러를 출력 하지 않기 위해서 2>/dev/null을 추가 해준다.

```
[kickscar@localhost log]$ find / -user "kickscar" 2>/dev/null
```


3.4 파일 관리

3. -perm “퍼미션”

명시된 퍼미션으로 된 파일을 찾을 때 사용한다.

실습 3

/home 디렉터리 아래에 755 퍼미션을 가지고 있는 파일을 찾아보자

```
[kickscar@localhost log]$ su -  
암호:  
마지막 로그인: 일 2월 14 21:58:57 KST 2016 일시 pts/0  
[root@localhost ~]# find /home -perm 755  
/home  
/home/kickscar/dowork  
[root@localhost ~]#
```

4. -type ?

? 형태의 파일을 찾는다.

실습4

/dev 디렉터리 밑에 문자열 디바이스 파일을 찾아 보자

```
[root@localhost ~]# find /dev -type c  
...  
/dev/sg1  
/dev/sg0  
/dev/ppp  
/dev/loop-control  
/dev/uhid  
/dev/btrfs-control  
/dev/mapper/control  
/dev/net/tun  
/dev/vfio/vfio
```

3.4 파일 관리

그 밖에 `-size` 파일크기, `-atime` 날짜, `-newer` 파일 등으로 파일을 찾을 수 있다.

3.4.9 grep

파일 내에서 또는 입력 값으로부터 특정 패턴을 검색한다.

사용법

`grep [옵션] 표현 [파일(들)]`

옵션

`-v` : 일치되는 내용이 없는 라인을 표시한다.

실습1

sshd 프로세스를 확인할 때, `$ ps -ef | grep sshd` 를 사용하게 되면 sshd 프로세스 뿐만 아니라 grep 프로세스도 표시된다.

```
[kickscar@localhost ~]$ ps -ef | grep sshd
root      704      1  0 20:46 ?        00:00:00 /usr/sbin/sshd -D
root      884     704  0 21:27 ?        00:00:00 sshd: kickscar [priv]
kickscar  889     884  0 21:27 ?        00:00:00 sshd: kickscar@nts/0
kickscar  910     890  0 21:28 pts/0    00:00:00 grep --color=auto sshd
```

`$ ps -ef | grep sshd | grep -v grep` 으로 grep process 를 없애자.

3.4 파일 관리

옵션

-c : 일치되는 내용이 있는 행의 개수를 표시한다.

실습2

/var/log/secure 에서 root와 관련된 로그의 개수를 세어 보자.

```
[kickscar@localhost ~]$ su -  
암호:  
마지막 로그인: 화 2월 16 00:47:59 KST 2016 일시 pts/0  
[root@localhost ~]# grep -c root /var/log/secure  
45  
[root@localhost ~]#
```

실습3

실습2 에서 옵션을 사용하지 말고 root를 포함하는 라인을 출력해 보자.

```
[root@localhost ~]# grep root /var/log/secure  
Feb 14 10:57:34 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)  
Feb 14 10:58:31 localhost su: pam_unix(su-l:session): session closed for user root  
Feb 14 11:53:29 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)  
Feb 14 11:54:11 localhost su: pam_unix(su-l:session): session closed for user root  
Feb 14 16:27:19 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)  
...  
...  
...  
Feb 15 22:41:00 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)  
Feb 16 00:52:04 localhost su: pam_unix(su-l:session): session closed for user root  
Feb 16 00:52:14 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)  
[root@localhost ~]#
```

3.4 파일 관리

옵션

-n : 일치되는 내용이 있는 행은 행 번호와 함께 표시된다.

실습4

/etc/profile.d 디렉터리에 있는 모든 파일에서 alias 설정되어 있는 행을 행 번호와 함께 출력해 보자.

```
[root@localhost ~]# grep -n alias /etc/profile.d/*
/etc/profile.d/colorgrep.csh:9:alias grep 'grep --color=auto'
/etc/profile.d/colorgrep.csh:10:alias egrep 'egrep --color=auto'
/etc/profile.d/colorgrep.csh:11:alias fgrep 'fgrep --color=auto'
/etc/profile.d/colorgrep.sh:5:alias grep='grep --color=auto' 2>/dev/null
/etc/profile.d/colorgrep.sh:6:alias egrep='egrep --color=auto' 2>/dev/null
/etc/profile.d/colorgrep.sh:7:alias fgrep='fgrep --color=auto' 2>/dev/null
/etc/profile.d/colorls.csh:13:alias ll 'ls -l'
/etc/profile.d/colorls.csh:14:alias l. 'ls -d .*'
/etc/profile.d/colorls.csh:66:alias ll 'ls -l --color=auto'
/etc/profile.d/colorls.csh:67:alias l. 'ls -d .* --color=auto'
/etc/profile.d/colorls.csh:68:alias ls 'ls --color=auto'
/etc/profile.d/colorls.sh:9: alias ll='ls -l' 2>/dev/null
/etc/profile.d/colorls.sh:10: alias l.='ls -d .*' 2>/dev/null
/etc/profile.d/colorls.sh:55:alias ll='ls -l --color=auto' 2>/dev/null
/etc/profile.d/colorls.sh:56:alias l.='ls -d .* --color=auto' 2>/dev/null
/etc/profile.d/colorls.sh:57:alias ls='ls --color=auto' 2>/dev/null
/etc/profile.d/which2.csh:5:# alias which 'alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
/etc/profile.d/which2.sh:4:alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
[root@localhost ~]#
```

3.4 파일 관리

3.4.10 파이프(pipe)

프로그램의 실행결과를 다른 프로그램의 입력으로 연결한다. 둘 이상의 명령을 함께 사용하고, 한 명령어의 출력결과를 다른 명령어의 입력으로 전환하는 것을 파이프(pipe)라 한다.

실습1

```
[root@localhost ~]# grep root /var/log/secure | less
```

```
Feb 14 10:57:34 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 10:58:31 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 11:53:29 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 11:54:11 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 16:27:19 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 16:28:58 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 18:25:04 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 20:55:51 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 21:13:59 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 21:14:51 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 21:19:16 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 21:22:25 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 21:22:32 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 21:52:59 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 21:53:54 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 21:57:29 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 21:58:57 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 22:01:28 localhost su: pam_unix(su-l:session): session closed for user root
Feb 14 22:15:12 localhost unix_chkpwd[6933]: password check failed for user (root)
Feb 14 22:15:12 localhost su: pam_unix(su-l:auth): authentication failure; logname=kickscar uid=1000 euid=0 tty=pts/0 ruser=kickscar rhost=
user=root
Feb 14 22:15:12 localhost su: pam_succeed_if(su-l:auth): requirement "uid >= 1000" not met by user "root"
Feb 14 22:15:19 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
Feb 14 22:51:12 localhost su: pam_unix(su-l:session): session closed for user root
Feb 15 21:35:34 localhost su: pam_unix(su-l:session): session opened for user root by kickscar(uid=1000)
:
```

3.4 파일 관리

3.4.11 리다이렉션(redirection)

리다이렉션을 이용하면 명령의 출력을 변경할 수 있다. 명령어의 결과는 기본적으로 터미널(stdout) 이다. 리다이렉션을 이용하면 파일에 기록할 수 있다.

연산자

명령어 > 파일 : 파일이 없으면 생성하고, 있다면 기존의 내용을 지운다.

명령어 >> 파일 : 파일이 없다면 생성하고, 있다면 기존의 내용에 추가한다.

명령어 < 파일 : 파일에서 표준 입력(stdin)을 받는다.

실습1

echo는 주어진 문장을 현재 터미널 화면에 출력한다.

```
[kickscar@localhost ~]$ echo "hello"
hello
[kickscar@localhost ~]$
```

실습2

파일에 저장하기 위해 > 리다이렉션 연산자를 사용한다.

```
[kickscar@localhost ~]$ echo "hello" > hello.txt
[kickscar@localhost ~]$ cat hello.txt
hello
[kickscar@localhost ~]$
```

3.4 파일 관리

실습3

hello.txt 파일에 World 라는 단어를 추가해 보자. 기존의 내용에 추가하기 위해 >> 리다이렉션 연산자를 사용하도록 한다.

```
[kickscar@localhost ~]$ echo "world" >> hello.txt
[kickscar@localhost ~]$ cat hello.txt
hello
world
[kickscar@localhost ~]$
```

실습4

몇 개의 단어를 더 추가 해 보도록 하자. (apple, linux, java, red, hat, gnu, unix)

실습5

sort 명령은 표준 입력으로 단어를 입력 받아 순서대로 정렬하는 명령어이다 hello.txt를 표준 입력으로 sort 명령어에 입력 시키고 결과를 확인해 보자.

```
[kickscar@localhost ~]$ sort < hello.txt
apple
gnu
hat
hello
java
linux
red
unix
world
[kickscar@localhost ~]$
```

3.4 파일 관리

실습6

실습5의 결과를 다시 파일 sort.txt로 출력해 보자.

```
[kickscar@localhost ~]$ sort < hello.txt > sort.txt
[kickscar@localhost ~]$ cat sort.txt
apple
gnu
hat
hello
java
linux
red
unix
world
[kickscar@localhost ~]$
```


3.4 파일 관리

3.4.12 chown

파일 소유자나 소유그룹을 변경하기 위한 명령어이다.

사용법

chmod [옵션] 소유자:소유그룹 파일명

옵션

-R : 경로와 그 하위 파일들을 모두 변경한다.

--help : 도움말을 출력한다.

--version : 버전 정보를 보여준다.

실습1

root 로 새로 파일 root.file 를 생성 하고 /home/kickscar 로 복사 한 후, 소유계정과 소유 그룹을 바꿔보자.

```
[root@localhost ~]# touch root.file
[root@localhost ~]# cp root.file /home/kickscar
[root@localhost ~]# chown kickscar /home/kickscar/root.file
[root@localhost ~]# ls -la /home/kickscar/root.file
-rw-r--r--. 1 kickscar root 0  2월 16 03:32 /home/kickscar/root.file
[root@localhost ~]# chown root:wheel /home/kickscar/root.file
[root@localhost ~]# ls -la /home/kickscar/root.file
-rw-r--r--. 1 root wheel 0  2월 16 03:32 /home/kickscar/root.file
[root@localhost ~]# chown kickscar:users /home/kickscar/root.file
[root@localhost ~]# ls -la /home/kickscar/root.file
-rw-r--r--. 1 kickscar users 0  2월 16 03:32 /home/kickscar/root.file
```

3.5 파일 압축 관리

압축 유틸리티 중 가장 많이 사용하는 것은 tar, gzip, bzip2 이다.

3.5.1 tar

tar 기본적으로 압축을 하지 않고 묶어 주는 역할(아카이브, archive)을 한다.

사용법

묶을 때 : tar [옵션] 생성파일.tar 묶을 파일

해제할 때 : tar [옵션] 파일.tar

옵션

-c : 새 저장 파일을 만든다. 즉 묶을 때 사용하는 옵션이다.

-x : 묶인 파일을 해제 한다.

-v : 처리 중인 파일을 자세하게 보여준다.

-f : 파일을 지정한다.

실습1

보통 tar를 사용해서 파일을 묶을 때는 -cvf 옵션을 사용한다. /home/kickscar 폴더를 kickscar.tar로 묶어 보자.

3.5 파일 압축 관리

실습1

보통 tar를 사용해서 파일을 묶을 때는 -cvf 옵션을 사용한다. /home/kickscar 폴더를 kickscar.tar로 묶어 보자.

```
[root@localhost ~]# tar -cvf kickscar.tar /home/kickscar
tar: Removing leading `/' from member names
/home/kickscar/
/home/kickscar/.bash_logout
/home/kickscar/.bash_profile
/home/kickscar/.bashrc
/home/kickscar/.bash_history
/home/kickscar/.mongorc.js
/home/kickscar/.dbshell
/home/kickscar/dowork/
/home/kickscar/.lessht
/home/kickscar/hello.txt
/home/kickscar/sort.txt
/home/kickscar/root.file
```

실습2

tar 파일을 풀때에는 보통 -xvf 옵션을 사용하게 된다. 압축을 풀어보자.

3.5 파일 압축 관리

실습3

tar는 압축을 하지 않고 묶기 때문에 tar 파일은 압축을 하게 된다. gzip으로 압축해 보자

```
[root@localhost ~]# gzip kickscar.tar
[root@localhost ~]# ls -l
합계 62088
-rw-----. 1 root root    1066  2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root   4539  2월 16 03:53 kickscar.tar.gz
-rw-r--r--. 1 root root      0  2월 16 03:32 root.file
[root@localhost ~]#
```

실습4

gz 파일을 압축 해제는 gzip -d 또는 gunzip 명령어를 사용한다.

```
[root@localhost ~]# gzip -d kickscar.tar.gz
[root@localhost ~]# ls -l
합계 62112
-rw-----. 1 root root    1066  2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root   30720  2월 16 03:53 kickscar.tar
-rw-r--r--. 1 root root      0  2월 16 03:32 root.file
[root@localhost ~]#
```

3.5 파일 압축 관리

실습5

tar의 -z 옵션을 사용하면 각 각 gzip 를 사용해서 묶인 파일에 대해 압축을 하거나 압축을 해제 할 수 있다.

```
[root@localhost ~]# tar -cvzf kickscar.tar.gz /home/kickscar
tar: Removing leading `/' from member names
/home/kickscar/
/home/kickscar/.bash_logout
/home/kickscar/.bash_profile
/home/kickscar/.bashrc
/home/kickscar/.bash_history
/home/kickscar/.mongorc.js
/home/kickscar/.dbshell
/home/kickscar/dowork/
/home/kickscar/.lessht
/home/kickscar/hello.txt
/home/kickscar/sort.txt
/home/kickscar/root.file
[root@localhost ~]# ls -l
합계 62120
-rw-----. 1 root root   1066  2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root   4526  2월 16 04:03 kickscar.tar.gz
-rw-r--r--. 1 root root      0  2월 16 03:32 root.file
[root@localhost ~]#
```

실습 6

-xvzf 를 사용해서 압축과 함께 묶인 것을 풀어 보자

```
[root@localhost ~]# tar -xvzf kickscar.tar.gz
```

3.5 파일 압축 관리

실습7

tar의 -j 옵션을 사용하면 bzip2 압축 또는 해제를 하게 된다.

```
[root@localhost ~]# tar -cvjf kickscar.tar.bz2 /home/kickscar/
tar: Removing leading `/' from member names
/home/kickscar/
/home/kickscar/.bash_logout
/home/kickscar/.bash_profile
/home/kickscar/.bashrc
/home/kickscar/.bash_history
/home/kickscar/.mongorc.js
/home/kickscar/.dbshell
/home/kickscar/dowork/
/home/kickscar/.lessht
/home/kickscar/hello.txt
/home/kickscar/sort.txt
/home/kickscar/root.file
[root@localhost ~]# ls -l
합계 62088
-rw-----. 1 root root    1066  2월 11 17:32 anaconda-ks.cfg
-rw-r--r--. 1 root root   4204  2월 16 04:12 kickscar.tar.bz2
-rw-r--r--. 1 root root      0  2월 16 03:32 root.file
[root@localhost ~]#
```

실습 8

-xvjf 를 사용해서 압축과 함께 묶인 것을 풀어 보자

```
[root@localhost ~]# tar -xvjf kickscar.tar.gz
```

3.5 파일 압축 관리

[실습과제]

JDK 다운받아 설치하고 Java 실행/개발 환경을 설정해 보자.

1. 다운로드는 <http://www.oracle.com/technetwork/java/javase/downloads/index.html> 에서 할 수 있다.
2. wget을 사용해서 jdk를 다운로드 한다.

```
[root@localhost ~]# wget --no-cookies --no-check-certificate --header "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" "http://download.oracle.com/otn-pub/java/jdk/8u72-b15/jdk-8u72-linux-x64.tar.gz" --2016-02-16 04:23:36-- http://download.oracle.com/otn-pub/java/jdk/8u72-b15/jdk-8u72-linux-x64.tar.gz
```

3. 압축을 푼다.
4. /usr/local 밑에 압축을 푼 디렉터리(jdk1.8.x_xx)를 복사한다.

```
[root@localhost local]# ls -l /usr/local
합계 4
drwxr-xr-x. 2 root root    6  8월 12 2015 bin
drwxr-xr-x. 2 root root    6  8월 12 2015 etc
drwxr-xr-x. 2 root root    6  8월 12 2015 games
drwxr-xr-x. 2 root root    6  8월 12 2015 include
lrwxrwxrwx. 1 root root   11  2월 16 04:31 jdk -> jdk1.8.0_72
drwxr-xr-x. 8 root root 4096 12월 23 15:13 jdk1.8.0_72
drwxr-xr-x. 2 root root    6  8월 12 2015 lib
drwxr-xr-x. 2 root root    6  8월 12 2015 lib64
drwxr-xr-x. 2 root root    6  8월 12 2015 libexec
drwxr-xr-x. 2 root root    6  8월 12 2015 sbin
drwxr-xr-x. 5 root root   46  2월 11 17:27 share
drwxr-xr-x. 2 root root    6  8월 12 2015 src
```

3.5 파일 압축 관리

[실습과제]

5. 환경 설정을 위해 vi /etc/profile 을 열어 다음을 추가 하고 저장 한다.

```
#java
export JAVA_HOME=/usr/local/jdk
export PATH=$PATH:$JAVA_HOME/bin
export CLASSPATH=.:$JAVA_HOME/lib/tools.jar
```

6. 변경 내용을 쉘 환경에 적용하기 위해서 source 명령 또는 다시 로그인 하자

7. java 개발환경/실행환경을 테스트 해보자.

```
[kickscar@localhost ~]$ javac -version
javac 1.8.0_72
[kickscar@localhost ~]$ java -version
java version "1.8.0_72"
Java(TM) SE Runtime Environment (build 1.8.0_72-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.72-b15, mixed mode)
[kickscar@localhost ~]$
```


3.5 파일 압축 관리

8. 적당한 작업 디렉토리에서 HelloWorld.java 를 생성하고 컴파일하고 실행해보자.

```
[kickscar@localhost dowork]$ vi HelloWorld.java
[kickscar@localhost dowork]$ ls -la
합계 8
drwxr-xr-x. 2 kickscar wheel  28  2월 16 04:43 .
drwx-----. 4 kickscar wheel 4096  2월 16 04:40 ..
-rw-r--r--. 1 kickscar wheel  112  2월 16 04:43 HelloWorld.java
[kickscar@localhost dowork]$ javac HelloWorld.java
[kickscar@localhost dowork]$ ls -la
합계 12
drwxr-xr-x. 2 kickscar wheel  51  2월 16 04:43 .
drwx-----. 4 kickscar wheel 4096  2월 16 04:40 ..
-rw-r--r--. 1 kickscar wheel  425  2월 16 04:43 HelloWorld.class
-rw-r--r--. 1 kickscar wheel  112  2월 16 04:43 HelloWorld.java
[kickscar@localhost dowork]$ java HelloWorld
Hello World
[kickscar@localhost dowork]$
```