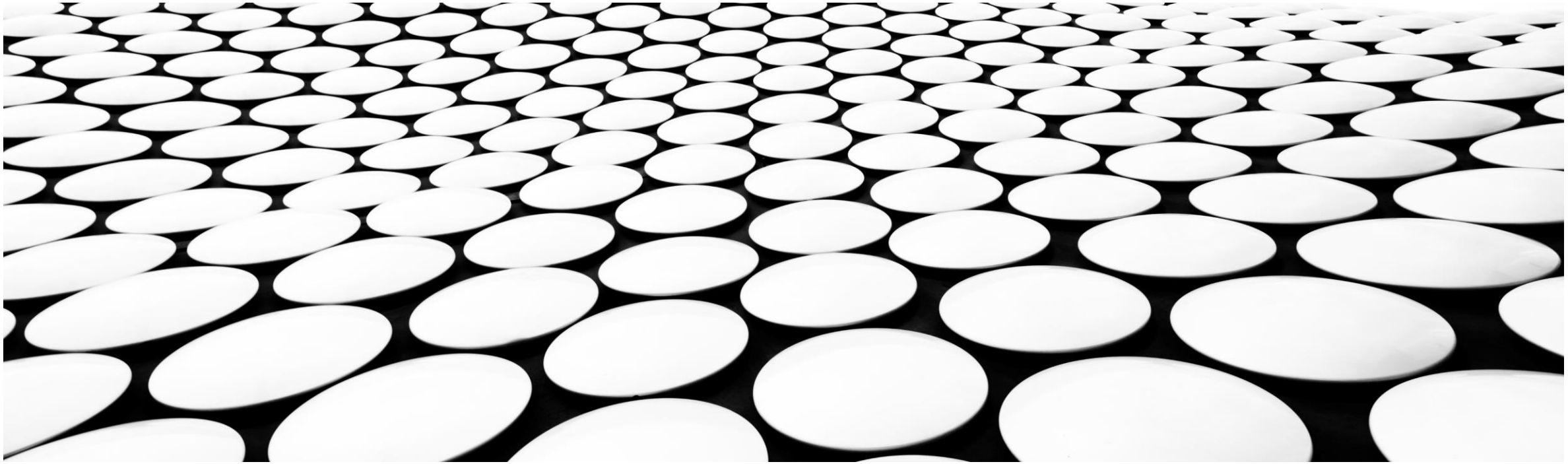
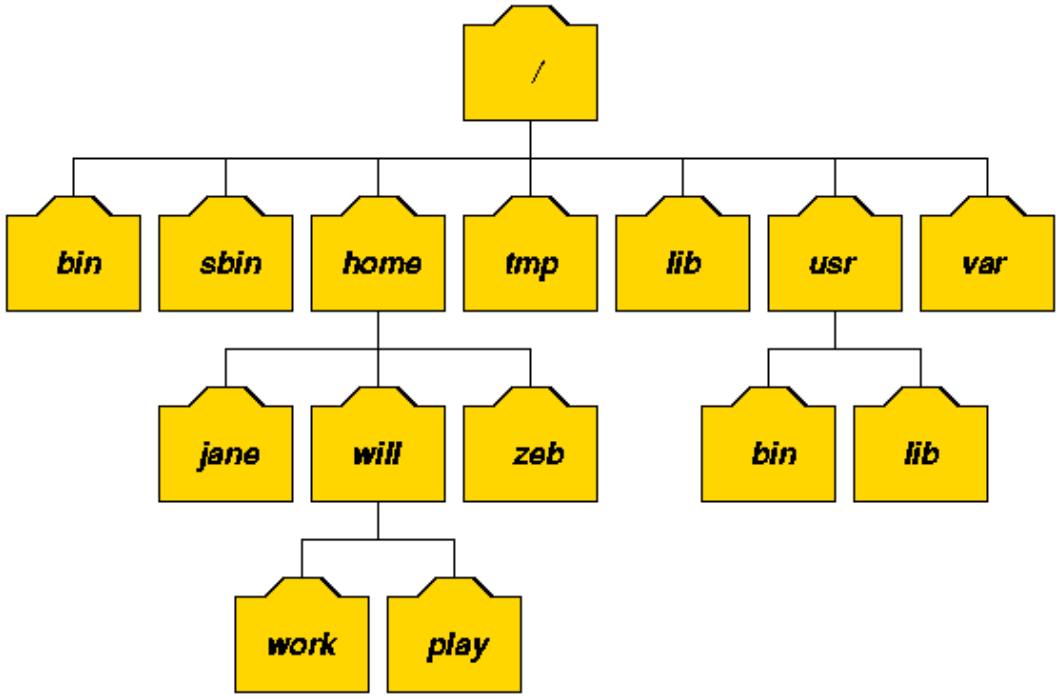
LINUX의 이해



LINUX – 디렉토리 구조



- **리눅스 파일 시스템 구조**
  - 리눅스 시스템의 디렉토리 구조는 전체적으로 역 트리(tree) 구조를 하고 있다. 그리고 명령어의 종류와 성격, 사용권한등에 따라 각각의 디렉토리들로 구분된다.
  - 리눅스 배포 판들은 ' 리눅스 파일시스템 표준 ' 인 **FSSTND (LINUX FILE System Standard)** 라는 표준을 준수하므로 대부분의 리눅스 배포 판들은 그 기본 골격이 같다.

LINUX – 디렉토리 구조

- **/(루트)**
  - 최상의 디렉토리인 **루트** 디렉토리를 의미하며, 리눅스의 모든 디렉토리들의 시작점이다. 즉, 모든 디렉토리들을 절대 경로로 표기할 때에 이 디렉토리로부터 시작해야 한다.
- **/bin**
  - 기본적인 명령어가 저장된 디렉토리. 즉, 리눅스 시스템사용에 있어 가장 기본적이라고 할 수 있는 mv, cp, rm 등과 같은 명령어들이 이 디렉토리에 존재하며 root 사용자와 일반사용자가 함께 사용할 수 있는 명령어 디렉토리이다.
- **/boot**
  - 리눅스 부트로더(Boot Loader)가 존재하는 디렉토리. 즉, GRUB 과 같은 부트로더에 관한 파일들(grub.conf 등)이 이 디렉토리에 존재한다.
- **/dev**
  - 시스템 디바이스(device)파일을 저장하고 있는 디렉토리. 즉, 하드디스크 장치파일 /dev/sda, CD-ROM 장치파일 /dev/cdrom 등과 같은 장치파일들이 존재하는 디렉토리이다.
- **/etc**
  - 시스템의 거의 모든 설정 파일이 존재하는 디렉토리. /etc/sysconfig(시스템 제어판용 설정파일), /etc/passwd(사용자관리 설정파일), /etc/named.conf(DNS 설정파일) 등과 같은 파일들이 존재한다.
- **/etc/mail/**
  - sendmail.cf 나 access 파일등의 sendmail 의 설정파일들이 존재하는 디렉토리.
- **/etc/ssh/**
  - SSH 서비스, 즉 sshd 데몬에서 사용하는 각종 설정파일들이 존재하는 디렉토리.
- **/etc/squid/**
  - squid 프락시서버의 설정파일들이 저장된 디렉토리.
- **/etc/samba/**
  - 삼바관련 설정파일들이 저장된 디렉토리
- **/etc/skel/**
  - 계정 사용자 생성시의 초기화파일들이 저장된 디렉토리(useradd 에서 사용함)

LINUX – 디렉토리 구조

- **/etc/rc.d/**
  - 부팅레벨별 부팅스크립트파일들이 존재하는 디렉토리.
- **/etc/rc.d/init.d/**
  - 시스템 초기화 파일들의 실제파일들이 존재함.
- **/etc/pam.d/**
  - PAM 설정 정보파일들이 저장된 디렉토리.
- **/etc/httpd/**
  - RPM 으로 설치된 아파치 설정파일(httpd.conf 등)들이 저장된 디렉토리.
- **/etc/cron.d/, /etc/cron.daily/, /etc/cron.hourly/, /etc/cron.monthly/, /etc/cron.weekly/**
  - 모두 크론 설정 파일이 존재하는 디렉토리임.
- **/etc/xinetd.d/**
  - xinetd 수퍼데몬에 의해 서비스되는 서비스설정파일이 존재함.
- **/home**
  - 사용자의 홈 디렉토리, useradd 명령어로 새로운 사용자를 생성하면 대부분 사용자의 ID와 동일한 이름의 디렉토리가 자동으로 생성됨.
- **/lib**
  - 커널모듈파일과 라이브러리파일 즉, 커널이 필요로 하는 커널모듈파일들과 프로그램(C, C++ 등)에 필요한 각종 라이브러리 파일들이 존재하는 디렉토리.
- **/media**
  - DVD, CD-ROM, USB 등과 같은 탈 부착이 가능한 장치들의 마운트포인트로 사용되는 디렉토리.
- **/mnt**
  - /media 디렉토리와 비슷한 용도로 탈 부착이 가능한 장치들에 대하여 일시적인 마운트포인트로 사용하는 디렉토리.

LINUX – 디렉토리 구조

- **/proc**
  - 일명 "가상파일시스템" 이라고 하는 곳으로 현재 메모리에 존재하는 모든 작업들이 파일형태로 존재하는 곳이다. 디스크상 에 실제 존재하는 것이 아니라 메모리상에 존재하기 때문에 가상 파일 시스템이라고 부른다. 실제 운용 상태를 정확하게 파악할 수 있는 중요한 정보를 제공하며 여기에 존재하는 파일들 가운데 현재 실행중인 커널(kernel)의 옵션 값을 즉시 변경할 수 있는 파라미터 파일들이 있기 때문에 시스템 운용에 있어 매우 중요한 의미를 가진다.
- **/root**
  - 시스템 최고관리자인 root 사용자의 개인 홈 디렉토리.
- **/sbin**
  - ifconfig, e2fsck, ethtool, halt 등과 같이 주로 시스템 관리자들이 사용하는 시스템관리자용 명령어를 저장하고 있는 디렉토리.
- **/tmp**
  - 일명 "공용디렉토리" . 시스템을 사용하는 모든 사용자들이 공동으로 사용하는 디렉토리. mysql에서 사용하는 mysql.sock 등과 같은 소켓파일, 또는 아파치에서 사용하는 세션파일등이 생성되기도 한다. 웹해킹에 사용되기도 해서 주의를 요망.
- **/usr**
  - 시스템이 아닌 일반사용자들이 주로 사용하는 디렉토리. 즉, c++, chsh, cpp, crontab, du, find등과 같이 일반사용자들용 명령어들은 /usr/bin 에 위치한다.
- **/usr/bin/**
  - 일반 사용자들이 사용가능한 명령어 파일들이 존재하는 디렉토리.
- **/usr/X11R6/**
  - X 윈도우 시스템의 루트 디렉토리.
- **/usr/include/**
  - C 프로그램에 필요한 헤드파일(*.h) 디렉토리.
- **/usr/lib/**
  - /lib 에 들어가지 않은 라이브러리 디렉토리.
- **/usr/sbin/**
  - /bin 에 제외된 명령어와 네트워크관련 명령어가 들어있는 디렉토리.
- **/usr/src/**
  - 프로그램 소스(주로 커널 소스)가 저장되는 디렉토리.
- **/usr/local/**
  - MySQL, Apache, PHP 등과 같은 어플리케이션들을 소스로 컴파일 설치할 때 사용되는 장소.
- **/usr/share/man/**
  - 명령어들의 도움말을 주는 메뉴얼(manual)페이지 디렉토리. 즉, 이 디렉토리에는 시스템에서 사용하는 모든 맨페이지파일(man page)이 존재함.

LINUX – 디렉토리 구조

- **/var**
  - 시스템운용중에 생성되었다가 삭제되는 데이터를 일시적으로 저장하기 위한 디렉토리. 거의 모든 시스템로그파일은 /var/log 에 저장되고, DNS 의 zone 설정 파일은 /var/named 에 저장되고, 메일 파일은 /var/spool/mail 에 저장되며, 크론 설정파일은 /var/spool/cron 디렉토리에 각각 저장됨.
- **/var/tmp/**
  - /tmp 디렉토리와 같은 공용디렉토리. 즉, /tmp 디렉토리와 /var/tmp 디렉토리의 퍼미션은 1777 로서 sticky bit 가 설정되어 있는 공용디렉토리이다. 리눅스 시스템에서 공용디렉토리는 /tmp 와 /var/tmp 둘뿐이다.
- **/var/log/**
  - 시스템로그파일(messages, secure, xferlog 파일등)이 저장되는 디렉토리.
- **/var/ftp/**
  - vsftp 등과 같은 FTP 서비스를 위한 다운로드 될 파일들 즉, FTP 홈 디렉토리.
- **/var/named/**
  - BIND 즉, DNS 에서 사용하는 zone 파일들이 저장되는 디렉토리.
- **/var/spool/mail/**
  - 각 계정사용자들의 메일 파일이 저장되는 디렉토리.
- **/var/spool/lpd/**
  - 프린트를 하기 위한 임시 디렉토리(스풀링 디렉토리).
- **/var/spool/mqueue/**
  - 발송을 위한 메일 일시 저장 디렉토리.
- **/var/spool/cron/**
  - 각 사용자들의 cron 설정파일들이 저장된 디렉토리.
- **/var/spool/at/**
  - atd 즉, 예약 작업에 관한 파일들이 저장되는 디렉토리.
- **/lost+found**
  - 최상위 디렉토리인 / 디렉토리에만 존재하는 것이 아니라 파일시스템마다 존재할 수 있는 디렉토리임. 이 디렉토리는 fsck 또는 e2fsck 등과 같은 파일시스템 체크 및 복구유틸리티 실행 후에 주로 생성이 되는 것으로서 복구되지 않은 채로 블록(block)만 존재하는 파일 즉, 연결이 끊어진 inode 들이 숫자파일형태로 존재하는 곳임. 숫자 형태로 존재하는 파일들은 mv 명령어로 파일이름만 바꾸면 바로 복구될 수 있다.

LINUX – CLI(COMMAND LINE INTERFACE)

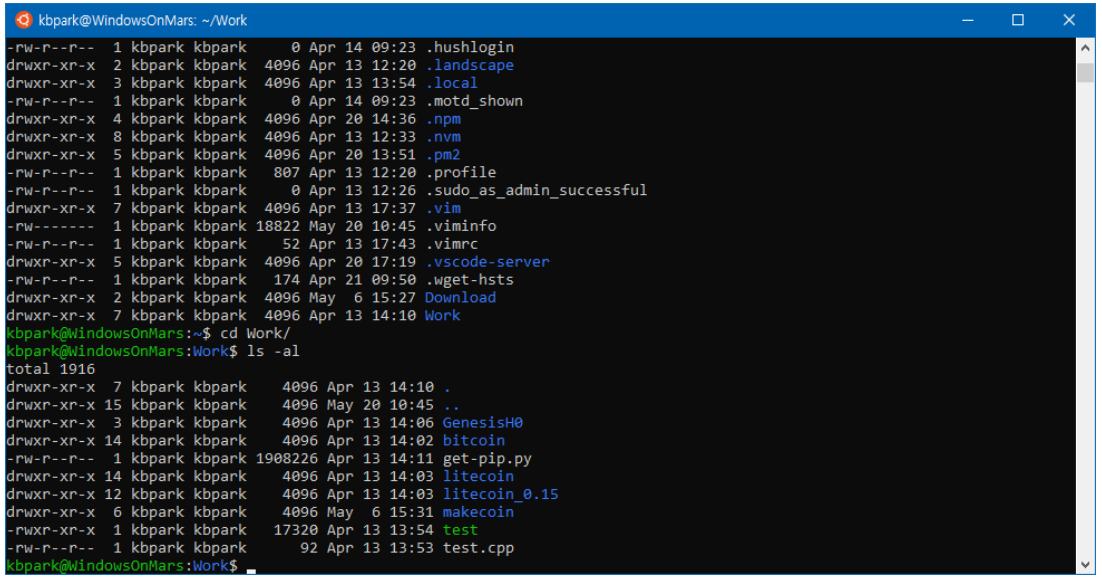
- **ls**, mkdir, cd, etc
- build-essential 설치, hello world, file attribute

LINUX – CLI(COMMAND LINE INTERFACE)

ls 옵션	의 미
-a	.(점)을 포함한 경로안의 모든 파일과 디렉토리 표시
-l	지정한 디렉토리의 내용을 자세히 출력
-d	지정된 디렉토리의 정보 출력
-n	파일 및 디렉토리 정보 출력시, UID,GID를 사용
-R	하위 경로와 그안에 있는 모든 파일들도 같이 나열
-F	파일 형식을 알리는 문자를 각 파일 뒤에 추가

- ls --help
  - Usage: ls [OPTION]... [FILE]...
  - List information about the FILEs (the current directory by default).
  - Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.
  - .....

LINUX – CLI(COMMAND LINE INTERFACE)



- drwxr-xr-x 2 kbpark kbpark 4096 May 6 15:31 makecoin

- 파일Type 퍼미션정보 링크수 소유자 소유그룹 용량 생성날짜 파일이름
- 파일 Type : "d" -> 디렉토리 , "l" -> 링크파일 , "-" -> 일반파일 등등..
- 퍼미션정보 : 해당 파일에 어떠한 퍼미션이 부여되어있는 지 표시!
- 링크수 : 해당 파일이 링크된 수! 링크는 윈도우의 "바로가기"와 같다. "in [대상파일] [링크파일]" 명령으로 링크 파일을 만든다.
- 소유자 : 해당 파일의 소유자이름!
- 소유그룹 : 해당 파일을 소유한 그룹이름! 특별한 변경이 없을 경우 소유자가 속한 그룹이 소유그룹으로 지정된다.
- 용량 : 파일의 용량
- 생성날짜 : 파일이 생성된 날짜!
- 파일이름 : 파일이름

LINUX – CLI(COMMAND LINE INTERFACE)

- 퍼미션 종류 , () 괄호에 있는 것이 해당 퍼미션 기호
  - 읽기 (r) : 파일의 읽기 권한
  - 쓰기 (w) : 파일의 쓰기 권한
  - 실행 (x) : 파일의 실행 권한
- 퍼미션의 사용자지정
  - 소유자 : 소유자에 대한 퍼미션 지정
  - 그룹 : 소유그룹에 대한 퍼미션 지정
  - 공개 : 모든 사용자들에 대한 퍼미션 지정
- 퍼미션 정보
  - rwxr-xr-x
  - 기호의 종류는 (r w x) 3개인데.. 퍼미션 정보에는 총9개가 표시
  - 일단 세 자리씩 끊어봅시다. (rwx r-x r-x)
  - "소유자 : rwx , 그룹 : r-x , 공개 : r-x" (이때 '-' 기호는 그 퍼미션은 없다는 기)
  - 해석해보면 " 이 파일에 대해서 소유자는 읽기(r),쓰기(w),실행(x)을 허용하고,
  - 파일의 소유 그룹에 속하고 있는 사용자들은 읽기(r),실행(x)만 허용하고,
  - 이외에 나머지 모든 사용자들도 읽기(r),실행(x)만 허용한다. "

LINUX – CLI(COMMAND LINE INTERFACE)

- 퍼미션 변경하기
  - 파일이 생성될때 기본적인 퍼미션이 부여된다.
  - "chmod" 명령을 사용하면 아주 손쉽게 퍼미션을 변경할 수 있다.
  - **chmod [변경될 퍼미션값] [변경할 파일]**
  - 각 퍼미션 기호를 숫자로 변환 한다. (r = 4 , w = 2 , x = 1)
  - 예) r - x 인 경우 4 0 1
  - 변환한 숫자를 합산한다.
  - 예) 4 0 1 인 경우 4+0+1 = 5
  - 이런 식으로 하나의 퍼미션을 숫자 값으로 변환하면 된다.
  - 예) rwxr-xr-x 이면 rwx r-x r-x 세자리씩 끊고, 4+2+1 | 4+0+1 | 4+0+1 숫자 변환 뒤 합산하면 "755" 라는 퍼미션 값이 나온다.
  - " chmod 755 conory.text " 명령을 실행하면 conory.text 파일이 755에 해당되는 퍼미션으로 변경된다.
  - 디렉토리의 경우 "-R" 옵션을 사용하면 하위 디렉토리의 모든 디렉토리 및 파일의 퍼미션이 변경된다.
  - 예) " chmod –R 777 conory " conory 디렉토리의 하위에 위치한 모든 파일및 디렉토리 퍼미션이 777로 변경된다.

필요 툴 설치

- Nodejs
  - Nvm(Node version manager) 설치 후 nodejs LTS 버전 설치

    curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
    또는
    wget -qO- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash

  - Nodejs scrypto 사용
- MySQL 혹은 MariaDB

과제

- Nodejs 및 MySQL 또는 MariadDB를 이용 한 사용자 가입 및 로그인 사이트 개발
  - Nodejs 및 DB 설치 필요
- 비밀번호 저장은 SHA-256으로 저장
- React와 함께 개발 하여도 무방 함

LINUX – CLI(COMMAND LINE INTERFACE)

- 소유자 변경하기
  - 파일의 소유자 양도를 해야 될 경우 소유자 변경을 할 수 있다.
  - chown [변경할 소유자] [변경할 파일]
  - 이 명령으로 소유자 뿐만 아니라 소유 그룹도 변경할 수 있다.
  - [변경할 소유자]란에 ".그룹이름" 형식으로 입력하면 된다. " .conory "
  - 예를 들어 “conory.text”의 소유자를 " conory " 로, 소유 그룹을 " conory2 " 로 동시에 변경할 경우 " chown conory.conory2 conory.text”

LINUX 사용자 권한

다중 사용자

- 유닉스 계열 운영체제는 여러 명이 함께 사용할 수 있는 기능을 가지고 있다.
- 다중 사용자 시스템이 되면 시스템의 복잡도가 높아진다.
- $ id: uid(유저 아이디), gid(그룹 아이디)
- $ who: 현재 접속한 사용자 리스트

관리자와 일반 사용자 (Superuser vs. User)

- 관리자: superuser(root user)
- $ sudo [명령어]: 일시적으로 super user의 권한으로 명령 실행
- super user 되는 방법
  - $ su -: superuser 되기
    - ✓ root user의 홈디렉토리는 /root (일반사용자의 홈디렉토리는 /home/username)
  - $ su : superuser 되기
    - ✓ 이때는 user의 홈디렉토리 /home/username 현재 디렉토리가 된다.
  - # exit: logout

LINUX 사용자 권한

사용자의 추가 (Add User)

- $ sudo useradd -m username: username으로 사용자 추가
  - -m: 홈 디렉토리를 같이 만들어줌 (`/home/username)
- $ sudo passwd username: username의 password 지정
- $ su - username: username으로 접속
- $ sudo usermod -a -G sudo username: (sudo 가능한 유저로 접속해서 실행할 것) username이 sudo 명령을 할 수 있도록 지정.
  - sudo group에 username을 추가하는 것
- sudo deluser username sudo: username의 sudo 권한을 삭제하기

LINUX 사용자 권한

권한 (Permission)

- User가 파일/디렉토리에 대해 읽기/쓰기/실행을 할 수 있는 권한을 지정
- ls -l: 파일의 권한, 소유자 등의 정보 확인 가능
- 권한: -rw-rw-r--와 같은 형태 (10자리가 아래와 같이 나눠짐)
  - 첫번째 자리=type: -(파일) d(디렉토리)
  - 다음 세자리=owner의 access mode: r(읽기), w(쓰기), x(실행)
  - 다음 세자리=group의 access mode: r(읽기), w(쓰기), x(실행)
  - 다음 세자리=other의 access mode: r(읽기), w(쓰기), x(실행)

- 권한 변경: chmod 명령어 사용
  - $ chmod [options] mode file 형태
  - $ chmod o-r [filename]: [filename]에 대해 other의 read 권한을 빼기
  - $ chmod o+r [filename]: [filename]에 대해 other의 read 권한을 추가하기
  - $ chmod o+w [filename]: [filename]에 대해 other의 write 권한을 추가하기
  - $ chmod u-r [filename]: [filename]에 대해 user(소유자)의 read 권한을 빼기
  - $ chmod u+r [filename]: [filename]에 대해 user(소유자)의 read 권한을 추가하기
- 실행(execute) 권한
  - $ chmod u+x [filename]: user가 해석기 프로그램 없이 [filename]을 바로 실행할 수 있게 함

LINUX 사용자 권한

권한 (Permission)

- directory에 대한 권한
  - read 권한: 해당 디렉토리 안의 파일/디렉토리를 볼 수 있는가
  - write 권한: 해당 디렉토리 안에 파일/디렉토리를 생성/삭제/변경할 수 있는가
  - execute 권한: 해당 디렉토리에 들어갈 수 있는가 (cd 명령)
  - $ chmod -R o+w [dirname]: [dirname] 안의 모든 디렉토리에 대해 재귀적으로 other의 write 권한을 추가하기
- chmod octal modes
  - Ex) $ chmod 111 [filename]: 모든 사용자가 실행만 가능하게 변경

#	Sum	rwx	Permission
7	4(r) + 2(w) + 1(x)	rwx	read, write and execute
6	4(r) + 2(w)	rw-	read and write
5	4(r) + 1(x)	r-x	read and execute
4	4(r)	r--	read only
3	2(w) + 1(x)	-wx	write and execute
2	2(w)	-w-	write only
1	1(x)	--x	execute only
0	0	---	none

LINUX 사용자 권한

권한 (Permission)

- 그룹 (Group)
- 특정한 사용자 그룹에 대해서 파일/디렉토리에 대한 권한을 지정할 수 있음
  - $ useradd -G [group_name] [username]: group에 사용자를 생성해서 추가
  - $ sudo groupadd [group_name]: 그룹 생성
  - /etc/group파일에 해당 그룹이 추가되었는지 확인
  - $ sudo usermod -a -G [group_name] [username]: username을 group_name에 추가하기
  - $ sudo chown [owner_name:group_name] [file or dir]: owner & group 변경
  - $ sudo chmod g+w [file or dir]: group에 w 권한 추가

LINUX – BASH PROMPT

쉘변수 기호	의미
\t	24시간의 단위로 현재시각을 HH:MM:SS 로 표시
\T	12시간의 단위로 현재시각을 HH:MM:SS 로 표시
\@	12시간의 단위로 현재시각을 오전/오후 로 표시
\d	현재 날짜를 나타냄. 요일, 월, 일 형식으로
\s	현재 사용중인 쉘의 이름을 나타냄 (C쉘이면 /bin/csh, bash쉘이면 /bin/bash)
\w	현재 디렉토리의 전체 절대경로를 모두 표시함
\W	현재 디렉토리의 전체 절대경로명 중 마지막 디렉토리명만을 표시함. 즉 현재디렉토리만 표시함
\u	사용자명을 표시함

쉘변수 기호	의미
\h	서버의 호스트명을 표시함 (www.uzuro.com에서 www 부분)
\H	서버의 도메인명을 표시함 (www.uzuro.com에서 uzuro.com 부분)
\#	접속한 순간부터 사용한 명령어의 번호를 1번부터 차례대로 표시함
\!	사용한 명령어의 history 번호를 표시함
\\$	현재 사용자가 root(uid 가 0)이면 # 을 표시하고 아니면 $ 를 표시함
\\	'\' 문자 자체를 표시함
\a	ASCII 종소리 문자 (07)
\e	ASCII 의 escape 문자 (033)
\n	개행문자 (줄바꿈)
\v	사용중인 bash 의 버전
\V	사용중인 bash 의 배포, 버전+패치수준으로 버전을 상세히 표시함
\r	Carrage retrun
\nnn	8진수 nnn 에 해당하는 문자

LINUX – PROCESS

- 프로세스 목록 확인
  - ~$ ps
- 프로세스 목록 확인 - 자세한 정보
  - ~$ ps -f
- 모든 프로세스 리스트 확인
  - ~$ ps -e
  - ~$ ps -ef
- 프로세스 목록 배열 및 시스템 자원 사용률 확인
  - ~$ ps -aux

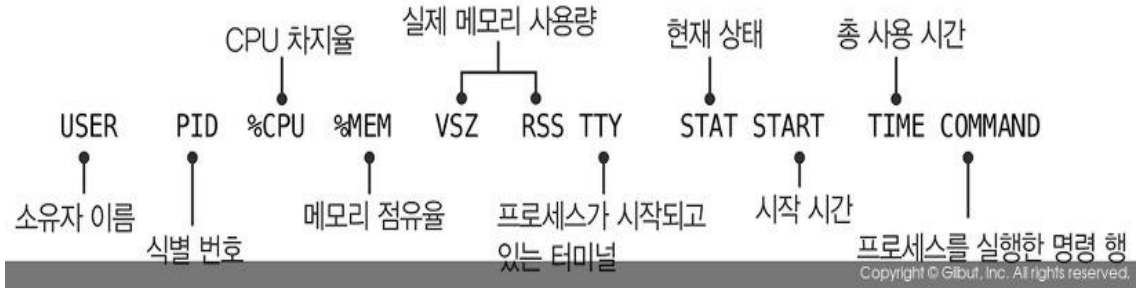
LINUX – PROCESS

- ps -f 명령으로 나타나는 항목들의 의미

항목	의미
UID	프로세스의 실행 / 소유자 아이디
PID	프로세스의 고유 번호(Process Identification Number)
PPID	부모 프로세스의 PID(Parent PID)
C	프로세스 우선순위
STIME	프로세스가 시작된 시간
TTY	프로세스와 연결된 터미널
TIME	실행에 걸린 시간
CMD	프로세스를 생성하는데 내린 명령

LINUX – PROCESS

- 프로세스 상태를 나타내는 STAT 항목
  - R(Runable) : 실행 대기 상태
  - S(Sleeping) : 수면 상태
  - D(inDIskwait) : 입출력을 기다리는 상태
  - T(sTopped) : 멈춰 있거나 흔적이 남아있는 상태
  - Z(Zombie) : 죽었지만 프로세스에 남아있는 상태 (자원 낭비)
- ps [-옵션]
  - a (All processes) : 프로세스 현황 표시
  - u (User) : 유저 지향적 (top 포맷)
  - x : 터미널 제어 없이 프로세스 현황 보기
  - e (All processes) : 현재 시스템 내에서 실행중인 모든 프로세스 정보를 출력
  - f (Full listing) : 모든 정보 확인
  - o (User) : 유저 포맷팅



LINUX – PROCESS

- 모든 프로세스의 트리 확인
  - ~$ pstree
- 특정 프로세스 정보 확인
  - ~$ ps -ef | grep 프로세스이름
- 프로세스의 PID 확인 (1)
  - ~$ pgrep 프로세스이름(일부 가능)
- 프로세스의 PID 확인 (2)
  - ~$ pidof 프로세스이름(일부 불가능, 전체 이름 입력)
- 지속적으로 현재 실행 중인 프로세스의 목록 확인
  - ~$ top
- 특정 프로세스 상태 확인
  - ~$ top | grep 프로세스이름
- 프로세스 죽이기
  - ~$ kill -9 PID
  - Ex : kill -9 1234
- 프로세스 시그널 종류 출력
  - ~$ kill –l

LINUX – PROCESS

- kill [-옵션]
- 자주쓰는 옵션
  - SIGHUP (1) : 대기
  - SIGKILL (9) : 강제 종료
  - SIGTERM (15) : 종료

신호	이식 가능한 번호	기본 동작	설명
SIGABRT	6	종료 (코어 덤프)	프로세스 중단 신호
SIGALRM	14	종료	알람 클럭
SIGBUS	없음	종료 (코어 덤프)	정의되지 않은 메모리 오브젝트의 일부분에 접근.
SIGCHLD	없음	무시	차일드 프로세스 종료, 중단, 계속
SIGCONT	없음	계속	정지하지 않으면 계속 실행.
SIGFPE	없음	종료 (코어 덤프)	오류가 있는 산술 조작.
SIGHUP	1	종료	행업(Hangup).
SIGILL	없음	종료 (코어 덤프)	유효하지 않은 명령.
SIGINT	2	종료	터미널 인터럽트 신호.
SIGKILL	9	종료	킬 (신호를 잡거나 무시할 수 없음).
SIGPIPE	없음	종료	신호를 읽는 사용자가 없는 상태에서 파이프에 기록.
SIGPOLL	없음	종료	폴링 가능한 이벤트.
SIGPROF	없음	종료	프로파일링 타이머 시간 초과.
SIGQUIT	3	종료 (코어 덤프)	터미널 종료 신호.
SIGSEGV	없음	종료 (코어 덤프)	잘못된 메모리 참조.
SIGSTOP	없음	정지	실행 정지 (신호를 잡거나 무시할 수 없음)
SIGSYS	없음	종료 (코어 덤프)	불량 시스템 호출.
SIGTERM	15	종료	종료 신호.
SIGTRAP	없음	종료 (코어 덤프)	트레이스/브레이크포인트 트랩.
SIGTSTP	없음	정지	터미널 정지 신호.
SIGTTIN	없음	정지	백그라운드 프로세스 읽기 시도.
SIGTTOU	없음	정지	백그라운드 프로세스 쓰기 시도.
SIGUSR1	없음	종료	사용자 정의 신호 1.
SIGUSR2	없음	종료	사용자 정의 신호 2.
SIGURG	없음	무시	높은 대역의 데이터를 소켓에서 이용 가능.
SIGVTALRM	없음	종료	가상 타이머 시간 초과.
SIGXCPU	없음	종료 (코어 덤프)	CPU 시간 제한 초과.
SIGXFSZ	없음	종료 (코어 덤프)	파일 크기 제한 초과.