기초 교육

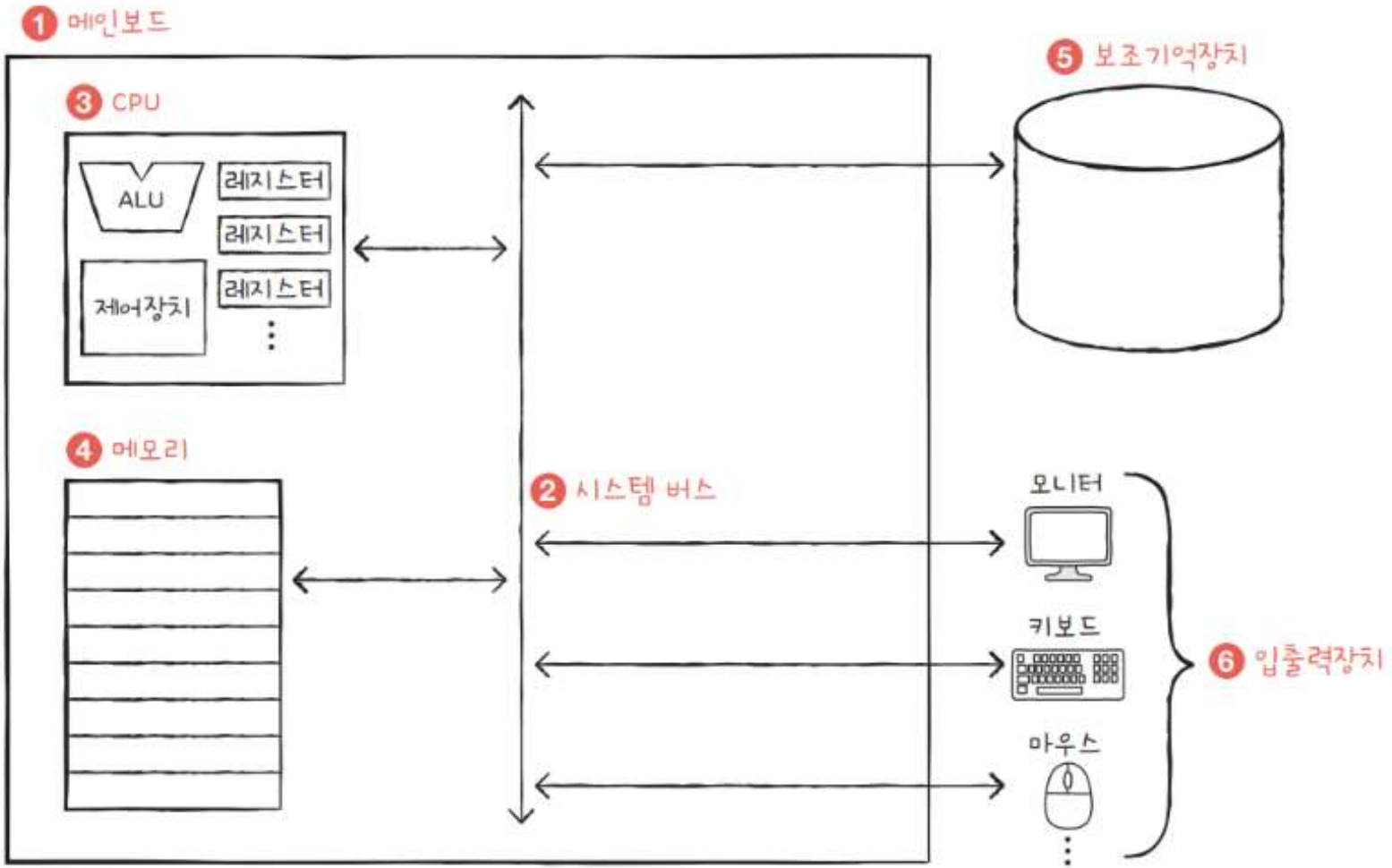
Contents

컴퓨터 구조

컴퓨터 구조

▪ 컴퓨터 구조

- 컴퓨터 구조에는 4가지 핵심 부품 : CPU, 메모리, 보조기억장치, 입출력장치
- 다양한 종류의 컴퓨터가 있으며 외관과 용도를 막론하고 컴퓨터를 이루는 핵심 부품은 크게 다르지 않습니다.



컴퓨터 구조

- **메모리(주기억장치)**
  - 컴퓨터가 이해하는 정보는 명령어와 데이터라고 한다.
  - 메모리는 현재 실행되는 프로그램의 명령어와 데이터를 정장하는 부품
  - 프로그램이 실행 되려면 반드시 메모리에 저장되어 있어야 합니다.
  - 전원이 꺼지면 저장된 내용을 잃는다 (휘발성)
- **CPU(중앙처리장치)**
  - CPU는 메모리에 저장된 명령어를 읽어 들이고, 읽어 들인 명령어를 해석하고, 실행하는 부품
- **보조기억장치**
  - 메모리보다 크기가 크고 전원이 꺼져도 저장된 내용을 일지 않는 메모리를 보조할 저장 장치
  - 하드 디스크, SSD, USB, DVD, CD-ROM과 같은 저장 장치가 보조기억장치
  - 메모리는 현재 '실행되는' 프로그램을 저장 한다면, 보조기억장치는 '보관할' 프로그램을 저장
- **입출력장치**
  - 입출력장치는 마이크, 스피커, 프린터, 마우스, 키보드처럼 컴퓨터 외부에 연결되어 컴퓨터 내부와 정보를 교환

네트워크 기초

네트워크 기초

▪ 네트워크란?

- 컴퓨터들이 통신 기술을 이용하여 그물망처럼 연결된 통신 이용 형태
- 정보나 자원들을 주고받을 수 있도록 하는 시스템
- 물리적네트워크는 네트워크를 구성하는 하드웨어(어댑터, 케이블 및 전화성과 같은 장비)

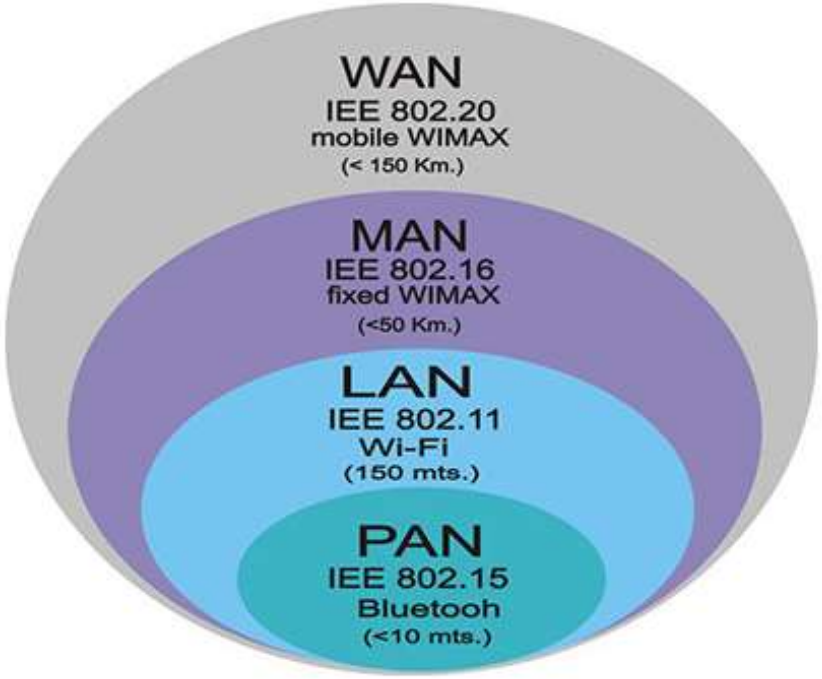
▪ 중요 용어

- IP주소 : 통신을 위해 인터넷 프로토콜을 사용하는 네트워크에 연결된 모든 디바이스에 할당된 고유 번호
- 노드 : 데이터를 송신, 수신, 작성 또는 저장할 수 있는 네트워크 내의 연결 지점
- 라우터 : 네트워크 간에 데이터 패킷에 포함된 정보를 전송하는 물리적 또는 가상 디바이스
- 라우팅 : 라우터가 패킷을 네트워크에서 목적지까지 보내는 최적의 경로를 선택하는 과정
- 스위치 : 다른 디바이스를 연결하고 네트워크 내의 노드 간 통신을 관리함으로써 데이터 패킷이 최종 목적지에 도달하도록 보장하는 디바이스
- 포트 : 네트워크 디바이스 간의 특정 연결 식별
- 프로토콜 : 컴퓨터가 다른 컴퓨터와 통신하는데 필요한 장비가 서로 통신을 위해 정해놓은 통신규약
- 패킷 교환 : 데이터를 일괄적으로 한 번에 보내지 않고 여럿으로 분할해서 송신하는 것

네트워크 기초

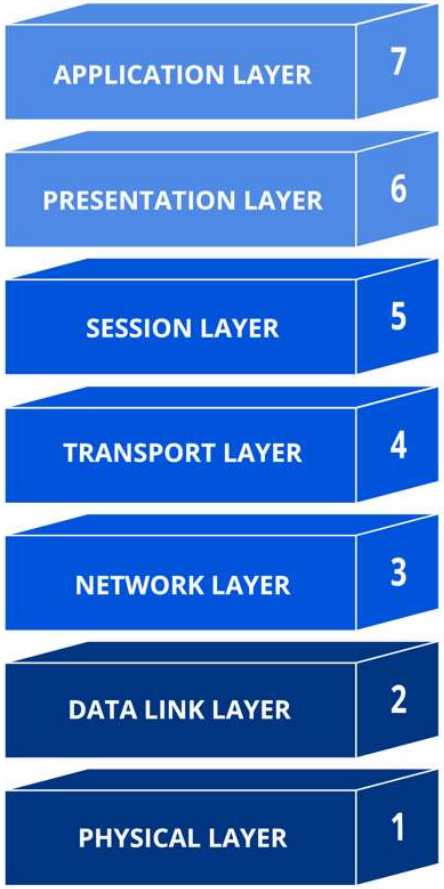
▪ 네트워크 종류

- WAN(Wide Area Network) : 지역 간 또는 대륙간의 넓은 지역의 컴퓨터를 연결, 인터넷은 전 세계 수십억 대의 컴퓨터를 가장 큰 WAN
- MAN(Metropolitan Area Network) : 일반적으로 도시 및 정부기관이 소유, 관리함
- LAN(Local Area Network) : 상대적으로 짧은 거리에 있는 컴퓨터를 연결, 예를 들어 사무실, 학원, 병원의 모든 컴퓨터 연결 가능
- PAN(Personal Area Network) : 약 5m전후의 인접 통신, 예를 들어 아이폰관 맥에서 정보를 공유하는 형태



네트워크 기초

- **OSI 7 계층**
  - 네트워크에서 통신이 일어나는 과정을 7단계로 나눈 것을 말한다,
  - 7단계로 정의한 이유는 통신이 일어나는 과정을 단계별로 파악하기 위함



- 응용 계층
- 표현 계층
- 세션 계층
- 전송 계층
- 네트워크 계층
- 데이터 링크 계층
- 물리 계층

네트워크 기초

- 1 물리 계층(Physical Layer)
  - 실제 장치를 연결하기 위한 전기적 및 물리적 세부 사항
  - 인터넷 케이블 전기적 신호가 물리적인 장치에 의해 통신하는 계층
  - 이 계층은 단지 데이터를 전달만 할 뿐 전송하려는(또는 받으려는) 데이터가 무엇인지, 어떤 에러가 있는지 등에는 신경 쓰지 않고 데이터 전기적인 신호로 변환해서 주고받는 기능
- 2 데이터 링크 계층(Data link Layer)
  - 해당 계층은 장치 간 신호를 전달하는 물리계층을 이용하여 네트워크 상의 주변 장치들 간의 데이터를 전송
  - 물리계층을 통해 송수신되는 정보의 오류와 흐름을 관리하여 안전한 정보의 전달을 수행
  - 통신에서의 오류도 찾아주고 재전송도 하는 기능을 가지고 있고 이 계층에서는 맥 주소를 가지고 통신
  - 두 지점(장치) 간의 신뢰성 있는 전송을 보장하기 위한 계층
  - 대표적인 장비로는 브리지, 스위치 등이 있다.
- 3 네트워크 계층(Network Layer)
  - 여러 개의 노드를 거칠 때마다 경로를 찾아주는 역할을 하는 계층
  - 경로를 선택하고 주소를 정하고 경로에 따라 패킷을 전달해주는 것이 이 계층의 주 역할이다.
  - 데이터를 연결하는 다른 네트워크를 통해 전달함으로써 인터넷이 가능하게 만드는 계층이다.
  - 대표적인 장비로는 라우터가 있다.

네트워크 기초

▪ 4 전송 계층(Transport Layer)

- 이 계층은 통신을 활성화하기 위한 계층이며 보통 TCP 프로토콜을 이용
- 4계층에서 해당 데이터를 하나로 통합하여 5계층으로 전달
- 종단 간 신뢰성 있고 정확한 데이터 전송을 담당
- 송신자와 수신자 간의 신뢰성있고 효율적인 데이터를 전송하기 위하여 오류검출 및 복구, 흐름제어와 중복검사 등을 수행
- 데이터 전송을 위해서 Port 번호를 사용함 (대표적인 프로토콜 TCP, UDP가 있다.)

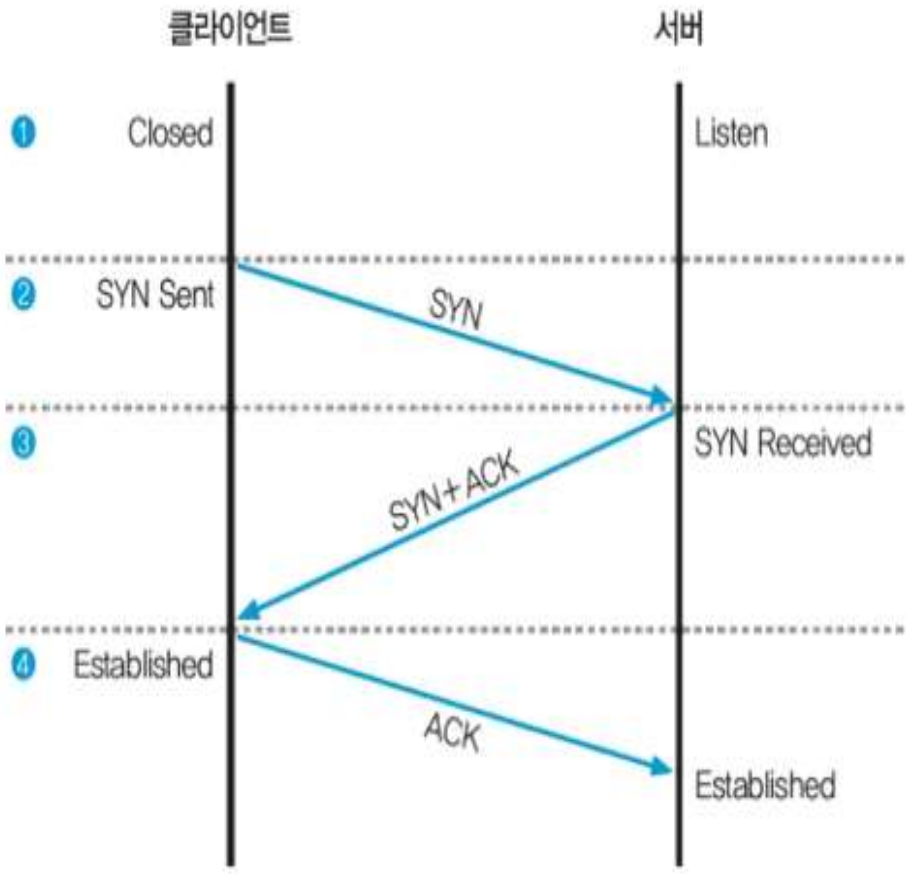
▪ TCP (Transmission Control Protocol)

- 서버와 클라이언트간에 데이터를 신뢰성 있게 전달하기 위해 만들어진 프로토콜
- 데이터를 전송하기 전에 데이터 전송을 위한 연결을 만드는 연결지향 프로토콜
- 연결 지향형인 TCP는 3-way handshaking 이라는 과정을 통해 연결 후 통신을 시작

▪ UDP (User Datagram Protocol)

- UDP는 전송계층의 비연결 지향적 프로토콜 입니다.
- 비연결 지향적이란 데이터를 주고받을 때 연결 절차를 거치지 않고 발신자가 일방적으로 데이터를 발신
- 연결과정이 없기 때문에 TCP보다는 빠른 전송을 할 수 있지만 데이터 신뢰성은 떨어집니다.

▪ 3-웨이 핸드셰이킹(3-Way Handshaking) – TCP 연결과정



3-웨이 핸드셰이킹(3-Way Handshaking) 과정

1단계
- 두 시스템이 통신을 하기 전에 클라이언트는 포트가 닫힌 Closed 상태
- 서버는 해당 포트로 항상 서비스를 제공할 수 있는 Listen 상태

2단계
- 클라이언트가 처음 통신을 하려면 임의의 포트 번호가 클라이언트 프로그램에 할당되고, 클라이언트는 서버에 연결하고 싶다는 의사 표시로 SYN Sent 상태가 된다.

3단계
- 클라이언트의 연결 요청을 받은 서버는 SYN Received 상태가 되고, 클라이언트에 연결을 해도 좋다는 의미로 SYN + ACK 패킷을 보낸다.

4단계
- 클라이언트는 연결 요청에 대한 서버의 응답을 확인했다는 표시로 ACK 패킷을 서버로 보냄

네트워크 기초

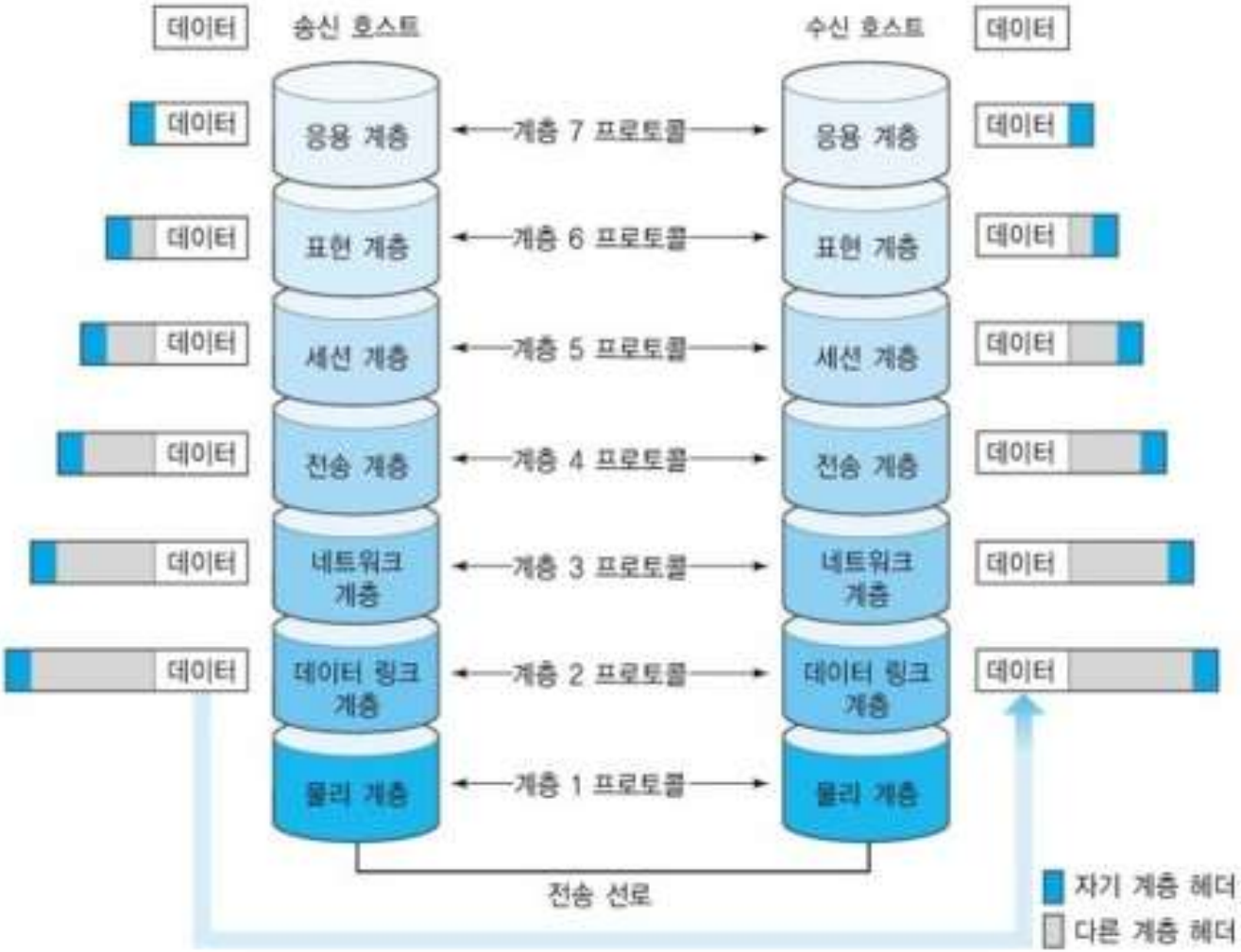
- **5 세션 계층(Session Layer)**
  - 양 끝단의 응용 프로세스가 통신을 관리하는 방법을 제공
  - 프로토콜은 통신 연결이 손실되는 경우 연결 복구 시도가 가능하며 연결 시도 중 장시간 연결이 되지 않았다면 세션 계층의 프로토콜이 연결을 닫고 다시 연결을 시도
  - 세션 계층의 중요한 기능인 동기화가 있다.

- **6 표현 계층(Presentation Layer)**
  - 데이터를 어떻게 표현할지를 정하는 역할을 하는 계층
  - 송신자에서 온 데이터를 해석하기 위한 응용계층 데이터 부호화, 변화
  - 수신자에서 데이터의 압축을 풀 수 있는 방식으로 된 데이터 압축
  - 데이터의 암호화와 복호화

- **7 응용 계층(Application Layer)**
  - 사용자와 가장 밀접한 계층으로 인터페이스 역할을 하는 계층
  - 응용 프로세스 간의 정보 교환을 담당

네트워크 기초

- OSI 7 계층 통신 과정



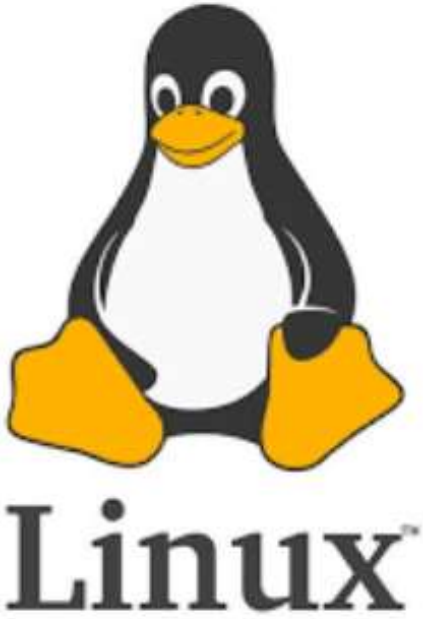
Linux 기초

▪ OS란?

- OS(Operating System)는 컴퓨터의 자원들을 효율적으로 관리
- 사용자가 컴퓨터를 편리하고, 효과적으로 사용할 수 있도록 환경을 제공
- OS는 컴퓨터 사용자와 하드웨어 간의 인터페이스로서 동작하는 시스템 소프트웨어
- 다른 응용프로그램이 유용한 작업을 할 수 있도록 환경을 제공

Linux 기초

- **Linux란**
  - 커널의 일종인 리눅스 커널, 또는 리눅스 커널을 사용하는 운영체제
  - 리눅스는 자유 소프트웨어와 오픈소스 개발의 가장 유명한 표본
  - 다수의 사용자가 네트워크를 통해 한시스템 자원에 접근하여 사용가능 (Multi-User)
  - 다수의 프로그램을 메모리에 적재하고 동시에 실행가능 (Multiprogramming)
  - 명령어 기반 프로그램을 제공 커널에게 명령을 내리고 명령어를 해석하여 맞는 프로그램을 실행
  - 전 세계적으로 가장 유명한 배포판 중 하나가 Red Hat Linux
  - 상용으로 판매되는 레드햇 소스코드를 그대로 가져와서 만든 것이 Centos

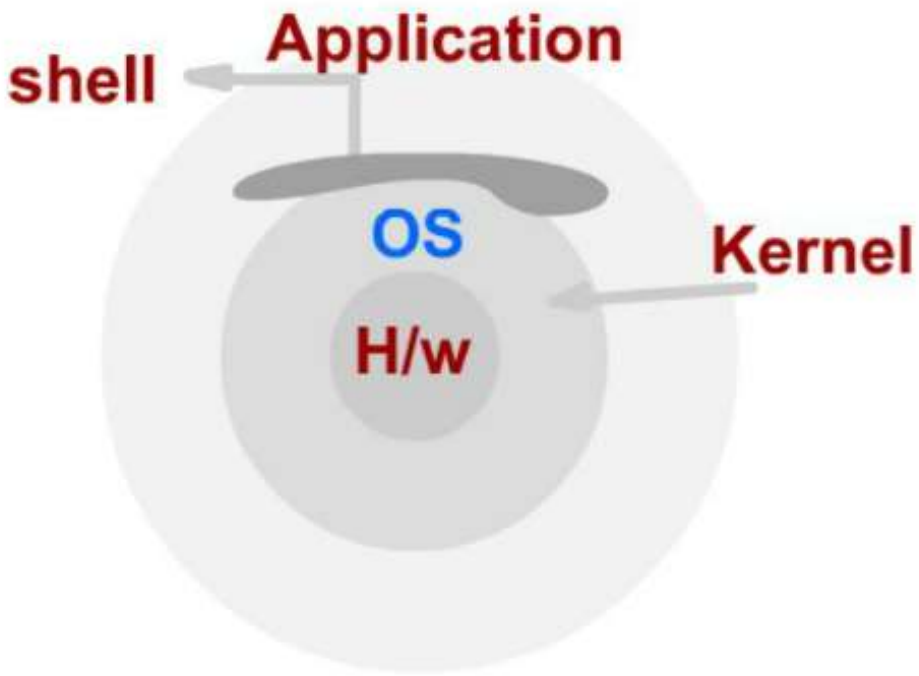


Red Hat
Enterprise Linux

CentOS

▪ Linux 구조

- 컴퓨터에 제일 내부에는 하드웨어가 있고, 그 하드웨어를 관리 해주는 것이 OS(운영체제) 이다.
- 실제로 하드웨어를 관리하는 OS부분을 Kernel 커널이라고 부른다.
- shell(명령어 해석기) OS 바깥부분에 위치하여 사용자로부터 명령을 받아들이고 그 명령을 해석
- Application은 소프트웨어/프로그램을 뜻한다.



▪ 프로세스 란?

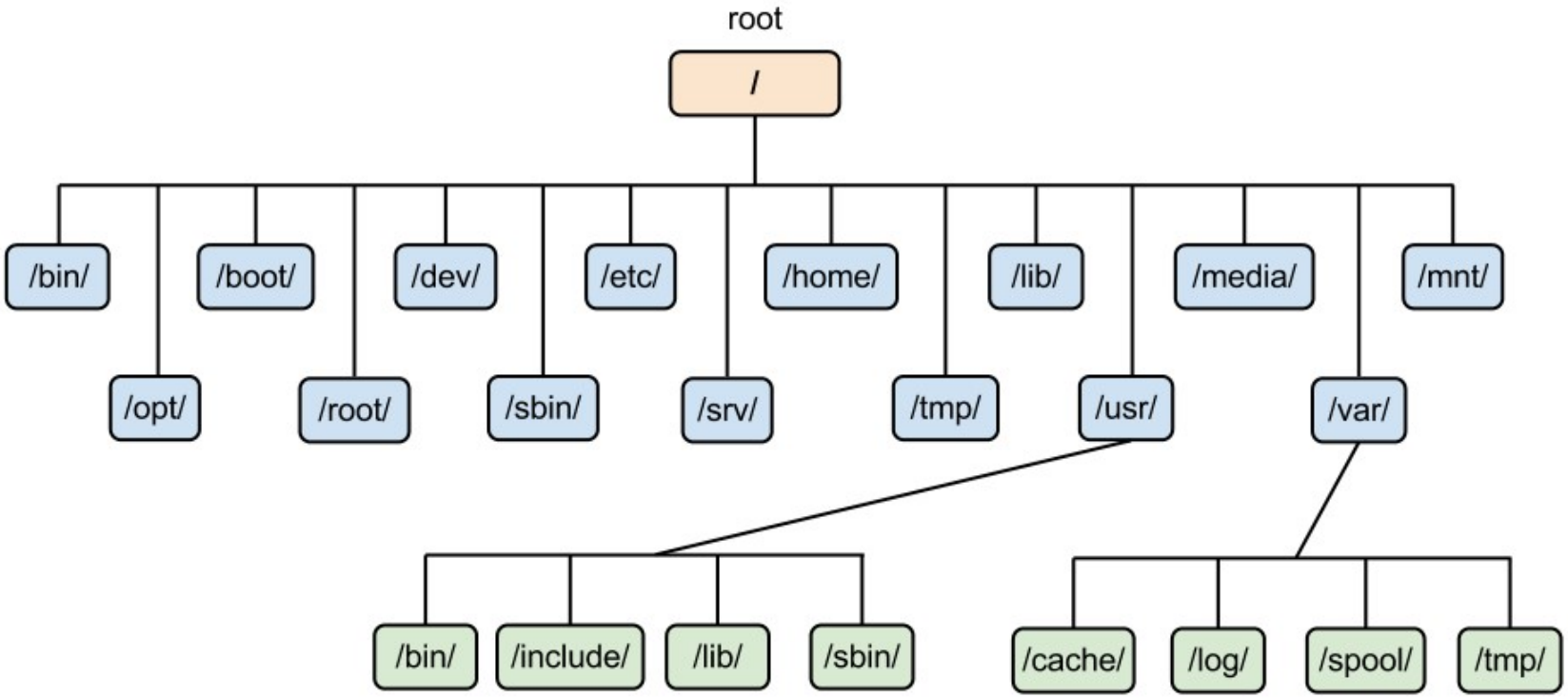
- 프로세스의 일반적인 정의는 실행중인 프로그램
- 서버의 CPU에서 실행되는 모든 프로그램을 프로세스라고 한다.
- 각각의 프로세스마다 고유 번호의 프로세스 ID(PID) 를 하나씩 증가하면서 부여
- 명령어 : ps -ef

UID	접속한 사용자		STIME	시작시간
PID	프로세스 ID		TTY	접속 도구
PPID	부모프로세스 ID		TIME	명령어 실행시간
C	CPU 점유율		CMD	명령어

```
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 00:18 ?        00:00:04 /usr/lib/systemd/systemd --switched-
root           2       0  0 00:18 ?        00:00:00 [kthreadd]
root           3       2  0 00:18 ?        00:00:00 [rcu_gp]
root           4       2  0 00:18 ?        00:00:00 [rcu_par_gp]
root           6       2  0 00:18 ?        00:00:00 [kworker/0:0H-events_highpri]
root           9       2  0 00:18 ?        00:00:00 [mm_percpu_wq]
root          10       2  0 00:18 ?        00:00:00 [rcu_tasks_rude_]
root          11       2  0 00:18 ?        00:00:00 [rcu_tasks_trace]
root          12       2  0 00:18 ?        00:00:00 [ksoftirqd/0]
root          13       2  0 00:18 ?        00:00:03 [rcu_sched]
root          14       2  0 00:18 ?        00:00:00 [migration/0]
root          15       2  0 00:18 ?        00:00:00 [watchdog/0]
```

▪ 디렉토리 구조

- 리눅스에서는 수많은 파일을 관리하기 위해 디렉토리를 사용
- 디렉토리 파일을 효율적으로 관리하기 위해 계층적으로 구성 (트리구조)
- 디렉토리는 그 밑으로 하위 디렉토리로 나누어지고 각 디렉토리에는 파일들이 저장
- 모든 디렉토리의 최상위 디렉토리를 root (/) 디렉토리라고 한다.



- 디렉토리 기능

디렉토리	설 명
home	사용자 홈 디렉토리가 생성되는 곳입니다.
media	CD_ROM이나 USB같은 외부 장치를 연결하는 디렉토리입니다.
opt	추가 패키지가 설치되는 디렉토리입니다.
dev	장치파일들이 저장되어 있는 디렉토리입니다.
root	root계정의 홈 디렉토리입니다. (/ 디렉토리와는 다릅니다.)
sys	리눅스 커널관련 정보가 있는 디렉토리입니다.
usr	기본 실행파일과 라이브러리 파일, 헤더 파일등의 파일이 저장되어있는 디렉토리입니다.
boot	부팅에 필요한 정보를 가진 파일들이 있는 디렉토리입니다.
var	시스템 운영중에 발생한 데이터와 로그가 저장되는 디렉토리입니다.
tmp	시스템 사용중에 발생한 임시데이터가 저장됩니다. (부팅 시 초기화)
proc	프로세스 정보 등 커널 관련 정보가 저장되는 디렉토리입니다.
mnt	파일 시스템을 임시로 연결하는 디렉토리입니다.
etc	리눅스 설정을 위한 각종 파일들을 가지고 있는 디렉토리입니다.
bin	기본 명령어들이 모여 있는 디렉토리 (일반 사용자 명령어)
sbin	시스템 관리를 위한 명령어 (슈퍼유저 root 명령어)

▪ 파일시스템 이란?

- 파일(자료)를 사용자가 쉽게 접근 및 발견 한 수 있도록 운영체제가 시스템의 디스크에 보관
- 리눅스 운영체제의 경우에는 파티션을 나누고 정리하는데 주로 사용된다.
- 리눅스 파일시스템 종류는 ext, ext2, ext3, ext4, xfs
- Redhat 7 이후로는 xfs 파일시스템이 표준화

▪ ext4 ?

- 대용량 파일 지원 및 디스크 공간의 빠른 할당
- 디렉토리에 있는 하위 디렉토리 수 제한이 없음
- 최대 16TB까지 지원

▪ xfs ?

- Redhat7 출시 이후에는 ext 시리즈가 아닌 XFS를 기본 파일시스템 채택
- 높은 확장성, 대용량 파일시스템(16TB 이상의 확장 가능한 고가용성 파일시스템)
- 대용량 파일시스템이라 작은 사이즈의 파일에서는 성능 저하
- 우수한 입출력 확장성 제공

▪ 스왑(SWAP) 이란?

- 물리 메모리 사용량이 가득차 프로그램을 메모리에 로드할 수 없는경우, 데이터나 프로그램을 디스크로 옮김
- 물리 메모리를 확보하여 프로그램을 메모리에 다시 로드
- 디스크상의 공간을 스왑공간 이라 부르며 전용파일이나 전용 파티션이 존재 하여야 함
- 속도면에서는 하드디스크를 이용하는 것이기 때문에 메모리 속도면에서는 떨어짐
- DB 또는 오픈 소스 대용량 처리 어플의 경우 swap을 많이 활용

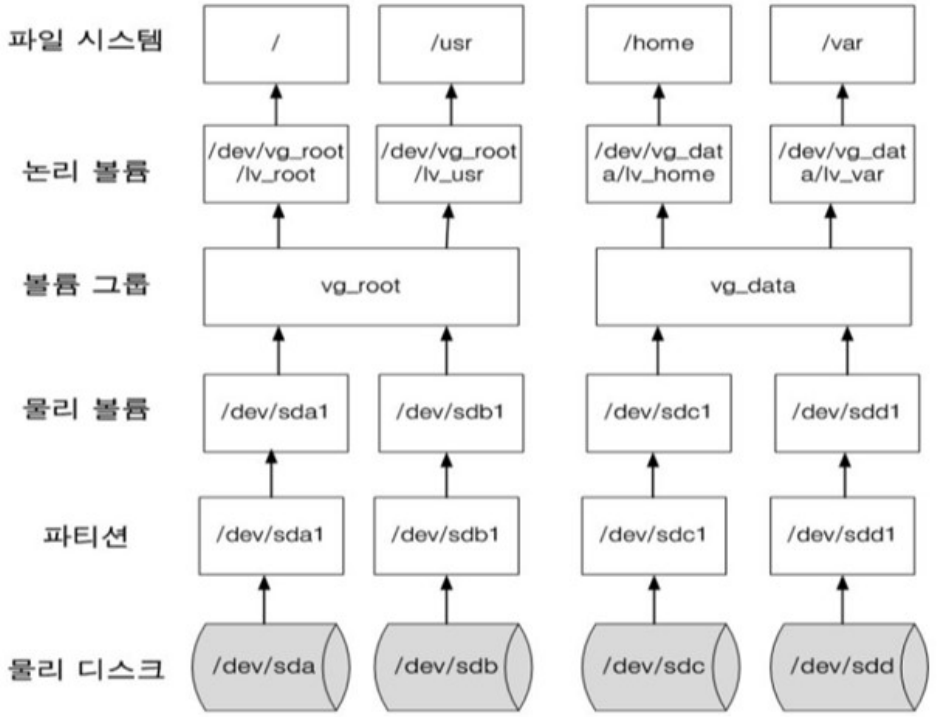
분 류	장 점	단 점
swap	- RAM이 가득 찼을 때 보조 공간 제공 - RAM처럼 빠르지는 않지만 하드보다 빠른 속도로 사용 가능, 더 많은 공간 사용	- swap 파티션은 크기를 유동성 있게 조정할 수 없기 때문에 하드 디스크의 공간을 차지 함

Linux 기초

- **LVM란?**
  - LVM(Logical Volume Manager)는 리눅스의 저장 공간을 효율적이고 유연하게 관리하기 위한 기능
  - 유연한 용량 조절
  - 크기 조절이 가능한 Storage pool
  - 편의에 따른 장치 이름 지정

- **용어**
  - PV(물리볼륨) : /dev/sda1, /dev/sdb1
  - VG(볼륨그룹) : 물리 볼륨을 합쳐서 1개의 볼륨 그룹
  - LV(논리 볼륨) : 볼륨 그룹을 1개 이상으로 나눈 논리 그룹

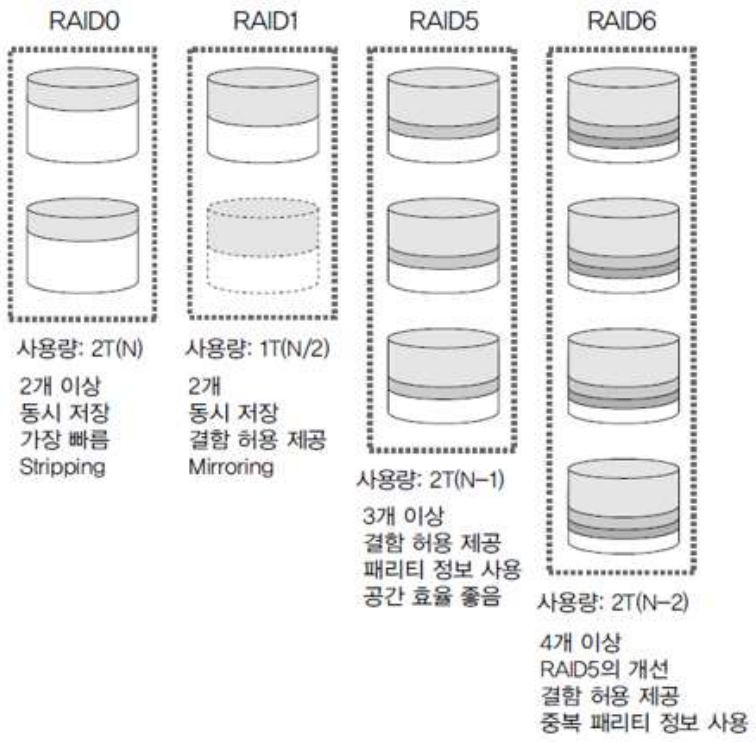
- **Mount**
  - 디스크 공간과 디렉토리를 연결
  - 생성된 LV 사용을 위해 디렉토리에 연결



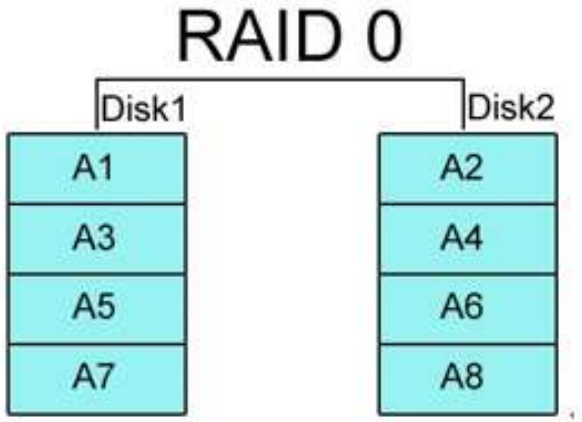
▪ RAID 란?

- 여러 개의 디스크를 배열하여 속도의 증대, 안정성의 증대, 효율성, 가용성의 증대를 하는데 쓰이는 기술
- 운용 가용성, 데이터 안정성 증대
- 디스크 용량 증설의 용이성
- 디스크 I/O 성능 향상

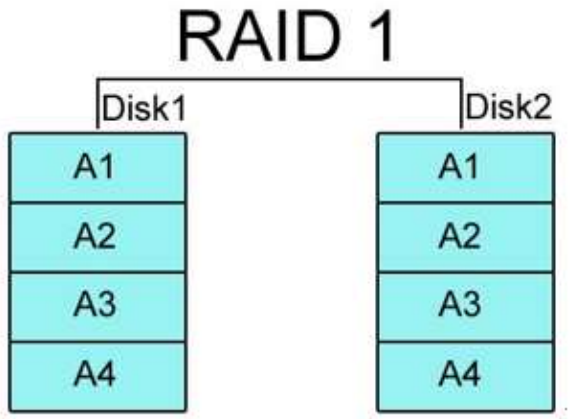
▪ RAID 종류



- **RAID 0**
  - 최소 2개의 하드디스크가 필요
  - 모든 디스크에 동시에 저장
  - 100%의 공간효율 ex) 1GB + 1GB = 2GB
  - 빠른 성능을 요구하되, 디스크 장애시 복구가 어려움

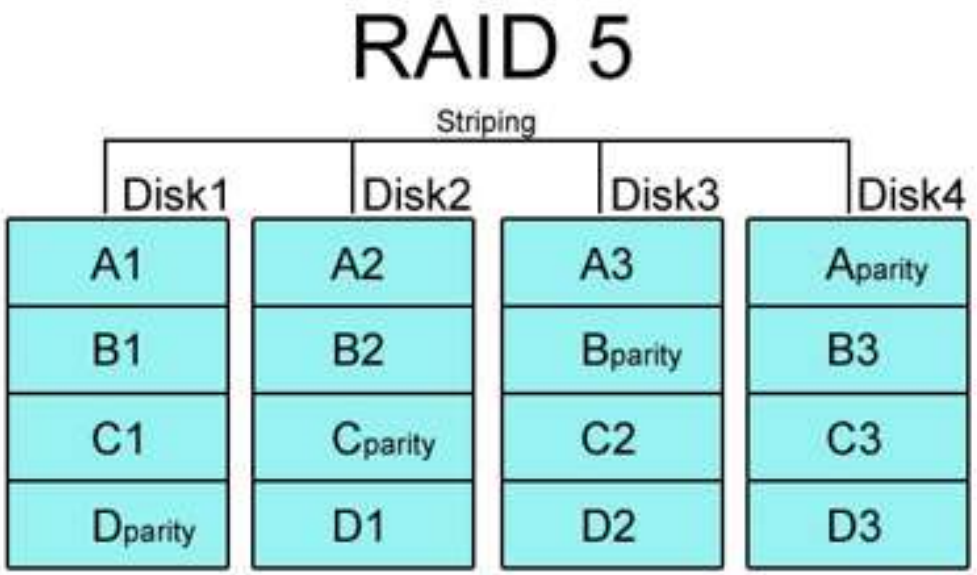


- **RAID 1**
  - 미러링(Mirroring) 이라 부르며 같은 데이터를 중복 기록하여 보존
  - 동일한 용량의 디스크 두 개가 필요
  - 볼륨 내 디스크 중 하나의 디스크만 정상이어도 데이터 보존되어 운영이 가능
  - 복원이 비교적 매우 간단함
  - 공간효율 나쁨 ex) 1GB + 1GB = 1GB



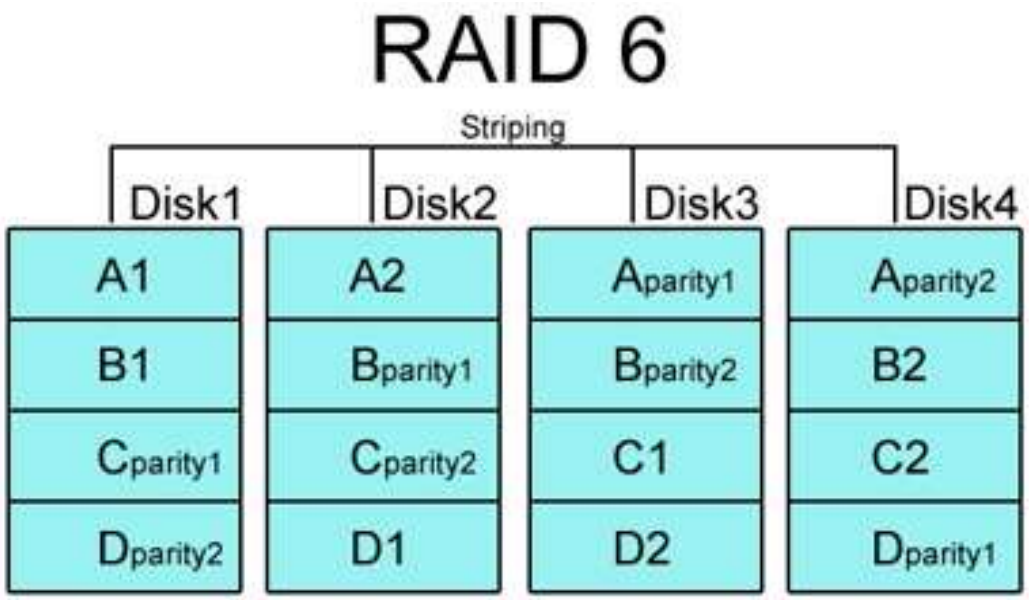
▪ RAID 5

- RAID 1처럼 데이터 안정성이 보장되며 RAID 0처럼 공간효율성도 좋은 방식
- RAID 5는 최소 3개 이상의 하드디스크가 있어야 구성이 가능
- 하드디스크에 오류가 발생하면 패리티를 이용해서 데이터를 복구
- RAID 5를 4개의 하드디스크로 구성했을 때 1개는 패리티로 사용하여 전체 용량 75% 사용
- 2개 이상의 하드디스크 장애시 데이터를 복구할 수 없음



- **RAID 6**

  - RAID 6은 5방식의 개선으로 2개의 패리티를 사용
  - 2개의 디스크가 동시에 장애가 나도 데이터에 이상이 없음
  - 최소 4개의 하드디스크를 사용
  - RAID 5보다 신뢰도는 높아지지만, 공간효율은 떨어진다.

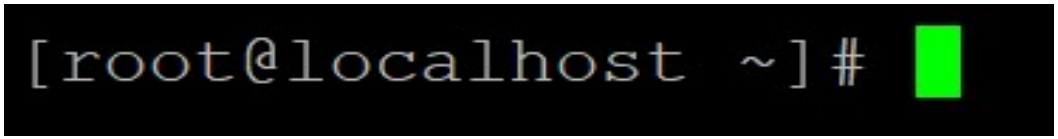


Linux 명령어

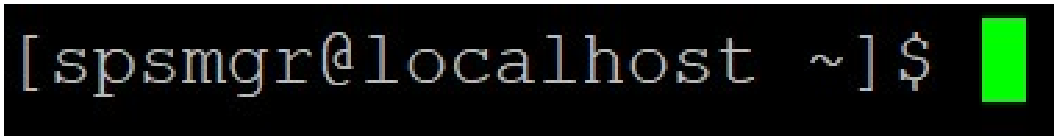
Linux 명령어

- 리눅스 Prompt 구조

구분자	의미
root	로그인한 사용자 계정명
localhost	리눅스 시스템의 호스트명
~	현재 작업 디렉토리 위치
#	관리자계정(#), 일반계정($)



[관리자계정]



[일반계정]

Linux 명령어

▪ Command : pwd

- 현재 경로 보기
- 현재 작업중인 디렉토리의 절대 경로를 보여준다.

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

▪ Command : cd

- 원하는 디렉토리로 이동

인자값	의미
directory	이동하기 원하는 디렉토리
.	현재 디렉토리
..	상위 디렉토리
~	로그인 된 사용자의 홈디렉토리로 이동

Linux 명령어

- **Command : ls**
  - 디렉토리의 목록 보기

옵션	의미
-a	. 을 포함한 경로안의 모든 파일과 디렉토리 표시
-l	지정한 디렉토리의 내용을 자세히 출력
-R	하위 경로와 그 안에 있는 모든 파일들도 같이 나열

```
[root@localhost /]# ls -a
.   .autorelabel  bin   db2  etc  home  lib64  media  opt   root  sbin  sys   tmp  var
..  app           boot  dev  fix  lib   logs   mnt    proc  run   srv   test  usr
[root@localhost /]# ls -l
합 계  34
drwxr-xr-x.   2 root root    6 12월  13 21:14 app
lrwxrwxrwx.   1 root root    7  6월  21  2021 bin -> usr/bin
dr-xr-xr-x.   5 root root 4096 12월  13 21:31 boot
drwxr-xr-x    8 root root  156  2월  21 11:07 db2
```

Linux 명령어

▪ Command : cp

- 파일이나 디렉토리를 복사하는 명령어

옵션	의미
-f	복사대상 파일이 있을 경우, 사용자에게 확인없이 강제로 복사
-r	디렉토리를 복사할 경우 하위 디렉토리와 파일을 모두복사
-p	원본 파일의 소유주, 그룹, 권한, 시간정보를 보존 하여 복사

▪ Command : mv

- 파일이나 디렉토리를 이동하거나 이름을 변경 할 때 사용

옵션	의미
-b	대상 파일이 이미 있어, 지워지는 것을 대비해 백업파일을 생성
-f	대상 파일이 이미 있어도 사용자에게 어떻게 처리할지 묻지 않는다.
-v	파일 을 옮기는 과정을 자세히 보여준다.

Linux 명령어

- **Command : mkdir**
  - 파일이나 디렉토리를 복사하는 명령어

옵션	의미
-p	필요한 경우 상위 경로까지 생성한다.

- **Command : rm**
  - 파일이나 디렉토리를 삭제하는 명령(권한이 있을 경우)

옵션	의미
-f	파일/디렉토리 삭제시 사용자에게 어떻게 처리할지 묻지 않고 삭제
-r	일반 파일이면 그냥 지우고, 디렉토리면 디렉토리를 포함한 하위 경로와 파일을 삭제
-i	삭제시 사용자에게 물어보면서 삭제

Linux 명령어

- **Command : cat**
  - 텍스트 파일 내용을 출력하는 명령어

```
[root@localhost /]# cat /etc/passwd
```

[파일내용 출력]

```
[root@localhost /]# cat /etc/passwd > /file
```

[기존의 파일 내용을 다른 파일로 입력]

```
[root@localhost /]# cat /etc/passwd >> /file
```

[기존의 파일에 내용을 추가]

Linux 명령어

- **Command : touch**
  - 크기가 0인 새로운 파일을 생성

```
[root@localhost /]# touch testfile
```

- **Command : head**
  - 파일의 내용중 처음부터 아래로 10줄 출력
  - head -5 [filename] // 5줄만 출력

- **Command : tail**
  - 파일의 내용중 마지막부터 위로 10줄 출력
  - tail -5 [filename] // 5줄만 출력

- **Command : shutdown**
  - 시스템을 종료하거나 재부팅하는 명령어
  - shutdown –h now // 시스템 즉시 종료
  - shutdown –r now // 시스템 즉시 재부팅

Linux 명령어

- **Command : df**
  - 파일시스템의 디스크 사용량을 표시

옵션	의미
-a	모든 파일시스템을 표시
-h	사용자가 읽을 수 있는 형태로 용량을 변환하여 표시
-T	파일시스템 타입을 표시

```
[root@localhost /]# df -Th
Filesystem                Type      Size  Used Avail Use% Mounted on
devtmpfs                  devtmpfs  3.8G     0  3.8G   0% /dev
tmpfs                     tmpfs     3.8G     0  3.8G   0% /dev/shm
tmpfs                     tmpfs     3.8G  9.6M  3.8G   1% /run
tmpfs                     tmpfs     3.8G     0  3.8G   0% /sys/fs/cgroup
/dev/mapper/rhel-root     xfs        20G  4.1G   16G  21% /
/dev/mapper/rhel-usr      xfs       8.0G  6.7G  1.4G  84% /usr
/dev/mapper/rhel-opt      xfs        10G  3.1G  7.0G  31% /opt
/dev/mapper/rhel-logs     xfs        10G  104M  9.9G   2% /logs
/dev/mapper/datavg-db2    xfs       3.0G   55M  3.0G   2% /db2
/dev/mapper/rhel-app      xfs       5.0G   68M  5.0G   2% /app
```

Linux 명령어

- **Command : free**
  - 메모리 사용량을 확인하는 명령어

옵션	의미
-h	사용자가 읽을 수 있는 GB, MB, KB 형태로 변경하여 출력

```
[root@localhost /]# free -h
              total        used        free      shared  buff/cache   available
Mem:          7.6Gi       710Mi       6.4Gi        10Mi       468Mi       6.6Gi
Swap:          15Gi          0B        15Gi
[root@localhost /]#
```

Linux 명령어

- **Command : top**
  - 시스템의 상태를 전반적으로 가장 빠르게 파악가능(CPU, Memory, Porcess)

옵션	의미
load average	CPU가 수행하는 작업 양 3개의 숫자는 1분, 5분, 15분 간의 평균 값
Tasks	Tasks는 현재 프로세스들의 상태를 나타내는 영역
%Cpu(s)	% us : 유저 레벨에서 사용하고 있는 CPU 비중
	% sy : 시스템 레벨에서 사용하고 있는 CPU 비중
	% id : 유휴 상태의 CPU 비중
	% wa : 시스템 I/O 요청을 처리하지 못한 상태에서의 CPU idle 상태인 비중

```
top - 16:58:55 up 7 days,  1:15,  2 users,  load average: 0.00, 0.00, 0.00
Tasks: 129 total,   1 running,  75 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.3 st
KiB Mem :   705132 total,    63724 free,   499460 used,   141948 buff/cache
KiB Swap:  8388604 total,  7725676 free,   662928 used.    89256 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 4160 opc       20   0 3060992 299840   6508 S   0.3 42.5   6:05.84 java
    1 root      20   0  216952   7396   5572 S   0.0  1.0   0:15.20 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.09 kthreadd
    3 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 rcu_par_gp
    6 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 kworker/0:0H-kb
    8 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 mm_percpu_wq
    9 root      20   0       0      0      0 S   0.0  0.0   0:06.96 ksoftirqd/0
```

Linux 데몬

Linux 데몬

- 데몬(daemon) 이란?
  - 데몬이란 리눅스 시스템이 처음 가동될 때 실행되는 백그라운드 프로세스의 일종
  - 메모리에 머무르고 있으면서 특정 요청이 오면 바로 그에 대한 대응을 할 수 있도록 대기중인 프로세스
  - 데몬은 프로세스 이름이 'd'로 끝나는 공통점이 있다.

기능	6버전	7, 8버전
서비스 상태보기	service [서비스명] status	systemctl status [서비스명]
서비스 시작	service [서비스명] start	systemctl start [서비스명]
서비스 정지	service [서비스명] stop	systemctl stop [서비스명]
서비스 재시작	service [서비스명] restart	systemctl restart [서비스명]
서비스 자동시작 활성화	chkconfig [서비스명] on	systemctl enable [서비스명]
서비스 자동시작 비활성화	chkconfig [서비스명] off	systemctl disable [서비스명]

Linux vi 편집기

Linux vi 편집기

- 명령모드

KEY	동작
i	현재 위치에서 입력 모드
I	행의 제일 처음에서 입력 모드
a	현재 위치에서 우측으로 한 칸 이동 후 입력 모드
A	행의 제일 마지막에서 입력모드
o	커서 아래에 새로운 행을 추가하고 입력 모드
x	커서가 있는 문자 삭제 (back space 로 전환)
dd	현재 커서의 행 삭제
숫자 + dd	현재 커서부터 숫자만큼 행 삭제
yy	현재 커서가 있는 라인을 복사
숫자 + yy	현재 커서부터 숫자만큼의 행을 복사
p	복사한 내용을 현재 라인 이후에 붙여넣기
P	복사한 내용을 현재 라인 이전에 붙여넣기
]]	문서의 마지막으로 이동 (shift + g)
[[	문서의 처음으로 이동
u	실행취소, 이전으로 돌아가기(ctrl + z 같은 기능)
ctrl + f	한 화면 단위로 다음 이동
ctrl + b	한 화면 단위로 뒤로 이동

Linux vi 편집기

- EX모드(검색)

KEY	동작
:/test	test를 검색

- 패턴이 검색 된 후 'n' 키를 통해 아래 방향으로 계속 찾기
- 패턴이 검색 된 후 'N' 키를 통해 위 방향으로 계속 찾기

- EX모드(치환)

KEY	동작
:[범위]s/[Old]/[New]	Old 를 New로 치환

- %s/Old/New // 문서 전체에서 Old를 New로 치환
- 8s/Old/New // 8번 라인만 Old를 New로 치환
- 10,15s/Old/New // 10 ~ 15번 라인만 Old를 New로 치환

- EX모드(행 번호)

KEY	동작
:se nu	문서에 행 번호를 출력
:se nonu	문서에 행 번호 비활성화

Linux vi 편집기

- EX모드(파일)

KEY	동작
:q	종료(변경된 내용이 없는 경우)
:q!	강제 종료(변경된 내용 있어도 무시)
:w	파일 저장
:wq	파일 저장 후 종료

- EX모드(명령어 실행)

KEY	동작
:! [command]	vi 를 잠시 중단하고 명령어 수행
:.! [command]	수행한 명령의 결과를 vi 편집기로 출력

Linux RPM & YUM

Linux RPM & YUM

▪ RPM란?

- RPM(RedHat Package Manager)
- 레드햇 계열의 리눅스 배포판에서 사용하는 프로그램 설치 관리 도구
- Windows의 setup.exe와 비슷하게 패키지를 자동으로 설치
- RPM을 이용하면 패키지 자동설치까지는 도와주나 패키지 사의에 의존하고 있는 패키지까지는 자동으로 설치 되지 않는다.

▪ 패키지명 형식

- mysql-connector-java-5.1.25-3.el7.noarch.rpm

패키지이름	-	소스버전	-	릴리즈 버전	.	OS 버전	.	아키텍처	.	rpm
mysql-connector-java	-	5.1.25	-	3	.	el7	.	noarch	.	rpm

- 아키텍처 : 패키지가 설치할 수 있는 CPU
- i386, i486, i586, i686 : 인텔 또는 AMD 계열의 32비트 CPU를 의미
- x86_64 : 인텔 또는 AMD 계열의 64비트 CPU를 의미
- noarch : 모든 CPU 설치 가능
- 아키텍처 확인 명령어 arch

Linux RPM & YUM

▪ RPM 명령어

- rpm [옵션] [패키지명]
- rpm –ivh [패키지명] -> 패키지설치
- rpm –Uvh [패키지명] -> 패키지 업데이트
- rpm –e [패키지명] -> 패키지 삭제
- rpm –qa -> 설치된 패키지 목록 출력
- rpm –qa | grep –i [패키지명] -> grep 명령어와 함께 사용하여 특정 패키지 설치 확인

옵션	의미
-i	패키지 설치
-U	패키지 업데이트
-v	설치 시 상세 내용을 출력
-h	설치 시 progress 를 # 으로 표시
-e	패키지 삭제
-q	시스템에 설치된 패키지 정보 확인
-a	시스템이 설치된 전체 패키지 목록

Linux RPM & YUM

- [실습 - 1]
  - iso이미지 /media 마운트 하기
  - cd /media/BaseOS/Packages
  - rpm –ivh ./net-tools-2.0-0.52.20160912git.el8.x86_64.rpm
  - rpm –qa | grep –i net-tools-2.0-0.52.20160912git.el8.x86_64.rpm
  - ifconfig
- [실습 - 2]
  - java 17 설치
  - java 17 패키지 서버 업로드
  - rpm –ivh [java 17 패키지]
  - rpm –qa | grep –i [java 17 패키지]
  - java --veriosn
- [실습 - 3]
  - java 17 -> 19 업그레이드
  - java 19 패키지 서버 업로드
  - rpm –ivh [java 19 패키지]
  - rpm –qa | grep –i [java 19 패키지]
  - java --veriosn

Linux RPM & YUM

▪ YUM란?

- 지정된 서버주소로부터 업데이트된 패키지들을 검사하여 다운 및 설치
- 의존성 문제도 같이 검사하여 관련 패키지들을 자동으로 설치
- RPM 기반의 프로그램 설치 및 업데이트를 대폭 개선
- 레드햇은 서브스크립 등록하여 레드햇 공식 repo 사용 (인터넷이 되어야함)
- iso 이미지를 이용하여 local repo 환경 구성하여 사용

▪ YUM 명령어

- yum –y install [패키지명] // 패키지 설치
- yum search [패키지명] // 패키지 찾기
- yum list // 패키지의 리스트를 보여줌

* yum –y remove [패키지명] // 삭제시 yum이 아닌 rpm으로 삭제 할 것 yum 이용시 의존성 관련 패키지도 같이 삭제가 될 우려가 있음

▪ [실습]

- local repo 환경 구성하기
- iso 이미지 /media 마운트
- mariadb 설치 하기

Linux 로그 파일

Linux 로그 파일

- 로그 파일

경로	설명
/var/log/messages	시스템에 문제가 생겼을 때 가장 먼저 찾아보는 로그
/var/log/secure	사용자 접속 정보가 기록되는 파일
/var/log/maillog	메일 송, 수신 내용이 기록하는 파일
/var/log/cron	cron이 실행된 것들에 대한 정보가 기록
/var/log/boot.log	서비스 데몬들의 부트에 관련된 정보가 기록
/var/log/dmesg	부팅 시의 시스템 로그가 기록 (명령어 dmesg)
/var/log/wtmp	최근의 접속 사항이 기록되는 파일
/var/log/lastlog	각 사용자의 마지막 로그인 내용이 기록

감사합니다.