Section 1. 권한 및 그룹 설정

- 리눅스 시스템은 모든 파일과 디렉터리에는 접근권한과 소유권이 부여된다.
- 명령어 'ls –l' 은 파일 속성을 나타낸다.
- 속성 필드 중 첫 번째 필드는 파일이나 디렉터리의 허가권, 세번째와 네번째 필드는 파일 이나 디렉터리의 소유권을 나타낸다.

- 파일의 허가권이나 소유권을 설정하는 명령어는 chmod, chown, chgrp, umask 등이 있다.

<파일종류>:일반파일, 디렉터리파일 특수파일.

- f 일반 파일
- d 디렉토리
- b 블록 디바이스(CD/DVD, 플로피디스크 등)
- c 캐릭터
- p Named PIPE (FIFO)
- l 심볼릭 링크
- s 소켓

>: 덮어 씌움

>>: 추가함

소유권 관련 명령어

- 소유권은 임의의 파일 또는 디렉터리에 대한 사용자와 그룹들의 소유 권한을 나타낸 것 이다. :으로 소유자랑 소유그룹 구분 해 줄 수 있음. ls명령어로 소유권 확인.
- 그룹은 사용자들의 시스템 운영 특성에 따라 묶어 놓은 것으로, 같은 그룹에 속한 사용자 들은 파일 또는 디렉터리에 대해 동일한 소유권과 직접 권한을 갖는다.

chown

- 파일과 디렉터리의 사용자 소유권과 그룹 소유권을 변경한다.

```
drwxrwxr-x 1 youngjin youngjin 53 2017-09-16 18:49 Young

chown root:root Young
chown root Young
chown :root Young    ==> chgrp Young
```

[chown -h] : 심볼릭 링크 파일의 소유자와 소유 그룹을 변경

--> [-h] 옵션 없이 심볼릭 링크 파일을 바꿀 시 원본 파일이 변경된다

[chown -**R**] : 하위 디렉터리의 파일까지 소유권을 변경한다.

chgrp

- 파일이나 디렉터리의 그룹 소유권을 변경한다.

[-c]:변경된 파일만 자세하게 보여준다.
[-**f**]:변경되지 않은 파일에 대해서 오류 메시지를 **보여주지 않는다**.
[-**R**]:경로와 그 하위 파일들을 **모두 변경**한다.
[-h]: **심볼릭 링크** 자체의 그룹을 변경
[-v] : 명령어의 버전 정보를 출력한다.

허가권 관련 명령어

- 명령어 'ls –l' 으로 파일 유형과 허가권을 알 수 있다. 디렉터리는 –d를 붙인다.
- 파일 허가권의 첫 번째 자리는 파일 유형을 기호로 정의한다.
- 파일은 일반 파일(-), 디렉터리 파일(d), 특수 파일(b-블록파일, c-문자파일, l-링크파일 등)로 나뉜다.
- 파일 권한을 읽기(read)4, 수정,삭제(write)2, 실행(execute)1이 있다.

읽기(read): 디렉터리 내부의 내용을 **볼 수 있는** 권한
쓰기,삭제(write)디렉터리 내부에 파일을 **생성, 삭제** 할 수 있는 권한
실행(execute):디렉터리는 **디렉터리 내부로 접근**. 파일은 **파일의 실행**

- 읽기, 쓰기 또는 실행의 접근 제한 표시는 하이픈(-)으로 나타낸다.

chmod

- 파일이나 디렉터리의 접근 허가권을 변경하는 명령어이다.
- 파일 기본권한 **666** 디렉터리의 기본권한 **777**

[-**R**] : 하위 디렉터리를 포함하여 디렉터리 내부의 **모든 파일**의 접근 권한을 변경

[-c] : 권한이 **바뀐** 파일만 자세히 기술한다.

[-f] : 파일의 권한이 바뀔 수 없어도 에러 메세지를 출력하지 않는다

```
chmod o+w TST.txt
```

+:**원래있던 권한에서 추가로 줄 때**, =**기존 권한에 상관없이** 권한부여(**숫자로부여**)

7->007, 77->077

- u:소유자, g:그룹소유자, o:기타사용자
- a: 모든사용자에게 허기권 부여 ex) a=rwx
- chmod 첫 번째 숫자는 SetUID-4,SetGID-2,stickey bit-1를 나타냄.
- **소유자의 s**: SetUID, **그룹소유자의 s**:SetGID, **기타 사용자의 t** : 스티키 비트

umask

- 새로 생성되는 파일이나 디렉터리의 기본 허가권 값을 지정한다.
- 파일의 기본 권한은 **666**, 디렉터리의 기본 권한은 **777** 이다.
- 파일이나 디렉터리 생성 시 디폴트 권한 값에서 설정한 umask를 뺀 값을 기본 허가권으로 설정한다.

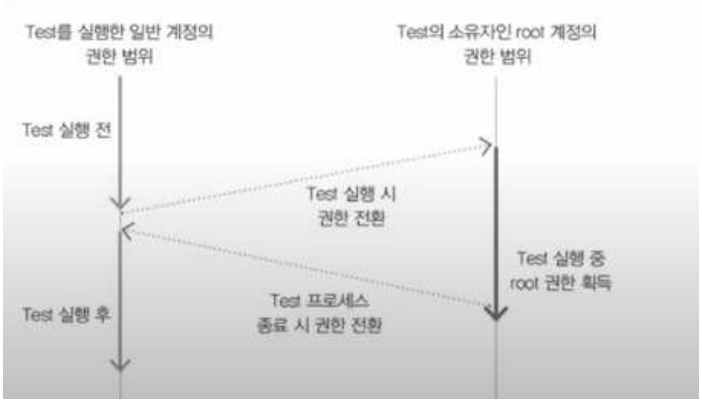
```
umask
0002
mkdir AAA
ls -l

0777-0002 = 0775
 umask -p
 => umask 0022

 umask -S
 => u=rwx,g=rx,o=rx
```

-p : umask값 **숫자**로 확인
-s : umask값 **문자**로 확인
-S : 디렉토리 기준 **퍼미션** 값 확인

특수 권한(Set-Bit) : usr/bin/passwd 에 설정됨

SetUID(4) 와 SetGID(2)

- 프로세스가 실행되는 동안 해당 프로세스의 root 권한을 임시로 가져오는 기능이다.
- 프로세스가 사용자보다 높은 수준의 접근을 요구할 때 파일 접근 제한 때문에 원활한 기 능을 제공할 수 없기 때문에 이러한 문제점을 해결하기 위한 방법이다.



- SetUID의 경우 사용자가 사용할 때만 소유자 권한으로 파일을 실행시키고, SetGID 의 경 우 사용자가 사용할 때만 그룹 권한으로 파일을 실행한다.

Sticky bit(1)

- 일반적으로 공용 디렉터리를 사용할 때 sticky bit 를 설정하여 사용한다.
- 사용자 권한을 지정하기 어려운 프로그램들이 **일시적**으로 특정 디렉터리(**/tmp**)에 파일을 생성 하고 삭제하도록 이용된다.
- 설정된 디렉터리에는 누구든 접근 가능하고 파일을 생성할 수 있다.
- Sticky bit 가 설정되어 있는 디렉터리 안의 내용은 해당 파일의 **소유자나 root 만이** 변경 이 가능하다.

허가권에서 t : sticky bit 원래 실행권한이 있으면 t 없었으면 T

디스크 쿼터(Disk Quota)

- 파일 시스템마다 사용자나 그룹이 생성할 수 있는 파일의 용량 및 개수를 제한하는 것이 다. 보통 블록 단위의 용량 제한과 inode의 개수를 제한한다.

  * inode:파일 시스템에 관한 정보를 담고 있는 특별한 종류의 디스크 블록으로 이름, 소유주, 시간, 위치 등 파일에 관한 전체적인 정보를 담고 있다.

- 사용자나 그룹이 가질 수 있는 inode의 수, 사용자나 그룹에게 할당된 디스크 블록 수를 제한한다.
- 쿼터는 사용자별, 파일 시스템별로 동작된다.
- 그룹 단위로도 용량을 제한할 수 있으며 웹호스팅 서비스를 하는 경우에 유용하다.

디스크 쿼터 지정 단계

- 단계 1: 파일 **/etc/fstab** 에 디스크 쿼터 관련 설정
- 단계 2: 재마운팅 실행 후 확인
- 단계 3: 마운트 된 커터를 끄고 생성된 쿼터 파일 삭제
- 단계 4: 쿼터 데이터베이스 생성

- 단계 5: 사용자별 쿼터 지정
- 단계 6: 쿼터 현재 상태

/etc/fstab 파일 (1301회) (1401회) (1404회)

어떤 장치를 어디에, 어떤 옵션으로 mount할 것인지 적어두고 있다.

Ex] **/dev/**hda1/mnt/hda1 ext4 default 0 0

[1번째 필드] : **디바이스 명**

[2번째 필드] : **마운트 될 디렉토리**

[3번째 필드] : **파일 시스템**의 종류

[4번째 필드] : 마운트될 파일 시스템의 고유 **옵션**

[5번째 필드] : **dump** 명령어가 그 파일 시스템을 덤프할 필요가 있는지를 지정

[**6**번째 필드] : fsck 명령어로 무결성의 **체크** 여부를 지정. 0인 파티션은 체크X

디스크 사용량(쿼터) 설정 관련 명령어

[quotacheck] : 파일 시스템의 디스크 사용 상태를 검색한다. 파일이 없을 경우 쿼터 기록 파일 생성.

[quotaon] : 파일 시스템의 쿼터 기능을 **활성화**한다.

[quotaoff] : 쿼터 서비스를 비활성화 한다.

[**edquota**] : 편집기(vi)를 이용하여 사용자나 그룹에 디스크 사용량을 **할당**하는 명령어이다.

[edquota **-p** <u1> <u2>] : u1의 쿼터 설정을 u2에 복사한다.

[etquota]: 편집기가 기반이 아닌 명령행에서 **직접** 사용자나 그룹에 디스크 사용량을 할당하는 명령어이다.

[**quota**] : 쿼터 정보를 출력한다. 주로 quota 사용자계정명

[**repquota**] : 쿼터 정보를 **요약**하여 출력한다. 주로 repquota 디렉토리명

사용자 쿼터 설정 : /etc/fstab -> **usr**quota/grpquota 옵션 지정

사용자 쿼터를 설정하는 단계 명령 순서

- quotacheck->edquota->quotaon

Section 2. 파일 시스템의 관리

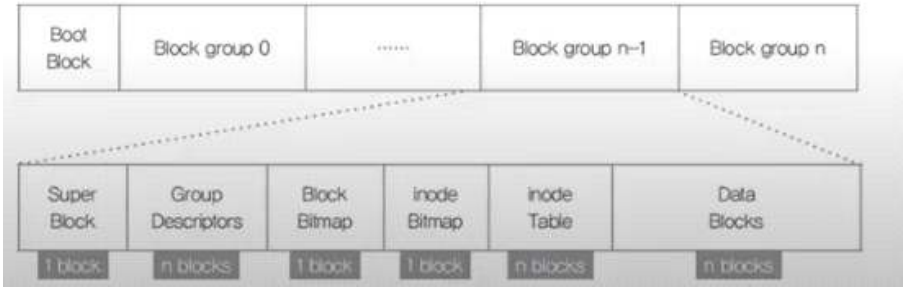
파일 시스템의 개요와 종류

개요

- 운영체제가 파일을 시스템의 디스크상에 구성하는 방식이다.
- 컴퓨터에서 파일이나 자료를 쉽게 발견 및 접근할 수 있도록 보관 또는

조직하는 체제이 다.

- 하드디스크나 CD-ROM 과 같은 물리적 저장소를 관리한다.
- 파일 서버상의 자료로의 접근을 제공하는 방식과 가상의 현태로서 접근 수단만이 존재하 는 방식도 파일 시스템의 범위에 포함된다.



종류

- **리눅스 전용** 디스크 기반 파일 시스템

파일 시스템	설명
ext(ext1)	리눅스 초기에 사용되던 파일 시스템이며 호환성이 없음 ext2의 원형 2GByte 의 데이터와 파일명을 255자까지 지정 가능
ext2	고용량 디스크 사용을 염두하고 설계된 파일 시스템 쉽게 호환되며 업그레이드도 쉽게 설계되어 있음
ext3	리눅스의 대표적인 **저널링**(저널이라고 부르는 로그에 변경사항을 저장)을 지원하도록 확장된 파일 시스템 ACL(Access control List) 를 통한 접근 제어 지원 **ext3부터 저널링 가능**
ext4	파일에 디스크 할당 시 물리적으로 연속적인 블록을 할당 저널링 가능 64비트 기억 공간 제한을 없앰 16 TeraByte의 파일을 지원

- **저널링** 파일 시스템

파일 시스템	설명
JFS	Journaling File System 의 약자 IBM 사의 독자적인 **저널링** 파일 시스템 GPL로 공개하여 현재 리눅스용으로 개발
xfs	eXetended File System 고성능 **저널링** 시스템 64비트 주소를 지원하며 확장성이 있는 자료 구조와 알고리즘 사용 데이터 읽기/쓰기 트랜잭션으로 성능 저하를 최소화 64비트 파일 시스템으로 큰 용량의 파일도 다룰 수 있음 **파일 크기 졸라큼.**
ReiserFS	독일의 한스 라이저가 개발한 파일 시스템 모든 파일 객체들을 B트리에 저장, 간결한 색인화 된 디렉터리 지원

● **네트워크** 파일 시스템

파일 시스템	설명
SMB	Server Message Block **삼바** 파일 시스템을 마운트 지정 윈도우 계열 OS 환경에서 사용되는 파일/프린터 공유 프로토콜 리눅스, 유닉스 계정 OS와 윈도우 OS와의 자료 및 하드웨어 공유
CIFS	Common Internet File System SMB를 확장한 파일 시스템 SMB를 기초로 응용하여 라우터를 뛰어넘어 마운트할 수 있는 프로토콜
NFS	Network File System 썬마이크로시스템이 개발한 네트워크 공유 프로토콜 파일 공유 및 파일 서버로 사용됨 NFS 서버에서 공유된 영역을 마운트할 때 지정 하드웨어, 운영체제 또는 네트워크 구조가 달라도 공유 가능 NFS 서버의 특정 디렉터리를 마운트하여 사용할 수 있음

● 기타 지원 가능한 파일 시스템

파일 시스템	설명
FAT	Windows NT가 지원하는 파일 시스템 중 가장 간단한 시스템 FAT 로 포맷된 디스크는 클러스터 단위로 할당 클러스터 크기는 볼륨 크기에 따라 결정 읽기 전용, 숨김, 시스템 및 보관 파일 특성만 지원 삼바 파일 시스템을 마운트 지정
VFAT	Virtual FAT FAT 파일 시스템이 확장된 것으로 FAT 보다 제한이 적음 파일 이름도 최고 255자까지 만들 수 있음 공백이나 여러 개의 구두점도 포함
FAT32	SMB를 확장한 파일 시스템 32GB 보다 큰 파티션을 만들 수 없고 파티션에 4GB를 초과하는 파일을 저장할 수 없음
NTFS	윈도우에서 사용하는 파일 시스템 안정성이 뛰어나고 대용량 파일도 저장 파일 크기 및 볼륨은 이론상으로 최대 16EB 이나 실질적으로는 2TB 로 한계가 있음
ISO 9660	CD-ROM 의 표준 파일 시스템 DVD를 마운트 할 때 지정하는 파일 시스템 1988년에 재정된 표준
UDF	Universal Disk Format 의 약자로 최신 파일 시스템 형식 광디스크 파일 시스템 표준 **ISO 9660파일** 시스템을 대체하기 위한 것으로 대부분 DVD에서 사용

HPFS	OS/2 의 운영체제를 위해 만들어진 파일 시스템

관련 명령어

mount 와 umount

- 마운트는 특정 디바이스를 특정 디렉터리처럼 사용하기 위해 **장치**와 디렉터리를 연결한 다.
- 리눅스는 PnP 기능을 지원하지만 지원하는 하드웨어가 많지 않으므로 시스템 부팅후에 수동으로 마운트해서 사용을 하고 사용이 끝난 후에는 언마운트를 시킨다.
- 파일 /etc/mtab 은 현재 마운트된 블록 시스템 정보를 표시한다.

```
#mount-o ( 괄호 ) CentOS-6.10-i386.bin-DVD.iso
/media
```

[-a]:**/etc/mstab**에 명시된 파일 시스템을 모두 **마운트**/언마운트

[**-t <파일시스템>** <마운트할 디렉터리>]: 파일 시스템을 지정

[**-o**]: 추가 옵션을 명시 **ro**: read only 읽기전용 **remount**:재마운트 **loop**: iso파일 마운트

noatime: access time 갱신 X

eject

- 이동식 보조기억장치(**CD-ROM**)등과 같은 미디어를 해제하고 **장치를 제거**하는

명령어이다. (트레이Tary를 연다)

fdisk

- **새로운 파티션의 생성**, 기존 파티션의 삭제, 파티션의 타입 결정 등의 작업을 수행할 수있다. 설정 후에는 파티션 테이블 업데이트가 필요하다.
- 한 번에 한 디스크에 대해서만 작업을 수행한다.

```
[root@www ~]#

Disk /dev/sda: 22.1 GB, 22078537728 bytes
255 heads, 63 sectors/track, 2684 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x000e030e

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1        1913    15360000       Linux
/dev/sda2            1913        2423     4096000       Linux swap / Solaris
/dev/sda3            2423        2684     2102206       Linux
[root@www ~]#
```

[**-n**] 파티션 추가

[-w]변경된 파티션 정보 저장]

[-q]:변경된 파티션의 정보를 저장하지 않고 종류

[-d] 파티션 삭제

[**-t**] 파티션 종류 **변경**

[-c] 플래그 설정

[-m] 설정 도움말 매뉴얼

[-s]:파티션의 크기를 출력

- fdisk명령어는 파티션테이블을 관리하는 명령어로 리눅스의 디스크파티션을 생성, 수정, 삭제할 수 있는 일종의 유틸리티라고 할 수 있습니다.

1.1 명령어의 위치
/sbin/fdisk

1.2 사용 형식
fdisk [-l] [-v] [-s 파티션] [장치이름]

2. 현재 모든 디스크의 파티션 설정 현황 파악하기
- 리눅스 쉘에서 fdisk -l 이라고 입력하면, 모든 디스크의 파티션 설정 현황을 알 수 있습니다.

3. SWAP 은 물리메모리의 메모리 부족시 사용할 수 있는 디스크를 메모리로 사용하는 파티션입니다.
- 각 파티션은 파티션에 따라 ID가 다르며,
82 는 Linux swap / Solaris
83 은 Linux
8e 는 논리 볼륨 관리자 파티션
fd 는 Linux RAID 입니다.

mkfs

- **리눅스 파일 시스템을 생성**한다.
- fdisk로 하드디스크를 파티션을 나눈 후 해당 파티션에 맞는 파일시스템을 생성한다.

[mkfs.ext3] : ext3 파티션 생성

mke2fs

- ext2, ext3, ext4 타입의 **리눅스 파일 시스템을 생성**하는 명령어이다. 그냥생성하면 ext2

형식 : mke2fs [–옵션] [파일시스템] [경로]

[mke2fs **-j**] : **ext3(저널링파일 시스템)** 파티션 생성

[**mke2fs -t <파일시스템>**] : <파일시스템> 파티션 생성

[mke2fs –T <파일크기>]: 파일 크기 설정 ex) largefile

[mke2fs –i <갯수>] : **I-node 개수** 지정

fsck

- 파일 시스템의 무결성을 점검하고 대화식으로 **복구**하는 명령어이다.
- 디렉터리 **/lost+found** 는 fsck 에서 사용하는 디렉터리이다.

[-**A**] 옵션을 사용하면 /etc/fstab의 모든 파일 시스템에 대해 기능을 수행한다.

[-**a**] 옵션을 사용하면 오류 발견 시 **자동**으로 복구를 시도한다(질문X)

[-s] : fsck 동작을 시리얼화 하는 명령어이며, 대화형 모드에서 여러 파일 시스템을 점검할 때 사용.

[-r] 옵션을 사용하면 복구 시도 전에 확인을 요청한다.

e2fsck

- ext2, ext3, ext4 타입의 리눅스 파일 시스템을 생성(포맷),**복구**하는 명령어이다.

du

- Disk Usage의 약자로 **디렉터리별**로 **디스크 사용량**을 확인할 수 있다.

[-D] : 심볼릭 링크 파일이 있을 경우 원본의 값을 보여준다.
[-l] : 하드 링크의 용량을 모두 계산한다.
[**-h**] : 디스크 사용량을 알기 쉬운 **단위**로 출력한다.
[**-s**] : **총 사용량**만 표시한다.

df

```
Filesystem Size Used Avail Use% mounted on
/dev/sda3  97G  56G  37G  61%  /
/dev/sda1  99M  12M  83M  13% /boot
/dev/sda4  50G   0G  50G  0%   /home
```

- **시스템**에 **마운트된** 하드 디스크의 남은 용량을 확인할 때 사용하는 명령어이다.
  시스템 전체 디스크의 정보
- 기본적으로 1024 Byte 블록 단위로 출력한다. 옵션을 통해 다른 단위로 출력이 가능하다.

[-a] : 모든 파일 시스템을 대상으로 디스크 사용량을 확인한다.
[**-h**] : 디스크 사용량을 알기 쉬운 **단위**로 출력한다.
[-t <파일시스템 종류>] : 지정한 파일 시스템 종류에 해당하는 디스크 사용량을 출력한다.
[**-T**] : 파일 시스템 **종류**도 출력한다.
[-i] :inode 사용정보를 보여줌 즉 파일이름,소유주,권한,시간,디스크 위치

[-m] : 메가바이트 단위로 남은 용량을 보여줌

Section 3. 셸 개념 및 종류

개념

- 명령어 해석기이다.
- 로그인할 때 실행되어 사용자별로 사용 환경 설정을 가능하게 한다.
- 강력한 스크립트 언어이다.
- 입출력 방향 재지정과 파이프 기능을 제공한다.
- 포어/백그라운드 프로세스를 실행한다.

종류

- 본셸계열과 C셸 계열로 나뉜다.
- 사용자 프롬프트가 $ 이면 본셸 계열, % 이면 C 셸 계열을 사용하고 있다는 것이다.
- 대부분의 셸은 본셸 계열의 기능을 포함하여 확대 발전한 형태이다.
- C셸은 본셸의 모든 기능과 명령어 **히스토리**, **별명(alias)**, **작업 제어** 기능을 추가로 가지고 있다. 버클리대학의 **빌 조이**가 개발함.
-
  - 1989년 브라이언 폭스(Brian Fox)가 GNU 프로젝트를 위해 개발한 셸
  - 명령 히스토리, 명령어 완성 기능, 히스토리 치환, 명령행 편집 등을 지원
  - 본 셸을 기반으로 하여 제작

  -> **bash shell**

- 셸 개발연도: 본셸->**C셸**->**korn셸**->Bash 셸
- tcsh:C셸 확장
- sh:응급복구쉘

셸 확인 및 변경

로그인 셸 **확인**

- 파일 /etc/shelsl 에서 사용할 수 있는 셸들을 확인할 수 있다. **cat /etc/shells**로 사용가능한 셸의 목록을 확인

- 파일 **/etc/passwd** 파일에서 계정마다 할당된 셸을 **확인**할 수 있다.
  grep 사용자 **/etc/passwd**
  echo $SHELL
  env | grep SHELL
  ps
  finger -l

- 명령어 echo $SHELL 은 현재 로그인한 사용자가 사용중인 셸을 **확인**할 수 있다.

셸 **변경**

- 로그인 셸 변경은 반영구적인 셸 변경 방법으로 관리자가 셸 변경 후 다음 변경을 하기 전까지 지정된 셸을 사용한다. /etc/passwd에서 가능하다.

- 명령어 **chsh**: 일반 사용자 환경에서 셸 **변경** 시 사용한다. 사용자가 사용 **가능**한 셸의 종류 확인 **cat /etc/shells**

[**-s = --shell**] : **지정하는 셸**을, 앞으로 사용할 로그인 셸로 변경

- 명령어 usermod: 관리자 환경에서 지정된 계정자의 정보를 변경할 때 사용하는 명령어이다.

Section 4. 셸 환경 설정

환경 변수와 셸 변수

환경 변수 (전역 변수)

- **전체 셸**에서 사용 가능한 전역 변수. 미리 예약된 변수명만 사용가능.
- 서브 셸에 기능 상속 가능
- **환경 변수 확인** 명령 **env**
- **export** [환경변수]=[**$**환경변수내용]:환경변수값(절대경로)

- echo $환경변수 : 환경변수를 호출 해 변수내용을 출력 echo만 쓰면 그냥 메아리

DISPLAY	**현재 X 윈도 디스플레이 위치** export DISPLAY="<IP주소>:<몇 번째 X 서버-**1**>.<보낼 모니터-**1**> Ex] export DISPLAY=:3.0 -> 4번째 X서버의 첫 번째 모니터로 전송
HISTFILE	history 파일의 위치
HISTFILESIZE	실질적인 **history 파일의 크기**
HISTSIZE	history에 저장되는 개수
HISTTIMEFORMAT	명령어를 수행할때마다 **현재시각**을 history에 기록
HOME	사용자 홈 디렉터리의 절대경로
LANG	쉘 사용 시 기본으로 지원되는 언어. (C로 지정 시 영어로 설정)
LOGNAME	사용자 계정 이름
MAIL	도착한 메일이 저장되는 경로
PATH	명령을 탐색할 경로
PS1	프롬프트(#-root, $-사용자) 수정 ₩(역슬래시)를 이용하여 명령줄 연장가능. ex) PS1='[₩**u@₩t ₩W**]₩$' => [ihd@09:30:21 ~]$ ₩u = user name ₩h = host name ₩w=**절대경로**표시 ₩W=**현재경로**표시 ₩t = 시간 표시 중 24시 표기법 ₩d = 요일 월 일 ₩s = 사용중인 셸이름 표시

PS2	제2의 프롬프트 (default = ' > ')
TMOUT	일정 시간 작업을 하지 않을 경우에 자동으로 로그아웃되는 시간 ex) TMOUT=/etc/profile
TERM	로그인한 터미널 종류의 이름 ex) TERM=xterm
SHELL	로그인한 셸

셸 변수 (지역 변수)

- **현재 로그인 셸에서만 사용 가능**한 지역 변수
- 서브 셸에 기능 상속 불가능
- 셸 변수 확인 명령 **set 변수 해제: unset**

환경 설정 파일

- 쉘 시작 시 자동으로 실행되는 고유의 시작 파일이 있다. 이 파일은 사용자 운영환경을 설정한다.
- 배쉬 쉘의 시작 파일은 /etc/**profile**(전역 환경변수.), /etc/**bashrc**(전역 alias와 함수설정), **~/.bash_profile**, **~/.bashrc, ~/.bash_logout : 개인 사용자가 로그아웃할 때 수행하는 설정 지(개인 사용자 환경파일))** 이다.
- 셸 파일은 전역적 파일과 지역적 파일로 나뉜다.
- 파일 /etc/profile.d 는 몇몇 응용 프로그램들이 시작 시 자동 실행할 스크립트 파일 경로를 넣어둔다.

배쉬셸의 주요 기능

* bash shell 내장 명령어 : alias,cd,export,se

History 기능

- 일정 개수 이상 사용했던 명령어를 **.bash_history** 에 저장해 두고 다시 불러서 사용할 수 있게 하는 기능이다. 로그아웃해도 삭제되지 않음.

- 대부분의 셸은 이전에 입력했던 명령어를 반복하거나 약간 변형하여 다시 사용할 수 있도록 하는 기능이다.

[**!<history Num>**] : **이전에 사용했던 명령**을 사용한다.

Ex] $!-1 : 바로 **이전에** 사용했던 명령을 사용

$!50 : *history에서* 50번째로 사용한 명령을 사용

[**!<명령어>**] : **가장 최근에 실행한 <명령어>**를 재실행한다.

[!?문자열?] : 가장 최근에 사용한 문자열을 포함하고있는 명령을 찾아서 실행

[**!!**] : **마지막**에 사용한 명령을 재실행한다.

[history num] : num까지 사용했던거 다보여줌

alias 기능

- 자주 사용하는 명령어를 특정 문자로 입력해 두고 간편하게 사용할 수 있게 하는 기능이 다.
- 형식: **alias [별명명령어] = [명령어]** 해제: **unalias [명령어]**

Section 5. 프로세스 개념 및 유형

개념

- 프로세스는 CPU와 메모리를 할당받아 현재 실행 중인 프로그램이다.
- 프로세스들마다 고유의 프로세스 ID를 할당받는다.

프로세스의 유형

포어그라운드 프로세스

- 사용자와 상호작용하는 프로세스
- 터미널에 직접 연결되어 입출력을 주고받는 프로세스
- 명령 입력 후 수행 종료까지 기다려야 하는 프로세스
- 화면에서 실행되는 것이 보이는 프로세스
- 응용프로그램이나 명령어 등

백그라운드 프로세스(**&**)

- 사용자와 직접적인 대화를 하지 않고 뒤에서 실행되는 프로세스
- 사용자의 입력에 관계없이 실행되는 프로세스
- 실행은 되지만 화면에 나타나지 않고 실행되는 프로세스
- 시스템 프로그램, 데몬 등

fork()

- 새로운 프로세스를 만들 때 기존 프로세스를 **복제**하여 생성하는 방식을 사용한다.

새로운 프로세스를 자식 프로세스로 관리하는 방식.

- 새로운 프로세스를 위한 메모리를 **할당**한다.
- 새로 생성된 프로세스는 원래의 프로세스와 똑같은 코드를 가지고 있다.
- 원본 프로세스를 **부모 프로세스**라 부르고, 새로 복제된 프로세스를 **자식 프로세스**라고 부른다.
- **보통 명령을 수행하면 fork() 방식으로 실행된다.**

exec()

- 호출하는 프로세스가 새로운 프로세스로 **변경**(대체)되는 방식이다.
- 새로운 프로세스를 위한 메모리를 할당하지 않는다.
- 호출한 프로세스의 메모리에 새로운 프로세스의 코드를 **덮어씌워** 버린다.

데몬

- 리눅스 시스템이 부팅 시 자동으로 실행되는 백그라운드 프로세스이다.
- 메모리에 상주하면서 사용자의 특정 요청이 오면 즉시 실행되는 대기 중인 서버 프로세스이다.
- 주기적이고 지속적인 서비스 요청을 처리하기 위해 사용된다.
- 사용자들은 이 프로세스들을 볼 수 있는 권한이 없다.

 # standalone 타입의 데몬 (1301회) (1403회)

- 시스템에 독자적으로 프로세스가 구동되어 서비스를 제공하는 데몬을 말한다.
- 메모리상에 항상 구동되어야하기 때문에 **자주 호출**되는 서비스(데몬)는 standalone타입의 데몬으로 사용하기에 적당하다. 빠르게 즉각처리 가능

<vs inetd 타입 데몬>

- 슈퍼 데몬(**xineted**)에 의해 관리된다. xineted는 환경설정파일을 이용해서 자체적으로 접근제어가 가능.

- 필요한 경우에만 메모리로 적재되어 실행되어 응답한다. 메모리 관리 효율적.
ex. telnet, pop3, finger

Section 6. 프로세스 유틸리티

프로세스 관련 명령어

ps

- 현재 실행중인 **프로세스**의 상태를 보여주는 명령어이다.
- CPU사용도가 낮은 **순서**로 출력된다.
- 옵션없이 명령어를 실행하면 자신의 터미널에서 실행되고 있는 프로세스들의 관련 정보만 나타낸다.
- 명령어 'ps'와 'grep'을 이용하여 특정 프로세스의 상태 정보를 확인 할 수 있다.
- 좀비 프로세스(Z)는 kill 시그널을 받아도 종료되지 않는다. 이 프로세스는 이미 현재 프로세스에 대한 모든 정보는 메모리에서 사라졌지만 부모 프로세스가 정상적인 종료 처리를 하지 못해 발생한다.

 # 명령어 kill 시그널

- **시그널**(signal) : 프로세스 간의 통신수단

1	SIGHUP	재시작
2	SIGINT	[Ctrl+C]의 시그널 / 프로세스 종료
3	SIGQUIT	[Ctrl+W]의 시그널 / 종료
9	SIGKILL	강제 종료
15	SIGTERM	정상 종료 [기본 시그널]
20	SIGTSTP	[Ctrl+Z]의 시그널 / 프로세스 중단

★★★★★★★★★★이 표 대박 중요함★★★★★★★★★★

형식 : kill –**시그널**번호 프로세스 번호 (-를 붙이지 않으면 default : **15번**)

[**kill -l**] : 기본적인 **시그널**의 종류(목록)를 번호 순서대로 확인

[kill –s] : 시그널의 이름을 지정하는 옵션

[**killall**] : 인자로 가진 이름의 프로세스를 모두 종료

[killall **-v**] : 시그널이 **전송된 결과**를 출력

● NICE값이 '프로세스의 실행 우선순위가 높다'라는 의미는 실행 우선순위가 낮은 프로세스보다 더 많은 시스템 자원을 할당하게 되므로 실행속도가 빨라지게 된다는 것을 뜻한다. 즉 NICE값으로 프로세스의 실행 우선순위를 설정한다.

ps - 옵션

[-a] : 세션 리더를 제외하고 터미널에 종속되지 않은 모든 프로세스의 정보를 출력

[**-e**] : **시스템**에서 실행 중인 **모든** 프로세스의 정보를 출력

[-f] : 프로세스의 **자세한** 정보를 출력 (PPID, 프로세스 시작 시간 등)

[-u] : **터미널**에서 실행한 프로세스의 자세한 정보를 출력 (PPID, CPU 및 메모리 사용량, RSS, VSZ 등)

[**-l**] : 프로세스의 자세한 정보를 출력 (UID, **PPID**, **PRI**(우선순위 값), **NI**(nice값) 등)

[a] : 터미널에서 실행한 프로세스의 정보를 출력

[-t] : 프로세스 항목 on/off

[-k] : PID값을 입력하여 종료신호를 보낸다.

[-m] : 메모리 관련 항목 on/off

[-W] : 바꾼 설정을 저장

aux 항목

a : 터미널과 연관된 프로세스 출력(x옵션과 연계하여 모든 프로세스를 출력할 때 사용, BSD계열)

u : 프로세스 **소유자** 기준으로 출력

x : **데몬** 프로세스처럼 터미널에 종속되지 않는 프로세스 출력

[**TIME**] : 프로세스 총 실행 시간

[**RSS**] : 사용하고 있는 **물리적 메모리의 크기**

[**VSZ**] : 사용하고 있는 **가상 메모리의 크기**

[**%MEM**] : 물리적 메모리 사용량을 %로 표시

[**%CPU**] : **CPU 사용비율**

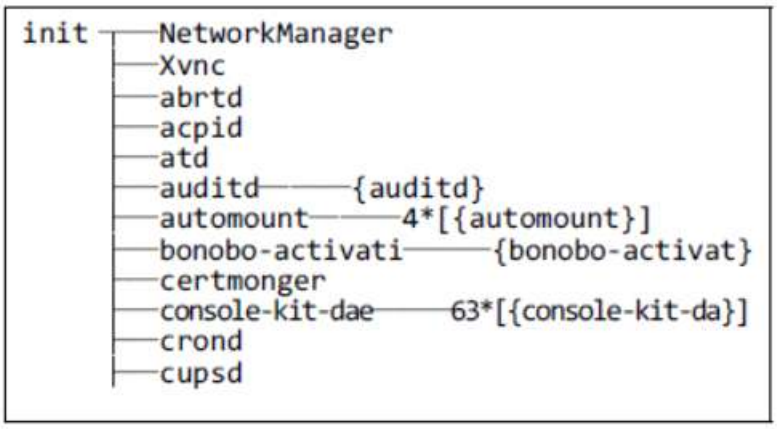
[**PID**] : 프로세스 식별번호 최초의 프로세스(init)은 PID가 1이다.

* PPID: 프로세스를 만든 부모 프로세스의 PID

[**STAT**] : 현재 프로세스 상태코드S(기다림),D(I/O에 대해 대기),R(특정이벤트가 끝나기를 기다리는상태),T(정지된상태),Z(부모프로세스가 정상적인 처리를 하지 못할 때)

pstree

- 실행 중인 프로세스들을 트리구조로 나타낸다. init프로세스가 모든 프로세스의 부모 프로세스 확인가능. (우선순위 X)



pstree **-a**: 각 프로세스의 명령행 인자까지 보여준다. arguments

pstree **-h**: 현재 프로세스와 그것의 조상 프로세스를 하이라이트로 강조해서 보여준다.

pstree –n: PID 출력

jobs

```
[1]     2998 Suspended (tty output)       vim a.txt
[2]   -  2999 Suspended (tty output)       vim b.txt
[3]   +  3001 Suspended (tty output)       vim c.txt
```

- 작업이 **중지**된 상태, **백그라운드(suspend or stopped)**로 진행 중인 상태, 변경되었지만 보고되지 않은 상태 등 을 표시한다.
- **백그라운드**로 실행중인 프로세스를 **확인**한다. [숫자] 는 작업번호이다.
- 출력된 목록에서 **+는 현재 작업** 실행, -는 앞으로 실행될 작업을 나타낸다.

jobs에서 작업번호없이 fg만 치면 +붙은 프로세스가 포어그라운드로 전환됨.

- 작업중인 프로세스를 대기 시키는 키조합은 [컨트롤+Z]이다

bg와 fg

- 전환 하기 전에 프로세스를 중지 [**ctrl+z**] 시켜주고 전환
- 포어그라운드에서 **백그라운드로** 전환: **bg %**작업번호 또는 bg 작업번호
- 백그라운드에서 **포어그라운드로** 전환: **fg %**작업번호 또는 fg 작업번호

kill

- 프로세스를 종료시킨다.

killall

- 같은 데몬의 여러 프로세서를 한 번에 종료시킬 때 사용한다.
- 프로세스명으로 연관된 프로세스들을 종료시킨다.

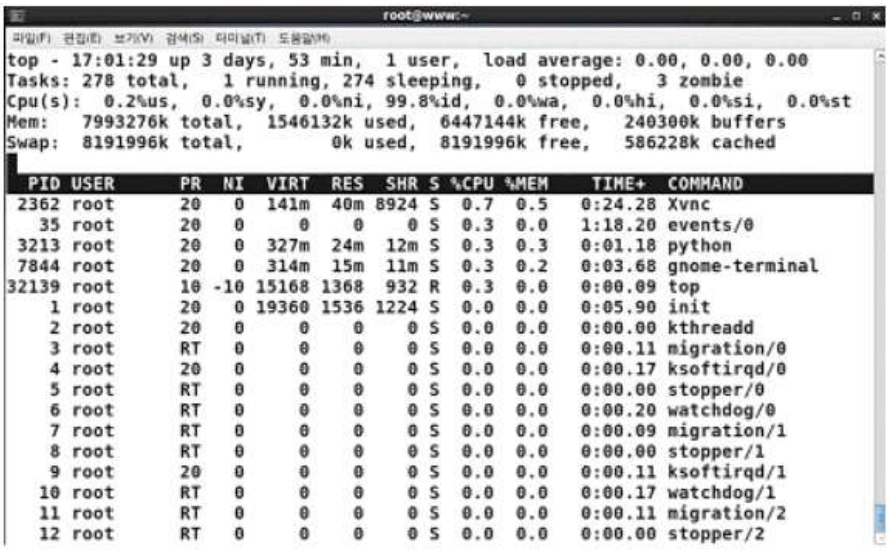
nice

- (프로세스를 **새로** 생성하는 경우) 프로세스 사이의 우선순위를 확인하고 우선순위를 변경할 수 있는 명령어이다.
- 조정할 수 있는 NI 값의 범위는 **-20 ~ 19** (우선순위 높음 -> 낮음)이다.
- 우선순위 0의 값(디폴트값)을 가지며 값이 작을수록 우선순위가 높다.
- 옵션 **–n**을 사용하지 않으면 디폴트는 **10**을 사용하고 IN값이 n으로 변경된다..
- 조정수치가 생략되면 명령의 우선권은 10만큼 **증가**한다.
- 명령어 'nice -10 bash' 는 bash 프로세스 NCE 값을 10만큼 **증가**시키는 것이다. 값이 증가한다는 것은 우선순위를 낮추는 것이다.(일반사용자는 증가만 가능0~20) 우선순위를 높이는 명령어는 'nice --10 bash' 이 다. 이것은 NI 값을 -10만큼 **증감**시켜 우선순위를 높인다.
- nice [옵션] **프로세스명**

renice

- **이미 실행중인 프로세스**(프로세스의 증가 X) 의 우선순위를 변경한다.
- nice는 기존 NI값을 **증감**하지만 명령어 renice는 **지정한 NI값을 설정**한다.
- **nice는 양수값은 –를, 음수값은 —를 사용하지만**, renice는 양수값에 –를 사용하지 않는다.(**양수값에 부호없음**)
- renice [옵션] NI값 **PID**

top



- 리눅스 시스템의 운영 상태를 실시간으로 모니터링하거나 프로세스 상태를 확인할 수 있다.

- 아무런 옵션 없이 실행하면 5초에 한 번씩 새로운 내용으로 갱신하여 출력한다. - **메모리 상태(RES)**, **CPU 상태**, **부하 상태**,**스왑 사용량** 등을 확인할 수 있다. (**디스크 사용량 X**)

- 옵션

[**-d**] : 갱신 딜레이(시간 간격) 설정

[-u <user>] : user 소유의 프로세스를 표시

[**-p** <PID num>] : PID가 <PID num>인 프로세스만을 실시간으로 화면에 출력

[-o <출력할 항목>] : <출력할 항목>으로 지정한 항목들로만 출력한다.

- 명령 실행 후 사용할 수 있는 옵션

[l (소문자)] : load average 줄 표시/해제

[P] : CPU 사용량(%CPU)에 따라 정렬하여 출력한다.

[**M**] : **메모리 사용량(RES)**에 따라 정렬하여 출력한다.

[m] : **m**emory 줄 표시/해제

[T] : 프로세스 실행 시간(**Time**)에 따라 정렬하여 출력한다

[**t**] :CPU항목 on/off.

[s] / [d]: 갱신되는 딜레이를 변경한다.

[c] : 명령인자 표시/해제

[**k** +PID]:특정 프로세스 종료

[-r] : NI값 조정

nohup

- **프로세스가 중단되지 않고 백그라운드로 작업을 수행할 수 있게 한다. (자동 X 사용자가 &명령어를 이용해 직접 명시해 주어야 한다.)**

- 표준출력과 표준에러는 '**nohup.out**'파일을 생성해 기록한다.

- 사용자가 로그아웃하거나 작업 중인 터미널 창이 닫혀도 실행 중인 프로세스를 백그라운드 프로세스로 계속 작업할 수 있도록 한다.

- 용량이 큰 데이터 압축 해제와 같은 실행 시간이 오래 걸리는 프로세스들에 대해 nohup 으로 처리하여 작업하면 작업 중단 없이 해당 업무를 완료할 수 있다.

- 백그라운드로 실행될 수 있도록 명령행 뒤에 **&**를 명시한다.

tail

- 파일의 마지막 행을 기준으로 지정한 행까지 파일 내용의 일부를 출력한다.

- 기본값으로 마지막 10줄을 출력한다. -n으로 읽어올 라인 수 지정.

스케줄링과 cron

- 주기적으로 반복되는 일은 자동적으로 실행될 수 있도록 설정한다.

- 스케줄링 데몬은 crond이며 관련 파일은 /etc/crontab이다.

- 파일 /etc/crontab은 7개의 필드로 구성되어 있다.

- 명령어 crontab, at은 사용자가 주기적인 작업을 등록할 수 있게 한다.

- **crontab 파일 형식**

분 시 일 월 요일(에만 이면 콤마(,) 부터까지(-) 일요일:0) 작업내용

(*는 없다는 표시)

- 옵션

[**-l**] : crontab 파일의 목록을 출력

[**-e**] : **사용자**의 crontab 파일을 **편집**

[**-r**] : crontab 파일을 삭제

[**-u**] : **특정 사용자**의 일정 수정

- **/var/spool**/cron : 시스템 **개별 사용자**를 위한 crontab 파일 위치이며 일반적으로 root 계정용 하나와 계정 사용자당 1개의 파일을 가진다.
- /etc/crontab : 관리자가 직접 지정한 작업들을 설정하며, 임의의 사용자 권한으로 실행할 수 있다. **시스템 관련 작업**들을 등록해 사용하는 곳이다.
- /etc/cron.d : 소프트웨어 **패키지**를 설치할 때 필요한 주기적인 작업을 등록하는 공간으로 사용한다. 임의의 사용자 권한으로 실행할 수 있다.

Section 7. 에디터 종류

개요

- 리눅스에서 지원하는 편집기로는 vi, emacs, pico, gedit, xedit 등이 있다.
- 리눅스 편집기는 편집기를 통해 파일을 수정한다.

종류

pico

- 워싱턴 대학의 **아보일 카사르**가 개발한 **유닉스 기반의 텍스트 에디터**이다.
- 메뉴 선택 방식의 텍스트 편집기로 기본 인터페이스가 윈도우의 메모장과 유사하여 간단하다. **단축키 명령어** 도움말이 같이 보여짐
- **자유 소프트웨어 라이선스가 아니기 때문에(아파치 라이센스)** 소스 수정이 불가능하다.
- 다른 편집기에 비해 사용하기 쉽고 사용하기 편리하지만 **기능이 부족하고 업데이트가 잘 되지 않는다**.
- **GNU프로젝트**에서는 pico의 복제 버전 에디너인 **nano**를 개발하였다.
- vi편집기처럼 입력모드와 명령모드가 존재하지 않고 바로 텍스트 입력이 가능하다.

emacs

- **리차드 스톨만**이 **매크로 기능**이 있는 텍**스트 교정 및 편집기**로 개발하였다.
- 최초의 개발자는 리차드 스톨만이며, 이후 제임스 슬링이 **LISP**언어를 기반하여 emacs 에 다양한 기능을 개발하여 추가하였다.
- **LISP** 에 기반을 둔 환경 설정 언어를 가지고 있다.
- xemacs : **GUI**기반으로 동작

vi

- 1976년 **빌 조이**가 초기 BSD 릴리즈에 포함될 편집기로 만들었다.
- 리눅스 배포판과 유닉스에 기본적으로 포함되어 있다.
- 유닉스 환경에서 가장 많이 쓰이는 문서 편집기이다.
- 다른 편집기들과 다르게 모드형 편집기이다.
- 명령모드, 입력모드, 편집모드로 구성되어 있다.

vim

- **브람 무레나르**가 만든 편집기이다. vi에 추가 기능을 강화해서 만듦.

vim을 **GUI기반**으로 개발한 편집기는 **gVim**

vi , vim=문법 강조 기능

- vi 편집기와 호환되면서 독자적으로 다양한 기능을 추가하여 만든 편집기이다.
- 편집 시 **다양한 색상**을 이용하여 가시성을 높일 수 있다.
- 패턴 검색 시 **하이라이트** 기능을 제공하여 빠른 검색을 가능하게 해준다.
- ex모드에서 **히스토리** 기능을 제공한다.
- 확장된 정규 표현식 문법과 강력한 **문법 강**조 기능을 갖는다.

gedit

- 그놈 데스크톱 환경으로 개발된 자유 소프트웨어 **GUI기반** 텍스트 편집기이다.
- 마이크로소프트, 윈도, 맥OS X에서도 사용할 수 있다.
- UTF-8과 호환하며 텍스트 문서를 편집하는 용도에 중점을 두었다.
- X-윈도우 시스템에 맞춰 개발되었다.
- GTK+와 **그놈** 라이브러리를 이용하여 개발되었다. (**gnome edit**)
- 텔넷 접속 시나 텍스트 기반 콘솔 창에서는 사용할 수 없다.

Section 8. 에디터 활용

에디터 기초 사용법

pico

[Ctrl] + [p]	커서를 **윗**줄로 이동한다
[Ctrl] + [f]	커서를 **앞**(오른쪽)으로 이동한다.
[Ctrl] + [b]	커서를 **뒤(**왼쪽)로 이동한다.
[Ctrl] + [I]	화면 갱신
메뉴	기능
[Ctrl] + [O]	파일 저장
[Ctrl] + [X]	파일 종료, 종료 시 저장이 안 되어 있으면 저장할 것인지 물어봄
[Ctrl] + [R]	현재 커서 위치에 다른 파일을 불러옴
[Ctrl] + [A]	현재 행의 맨 앞으로 이동
[Ctrl] + [E]	현재 행의 맨 끝으로 이동
[Ctrl] + [V]	이전 페이지로 이동
[Ctrl] + [Y]	다음 페이지로 이동
[Ctrl] + [C]	현재 커서의 위치를 표시
[Ctrl] + [T]	영문자의 철자를 확인
[Ctrl] + [W]	키를 누르고 문자열을 입력하면 원하는 문자열을 찾음
[Ctrl] + [K]	현재 라인을 삭제
[Ctrl] + [U]	마지막으로 삭제된 라인을 복구

emacs

메뉴	기능
[Ctrl] + [X] [Ctrl] + [S]	**파일 저장**
[Ctrl] + [X] [Ctrl] + [C]	편집 종료
마크 설정 후 [Ctrl] + [W]	잘라내기
[Ctrl] + [K]	커서 뒤에 있는 한 줄이 **모두 지워짐**
[Ctrl] + [A]	커서를 줄의 맨 앞으로 이동
[Ctrl] + [E]	커서를 줄의 맨 뒤로 이동
[Ctrl] + [N]	커서를 한 줄 아래로 이동
[Ctrl] + [S] 찾을 문자열	커서의 아랫부분에서 찾을 문자열을 검색
[Ctrl] + [R] 찾을 문자열	커서의 윗부분에서 찾을 문자열을 검색
[Ctrl] + [G]	진행되고 있는 명령을 끔

- 사용법

pico	emacs	기능
Ctrl + A		줄의 처음으로 이동
Ctrl + E		줄의 끝으로 이동 (End)
Ctrl + P		커서를 윗줄로 이동 (Previous)
Ctrl + N		커서를 아랫줄로 이동 (Next)
Ctrl + Y	Alt + V	이전 페이지로 이동
Ctrl + V		다음 페이지로 이동
Ctrl + K		한줄 삭제 (Kill)
	Ctrl + K	현재 커서 위치에서 줄 끝까지 삭제
Ctrl + U	Ctrl + Y	붙여넣기
Ctrl + O	Ctrl + X + S	저장
Ctrl + X	Ctrl + X + C	종료

vi 에디터 사용법

-명령모드: 문자가 아닌 단축키나 명령어로서 실행되는 모드. 처음 vi를 켰을 EO는 명령모드. 파일 삭제,복사,붙여넣기 작업

-입력모드:글을 입력하는 편집모드.

-EX명령모드(마지막 행 모드):저장, 저장할 때 설정작업

형식 : vi [옵션][파일] :파일이 있으면 파일 열고 없으면 파일 생성.

- Ex 명령모드(콜론 모드=마지막행모드) : ESC를 누르고, :(콜론)을 입력한 상태

[:!<명령어>] : Ex 모드에서 외부 명령어를 실행하는 명령

[:%d] : 파일 안의 모든 내용을 삭제

- vi 에디터에서 파일을 편집 중 비정상 종료가 되었을 때 생기는 파일

Ex] file.txt 편집 중 강제종료 -> **.**file.txt.**swp** (숨김파일이라 앞에**.**이 붙는다)

: set nu 행번호 붙여서 출력해줌= set number =se nu

: set ic :검색시 대소문자 **무시**

: set ai **:자동으로 커서위치 맞춰줌.**

: set ts=num : 탭의 크기 지정

: set sm: showmatch { (입력시 짝이 되는) }를 찾아 머무른다음 돌아옴.

: map : 매크로 지정

: **ab** [약어] 약어 설정

- 환경설정 적용방법 (영구 적용) (1404회)
사용자 홈 디렉터리에 **.exrc** 파일에 저장
환경변수 EXINIT에 지정
- **옵션** (1404회)
[-c] : 시작하면서 **명령**(Ex 모드의 명령)을 실행한다. (뒤에 숫자 입력하면+랑 동일)
[-**R**] : **읽기전용**(Readonly)으로 파일을 실행
[**-r**] : 파일이 손상되었을 경우 **복구** .swp파일의 내용으 불러온다. vi –r [파일명]
[-u] : undo 되감기 실행취소.
[**+**<NUM>] : 파일을 열면서 커서를 <NUM>번째 줄로 위치시킨다. **num이 생략되면 마지막줄**
- 바꾸기(대체)
[:s/문자열1/문자열2/] : 커서가 위치한 행에서 첫 번째로 나오는 문자열1을 문자열2로 바꾼다.
[:<범위>s/문자열1/문자열2/] : 범위 내 모든 행의 각 행에서 첫 번째로 나오는 문자열1을 문자열2로 바꾼다.
[:<범위>s/문자열1/문자열2/**g**] : 범위 내 모든 행에서 문자열1을 문자열2로 바꾼다.
-> 범위 % : **전체**
-> 범위 2,$: 2행부터 **마지막 행**까지
-> 범위 .,$: 현재줄부터 **마지막행**까지
^ : 줄의 시작
.exrc 파일에 [ab <문자열1> <문자열2>]을 추가하면 문자열1이 문자열2로 **자동**으로 바뀐다.
- **복사/잘라내기/붙여넣기** (1301회) (1304회) (1401회)
[yy] : 커서가 위치한 행을 복사. // 3yy는 현재 행부터 세 행을 복사
[**dd**] : 커서가 위치한 행을 잘라내기
[**D**] : 커서가 위치한 곳부터 줄의 끝까지 잘라내기
[p] : 커서가 위치한 행의 아래쪽에 붙여넣기. //
[P]는 위쪽에 붙여넣기
[J] : 커서 아래 행을 커서 행에다가 붙인다. - 11 -
- **검색하기** (1301회) (1302회)
[**/**문자열] : 커서 위치부터 **순방향**으로 문자열 검색
[**?**문자열] : 커서 위치부터 **역방향**으로 문자열 검색
[₩<]: 해당 문자열로 시작 [>₩]: 해당 문자열로 끝
[**/**] : 가장 최근에 검색한 문자열 순방향으로 검색
[**?**] : 가장 최근에 검색한 문자열 역방향으로 검색
[**n**] : 원래 찾던 방향으로 다음 문자열을 찾는다.
[**N**] : 역방향으로 다음 문자열을 찾는다.
- **이동 단축키** (1303회) (1401회)
[Ctrl + U] : 반화면 위로 이동 (Up)

[Ctrl + D] : 반화면 아래로 이동 (Down)
[Ctrl + B] : 한화면 위로 이동 (Before)
[Ctrl + F] : 한화면 아래로 이동 (Forward)
[**h**] : ← 커서를 한 글자 왼쪽으로 이동
[**j**] : ↓ 커서를 한 행 아래로 이동
[**k**] : → 커서를 한 글자 오른쪽으로 이동
[**l**] : ↑ 커서를 한 행 위로 이동
h j k l -> 왼 아래 오른 위
[w] : 현재 커서 위치한 곳의 다음 단어로 이동 (word)
[b] : 현재 커서 위치한 곳의 이전 단어로 이동 (back)
[G] : 현재 문서의 마지막 라인으로 이동
- **입력모드 전환 -- INSERT --**
- **i**:현재커서 **앞**에 삽입하면서 입력모드전환
- I: 현재커서가 위치한 줄의 맨 앞에 삽입하면서 입력모드 전환
- **a**:현재 커서**뒤**에 삽입하면서 입력모드 전환
- A:현재 커서가 위치한 줄의 맨 뒤에 삽입하면서 입력모드 전환
- **o**:현재 커서가 위치한 곳의 **아래줄**에 삽입하면서 입력모드 전환
- O:현재 커서가 위치한곳의 **윗줄**에 삽입하면서 입력모드 전환
- **s**:현재 커서가 위치한 곳의 문자를 지우면서 입력보드로 전환

Section 9. 소프트웨어 프로그램 설치

계열

데미안 계열

- 배포 업체: **Debian.deb**, **Ubuntu**, Xandros, Linspire
- 패키지 툴: dpkg, apt-get, aptitude

레드햇 계열

- 배포 업체: Fedora(d n f), **CentOS**, RHEL, open**SUSE**(yaST,저장소 기반 패키지 관리기법:**zypper**), Mandirva
- 패키지 툴: rpm, yum

```
  vsftpd - 2.2.2-24 .el6    .i686   .rpm
[패키지이름]-[버전]-[릴리즈]-[아키텍처].rpmi

릴리즈 : 한버전의 패키지를 몇번 빌드했나
        (ex)fc23(fedora23), el6(enterprise linux6)
아키텍처 : 패키지가 사용가능한 시스템
        (ex)i386,i486,i586,i686 :인텔x86
            ia64 :IA-64(Itanium)
            x86_64계열 :64bit CPU사용
```

-

rpm

- **레드햇** 사에서 만들어낸 패키지 관리 툴이다.
- 새로운 패키지를 설치하거나 업그레이드, 삭제 시 사용한다.
- Windows의 setup.exe와 유사하게 만든 프로그램이다.
- 레드햇 계열의 패키지 파일 확장명은 *.rpm 이다.

- rpm 패키지 구조 :

pkg**이름** pkg**버전** pkg**릴리즈** 아키텍처 확장자

- 옵션

[**-i**] : 패키지 **설치**

[-f] : 설치관련 옵션

[-**F**] : 이전버전 패키지 **있는 경우에만** 패키지 설치

[-h] : 설치과정 ############

[**-U**] : 패키지 **업데이트**

[-**e**] : 패키지 **삭제 (--erase)**

[**-qa**] : **전체** 패키지 목록 출력 ' **'문자열이 들어가있는 패키지 출력**

[-**qd**] : 패키지 관련 **문서** 파일 정보 출력

[-**qi**]: 자세한 정보를 출력/ **패키지 이름과 버전만** 표시

[-**ql**]: 패키지 내의 모든 **파일**을 출력

[**-qf**] : 사용자가 지정한 **형태**로 출력. 파일에 포함하는rpm패키지 출력

[**-V**] : 설치된 패키지들이 보안상 침입자에 의해 권한 획득이나 변조가 되었는지를

```
S.5....T.    c /etc/mail/sendmai.cf
```

검사

[--test] : 패키지를 설치하지 않고 제대로 설치되는 지 확인

[--force]: 강제 설치 replacepkgs, replacefiles, oldpackages

[**--nodeps**]:**의존성**무시하고 제거

- 검증 옵션 [rpm -V <검증옵션>] (1404회)

S : 파일 크기

M : 권한, 파일 타입을 포함한 모드

L : 심벌릭 링크

U : 사용자

G : 그룹

yum /etc/yum.repos.d에 저장

- 네트워크를 통해 기존 RPM 패키지 파일의 업데이트 자동 수행, 새로운 패키지 설치 및 제거를 수행한다.
- RPM(레드햇)의 **의존성 문제를 해결**하기 위한 유틸리티이다.
- 인터넷을 기반으로 설치하므로 네트워크가 정상적으로 연결된 상태여야만 한다.
- YUM은 페도라 22버전 이후부터 YUM의 문제점을 보안한 DNF로 전환되고 있다.

- 옵션

[-y] : 설치과정의 모든 질문에 yes로 답한다.

[**info**] : **해당** 패키지 정보를 확인

[**list**] : 패키지 **목록**을 확인

[**install**] : 패키지 설치 의존성이 걸려있으면 해당패키지 까지 설치

[**update**] : 패키지 업데이트

[check-update] : 업데이트가 **가능한** 패키지 목록

[**remove**] : 패키지 삭제 = [**erase**]

[**groupinstall**] : 그룹 패키지 설치

[**grouplist**] : 설치된 패키지의 그룹별 정보를 확인

[**search**] : 문자열이 있는 패키지를 찾음

[**history**] : 작업 이력을 출력

dpkg

- 데비안의 저레벨 패키지 관리 툴이다.
- deb 패키지의 설치, 삭제, 정보 제공을 위해 사용된다.
- 확장자 .deb 파일은 데비안 패키지 파일이다.
- 패키지 설치 및 제거 시 RPM과 같은 의존성 문제를 일으킨다. ->apt-get 사용
- 명령어 'dpkg **–s** 패키지'는 지정된 패키지에 대한 자세한 정보를 나타낸다.

[**-r**] : 패키지 삭제 (환경 설정 파일을 **남겨둠**)

[-P] : 환경설정 파일까지 전부 제거(Purge)

[-i] : 패키지 **설치**

[-l] : 설치된 **패키지** 리스트 출력

[-L] : 패키지가 설치한 **파일** 리스트를 출력

[-c] : 패키지 파일에 포함된 파일 정보를 출력

apt-get

- **데미안** 리눅스에서 소프트웨어 설치와 제거를 위한 패키지 관리 유틸리티이다.
- 패키지 관련 정보를 확인하거나 패키지 설치 시 발생할 수 있는 의존성과 충돌문제를 해 결하기 위해 **/etc/apt/source.list** 파일을 참조한다 이 파일에는 패키지 유형(바이너리소스), 저장소 주소(URL), 우분투 버전정보, 카테고리로 구성

[-- purge] : **remove** 명령을 수행할 때 환경설정까지 같이 제거

[clean] : /var/cache/**apt**/archive 패키지 파일 내용을 제거

aptitude

- 우분투 패키지 관리 유틸리티로 APT처럼 패키지를 관리를 자동화한다.

Section 10. 소스 파일 설치

파일 아카이브와 압축

파일 아카이브

- 아카이브는 다수 개의 파일이나 디렉터리를 하나의 파일로 묶는 것이다.
- 아카이브 파일은 다른 시스템으로 다수 개의 파일을 한 번에 전송하거나 파일 백업용으로 사용한다.

파일압축과 해제

- 대표적인 파일 압축 명은 compress< gzip< bzip2< xz가 있다.
- 일반적으로 많이 사용되는 압축 명령어는 **gzip**과 **bzip2**이다.
- 압축률이 가장 낮은 것은 명령어 compress(압축해제는 uncompress)이며, 압축률이 가장 높은 것은 명령어xz이다.

- compress : 유닉스에서 사용. LZW 압축알고리즘을 이용해 만듦
- gzip2 : 전통적으로 유닉스에서 사용. LZ77알고리즘과 Huffman coding을 이용하며 현재도 자주쓰인다.

- bzip2: : 블록정렬 알고리즘과 허브만 부호화를 사용하여 압축률이 좋다.
- xz : LZMA2 알고리즘을 사용했으며 리눅스 계열 운영체제에서 자주 쓰인다.

- **명령어 tar(테이프 아카이브) 옵션**

[**-cvf <*.tar> <file>**] : 아카이브 **생성**

[**-x**] : 아카이브 **풀기** (.xz)->최근에 배포 된 것. 압축률이 가장좋음

[-d] :압축해제

[**-t**] : 아카이브 **내용 확인, 파일 목록확인**

[**-r**] : 기존 아카이브에 **새로운 파일 추가**

[**-j**] : bzip2(.bz) 압축/ bunzip2 압축해제

[**-J**] : **xz** 압축/ unxz 압축해제

[**-z**] : gzip(.gz) 압축/ gunzip 해제 zcat : gzip으로 된 텍스트 파일을 내용 확인

[-Z] : compress 형식으로 압축/해제

[-v] : 처리하고 있는 **파일의 정보**를 자세하게 **출력**

[**-f**] : **이름 지정**

[**-C**] : **다른 곳에 압축을 해제**

소스 코드 설치

- 소스 코드를 압축 해제 후 컴파일 순서에 따라 프로그램을 설치한다.
- 컴파일 순서는 설치 파일의 환경설정, 컴파일, 파일 설치이다.
- 1단계 환경설정: **./configure** 프로그램 설치 과정에서 필요로 하는 환경파일 **makefile** 생성

  configure작업으로 생성된 파일 제거할 때는 make clean
- 2단계 컴파일: **make** **makefile**을 기반으로 소스 파일을 **컴파일(실행파일 생성)**

[make] : 유닉스 계열 OS만 지원

[**cmake**] : 소스 컴파일시 사용되는 make의 대체 프로그램. **멀티 플랫폼**을 지원 MySQL 설치시 사용. configure과정이 필요하지 않다. 크로스 컴파일.마이크로소프트 Visual Studio.Net 및 Visual Studio를 지원한다. 타임스탬프를 통해 파일 내용 변화를 알 수 있으며 이클립스용 빌드 파일도 생성 가능하다. 또한 평행 빌드(Parallel builds)도 지원한다.

[gcc]

[tar]

- 3단계 파일 설치: **make install** 컴파일 된 실행파일을 지정된 속성으로 지정된 디렉터리에 설치(--prefix = 절대경로)

Section 11. 주변 장치 연결 및 설정

프린터 인쇄 시스템 설치 및 설정

LPRng

- **리눅스 초기에 사용되었던 인쇄 시스템**이다.
- **버클리** 프린팅 시스템으로 **BSD 계열** 유닉스에서 사용하기 위해 개발되었다.
- **라인 프린터** 데몬 프로토콜을 사용하여 프린터 **스폴링**과 **네트워크 프린터** 서버를 지원한 다.
- LPRng설정 파일은 **/etc/printcap**이다.

CUPS

- **애플**이 개발한 오픈 소스 프린팅 시스템이다.
- 유닉스 계열 운영체제의 시스템을 프린터 서버로 사용 가능하게 해준다.
- 매킨토시나 윈도우 등 시중에 시판되는 대부분의 프린트를 지원한다.
- **HTTP** 기반의 IPP(Internet Printing Protocol)를 사용하여 프린터를 웹 기반으로 제어한다.
- CUPS 설정 파일은 /etc/cups이다.
- 사용자 및 호스트 기반의 인증을 제공한다.
- CUPS 관련 파일은 **cupsd.conf**(**데몬** 환경 설정 파일), **printers.conf**(**프린터 큐** 관련 환경 설정 파일), **classes.conf**(프린터 데몬의 **클래스** 설정 파일), **cpused**(**프린터 데몬**) 등이 있다.

프린터 설정

- 일반적으로 X-Windows상에서 '프린터 설정 도구'로 프린터를 설치한다.
- '로컬 접속'으로 프린터를 직접 연결할 수 있다.
- 네트워크 프린터를 설정할 경우 5가지 방법을 제공하고 있다.

AppSocket/HP jecDirect	프린터가 컴퓨터에 연결되어 있지 않고 네트워크에 연결된 경우 사용
LPD/LPR 호스트 또는 프린터	IPP 프로토콜 기반의 프린터 설정 시 사용
Windows Printer vis SAMBA	윈도우 시스템에 연결된 프린터 설정 시 사용 삼바 기반의 SMB 프로토콜 사용
인터넷 프린터 프로토콜 https	https 프로토콜 기반의 프린터 설정 시 사용
인터넷 프린터 프로토콜 ipp	IPP 프로토콜 기반의 프린터 설정 시 사용

사운드 카드 설치 및 설정

OSS(Open Sound System)

- 리눅스 및 유닉스 계열 운영체제에서 사운드를 만들고 캡쳐하는 인터페이스이다.
- **표준 유닉스** 장치 시스템콜에 기반을 둔 것이다.
- 현재 리눅스 커뮤니티에서는 **ALSA**로 대체되었다.

```
- 1992년에 Hannu Savolainen에 의해 개발
- 2007년 7월에 Linux의 GPL기반 라이선스로 소
  스를 공개
- 2008년 4월에 BSD라이선스 기반으로 소스를
  추가로 공개
```

ALSA(Advanced Linux Sound Architecture)

- **사운드 카드용 장치 드라이버**를 제공하기 위한 **리눅스 커널 요소**이다.
- GPL 및 LGPL 라이선스 기반으로 배포되고 있다.
- 사운드 카드를 **자동**으로 구성하고 시스템에 여러 개의 사운드 장치를 관리하는 것이 목적이다.
- OSS의 지원을 받아서 하드웨어 기반 미디합성, 다중 채널 하드웨어 믹싱, 전이중 통신, 다중 프로세서와의 조화, 스레드 안전장치 드라이브 등의 기능을 지원한다.
- 현결 설정 파일은 /etc/asound.state이다.

[**alsactl**] : 사운드 카드의 설정 정보를 초기화하거나 저장하고 읽는다.
[-v] : 버전 출력
[-d] : 디버그 모드
[-f] : 환경설정 파일 설정 (Default값 : **/etc/asound.state**)
[**init**] : 사운드 장치를 초기화
[**alsamixer**] : 사운드카드의 볼륨을 조정
[**cdparanoia**] : 오디오 CD에서 음악 파일을 추출할 때 사용하는 프로그램

스캐너 설치 및 설정

SANE(Scanner Access Now Easy)

- 평판 스캐너, 핸드 스캐너, 비디오 캠 등 이미지 관련 하드웨어를 제어하는 API이다.
- **GPL** 라이선스, 리눅스 및 유닉스 계열, OS2, Windows도 지원한다.

[scanimage] : 이미지 스캔

```
# scanimage - 150 - 180 > scan.log
```

가로 150mm 세로 180mm 스캔한 이미지를 > scan.log 파일로 저장
[**sane-find-scanner**] : USB 및 SCSI 스캐너와 관련 장치 파일을 찾아주는 명령어

-p: **병렬(Parallel)** 포트에 연결된 스캐너 찾기
-q: 스캐너 장치만 출력
-v: 상세한 정보 출력

XSANE(**X based** interface for the SANE)

- SANE 스캐너 인터페치스를 이용하여 **X-Windows 기반**(GTK+ 라이브러리 기반)의 스캐너 프로그램이다. xsane 명령으로 실행
- 스캐너, 디지털 카메라, 디지털 캠 등 다양한 장치에서 사용이 가능하다.
- 스캔 작업뿐만 아니라 캡쳐한 이미지에 수정 작업을 할 수도 있다.
- GPL 라이선스, 리눅스 및 유닉스 계열, OS2, Windows도 지원한다.

Section 12. 주변 장치 활용

프린터 설치 및 설정

BSD 계열 프린터 명령어들

- **lpr:** 프린터 작업 요청을 한다. **인쇄**.
  [lpr -**#**] : 출력할 문서의 **장수** 지정
  [lpr **-T**] : 타이틀 페이지의 타이틀 명 설정
  [lpr -**P**] : 프린터 지정
  [lpr – r]: **인쇄후** 삭제

- **lpq**: 프린터 큐에 있는 작업 목록을 출력한다. 프린트 작업상태 점검
- **lprm**: 프린터 큐에 대기 중인 작업을 **삭제**한다. 취소할 프린트 작업 번호를 입력한다. 작업번호 입력안하면 마지막에 작업한게 삭제된다.
  [**lprm -**] : 프린터 작업을 모두 취소
- **lpc**: 라인 프린터 컨트롤 프로그램이다. 프린터나 프린터 큐를 제어. 프린터 관리자

System V 계열 프린터 명령어들

- **lp**: 프린터 작업 요청(명령어 lpr과 유사한 기능)을 한다. **인쇄**
  [lp –**n**]: 출력할 문서의 **장수** 지정
- **lpstat**: 프린터 큐의 상태를 확인한다.(lpq와 유사)
- **cancel**: 프린트 작업을 취소한다. 취소할 요청 ID를 **lpstat로 확인 후 삭제**한다.

사운드 카드 관련 명령어

[-d] : 디버그 모드

[-E] : 환경 변수를 설정

[-f] : 환경설정 파일 설정 (Default값 : /etc/sound.state)

[init] : 사운드 장치를 초기화

[-i] : init을 위한 설정 파일을 지정

- **alsamixer**: **커서** 라이브러리 기반의 오디오 프로그램이다.
- **cdparanoia**: 오디오 CD로부터 음악 파일을 추출 시 사용한다. (.wav)

스캐너 관련 명령어

- **sane-find-scanner**: SCSI 스캐너와 USB 스캐너 관련 장치 파일을 **찾아주는** 명령어이다.
- scanimage: 기본 이미지.pnm를 **스캔**한다.
- scanadf: 자동 **문서 공급 장치**가 장착된 스캐너에서 여러 개의 사진을 스캔한다.
- **xcam**: **GUI 기반**으로 평판 스캐너나 카메라로부터 이미지 스캔한다.T

[lsmod] : 커널이 현재 사용 중인 **모듈**을 출력

[lsof] : list open files의 약자. 시스템에서 **열린 파일** 목록을 알려준다.

[lspci] : 설치된 PCI 관련 **장치**의 목록을 사용할 때 사용