1과목 리눅스 실무의 이해

CH.1 리눅스의 개요

1. 운영체제의 개요

1-1. 운영체제의 이해

운영체제의 주요 역할
- HW 제어, 사용자들 간에 HW 자원을 공유
- 스케줄링을 통해 자원의 효율적 사용
- I/O 용이
- 오류 방지/복구
- 편리한 사용자 인터페이스 제공

운영체제 유형 2014(2) 2014(2) 2015(2) 2015(2)
- 다중 교환(Multi-switching) : 다수의 작업이 동시 실행. 포그라운드 프로그램만 동작
- 단일 작업(Single-tasking) : 한 번에 하나의 작업만 처리
- 다중 작업(Multi-tasking) : 여러 개의 작업을 동시에 수행
- 다중 사용자(Multi-user) : 단일 프로세서에서 여러 사용자의 프로그램이 실행
- 대화형 처리(Interactive Processing) : 대화형으로 작업을 처리 (시분할 처리 기능 필요)
- 일괄 처리(Batch Processing) : 여러 개의 작업을 묶어 한 번에 처리
- 실시간 처리(Real Time Processing) : 작업의 처리가 지연없이 즉각적으로 처리
- 분산 처리(Distributed Processing) : 여러 시스템을 연결하여 작업을 나누어 처리

시스템 성능을 나타내는 4가지 요소 2015(2)
- 처리 능력(Throughput) : 단위 시간당 처리 능력
- 반환 시간(Turnaround Time) : 작업이 제출되어 결과를 얻을 때까지의 총 소요시간
- 신뢰도(Reliability) : 시스템이 얼마나 정확하게 작동되는지를 나타냄
- 사용 가능도(Availability) : 시스템에서 곧 사용할 수 있는 정도를 나타냄

가상메모리 2016(1) 2016(2) 2018(1)
- 하드디스크의 일부를 메모리(RAM)처럼 사용하는 것
- 하드디스크에서 가상메모리로 쓰이는 영역을 **스왑 영역(swap space)**이라고 한다.
- **스와핑(Swapping)** : 메모리와 하드디스크 사이의 데이터 교환

페이지 교체 알고리즘 2014(1)
- FIFO(First-In First-Out) : 먼저 적재된 페이지를 제거하는 알고리즘
- LRU(Least Recently Used) : 가장 오랫동안 참조되지 않은 페이지를 제거하는 알고리즘
- LFU(Least Frequently Used) : 최근에 가장 적게 사용된(참조된) 페이지를 제거하는 알고리즘
- NUR(Not Used Recently) : 참조비트와 변형비트를 사용하여 가장 오랫동안 참조되지 않은 페이지를 제거하는 알고리즘(LRU 알고리즘의 변형)
- SCR(Second Chance Replacement) : FIFO 알고리즘의 단점을 보완한 알고리즘. 참조비트로 한번의 기회를 더 부여하는 알고리즘

1-2. 운영체제의 종류

리눅스의 특징 2014(2) 2016(1) 2018(1)
- CPU 최적화 : 인텔 CPU 뿐만 아니라 AMD와 Cyrix CPU에도 최적화되어 있음
- 뛰어난 네트워킹 : TCP/IP 뿐만 아니라 IPX/SPX, SLIP, PPP 등의 여러 네트워킹 프로토콜을 지원
- 뛰어난 이식성/확장성 : 다른 OS에 비해 이식성, 확장성이 뛰어남
- 6개의 가상 콘솔 제공 : 하나의 모니터를 장착한 시스템에서 기본적으로 6개의 가상 콘솔을 제공
- 효율적인 가상메모리

2. 리눅스 기초

2-1. 리눅스의 철학

GNU 프로젝트 2015(1)

- GNU : Gnu is Not Unix 라는 뜻의 재귀적 약어
- 유닉스 호환 자유 소프트웨어들을 개발하는 프로젝트
- GNU 소프트웨어에는 gcc, emacs, Bash 쉘, GNOME 등이 있다.

GPL(General Public License) 2018(1)

FSF에 의해 만들어진 특별한 오픈소스 라이선스. "카피레프트(copyleft)"

FSF(Free Software Foundation, 자유 소프트웨어 재단) 2016(2)

- 리차드 스톨만이 설립한 비영리 조직.

• **주요 특징** 2016(1) 2017(2)

- 해당 프로그램을 마음대로 배포, 복사, 소스코드 수정이 가능
- GPL 프로그램의 소스코드를 수정해 만든 프로그램 또한 GPL을 가진다.
- 다른 소프트웨어와의 완전한 통합은 해당 소프트웨어가 GPL을 수용한다는 조건하에서만 허용
  (독점 소프트웨어와 결합이 불가능)
- 독점 소프트웨어와 결합 시 해당 영역의 소스를 공개해야 한다.

LGPL(Lesser General Public License) 2018(1)

- GPL의 조건을 완화시킨 라이선스 (GPL 특징 중 전염성을 완화)
- LGPL 프로그램의 소스코드를 이용하여 만든 프로그램을 판매/배포 시 프로그램 소스 공개 의무가 없고 LGPL 소스코드를 사용했음을 명시하면 된다. 하지만 수정한 경우는 전체 코드를 공개해야 한다.
- 독점 소프트웨어와 결합이 가능하다.

BSD(Berkeley Software Distribution) 라이선스

• BSD 계열 소프트웨어에 적용되는 오픈소스 라이선스

• 주요 특징 2015(1)

- 소스코드 공개의 의무가 없음.
- 저작권자 표기. BSD 라이선스의 소프트웨어를 이용하는 경우, 해당 저작권자의 이름과 BSD 라이센스의 내용을 같이 배포해야 한다.
- 독점 소프트웨어와 결합 시 원래 소스의 저작권자와 관련 사항을 표기하면 소스 공개 의무는 없다.

GNU/FSF에서 규정한 자유 소프트웨어 조건 2014(1) 2016(1)

- 프로그램을 어떠한 목적으로도 실행할 있는 자유
- 프로그램의 작동 원리를 연구하고, 이를 자신의 필요에 맞게 수정할 수 있는 자유
- 이웃을 돕기 위해서 프로그램을 복제하고 배포할 수 있는 자유
- 프로그램을 향상시키고 이를 공동체 전체의 이익을 위해서 다시 환원시킬 수 있는 자유

2-2. 리눅스의 역사

리눅스 배포판의 종류 2014(1) 2014(2) 2015(1) 2016(1) 2017(2)

- **레드햇(RedHat)**
  - RPM 기반으로 제작된 리눅스 배포판
  - 현재는 기업용 배포판으로 상업적으로 배포하고 있으며, 무료 버전으로는 페도라가 있다.
- **페도라(Fedora)**
  - RPM 기반. 레드햇 계열
  - 레드햇의 지원을 받아 개발 및 유지보수가 이루어진다.
  - 페도라의 업데이트 후에 문제점을 파악하여 레드햇 리눅스(RHEL)에 업데이트를 반영하는 방식으로 운영되고 있다.
- **데비안(Debian)**
  - GNU의 후원을 받는 리눅스 배포판
  - 패키지 설치 및 업그레이드가 단순하다. (패키지 관리 – dpkg, apt)
- **우분투(Ubuntu)**
  - 데비안 계열
  - 영국 회사인 캐노니컬의 지원을 받음
  - 유니티(Unity)라는 고유한 데스크톱 환경을 사용
- **슬랙웨어(Slackware)**
  - 초창기에 나온 배포판으로 현재까지 살아있는 가장 오래된 배포판이다.
  - 패트릭 볼커딩에 의해 만들어짐.
  - 구조가 간결하지만 설치과정이 어렵고 패키지 관리가 어려워 많이 사용되지는 않는다.
- **오픈 수세(openSUSE)** 2017(2)
  - 대표 기능으로 YaST 유틸리티가 있다.
  - 여러가지 데스크탑 환경(KDE, GNOME 등)의 버전이 있으며, KDE 판이 가장 유명하다.
- **맨드레이크(Mandrake)**
  - RPM 기반.
  - 단순함을 추구하며 다양한 데스크탑 환경을 제공한다.

  우리나라의 리눅스 배포판 2015(2)
  - **SULinux, 안녕 리눅스, 아시아눅스** 등
- Chrome OS
  - 구글에서 개발한 리눅스 기반 OS
  - 인터페이스는 웹 브라우저인 크롬과 비슷하다.

리눅스 커널이 사용된 모바일 OS 2014(1)

- **안드로이드(Android)** 2018(1)
  - C/C++ 라이브러리들을 포함하는 오픈소스 플랫폼
  - 가상머신은 Java 가상머신이 아니라 구글에서 자체 개발한 달빅(Dalvik) 가상머신을 사용
- **바다(Bada) OS** 2016(2)
  - 삼성에서 개발한 리눅스 커널 기반의 OS
  - **미고(MeeGo)**와 **리모(LiMo)**가 통합된 타이젠(Tizen)과 통합되었다.
- **타이젠(Tizen) OS** 2017(2)
  - 삼성과 인텔의 주축으로 개발된 오픈소스 OS이다.
  - HTML5 기반으로 만들어졌으며, 자바스크립트, CSS와 같은 웹 표준을 지원

CH.2 리눅스 시스템의 이해

1. 리눅스와 하드웨어

1-1. 하드웨어의 이해

1-2. 하드웨어의 선택

RAID(Redundant Array of Independent Disks) 분류 2014(2) 2015(1) 2015(2) 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)

- **RAID 0** : **스트라이핑** 기능(분배 기록) 사용. 빠른 I/O 성능, **고장 대비 능력 X**
- **RAID 1** : 두 개 이상의 디스크를 **미러링**을 통해 하나의 디스크처럼 사용.
- **RAID 2** : **ECC(에러 검출 기능)** 탑재
- **RAID 3** : **하나의 디스크를 에러검출을 위한 패리티 정보 저장용**으로 사용하고 나머지 디스크에 데이터를 균등하게 분산 저장
- **RAID 4** : RAID 3 방식과 같지만 블록 단위로 분산 저장
- **RAID 5** : 하나의 디스크에 **패리티 정보를** 저장하지 않고 **분산 저장** (회전식 패리티 어레이)
- **RAID 6** : **하나의 패리티 정보를 두개의 디스크에 분산 저장**. 쓰기 능력은 저하될 수 있지만 **고장 대비 능력이 매우 높음**. 두 개의 오류까지 검출 가능.
- **RAID 7** : 실시간 운영체계를 사용
- **RAID 0+1** : 최소 4개 이상의 디스크를 2개씩 스트라이핑으로 묶고(RAID 0) 미러링으로 결합(RAID 1)한 방식
- **RAID 10** : 두 개의 디스크를 미러링으로 묶고(RAID 1) 스트라이핑(RAID 0)으로 결합한 방식

SCSI(Small Computer System Interface) 2014(1)

- 주변기기를 연결하기 위한 표준 인터페이스
- **고성능**이고 **호환성/확장성**이 뛰어남.

2. 리눅스의 구조

2-1. 부트 매니저

리눅스 부트로더 LILO/GRUB

• 부트로더 : OS가 시동되기 이전에 미리 실행되면서 커널이 올바르게 시동되기 위해 필요한 모든 관련 작업을 마무리하고 최종적으로 OS를 시동하기 위한 프로그램

• **LILO(Linux Loader)**
  - 리눅스 배포판의 표준 부트로더
  - 오래된 리눅스용 기본 부트로더
  - 대화형 명령어 인터페이스가 없고, 네트워크 부팅 지원하지 않음.

• **GRUB(Grand Unified Boot loader)**
  - 리눅스, vSTA, DOS 및 기타 OS에서 사용할 수 있는 부트로더
  - 새로운 기본 부트로더
  - 대화형 명령어 인터페이스가 있고, 네트워크 부팅 지원함.
  - 부트로더에 문제가 있을 시 **grub-install 명령어**를 사용하여 복구 2014(2)

2-2. 디렉터리 구조 및 디렉터리 별 기능

디렉터리 분류 2014(1) 2014(2) 2015(2) 2016(2) 2017(1)

• **/bin** : 기본 명령어들이 저장

• **/sbin** : 시스템 관리를 위한 명령어들이 저장

• **/etc** : 환경설정에 연관된 파일들과 디렉터리들이 저장 2016(1)
  - /etc/rc.d : 시스템의 부팅과 런 레벨 관련 스크립트들이 저장
  - /etc/inittab : init을 설정하는 파일
  - /etc/issue : 로그인을 위한 프롬프트가 뜨기 전에 출력되는 메시지를 설정하는 파일
  - /etc/issue.net : issue 파일과 기능은 같으나, 원격지 상에서 접속(telnet 등)할 경우에 출력되는 메시지 설정
  - /etc/motd : 로그인 성공 후 쉘이 뜨기 전에 출력되는 메시지를 설정하는 파일
  - /etc/nologin.txt : 사용자의 쉘이 /sbin/nologin으로 지정되어 있을 때, 로그인 거부 메시지를 설정하는 파일

• **/boot** : 리눅스 부트에 필요한 부팅 지원 파일들이 저장

- **/mnt** : 외부장치를 마운트하기 위해 제공되는 디렉터리
- **/usr** : 각종 응용프로그램들이 설치되는 디렉터리
  - /usr/bin : /bin 디렉터리에 없는 다양한 실행파일이 저장
  - /usr/src : 시스템에서 사용하는 각종 프로그램들의 컴파일되지 않은 소스파일들이 저장 2015(2)
- **/lib** : 각종 라이브러리들이 저장
- **/dev** : 장치 드라이버들이 저장되는 가상 디렉터리
  - /dev/console : 시스템의 콘솔
  - /dev/null : 폐기를 위한 디렉터리. 이 디렉터리로 파일이나 데이터를 보내면 폐기된다.
    EX) # find / -perm -4000 2>/dev/null : 명령어 실행결과에서 오류메시지를 출력하지 않고 폐기 2016(2)
- **/proc** : 시스템의 각종 프로세서, 프로그램 정보, 하드웨어적인 정보들이 저장되는 가상 디렉터리
  - /proc/partitions : 파티션 정보 파일 2014(1)
- **/var** : 시스템에서 사용되는 동적인 파일들이 저장
  - /var/log : 프로그램들의 로그 파일들이 저장

2-3. 부팅과 셧다운

관련 명령어

- **halt** : 시스템 종료. [-f] 옵션으로 강제 종료 가능
- **reboot** : 시스템 재부팅. [-f] 옵션으로 강제 재부팅 가능 2014(1)
- **shutdown**
  - [-h] : 종료 옵션
    EX) # shutdown -h now : 즉시 종료
  - [-r] : 재부팅 옵션
    EX) # shutdown -r +30 : 30분 후에 재부팅
  - [-c] : 명령 취소
  - [-k] : 실제로 종료하지 않고 종료하겠다는 메시지만 사용자들에게 보냄
- **init** : 런레벨(RunLevel)을 변경하는 명령어 2014(2)
  EX) # init 0 : 시스템 종료
  # init 6 : 시스템 재부팅
- **ntsysv** 명령어 2016(1) 2017(2)
  - 부팅할 때 각 런레벨에 따라 자동으로 실행시킬 서비스를 설정하는 명령어

부팅과정 2014(2) 2015(1)

1. **BIOS**(Basic Input Output System) 실행, **POST**(Power On Self Test) 수행
2. **MBR**(Master Boot Record) 읽고, **부트로더(LILO/GRUB)** 로드
3. **커널 이미지** 로드
4. 각 장치 드라이버 초기화, 파일 시스템 검사, init 프로세스 호출
5. **/etc/inittab** 파일 참조
   - 런레벨 관련 파일
6. **/etc/rc.d/rc.sysinit** 스크립트 실행
   - **/etc/init/rcS.conf** 파일 적용 : 시스템 초기화 관련 설정 파일. rc.sysinit 스크립트를 실행
7. 해당 런레벨에 맞는 **/etc/rc.d/rc#.d/*** 스크립트 실행
   - **/etc/init/rc.conf** 파일 적용 : 런레벨별로 진행되는 내용이 설정된 파일. rc#.d/* 스크립트 실행
8. **/etc/rc.d/rc.local** 스크립트 실행 2016(2)
   - 부팅 시 필요한 서비스를 이 스크립트 안에 등록
9. **로그인 프롬프트** 출력

2-4. 파일시스템의 이해

파일시스템 종류

- **ext3**
  - ext2에서 fsck의 단점을 보완하기 위해 저널링 기술을 도입한 파일시스템
  - **저널링(Journaling) 기술** : 데이터를 디스크에 쓰기 전에 로그에 데이터를 남겨 fsck보다 빠르고 안정적인 복구 기능을 제공하는 기술 2018(1)
  - 최대 볼륨크기 2TB ~ 16TB / 최대 파일크기 16GB ~ 2TB 지원 / 하위 디렉터리 수 : 32000개
- **ext4** 2014(2) 2017(1)
  - ext3의 기능을 향상시킨 파일시스템
  - ext2/ext3 파일시스템과 호환 가능
  - 지연된 할당 : 데이터가 디스크에 쓰여지기 전까지 블록 할당을 지연시켜 향상된 블록 할당이 가능
  - 최대 볼륨크기 1EB / 최대 파일크기 16TB 지원 / 하위 디렉터리 수 : 64000개
- **XFS** 2014(1) 2015(1)
  - 고성능 64비트 저널링 파일 시스템
  - 리눅스 커널 2.4.20 버전에서 커널로 포팅되었다.

3. X 윈도

3-1. X 윈도의 개념 및 특징

X 윈도 시스템의 4가지 요소 2015(2)

- 서버/클라이언트
- **X 프로토콜** 2017(1)
  - X 서버와 X 클라이언트의 상호 작용을 위한 메시지 교환에서 메시지 형태와 사용법을 X 프로토콜이라 함.
- **Xlib** 2015(1) 2015(2)
  - C언어로 작성된 X 클라이언트 라이브러리
  - 저수준 인터페이스이기 때문에 상위 라이브러리인 Xtoolkit을 사용한다.
  - X.org에서는 Xlib 대신 XCB를 사용하고 있다.
- **Xtoolkit**
  - 상위 라이브러리
  - Qt, GTX 등이 있음.

XFree86 와 **X.org** 2016(2) 2017(1)

- 리눅스/유닉스 계열의 X 윈도 시스템 프로젝트
- XFree86의 라이선스 논란 이후에 X.org 서버가 사용되고 있다.
- 현재 대부분의 리눅스 배포판은 X.org를 사용한다.

관련 명령어 2018(1)

- **startx** 2016(2)
  - X 윈도 구동 명령어
    EX) # startx -- :1 : 두 번째 윈도 터미널에 X 윈도를 구동
  - 명령어 오류 발생 시 **Xconfigurator**을 실행하여 설정 2014(1)
- **xhost** 2016(1)
  - X 윈도 서버의 호스트 접근 제어를 하기 위한 명령어
    EX) # xhost + 192.168.100.100 : 해당 호스트에 대한 접근을 허용
- **xauth**
  - X 서버 연결에 사용되는 권한 부여 정보(.Xauthority 파일의 MIT-MAGIC-COOKIEs 값) 편집/출력 명령어
    EX) # xauth list $DISPLAY : 현재 MIT-MAGIC-COOKIEs 값을 출력
    # xauth add $DISPLAY . '쿠키 값' : .Xauthority 파일에 MIT-MAGIC-COOKIEs 값을 추가

X 윈도 데스크톱 환경 종류

- KDE(K Desktop Environment) 2015(1)
  - Qt 라이브러리를 사용
- GNOME(GNU Network Object Model Environment)
  - GTK 라이브러리를 사용
  - GNU 프로젝트의 일부이며, 리눅스 계열에서 가장 많이 쓰인다.

디스플레이 매니저 2017(2)

- X 윈도 상에서 작동하는 프로그램
- 로그인 창을 통해 사용자 인증을 수행한다.

X 윈도 소프트웨어

- Evince(에빈스) 2017(2)
  - 문서 뷰어 프로그램
  - 지원 파일 형식 : PDF, PS, XPS, TIFF 등
- LibreOffice(리브레 오피스) 2016(1)
  - 오피스 프로그램
  - MS Office 등의 오피스 프로그램과 호환
  - Writer(워드), Calc(스프레드시트/엑셀), Impress(프레젠테이션/파워포인트), Base(DB 관리) 등의 프로그램 지원
- Cheese Photo Booth(치즈) : 웹캠 프로그램
- Rhythmbox(리듬박스) : 오디오 플레이어
- Shotwell(샷웰) : 사진 관리 프로그램

4. 쉘

4-1. 쉘의 이해

쉘 기능 및 종류 2018(1)

- 쉘(Shell) : 명령어 해석기. 커널과 직접적으로 연결되어 해석 결과를 커널로 보냄
- **본 쉘(Bourne Shell, sh)**
  - 스티븐 본이 개발하였으며, 강력한 명령 프로그래밍 언어 기능을 갖추고 있음
  - 상호 대화형(Interactive) 방식을 사용하지 않음
- **C 쉘(csh)** 2017(2)
  - 빌 조이가 개발하였으며, C 언어와 유사한 언어를 사용
  - 상호 대화형 방식으로 구성
- **콘 쉘(Korn Shell, ksh)**
  - 데이브 콘이 개발하였으며, 사용하기 편리하고 기능이 탁월하다.
  - 명령행 편집 기능을 제공
- **배시(Bourne-Again Shell, bash)**
  - 브라이언 폭스가 개발하였으며, sh 호환의 명령어 해석기
  - 처음 로그인 했을 때 디폴트로 주어지는 쉘이다.

특수문자 2017(1) 2017(2)

- **$** : 변수 접근 기호
  EX) $value : 변수 value
  $SHELL : 환경변수 SHELL. 사용하는 쉘의 위치가 저장되어 있음
- **₩** : 이 문자 뒤에 나오는 특수문자는 문자로 처리된다. (escape 처리된다고 말한다.)
  EX) # echo ₩$a → $a 문자 출력
  # echo $a → 변수 a의 값 출력
- **#** : 주석 처리 문자

- * : 0개 이상의 문자가 일치함을 나타내는 치환 문자
  EX) a*e : apple, ace 등의 문자가 포함됨.
- ? : 1개의 문자가 일치함을 나타내는 치환 문자
  EX) a?e : ace, are, age 등의 문자가 포함됨.
- " (큰따옴표) : `(역따옴표), ₩ 를 제외한 모든 특수문자를 일반문자로 처리
  EX) # echo "$HOME, $USER" → 환경변수 HOME과 USER의 값을 출력한다.
  (실행결과 예) /home/fedora, fedorauser
- ' (작은따옴표) : 모든 특수문자를 일반문자로 처리
  EX) # echo '$HOME, $USER' → $HOME, $USER 을 그대로 출력
- ` (역따옴표) : 역따옴표로 감싼 문자열을 명령어로 해석
  EX) # echo `pwd` → pwd 를 명령어로 해석하여 pwd 명령의 결과를 출력

4-2. 쉘 프로그래밍

프로그래밍에 쓰이는 명령어 정리

- **echo 명령어** : 인수로 지정된 문자열이나 환경변수를 출력
  - [-n] : 개행없이 출력
- **read 명령어** : 인수로 지정된 변수에 값을 입력 받음.
- **let / expr 명령어** : 수식 연산을 위한 명령어

특수변수 / 매개변수 확장 2015(2) 2016(2) 2017(1)

- $? : 마지막으로 실행된 프로세스의 상태값을 나타냄
- $0 : 현재 스크립트의 이름
- $1 : 첫번째 인수

- ${#변수} : 문자열의 길이
  EX) # echo ${#value} → value="333" 이라면 3이 출력
- ${변수:위치} : 위치부터 문자열 출력 (0부터 시작)
  EX) # echo ${value:3} → value="string" 이라면 ing 출력

여러가지 함수들 2014(1) 2015(1) 2016(1)

- [조건문] **if 문**

1. 형식 :
```
if [ 조건 ] then
    명령어
elif [ 조건 ] then
    명령어
else
    명령어
fi
```

2. 조건문 작성법
   - [A -lt B] : A가 B보다 작다
   - [A -eq B] : A와 B가 같다
   - [A -qt B] : A가 B보다 크다
   - [A -qe B] : A가 B보다 크거나 같다
   - [A -le B] : A가 B보다 작거나 같다
   - [A -ne B] : A가 B와 다르다

• [조건문] **case 문**
- 형식 : case 변수 in
패턴1) 명령어 ;;
패턴2) 명령어 ;;
...
*) 명령어 ;;
esac
- " ***)** " 은 아무 패턴과 일치하지 않을 때 수행되어지는 구간이다.

• [반복문] **for 문**
- 형식 : for 변수 in 값
do
명령어
done

• [반복문] **while 문**
- 형식 : while [조건]
do
명령어
done

• [반복문] **until 문**
- 형식 : until [조건]
do
명령어
done

4-3. 책에 없는 내장 명령어와 기타 내용

출력 명령어

• **cut 명령어** : 파일에서 필드를 뽑아내서 출력 2017(1)
- [-f] : 잘라낼 필드를 지정
- [-d] : 필드를 구분하는 문자를 지정
- [-c] : 잘라낼 곳의 글자 위치를 지정
EX) # cut -f 1,3,4 -d : /etc/passwd → /etc/passwd 파일에서 필드를 :로 구분하여 1,3,4번째 필드를 출력
cut -c 1-10 /etc/passwd → /etc/passwd 파일에서 첫번째 문자부터 10번째 문자 까지만 출력

• **more/less 명령어** : 내용이 많은 파일을 출력할 때 사용하는 명령어
- [-f] / [SpaceBar] : 한 페이지 뒤로 이동
- [-b] : 한 페이지 앞으로 이동

• **tail 명령어** : 파일의 내용을 뒷부분부터 출력
- [-n] : 지정한 줄만큼 출력

• **head 명령어** : 파일의 내용을 앞부분부터 출력

• **cat 명령어** : 파일의 내용을 화면에 출력 2015(2)
- [-n] : 행 번호를 붙여서 출력
- [-b] : 행 번호를 붙여서 출력하되, 비어있는 행은 제외
- [-s] : 비어있는 2개 이상의 빈 행은 하나의 행으로 출력
- [-v] : 탭 문자와 End 문자를 제외한 제어 문자를 '^'로 출력
- [-T] : 탭(tab) 문자('^')를 출력
- [-E] : (End) 행마다 끝에 '$' 문자를 출력

- [-A] : 모든 제어문자를 출력
- 옵션 사용 예제

```
[root@localhost linuxmaster]# cat -n cattest.txt
     1  abcd asde14
     2  aaa      bbb      ccc
     3
     4
     5
     6  ccc      ddd      eee
     7  dfeqwe-asdc
     8
     9
    10
[root@localhost linuxmaster]#
[root@localhost linuxmaster]# cat -nA cattest.txt
     1  abcd asde14$
     2  aaa^Ibbb^Iccc$
     3  $
     4  $
     5  $
     6  ccc^Iddd^Ieee$
     7  dfeqwe-asdc$
     8  $
     9  $
    10  $
[root@localhost linuxmaster]#
```

find 명령어 2015(1) 2015(2)

- 형식 : # find [검색 디렉터리] [옵션]
- 주요 옵션
  - [-name] : 파일 이름으로 검색
  - [-perm] : 권한(permission)으로 검색
    EX) # find / -perm -4000 → 루트 디렉터리 이하에 있는 권한이 최소 4000을 만족하는 파일을 검색
    EX) # find /etc -perm 777 → /etc 디렉터리 이하에 있는 권한이 777인 파일을 검색
  - [-user] : 사용자 이름으로 검색
  - [-type] : 파일 종류로 검색 (d/f/l/s/…)
  - [-exec] : 검색한 파일들에 대해 특정 명령을 수행
    EX) # find / -name "*.c" -exec rm -rf {} ₩; → 루트 디렉터리 이하에 있는 파일 이름이 .c로 끝나는 파일을 모두 삭제
  - [-ok] : exec와 같은 기능을 수행하지만 명령 실행할 때마다 실행 의사를 물어 봄.
  - [-mtime] : 파일이 마지막으로 수정된 날짜에 대해 검색을 수행
  - [-atime] : 파일에 마지막으로 접근한 날짜에 대해 검색을 수행
  - [-ctime] : 파일이 마지막으로 권한이 변경된 날짜에 대해 검색을 수행
    → +n : n일 이전 / -n : 오늘부터 n일 전 사이 / n : 정확히 n일 전
    EX) # find . -name "*.sh" -mtime +1 → 현재 디렉터리 이하에 있는 파일 이름이 .sh로 끝나는 파일 중 수정시간이 1일 이전인 파일을 검색 (오늘이 7월 26일이면 25일 이전에 마지막으로 수정된 파일을 검색)

grep 명령어 2015(2) 2017(2)

- 형식 : # grep [옵션] [정규표현식] [검색 대상 파일/디렉터리]
- 주요 옵션
  - [-n] : 행번호를 같이 출력
  - [-v] : 해당 패턴이 들어가지 않은 행을 출력
  - [-i] : 대소문자 구분 안함
  - [-e] : 정규표현식 사용을 명시

• 정규표현식 2018(1)
- ^ : 해당 문자로 시작
EX) # grep '^ap' test.txt → test.txt 파일 내에서 ap로 시작하는 행을 출력
--- apple, api 등이 해당
- $: 해당 문자로 끝
EX) # grep 'le$' test.txt → test.txt 파일 내에서 le로 끝나는 행을 출력
--- apple, badedale 등이 해당
- [] : 대괄호 안에 있는 문자 집합 중 하나와 일치
EX) # grep '^[A-Z]' test.txt → test.txt 파일 내에서 영어 대문자로 시작하는 행을 출력
--- Apple, Banana, Car 등이 해당
- . : 하나의 문자를 의미
EX) # grep 'A..le$' test.txt → test.txt 파일 내에서 A 다음에 두개의 문자가 나오고 le로 끝나는 문자열로 끝나는 행을 출력
--- abc Apple, def Alele등이 해당
- * : 선행하는 문자열이 0개 이상 일치
EX) # grep 'A..le.* end$' test.txt → test.txt 파일 내에서 A 다음에 두개의 문자가 나오고 le로 끝나는 문자열 다음에 임의의 문자가 0개 이상 존재하고 하나의 공백 후에 end로 끝나는 행을 출력
--- Apple Apple end, Apaleace end 등이 해당

history 명령어
• 콘솔에 입력하였던 명령어들의 히스토리를 출력
• [-c] : 기존 히스토리를 모두 삭제
EX) history 10 : 최근 10개의 히스토리를 출력
• 관련 명령
- !! : 바로 전에 사용한 명령을 다시 수행
- ![숫자] : 해당 history 번호로 명령을 다시 수행
- ![문자열] : 해당 문자열이 들어간 가장 최근 명령을 다시 수행
EX) # !find → find 가 들어간 가장 최근 명령을 다시 수행

date 명령어 2014(2)
• 형식 : # date [옵션] [포맷]
• 주요 포맷
- +%Y : 년도를 출력
- +%m : 월을 출력
- +%d : 일을 출력
- +%H : 시를 출력
- +%M : 분을 출력
EX) echo `date +%Y%m%d%H%M` → 현재 날짜가 2018년 7월 26일 22시 06분이면 201807262206 출력

5. 프로세스
5-1. 프로세스의 개념 및 종류
프로세스(process) 정의
• 프로세스 : 실행중인 프로그램
• 관련 용어
- 프로그램(program) : 특정 기능을 수행하기 위한 명령어의 조합
- 작업(job) : 프로그램과 프로그램 실행에 필요한 입력 데이터
- 프로세서(processor) : 연산을 수행하고 처리하기 위한 자원 (CPU)
- 프로시저(procedure) : 프로그램의 일부로 공통적으로 사용되는 특정 루틴
- 스레드(thread) : 프로세스의 일부 특정 데이터만 갖고 있는 가벼운 프로세스

- 좀비 프로세스(Zombie Process) 2017(1)
  - 프로그램 수행을 마치고 자원도 모두 반납한 상태지만 프로세스는 존재하는 상태
  - 자식 프로세스가 종료되었지만 부모 프로세스가 확인하지(wait()) 못하여 남아있는 상태
- 고아 프로세스(Orphan Process)
  - 자식 프로세스보다 부모 프로세스가 먼저 종료된 상태
  - 이 때 자식 프로세스의 부모 프로세스는 init 프로세스가 된다.

프로세스 제어 블록(Process Control Block, PCB)
- 커널에 등록된 각 프로세스들에 대한 정보를 저장하고 있는 영역
- 프로세스들은 커널 공간에 자신의 PCB를 하나씩 갖는다.
- PCB에 저장되는 정보 : 프로세스 고유 번호, 프로세스 우선 순위, 프로세스 현재 상태, 문맥 저장 영역 등

데몬 프로세스(Demon Process)
- 메모리에 상주하면서 요청이 들어올 때마다 명령을 수행하는 프로세스
- 백그라운드에서 작동하는 프로세스이다.

데몬 프로세스 작동 방식 2015(1), 2017(2)
- **standalone 방식**
  - 각 데몬 프로세스들이 독립적으로 수행되며, 항상 메모리에 상주하는 방식
  - 자주 실행되는 데몬 프로세스에 적용되는 방식이다.
  - /etc/rc.d/init.d 디렉터리에 위치
  - 웹 서비스 데몬(apached, httpd, mysqld 등), 메일 서비스 데몬(sendmail 등), NFS 등 서비스 요청이 많은 프로세스들이 standalone 방식으로 작동한다.
- **(x)inetd 방식 (슈퍼 데몬)** → 55 페이지 참고
  - (x)inted 이라는 슈퍼 데몬이 서비스 요청을 받아 해당 데몬을 실행시켜 요청을 처리하는 방식
  - 서비스 속도는 standalone 방식보다는 느리지만, (x)inted 데몬만 메모리에 상주해 있기 때문에 메모리를 많이 필요로 하지 않는다.
  - /etc/xinetd.d 디렉터리에 위치
  - telnetd, ftpd, pop3d, rsyncd 등의 서비스들이 슈퍼 데몬 방식으로 작동한다.

시스템 호출 fork(), exec() 2015(1)
- 프로세스가 다른 프로세스를 실행하기 위한 시스템 호출 방법
- **fork()** 2014(2)
  - 새로운 프로세스를 위한 메모리를 할당 받음
  - 자식 프로세스를 생성하는 시스템 호출 함수
  - 새롭게 생성된 프로세스는 똑같은 코드를 기반으로 실행됨 (복사)
- **exec()**
  - 원래의 프로세스를 새로운 프로세스로 대체
  - 새로운 프로세스를 위한 메모리를 할당하지 않고, exec()에 의해 호출된 프로세스만 메모리에 남음.

5-2. 프로세스 관리의 이해

프로세스 상태 2015(2)
- 생성 상태 : 프로세스가 처음 생성되는 상태
- 준비 상태 : 프로세스가 필요한 자원들(기억장치 포함)을 할당 받은 상태에서 프로세서를 할당 받기 위해 기다리고 있는 상태
- 실행 상태 : 프로세스가 실행되고 있는 상태
- 대기 상태 : 프로세스가 임의의 자원을 요청한 후 할당 받기 전에 할당 받을 때까지 기다리고 있는 상태
- 지연 상태 : 프로세스가 기억 장치를 할당 받지 못하고 있는 상태

시그널(Signal) 분류 2015(1) 2016(1) 2016(2) 2016(2) 2017(1) 2017(2)
- **1 / SIGHUP** : 연결 끊기. 프로세스의 설정파일을 다시 읽는데 사용된다.
- **2 / SIGINT** : 인터럽트 발생 [Ctrl + C]
- **9 / SIGKILL** : 강제 종료
- **15 / SIGTERM** : kill 명령어로 프로세스를 종료할 경우
- **19 / SIGTSTP** : 프로세스를 백그라운드로 전환하는 경우(정지) [Ctrl + Z]

CH.3 네트워크의 이해

1. 네트워크 기초

1-1. OSI 7계층

OSI 7계층 요약 2015(1) 2016(1) 2018(1)

계층		특징	데이터 종류	프로토콜/서비스
7	응용 (Application)	· 각종 응용서비스 제공	메시지 (Message)	FTP, SMTP, HTTP등
6	표현 (Presentation)	· 압축, 암호화		ASCII, jpg
5	세션 (Session)	· 동기화, 세션 연결/관리/종료		전송모드 결정
4	전송 (Transport)	· 데이터 전송보장 · 흐름제어	세그먼트 (Segment)	TCP, UDP
3	네트워크 (Network)	· 라우팅	패킷 (Packet)	IP, ICMP, IGMP, ARP
2	데이터 링크 (Data Link)	· 오류제어 · MAC	프레임 (Frame)	PPP
1	물리 (Physical)	· 물리적 연결 성정	비트 스트림 (Bit Stream)	

1-2. 네트워크 장비

허브(Hub)
- 물리 계층에서 동작하는 장비
- **더미 허브(Dummy Hub)**
  - 일반적인 허브.
  - 허브로 들어온 데이터를 모든 포트로 뿌려주는 역할
  - 충돌이 발생할 확률이 높고, 연결된 노드가 많을수록 속도가 저하됨
- **스위칭 허브(Switching Hub)**
  - 허브에 스위칭 개념을 도입한 장비
  - 허브로 들어온 데이터를 해당 목적지에 해당하는 포트로 전송함
  - MAC 주소를 이용하기 때문에 2계층 장비로 분류되고 충돌 도메인을 나누는 역할을 함.
- **리피터(Repeater)** 2017(1)
  - 약해진 신호를 수신하여 원래의 형태로 재생하고 증폭하는 장비
  - 최근에는 모든 네트워크 장비에 기본적으로 들어가 있는 기능이다.

브리지(Bridge)
- 데이터링크 계층인 MAC에서 동작하는 장비
- 네트워크를 확장시키고 통신을 격리시키기 위해 사용
- 충돌 도메인(Collision Domain)을 나누는 역할을 함

라우터(Router)
- 3계층 장비로 물리 / 데이터 링크 / 네트워크 계층에서 동작
- 라우터는 각 인터페이스마다 MAC 주소와 IP주소를 갖는다.
- 브로드캐스트 도메인(Broadcast Domain)을 나누는 역할을 함

게이트웨이(Gateway) 2017(1)
- 서로 다른 프로토콜을 사용하는 네트워크를 상호 접속하기 위한 장비
- 4계층 장비로 분류된다.

1-3. 이더넷/LAN의 기본 이해

토폴로지(topology)에 따른 분류 2018(1)

- **그물(mesh)형** 토폴로지
  - 모든 장치들끼리 점-대-점 링크를 갖는다. → 많은 링크가 필요
  - 통신량 문제 해결, 안전성/보안성 ↑
- **스타형(성형)** 토폴로지
  - 각 장치들은 허브라고 불리는 중앙제어장치와 점-대-점 링크를 갖는다.
  - 그물형 토폴로지보다 비용 ↓, 설치/재구성 용이
- **버스형** 토폴로지
  - 하나의 긴 케이블이 네트워크 상의 모든 장치를 연결하는 형태 → 적은 양의 링크 수
  - 설치가 용이하지만 재구성/결합 분리는 어려움
- **링형** 토폴로지
  - 자신의 양쪽에 있는 장치와 점-대-점 링크를 갖고, 각 장치는 중계기를 포함
  - 비교적 설치/재구성이 용이하지만 장치 추가가 어렵고 단방향 전송의 단점을 갖는다.

규모에 따른 분류

- **LAN**(Local Area Network)
  - 근거리 통신망. 단일 건물 같은 소규모 지역을 묶는 네트워크
- **MAN**(Metropolitan Area network)
  - LAN보다 크고 WAN보다는 작은 규모의 네트워크
  - 하나의 도시 정도를 묶는 네트워크
- **WAN**(Wide Area Network)
  - 광역 통신망
  - 하나의 국가 정도를 묶는 네트워크

1-4. TCP/IP 및 네트워크 프로토콜의 이해

프로토콜(Protocol)과 포트(Port) 2016(2)

- 프로토콜 2014(1)
  - 컴퓨터 간의 통신을 위한 규격
  - /etc/protocols 에서 주요 프로토콜 번호를 확인할 수 있다.
- 포트 번호 2014(2) 2017(2)
  - 포트 : TCP와 UDP에서 어플리케이션의 상호 통신을 사용
  - OSI 7계층의 4계층(전송계층)에서 사용되는 논리적인 주소
  - 0~65535번의 범위(2^16) 중에서 0~1023번 까지는 특정 프로토콜이 지정되어 있으며, 잘 알려진 포트(well-known port)라고 한다.
  - /etc/services 에서 주요 포트 번호를 확인할 수 있다.

• 주요 포트 번호 2016(2)

FTP	TCP/20 TCP/21	데이터 포트(20), 제어 포트(21)
SSH	TCP/22	암호화된 원격 터미널 접속 프로토콜
SFTP	TCP/22	SSH를 이용한 암호화된 FTP 프로토콜
TELNET	TCP/23	암호화를 하지 않는 원격 터미널 접속 프로토콜
SMTP	TCP/25	메일 전송 프로토콜
HTTP	TCP/80	Hyper-Text Transfer Protocol
POP3	TCP/110	메일 수신용 프로토콜
IMAP4	TCP/143	메일 수신용 프로토콜
SMB	TCP/445	서버 메시지 블록 프로토콜
DNS	UDP/53 TCP/53	도메인에 대한 호스트 정보 제공하는 프로토콜
DHCP	UDP/67 UDP/68	동적 호스트 네트워크 설정 프로토콜 서버(67) / 클라이언트(68)
TFTP	UDP/69	단순 파일 송수신 프로토콜
NetBIOS	TCP/135~139 UDP/135~139	별개의 컴퓨터상에 있는 어플리케이션들이 LAN내에서 통신할 수 있게 해주는 프로그램
SNMP	UDP/161 UDP/162	네트워크 관리 프로토콜 서버(161) / 클라이언트(162)

IP(Internet Protocol)

• IP 주소 : OSI 7계층의 3계층(네트워크 계층)에서 사용되는 논리적인 주소

• 32bits의 길이를 가지며, 네트워크 ID와 호스트 ID로 구성되어 있다.

• 클래스(class)

	8bits	8bits	8bits	8bits	
Class A	Network	Host			큰 규모의 네트워크에서 사용
Class B	Network		Host		중간 규모의 네트워크에서 사용
Class C	Network			Host	작은 규모의 네트워크에서 사용
Class D	Multicast Address				멀티캐스트용으로 사용되는 주소
Class E	Reserved Address				추후 사용을 위해 예약된 주소

• 예약된 주소 2017(1)

- **0.0.0.0/32** : 현재 네트워크를 뜻하는 주소로, 자신의 IP 주소를 모를 때 사용
- **10.0.0.0/8** : A 클래스의 사설 주소
- **127.0.0.0/8** : 루프백(loopback) 주소
- **172.16.0.0/12** : B 클래스의 사설 주소
- **192.168.0.0/16** : C 클래스의 사설 주소
- **255.255.255.255/32** : 브로드캐스트 주소. 같은 네트워크 상의 모든 장치에게 패킷을 전송할 때 사용

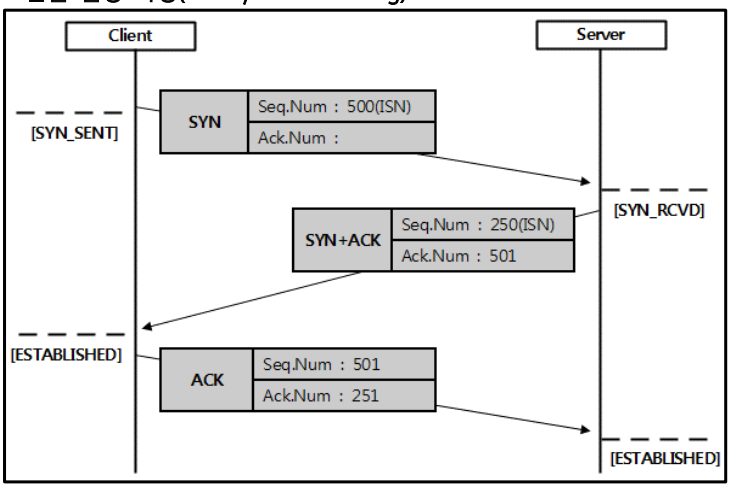
• **서브네팅(Subnetting)** 2014(1) 2015(1) 2016(1) 2016(2) 2017(2) 2018(1)
- IPv4 주소의 고갈로 인해 하나의 네트워크를 여러 개의 서브 네트워크로 나누어 낭비를 막기위한 방법
- 서브넷 마스크(Subnet mask) : IP 주소를 네트워크 주소와 호스트 주소로 구분하기 위한 주소

• **IPv6** 2018(1)
- IPv4 주소의 고갈로 더 큰 주소 길이를 갖는 IPv6 주소 체계가 등장
- 크게 확장된 주소 공간, 향상된 서비스, 보안성 ↑
- IPv4 vs IPv6

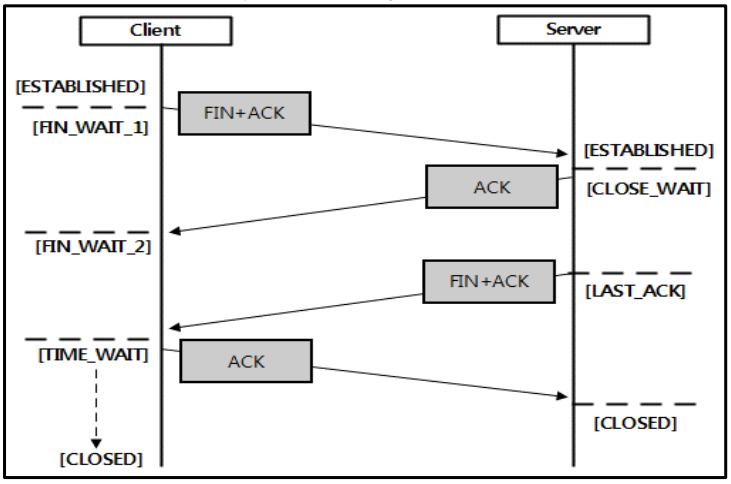
구분	IPv4	IPv6
주소 길이	32bits (4개의 필드)	128bits (8개의 필드)
주소할당 방식	클래스 단위	네트워크 규모
브로드캐스트 주소	있음	없음
헤더 크기	가변	고정
QoS 제공	미흡	제공
보안	IPSec 프로토콜 별도로 설치	IPSec 자체 지원
Plug&Play	불가(DHCP 이용 가능)	가능

ARP 프로토콜 2016(1) 2017(2)
- IP 주소를 이용해 해당 MAC 주소를 요청하는 프로토콜
- 보통은 3계층 프로토콜로 분류된다. (2계층으로도 분류되는 경우도 있음)
- RARP : MAC 주소를 이용해 IP 주소를 요청하는 프로토콜

TCP(Transmission Control Protocol)
• 신뢰적이고 연결지향적인 프로토콜
• 전이중(full-duplex) 통신 : 동시에 양방향 전달 가능
• **연결 설정 과정(3-way Handshaking)**



• **연결 종료 과정(4-way handshaking)** 2014(2)



2. 네트워크 설정

2-1. 환경 설정

IP 주소 설정

• LAN 카드를 OS에 인식시킨 후 IP를 설정해야 한다.

• **ifconfig 명령어** 2014(1) 2014(2) 2015(2)

- 형식 : # ifconfig [인터페이스 명] [옵션]
- [-a] : 시스템의 전체 인터페이스에 대한 정보를 출력
- 옵션 없이 사용 시 현재 설치된 네트워크 인터페이스의 설정 내용을 출력
- IP 주소, 서브넷 마스크, 브로드캐스트 주소, MAC 주소, MTU, RX/TX 패킷 등의 정보를 알 수 있다.

EX) IP 설정 예제

ifconfig eth0 192.168.1.33 netmask 255.255.255.0 → eth0 인터페이스에 IP, 넷마스크 설정
ifconfig eth0 up → eth0 인터페이스 활성화

• **ip 명령어** 2015(1) 2018(1)

- [addr] : 네트워크 인터페이스에 대한 정보를 **출력(show) / 추가(add) / 삭제(del)**

EX) # ip addr show → 인터페이스 정보 출력
ip addr add 192.168.1.33/24 dev eth0 → eth0 인터페이스에 IP 설정
ip addr del 192.168.1.33/24 dev eth0 → eth0 인터페이스에 설정된 IP 삭제

- [route] : 라우팅 테이블 출력(show) / 정적 라우팅 추가(add) / 삭제(del)

EX) # ip route show → 라우팅 테이블 출력
ip route add 10.90.21.1/24 via 192.168.1.33 dev eth 0 → eth0 인터페이스에 대한 정적 라우팅 추가
ip route del 10.90.21.1/24 → 정적 라우팅 삭제
ip route add default gateway 192.168.1.1 → 디폴트 게이트웨이 추가

- [link] : 네트워크 인터페이스 상태 출력(show) / 관리(set)

EX) # ip link show → 인터페이스 상태 출력
ip link set eth0 up → eth0 인터페이스 활성화

2-2. 네트워크 관련 명령어

netstat 명령어 2014(1) 2014(2) 2015(2)

- 네트워크 상태 정보를 출력하는 명령어
- 주요 옵션
  - [-a] : 모든 소켓 정보 출력
  - [-r] : 라우팅 정보 출력
  - [-n] : 호스트 이름을 IP 주소로 출력
  - [-l] : LISTEN 상태인 소켓 정보만 출력
  - [-p] : 해당 소켓과 관련된 프로세스의 이름과 PID 출력
  - [-t] : TCP 프로토콜만 출력
  - [-u] : UDP 프로토콜만 출력

네트워크 상태 진단 명령어

- **traceroute 명령어** : 목적지까지 패킷이 지나가는 경로를 출력 2017(2)
- **ping 명령어** : 목적지에 ICMP 패킷(echo_request)을 보내 상태를 점검

DNS 관련 명령어

- **nslookup 명령어** : DNS 서버와 질의/응답하는 명령어
- **dig 명령어** : nslookup 명령어와 비슷하지만 더 간결하고 사용하기 편리
- 관련 파일 2014(2)
  - **/etc/resolv.conf 파일** : 네임서버가 정의되어 있는 파일 2014(1) 2016(1) 2016(2) 2017(1) 2017(2)
  - **/etc/hosts 파일** : 도메인/호스트명과 IP 주소 매핑 정보를 저장하고 있는 파일

route 명령어

- 라우팅 테이블을 출력/수정 하는 명령어
- 주요 옵션
  - [add] : 라우팅 경로나 디폴트 게이트웨이를 추가
  - [del] : 라우팅 경로나 디폴트 게이트웨이를 삭제
  - [-n] : 호스트 이름을 IP 주소로 출력
- 옵션 없이 사용시 라우팅 테이블을 출력

nmcli 명령어

- 네트워크 관리자(Network Manager) 서비스를 커맨드라인에서 수행하는 명령어
- 주요 옵션
  - [con show] : 네트워크의 모든 연결에 대한 정보를 출력
  - [con add] : 네트워크 연결 설정 추가
    EX) # nmcli con add con-name ens33 ifname test-net type eth ip4 192.168.1.33/24 gw4 192.168.1.1
  - [con del] : 네트워크 연결 설정 제거
  - [con up] : 네트워크 연결 활성화
  - [con mod] : 네트워크 연결 설정 수정

2과목 리눅스 시스템 관리

CH.4 일반 운영 관리

1. 사용자 관리

1-1. root 사용자 관리

 # su 명령어 2016(1) 2017(1) 2018(1)

- 다른 사용자로 전환하는 명령어
- 주요 옵션
  - [-] : 전환하는 사용자의 초기화 파일을 실행
    EX) # su - root → root의 환경변수를 적용하고 전환
  - [-c] : 계정 변환 없이 특정 명령어만 실행 (sudo 명령어와 같은 기능)
    EX) # su -c 'cat /etc/passwd' - root → root 권한으로 해당 명령을 실행
- su 명령어로 사용자 전환 후의 위치는 동일

root 계정 관리 2014(2) 2015(2) 2016(1) 2016(2) 2018(1)

- root 계정의 UID 값은 0이다.
- root 이외에 UID가 0인 사용자가 없도록 해야 한다.
- TMOUT 환경 변수를 사용해 일정시간 미사용시 자동으로 로그아웃 되도록 설정하여 보안을 강화한다.
- 사용자 인증 모듈인 PAM을 이용해 root 계정으로의 직접 로그인을 차단한다.
- 일반 사용자에게 특정 명령어에 대한 root 권한이 필요할 때는 su 명령어보다는 sudo 명령어를 이용하도록 한다.

1-2. 사용자 계정 관리

계정 관리 관련 파일 2017(1)

- **/etc/passwd 파일** 2014(1) 2014(2) 2015(1)
  - 파일 내용 구성

    [user_account] : [user_password] : [UID] : [GID] : [comment] : [home_directory] : [login_shell]

    1) user_account : 사용자 계정
    2) user_password : /etc/shadow 파일에 암호화되어 저장되어 있다.
    3) UID : User ID. 보통 100번 이하는 시스템이 사용, 0번은 시스템 관리자를 나타낸다.
    4) GID : Group ID
    5) comment
    6) home_directory : 로그인 성공 후에 사용자가 위치할 홈 디렉터리의 절대경로
    7) login_shell : 로그인 셸의 절대경로

- **/etc/shadow 파일** 2015(1) 2016(1)
  - 파일 내용 구성

    [user_id] : [encryption_pw] : [last_change] : [minlife] : [maxlife] : [warn] : [inactive] : [expires]

    1) user_id : 사용자 계정
    2) encryption_pw : 일방향 해시 알고리즘을 이용해 암호화한 패스워드
       - 형식 : $ id $ salt $ encrypted_password
       - id : 적용된 일방향 해시 알고리즘 (1 : MD5 / 5 : SHA-256 / 6 : SHA-512 등)
    3) last_change : 마지막으로 패스워드를 변경한 날(1970.01.01.부터 지난 일수로 표시)
    4) minlife : 최소 패스워드 변경 일수(패스워드를 변경할 수 없는 기간)
    5) maxlife : 최대 패스워드 변경 일수(패스워드 변경 없이 사용할 수 있는 일수)
    6) warn : 경고 일수(maxlife 필드에 지정한 일수가 얼마 남지 않았음을 알림)
    7) inactive : 최대 비활성 일수
    8) expires : 계정이 만료되는 날(1970.01.01.부터 지난 일수로 표시)

- **encryption_pw 필드**의 기호 뜻

기호	설명
*	패스워드 잠긴 상태. 별도의 인증방식을 사용하여 로그인
!!	패스워드 잠긴 상태. 모든 로그인이 불가능.
빈값	패스워드가 설정되지 않은 상태

• **/etc/login.def 파일** 2017(2)
- 사용자 계정의 설정과 관련된 기본 값을 정의한 파일
- 기본 메일 디렉터리, 패스워드 에이징, 사용자 계정의 UID/GID 값 범위 등의 기본값을 설정할 수 있다.

• **/etc/skel 디렉터리** 2015(2)
- 사용자 계정 생성 시 공통으로 배포할 파일이나 디렉터리를 저장하는 디렉터리

• **/etc/default/useradd 파일** 2014(2)
- useradd 명령어로 계정 생성 시 기본 값을 지정한 파일

1-3. 그룹 계정 관리

그룹 관리 관련 파일

• **/etc/group 파일** 2018(1)
- 파일 내용 구성 : **그룹 명 : 그룹 패스워드 : GID : 그룹 멤버**
- /etc/passwd 파일에는 기본 그룹의 GID가 저장되고, /etc/group 파일에는 **2차 그룹의 정보**가 저장

• **/etc/gshadow 파일**
- 파일 내용 구성 : **그룹 명 : 암호화 된 그룹 패스워드 : 관리자 : 그룹 멤버**

1-4. 관련 명령어

useradd 명령어 2014(1) 2015(1) 2016(2) 2017(1)

• 사용자 계정을 생성하는 명령어

• 주요 옵션
- [-u] : UID 지정
- [-o] : UID의 중복을 허용
- [-g] : 기본 그룹의 GID 지정
- [-G] : 2차 그룹의 GID 지정
- [-d] : 홈 디렉터리 지정
- [-s] : 기본 쉘을 지정
- [-c] : 코멘트 지정
- [-D] : 기본 값을 설정하거나 출력 (/etc/default/useradd 파일 내용)
- [-e] : 유효 기간 지정 (/etc/shadow 파일의 expires 항목)
- [-f] : 비활성 일수 지정 (/etc/shadow 파일의 inactive 항목)

EX) # useradd -u 500 -G 10 -s /bin/bash -e 2018-12-31 testuser → testuser 사용자의 UID는 500, GID가 10인 2차그룹, 기본 쉘은 bash, 유효기간은 2018-12-31 까지로 설정하여 생성

• 옵션 없이 계정 생성할 경우 패스워드를 설정하지 않았기 때문에 /etc/shadow 파일에 패스워드 항목이 !!로 지정되어 있다. (패스워드가 잠겨 있다는 기호)

usermod 명령어 2014(1) 2014(2) 2015(2) 2016(1) 2017(2)

• 사용자 계정의 정보를 변경하는 명령어

• 주요 옵션
- useradd 명령어와 옵션이 동일
- [-l] : 계정 이름 변경

EX) # useradd -l usertest testuser → testuser 사용자의 이름을 usertest로 변경

userdel 명령어 2014(1) 2014(2) 2015(1)
- 사용자 계정을 삭제하는 명령어
- [-r] 옵션 : 홈 디렉터리 제거

chage 명령어 2016(1) 2016(2)
- 패스워드 에이징에 관한 설정을 하는 명령어
- 주요 옵션
  - [-m] : /etc/shadow 파일의 minlife 항목 설정 (passwd -n)
  - [-M] : /etc/shadow 파일의 maxlife 항목 설정 (passwd -x)
  - [-W] : /etc/shadow 파일의 warn 항목 설정 (passwd -w)
  - [-I] : /etc/shadow 파일의 inactive 항목 설정 (usermod -f)
  - [-E] : /etc/shadow 파일의 expires 항목 설정 (usermod -e)

passwd 명령어
- 사용자 계정의 패스워드 변경/관리하는 명령어
- 주요 옵션
  - [-n] : /etc/shadow 파일의 minlife 항목 설정
  - [-x] : /etc/shadow 파일의 maxlife 항목 설정
  - [-w] : /etc/shadow 파일의 warn 항목 설정
  - [-l] : 계정 잠금(lock)
  - [-u] : 계정 잠금 해제(unlock)

groupadd 명령어
- 그룹을 생성하는 명령어
- 주요 옵션
  - [-g] : GID 지정
  - [-o] : GID의 중복을 허용

groupmod 명령어 2014(1) 2015(1) 2017(2) 2018(1)
- 그룹의 정보를 변경하는 명령어
- 주요 옵션
  - groupadd 명령어와 옵션이 동일
  - [-n] : 그룹 이름 변경

groupdel 명령어 2016(2)
- 그룹을 삭제하는 명령어

gpasswd 명령어 2015(2)
- 그룹의 패스워드를 변경하거나 그룹에 계정을 추가/삭제하는 명령어
- 주요 옵션
  - [-a] : 사용자 계정을 그룹에 추가
    EX) # gpasswd -a testuser testgroup → testgroup 그룹에 testuser 사용자를 추가
  - [-d] : 사용자 계정을 그룹에서 삭제
  - [-r] : 그룹 암호를 삭제
  - [-A] : 그룹의 관리자를 지정
    EX) # gpasswd -A testuser testgroup → testgroup 그룹의 관리자를 testuser 사용자를 관리자로 설정

newgrp 명령어
- 계정의 소속 그룹을 변경하는 명령어
  EX) # newgrp grp01 → grp01로 그룹을 변경

사용자 정보 확인 명령어 2016(2) 2017(1)

(예제는 fedora, fedora2 사용자가 로그인 중인 상태에서 fedora 사용자가 명령 수행)

- **who 명령어** : 현재 시스템을 사용하는 사용자의 정보를 출력하는 명령어
  - [-q] : 사용자의 이름만 출력
  - [-H] : 출력 항목의 제목도 함께 출력 (Header)
  - [-b] : 마지막으로 재시작한 날짜와 시간을 출력 (boot)
  - [-r] : 현재 런레벨을 출력 (runlevel)

```
[fedora@localhost ~]$ who
fedora   tty2         2018-08-01 17:29 (/dev/tty2)
fedora2  tty3         2018-08-01 17:41 (/dev/tty3)
[fedora@localhost ~]$
```

- **w 명령어** 2015(2)
  - 현재 시스템을 사용하는 사용자의 정보와 작업 정보를 출력하는 명령어

```
[fedora@localhost ~]$ w
 17:44:18 up 15 min,  2 users,  load average: 1.85, 1.72, 0.99
USER     TTY        LOGIN@   IDLE   JCPU   PCPU WHAT
fedora   tty2      17:29    15:31  26.15s  0.05s /usr/bin/seapplet
fedora2  tty3      17:41    15:31  31.73s  0.05s /usr/bin/seapplet
[fedora@localhost ~]$
```

- **whoami 명령어**
  - 현재 작업하고 있는 자신의 계정을 출력

```
[fedora@localhost ~]$ whoami
fedora
[fedora@localhost ~]$
```

- **id 명령어** 2017(1)
  - 현재 작업하고 있는 자신의 계정명, 그룹명, UID, GID를 출력하는 명령어
- **users 명령어** 2018(1)
  - 현재 로그인 되어 있는 사용자의 계정을 출력

```
[fedora@localhost ~]$ users
fedora fedora2
[fedora@localhost ~]$
```

- **groups 명령어**
  - 사용자 계정이 속한 그룹을 출력
- **lslogins 명령어** 2017(2)
  - 시스템 내에 있는 사용자 계정에 대한 정보를 출력

```
[root@localhost fedora]# lslogins | tail -10
 994 openvpn                0        0        1              OpenVPN
 995 chrony                 1        0        1
 996 gluster                0        0        1              GlusterFS daemons
 997 polkitd                1        0        1              User for polkitd
 998 systemd-timesync       0        0        1              systemd Time Synchronizati
on
 999 systemd-coredump       0        0        1              systemd Core Dumper
1000 fedora                68        0        0        17:29 fedora
1001 fedora2               69        0        0        17:41
2010 test33                 0        0        1              test03 user
2011 test11                 0        0        1
[root@localhost fedora]#
```

무결성 검사 명령어

- **pwck 명령어** 2016(1) 2017(2)
  - /etc/passwd 파일과 /etc/shadow 파일 내용의 무결성을 검사하는 명령어

```
[root@localhost fedora]# pwck
사용자 'pulse': '/var/run/pulse' 디렉터리가 없습니다
사용자 'pipewire': '/var/run/pipewire' 디렉터리가 없습니다
사용자 'rpc': '/var/lib/rpcbind' 디렉터리가 없습니다
사용자 'saslauth': '/run/saslauthd' 디렉터리가 없습니다
사용자 'gnome-initial-setup': '/run/gnome-initial-setup/' 디렉터리가 없습니다
pwck: 바뀐 점이 없음
[root@localhost fedora]#
```

- **grpck 명령어** 2018(1)
  - /etc/group 파일과 /etc/gshadow 파일 내용의 무결성을 검사하는 명령어

2. 파일 시스템 관리

2-1. 파일 및 디렉터리 관리

ls -l 명령어 수행 시 각 필드 항목 2014(1)

파일 종류	파일 권한	하드링크 수	소유자	소유 그룹	크기	마지막 수정 시간	파일 명
d	rwxr-xr-x	3	fedora	fedora	4096	7월 26 21:41	linuxmaster

파일의 종류

- "ls -l" 명령어 사용 시 첫번째 필드의 첫번째 비트
- 일반 파일 : "-" 로 표기
- 디렉터리(directory) : "d" 로 표기
- 심벌릭링크 파일 : "l" 로 표기
  - **ln 명령어** : 링크 파일 생성 명령어 2015(2) 2017(2)
  - 옵션 없이 수행 시 하드링크 파일 생성, [-s] 옵션 사용 시 심벌릭링크 파일 생성
    - 하드링크 : inode number가 같음. 복사 개념
    - 심벌릭링크 : inode number가 다름. 바로가기 개념

    EX) # ln -s file.txt file.txt.sl → file.txt의 심벌릭링크 파일로 file.txt.sl 을 생성
- 블록 장치 파일(/dev/hda, /dev/fd0 등) : "b" 로 표기
- 문자 디바이스 파일(입출력 장치) : "c" 로 표기

파일의 권한(Permission)

- "ls -l" 명령어 사용 시의 첫번째 필드에서 첫번째 비트를 제외한 9개 비트
- 소유자(owner) / 그룹(group) / 기타 사용자(others) 순으로 표기

  EX) # ls -l

  drwxr-xr-x … directory → directory 파일은 디렉터리이며 소유자에 대해 모든 권한을 부여하며, 그룹/기타 사용자에게는 읽기와 실행 권한만 부여

  -rw-r—r-- … file.txt → file.txt 파일은 일반 파일이며 소유자에 대해 읽기와 쓰기 권한을 부여하며, 그룹/기타 사용자에게는 읽기 권한만 부여
- 특수 접근 권한 2015(2) 2016(1) 2016(2)
  - SetUID : 이 권한이 있는 파일을 실행하는 동안에는 파일 소유자의 권한으로 실행되도록 함.

    EX) **-rwsr-xr-x 1 root root … /bin/passwd** → passwd 명령어를 실행하는 동안은 root 권한으로 동작
  - SetGID : 이 권한이 있는 파일을 실행하는 동안에는 파일 소유 그룹의 권한으로 실행되도록 함.
  - Sticky bits : 이 권한이 있는 디렉터리에는 누구나 파일을 생성할 수 있다. (삭제는 소유자와 root만 가능)

    EX) **drwxrwxrwt 16 root root … /tmp** → tmp 디렉터리에는 누구나 파일을 생성할 수 있음.

• **umask 명령어** 2017(2) 2018(1)
- 기본 접근 권한을 출력/변경하는 명령어
EX) # umask 022 → 그룹, 기타 사용자에게는 쓰기 권한을 부여하지 않도록 기본 접근 권한 설정
- 일반 파일의 최대 접근 권한은 666이고, 디렉터리의 최대 접근 권한은 777이다.
- umask가 022일 때 파일을 생성한 직후의 권한은 644이고, 디렉터리를 생성한 직후의 권한은 755이다.

권한/소유권 변경 관련 명령어

• **chown 명령어** 2016(2)
- 파일과 디렉터리의 소유자와 소유 그룹을 변경하는 명령어
- [-R] : 서브 디렉터리의 소유자와 소유 그룹도 같이 변경
EX) # chown fedora2 /linuxmaster/fedora.txt → fedora.txt 파일의 소유자를 fedora2로 변경
chown fedora2:grp01 /linuxmaster/fedora.txt → fedora.txt 파일의 소유자를 fedora2로, 소유 그룹을 grp01로 변경

• **chgrp 명령어** : 파일과 디렉터리의 소유 그룹을 변경하는 명령어

• **chmod 명령어** 2015(1) 2017(1)
- 파일과 디렉터리의 접근권한을 변경하는 명령어
- [-R] : 서브 디렉터리의 접근권한도 모두 변경
EX) # chmod 644 perm.tmp → perm.tmp 파일의 접근권한을 644(rw-r—r--)로 변경
chmod go-rwx perm.tmp → perm.tmp 파일의 접근권한에서 그룹과 기타 사용자의 권한을 모두 제거

파일 속성 관련 명령어

• 파일 속성 기호
- a : 해당 파일에 추가만 가능
- c : 자동으로 압축된 상태로 저장
- d : dump로 백업이 되지 않음
- i : 해당 파일에 대해 변경/삭제 등을 할 수 없음
- u : 해당 파일이 삭제될 경우 그 내용이 저장이 되며, 데이터 복구가 가능

• **lsattr 명령어**
- 디렉터리/파일의 속성을 출력하는 명령어

• **chattr 명령어** 2016(1)
- 디렉터리/파일의 속성을 변경하는 명령어
EX) # chattr +i attr.tmp → attr.tmp 파일에 i 속성을 추가

2-2. 파일 시스템 복구

/etc/fstab 파일 2014(1) 2014(1) 2015(2) 2017(1)

• 파일 시스템의 마운트 설정 정보를 가지고 있는 파일

• 파일 구조 2014(2) 2015(1) 2015(2) 2016(1)

장치 이름	마운트 포인트	파일 시스템 종류	옵션	덤프 설정	파일 점검 옵션
/dev/mapper/fedora-root	/	ext4	defaults	1	1

- 장치 이름에는 장치 파일명 외에도 UUID나 라벨명으로 설정 가능

• 옵션 필드 종류
- defaults : 기본 값. rw, nouser, auto, exec, suid 속성을 포함
- auto : 부팅 시 자동으로 마운트 ↔ noauto
- exec : 실행 파일이 실행되는 것을 허용 ↔ noexec
- suid : SUID/SGID의 사용 허용 ↔ nosuid
- ro : 읽기 전용 ↔ rw : 읽기/쓰기 허용
- user : 일반 사용자도 마운트 가능 ↔ nouser : 일반 사용자의 마운트 불가능(root만 가능)
- usrquota : 사용자별로 디스크 쿼터 설정 허용
- grpquota : 그룹별로 디스크 쿼터 설정 허용

- 덤프 설정
  - 1 : dump 등의 데이터 백업 명령어 사용으로 파일 시스템의 내용 덤프 가능
  - 0 : 파일 시스템의 내용을 덤프 불가능
- 파일 점검 옵션
  - 0 : 부팅 시 fsck 명령어로 파일 시스템 점검 미수행
  - 1 : 루트 파일 시스템에 설정. 부팅 시 파일 시스템 점검 수행
  - 2 : 루트 파일 시스템 이외의 파일 시스템 점검 수행

mount 명령어 2014(1) 2016(1) 2016(2)

- 파일 시스템을 마운트하는 명령어
- 형식 : # mount [옵션] [장치 이름] [마운트 포인트]
- 주요 옵션
  - [-t] : 파일 시스템 종류를 지정
    EX) # mount -t iso9660 /dev/cdrom /mnt/cdrom → CD-ROM을 /mnt/cdrom 디렉터리에 마운트
  - [-f] : 마운트가 가능한지 확인만 수행
  - [-o] : 마운트 옵션을 지정
    EX) # mount -t ext4 -o ro,user /dev/sdb /mnt → /dev/sdb 장치를 /mnt에 마운트.
    읽기전용, 일반 사용자도 마운트할 수 있도록 허용
    # mount -t iso9660 -o loop /linuxmaster/temp.iso /mnt/iso → temp.iso ISO 이미지를 /mnt/iso에 마운트
- 언마운트는 **umount 명령어**를 이용
- CD-ROM 장치 언마운트 후 CD를 꺼낼 때 **eject 명령어**를 이용 2018(1)

fdisk 명령어 2016(2) 2017(1)

- 디스크의 파티션 생성/삭제, 정보 출력 등 파티션을 관리하는 명령어
- 주요 옵션
  - [-l] : 파티션 테이블 출력
- **주요 내부 명령어** 2014(1) 2014(2) 2016(1) 2018(1)
  - d : 파티션을 삭제
  - l : 사용 가능한 파티션 시스템 종류를 출력
    * 주요 시스템 종류 : **82(linux swap), 83(linux), 8e(linux LVM)** 2015(1) 2017(2)
  - n : 새로운 파티션 생성
  - p : 파티션 테이블 출력
  - t : 파티션의 시스템 종류를 변경
  - w : 파티션 정보를 디스크에 저장하고 종료

파일 시스템 생성(포맷) 명령어

- **mkfs 명령어**
  - [-t] : 파일 시스템 종류 지정 (default : ext2)
  EX) mkfs -t ext4 /dev/sdb1 → /dev/sdb1 파일 시스템을 ext4로 생성
  - **mkfs.ext2 / mkfs.ext3 / mkfs.ext4 명령어**도 제공
- **mke2fs 명령어** 2014(1) 2016(1)
  - [-t] : 파일 시스템 종류 지정 (default : ext2)
  - [-c] : 배드블록을 체크
  - [-b] : 블록크기를 바이트 수로 지정
  - [-j] : ext3 형식으로 생성 (journaling)
  - 기본 설정 값은 **/etc/mke2fs.conf 파일**에서 정의

LVM(Logical Volume Manager) 2014(2)

- 독립적으로 구성된 디스크 파티션을 하나로 연결하여 한 파티션처럼 사용할 수 있도록 해주는 관리도구
- 관련 용어 2018(1)
  - PV(Physical Volume) : 실제 하드디스크의 파티션을 의미
  - VG(Volume Group) : 여러 개의 PV를 그룹으로 묶은 것을 의미
  - LV(Logical Volume) : VG를 다시 적절한 크기로 나눈 파티션을 의미
  - PE(Physical Extent) : PV가 가진 일정한 블록을 의미
  - LE(Logical Extent) : LV가 가진 일정한 블록을 의미
- 주요 명령어 (LVM 생성 시 사용 순)
  1) 해당 파티션의 종류 변경
     - fdisk → t → 기존 83(linux) 를 8e(linux LVM)로 변경
  2) PV 생성
     - **pvcreate 명령어**
     - **pvscan 명령어** : PV 상태를 확인
  3) VG 생성 : **vgcreate 명령어**
  4) VG 활성화
     - **vgchange -a y** : 활성화
     - **vgchange -a n** : 비활성화
     - **vgdisplay -v** : VG 상태 확인
  5) LV 생성
     - **lvcreate 명령어**
     - **lvscan 명령어** : LV 상태 확인
  6) LV에 파일 시스템 생성 : mkfs / mke2fs 명령어 사용
  7) LV 마운트 : mount 명령어 사용

파일 시스템 점검/복구 관련 명령어 2014(1) 2015(1)

- **fsck 명령어**
  - 파일 시스템을 검사/복구하는 명령어
  - [-f] : 강제 검사
  - [-b] : 지정한 백업 슈퍼블록을 사용하여 복구
  - **fsck.ext2 / fsck.ext3 / fsck.ext4 명령어**도 제공
- **e2fsck 명령어**
  - 파일 시스템을 점검하기 전에 해당 파일 시스템을 umount 후 진행
  - [-j ext3/ext4] : ext3나 ext4 파일 시스템을 검사할 때 지정
- **badblocks 명령어**
  - 지정한 장치의 배드 블록을 검사하는 명령어
  - [-v] : 검사 결과 자세히 출력
  - [-o] : 검사 결과를 지정한 출력 파일에 저장
- **tune2fs 명령어**
  - ext2 파일 시스템을 설정(튜닝)하는 명령어
  - [-i] : ext2 파일 시스템을 ext3 파일 시스템으로 바꿈
  - [-l] : 파일 시스템 슈퍼 블록 내용을 출력

새로운 디스크 추가 과정 2014(1) 2014(2) 2015(1) 2016(1) 2018(1)

1. 새로운 디스크 설치
2. 파티션 생성 : fdisk 명령어
3. 파일 시스템 생성 : mkfs / mke2fs 명령어
4. 마운트할 디렉터리(마운트 포인트) 생성
5. 파일 시스템 마운트 : mount 명령어

2-3. 관련 명령어

ls 명령어 2017(1) 2017(1)

- 주요 옵션
  - [-a] : 숨겨진 파일을 포함한 모든 파일의 목록을 출력
  - [-d] : 지정한 디렉터리 자체의 정보를 출력
  - [-i] : inode 번호를 출력
  - [-l] : 파일의 상세한 정보를 출력
  - [-A] : .(현재 디렉터리), ..(이전 디렉터리)를 제외하고 출력
  - [-F] : 파일의 종류를 함께 출력 (* : 실행파일, / : 디렉터리, @ : 심벌릭 링크)
  - [-L] : 심벌릭 링크 파일의 원본 파일 정보를 함께 출력
  - [-R] : 하위 디렉터리들의 파일 목록까지 함께 출력
- grep 명령어와 조합
  - 디렉터리만 출력
    # ls -al | grep "^d"
  - 일반 파일만 출력
    # ls -al | grep "^-"
    # ls -al | grep -v "^[a-zA-Z]"
  - 특정 문자열이 들어간 파일 검색
    # ls -al | grep linux

디스크 사용량 출력 명령어

- **df 명령어** 2015(2)
  - 파일 시스템별 디스크 사용량을 확인하는 명령어
  - [-a] : 모든 파일시스템의 디스크 사용량을 출력
  - [-h] : 디스크 사용량을 알기 쉬운 단위로 출력
  - [-t] : 지정한 파일 시스템에 해당하는 디스크 사용량을 출력
  - [-T] : 파일 시스템의 종류도 함께 출력
  - 옵션 없이 사용 시 vs [-h] 옵션 사용 시 비교

```
[fedora@localhost ~]$ df
Filesystem              1K-blocks    Used Available Use% Mounted on
devtmpfs                  1999556       0   1999556   0% /dev
tmpfs                     2012376       0   2012376   0% /dev/shm
tmpfs                     2012376    4760   2007616   1% /run
tmpfs                     2012376       0   2012376   0% /sys/fs/cgroup
/dev/mapper/fedora-root  17410832 8246236   8257128  50% /
tmpfs                     2012376     140   2012236   1% /tmp
/dev/sda1                  999320  146160    784348  16% /boot
tmpfs                      402472      16    402456   1% /run/user/42
tmpfs                      402472    4644    397828   2% /run/user/1000
[fedora@localhost ~]$
[fedora@localhost ~]$ df -h
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                2.0G     0  2.0G   0% /dev
tmpfs                   2.0G     0  2.0G   0% /dev/shm
tmpfs                   2.0G  4.7M  2.0G   1% /run
tmpfs                   2.0G     0  2.0G   0% /sys/fs/cgroup
/dev/mapper/fedora-root   17G  7.9G  7.9G  50% /
tmpfs                   2.0G  140K  2.0G   1% /tmp
/dev/sda1               976M  143M  766M  16% /boot
tmpfs                   394M   16K  394M   1% /run/user/42
tmpfs                   394M  4.6M  389M   2% /run/user/1000
[fedora@localhost ~]$
```

• **du 명령어** 2017(1)
- 디렉터리나 사용자 별로 디스크 사용량을 확인하는 명령어
- 형식 : # du [옵션] [디렉터리]
- [-s] : 특정 디렉터리의 전체 사용량을 출력
- [-h] : 디스크 사용량을 알기 쉬운 단위로 출력
- [-d NUM] : (--max-depth=NUM) 현재 디렉터리 기준으로 하위 NUM 단계 아래의 디렉터리까지의 디스크 사용량 출력
- 옵션 없이 사용 시 현재 디렉터리의 디스크 사용량을 출력

디스크 사용량(quota) 관련 명령어

• 쿼터 설정 전에 /etc/fstab 파일의 옵션 필드에 쿼터 관련 설정을 해주어야 한다.

• **quotacheck 명령어**
- 쿼터 파일을 생성/확인/수정을 위해 파일 시스템을 검사하는 명령어
- [-a] : 전체 파일 시스템 검사
- [-u] : 사용자 쿼터 확인
- [-g] : 그룹 쿼터 확인
- [-m] : 파일 시스템을 다시 마운트하지 않는다.
- [-v] : 자세하게 출력

EX) # quotacheck -avugm

• **quotaon 명령어**
- 파일 시스템의 쿼터 기능을 활성화하는 명령어
- [-a] : 전체 파일 시스템의 쿼터를 활성화
- [-u] : 사용자 쿼터 활성화
- [-g] : 그룹 쿼터 활성화
- [-v] : 자세하게 출력

EX) # quotaon -augv

• **edquota 명령어** 2017(2)
- 쿼터를 설정하는 명령어 (vi 형식)
- [-u] : 사용자 쿼터 설정

EX) # edquota -u fedora → fedora 유저의 쿼터 설정

- [-g] : 그룹 쿼터 설정
- [-p] : 쿼터 설정 복사

EX) # edquota -p fedora fedora2 → fedora 유저의 쿼터 설정 내용을 fedora2 유저에게 적용

• **setquota 명령어** 2016(2)
- 쿼터를 설정하는 명령어 (커맨드 라인에서 옵션으로 설정하는 형식)
- 형식 : **# setquota [옵션] [이름] [block soft limit] [block hard limit] [inode soft limit] [inode hard limit] [장치명]**

EX) # setquota -u fedora 1000 1100 0 0 /

• **quota 명령어**
- 쿼터 정보를 출력하는 명령어
- [-u], [-g] 옵션 포함

EX) # quota -u fedora → fedora 유저의 쿼터 설정 내용 출력

• **repquota 명령어**
- 쿼터 정보를 요약하여 출력하는 명령어
- [-a], [-u], [-g] 옵션 포함
- [-v] : 사용량이 없는 쿼터의 정보 출력

파일 시스템 ACL(Access Control List) 관련 명령어 2014(2)

• **getfacl 명령어**
- 접근 권한을 확인하는 명령어

```
[root@localhost fedora]# getfacl linuxmaster/
# file: linuxmaster/
# owner: fedora
# group: fedora
user::rwx
group::rwx
other::r-x

[root@localhost fedora]#
```

• **setfacl 명령어**
- 접근 권한을 설정하는 명령어
- [-m] : 권한을 수정
- [-x] : 권한을 삭제
- [-R] : 하위 디렉터리의 권한까지 변경

```
[root@localhost linuxmaster]# setfacl -m u:fedora2:6 permtest
[root@localhost linuxmaster]# getfacl permtest
# file: permtest
# owner: root
# group: root
user::rw-
user:fedora2:rw-
group::r--
mask::rw-
other::r--
```

스왑 영역 관련 옵션 2014(1) 2017(2)

• 스왑(swap) : 1 페이지 참고

• **mkswap 명령어**
- 스왑 영역을 생성하는 명령어
- [-c] : 스왑 영역을 만들기 전에 먼저 배드블록을 검사

• **swapon / swapoff 명령어**
- 스왑 영역을 활성화/비활성화 하는 명령어

• **dd 명령어**
- 블록 단위로 파일을 복사/변환하는 명령어
- [if=FILE] : 지정한 파일을 입력으로 받는다
- [of=FILE] : 지정한 파일을 출력으로 설정
- [bs=BYTES] : 한 번에 읽어 들이거나(read) 출력(write)할 바이트 수 지정
- [count=BLOCKS] : 지정한 블록 수만큼 복사

EX) # dd if=/dev/zero of=swapfile bs=1024 count=1000 → 1024바이트씩 1000번을 /dev/zero에 읽어서 /swapfile에 기록

• 스왑 파일 생성 절차 2014(2) 2015(1) 2016(1)
1) dd 명령어로 파일 생성
2) mkswap 명령어로 스왑 파일 생성
3) sync 명령어로 현재 메모리에 있는 데이터를 디스크에 저장
4) swapon 명령어로 스왑 영역 활성화

parted 명령어 2017(2)
- 파티션 생성, 삭제, 크기 변경 등의 파티션 관리를 위한 명령어

3. 프로세스 관리

3-1. 프로세스의 제어

런레벨(Run Level) 분류 2014(1)

Run Level 0	시스템 종료
Run Level 1	단일 사용자 모드
Run Level 2	다중 사용자 모드 (NFS 지원 X)
Run Level 3	다중 사용자 모드 (텍스트 유저 모드)
Run Level 4	사용자 정의 모드
Run Level 5	그래픽 유저 모드
Run Level 6	시스템 재부팅

포그라운드 / 백그라운드

- 포그라운드(foreground) : 포그라운드로 실행시킨 프로세스는 종료되기 전까지는 다른 작업을 할 수 없다.
- 백그라운드(background) : 프로세스를 백그라운드로 실행시키면 다른 작업 수행이 가능하다. 2015(1)
  - 명령어 마지막에 '&' 를 붙여 실행하면 백그라운드로 실행시킬 수 있다.

3-2. 관련 명령어 2014(1) 2016(2)

top 명령어 2015(2)

- CPU 프로세서의 현황을 실시간으로 출력해주는 명령어
- 주요 내부 명령어 2018(1)
  - l : 상단의 load average 항목 켜기/끄기
  - t : 상단의 프로세스와 CPU 항목 켜기/끄기
  - m : 상단의 메모리 항목 켜기/끄기
  - M : 메모리 사용량이 큰 순서로 정렬
  - P : CPU 사용량이 큰 순서로 정렬
  - k : 입력한 PID의 프로세스 종료
  - q : 종료
- 각 필드 설명
  - PID : 프로세스 ID
  - USER : 프로세스를 실행시킨 사용자
  - PR : 프로세스 우선순위
  - NI : NICE 값
  - VIRT : 프로세스가 사용하는 가상 메모리 크기
  - RES : 프로세스가 사용하는 메모리 크기
  - SHR : 프로세스가 사용하는 공유 메모리 크기
  - S : 프로세스의 상태 (S : Sleeping, 휴먼상태 / R : Running, 실행상태 / Z : Zombie, 좀비 프로세스 / …)
  - %CPU : 프로세스가 사용하는 CPU의 사용률
  - %MEM : 프로세스가 사용하는 메모리의 사용률
  - TIME+ : 프로세스가 CPU를 사용한 시간
  - COMMAND : 실행된 명령어

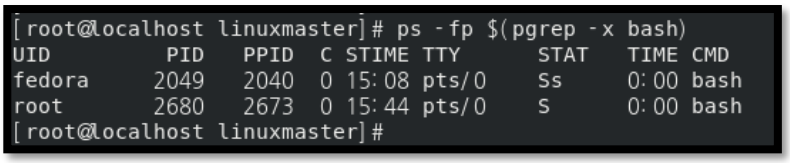
ps 명령어 2017(1)

- 현재 실행중인 작업의 사용자와 PID를 출력하는 명령어
- 주요 옵션
  - [-a] : 모든 프로세스 출력
  - [-e] : 해당 프로세스에 관련된 환경변수 정보를 함께 출력
  - [-f] : 여러 항목들에 관한 내용을 출력
  - [-u USERNAME|UID] : 지정한 사용자가 실행한 프로세스를 출력
  - [-p PID] : 지정한 프로세스의 정보를 출력

- 각 필드 설명
  - PID : 프로세스 ID
  - PPID : 부모 프로세스 ID
  - TTY : 프로세스와 연결된 터미널 포트
  - C : 프로세스가 사용하는 CPU의 사용률
  - STIME : 프로세스가 시작된 시간
  - STAT : 프로세스의 상태 (top 명령어의 S 필드와 같다) 2016(1)

pstree 명령어 2014(1) 2016(2)
- 실행 중인 프로세스를 트리형태로 출력하는 명령어
- 주요 옵션
  - [-c] : 중복된 프로세스도 출력
  - [-n] : PID를 기준으로 정렬하여 출력
  - [-p] : PID도 같이 출력
  - [-u] : UID도 같이 출력

pgrep 명령어 2016(2)
- 지정한 패턴과 일치하는 프로세스에 대한 정보를 출력
- 주요 옵션
  - [-x] : 패턴과 정확히 일치하는 프로세스 정보를 출력
  - [-u] : 지정한 사용자에 대한 모든 프로세스 출력
- 출력되는 정보가 많지 않기 때문에 ps 명령어와 같이 사용된다.



프로세스 종료 명령어
- **kill 명령어**
  - 프로세스에 시그널을 전송하는 명령어
  - 형식 : # kill [시그널] [PID]
  - 옵션을 지정하지 않으면 15번 시그널로 간주되어 실행된다.
  - 주요 시그널 : 13 페이지 참고
  - pgrep 명령어를 이용하면 pkill/killall 명령어와 같은 효과를 볼 수 있다.
    EX) # kill `pgrep sleep`
    # kill $(pgrep sleep)
    # killall sleep → 전부 같은 결과를 나타낸다.
- **pkill 명령어**
  - kill 명령어와 역할이 같지만 PID 대신 해당 명령어의 프로세스를 종료하는 명령어
  - kill 명령어는 지정한 PID의 프로세스 하나만 종료되지만, pkill 명령어는 지정한 명령어에 관련된 여러 프로세스들이 한꺼번에 종료된다.
- **killall 명령어** 2014(2) 2017(2)
  - pkill 명령어와 기능이 같다.

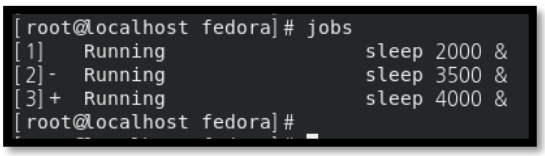
우선순위 관련 명령어

• **nice 명령어** 2014(2) 2015(2) 2017(2)
- 지정한 프로세스의 우선순위(top 명령어의 NI 값)를 조정하는 명령어
- [-n] 옵션으로 우선순위를 조정할 수 있다.
  EX) # nice -n 15 top → top 명령어의 우선순위를 15 증가
- 옵션없이 실행 시 지정한 프로세스의 우선순위를 10 증가시킨다.
- nice 명령어로 우선순위를 조정하면 새로운 프로세스가 추가된다.
- 우선순위 범위는 -20 ~ 19 사이의 값으로, 우선순위 값이 작을수록 우선순위가 높다.
- 우선순위 값을 낮추는 것(우선순위가 높아지는 것)은 root 만 가능하고 일반 계정은 우선순위 값을 높이는 것만 가능하다.

• **renice 명령어** 2016(1) 2017(2)
- 지정한 PID/UID/GID의 우선순위를 지정하는 명령어
- [-u UID|USERNAME] : 지정한 UID 또는 사용자의 모든 프로세스의 우선순위를 지정
- [-g GID] : 지정한 GID의 모든 프로세스의 우선순위를 지정
- [-p PID] : 지정한 PID의 우선순위를 지정
- nice 명령어와 다르게 지정한 우선순위의 값이 바로 설정되고, 새로운 프로세스가 추가되지 않는다.
  EX) # renice 10 -u fedora → fedora 사용자의 모든 프로세스의 우선순위를 10으로 지정

백그라운드 작업 관련 명령어

• **jobs 명령어** 2014(1) 2017(1)
- 백그라운드 작업을 출력하는 명령어
- [-l] : 작업의 PID도 같이 출력
- [-p PID] : 해당 PID의 작업을 출력
- [-r] : Running 상태의 작업만 출력
- [-s] : Stopped 상태의 작업만 출력
- %[작업번호] 로 해당 작업의 정보만 출력할 수 있다.
- 작업번호 뒤에 기호 '+'는 가장 최근에 접근한 작업을 의미하고 '-'는 '+' 작업 바로 이전에 접근한 작업을 의미



• **nohup 명령어** 2014(2) 2015(2)
- 로그아웃한 후에도 백그라운드에서 작업을 계속 실행하도록 하는 명령어
- 형식 : # nohup [명령어] &
  EX) # nohup find / -name shell & → 'find / -name shell' 명령어를 백그라운드로 작업을 계속 실행하도록 함

• **fg / bg 명령어** 2014(2) 2017(1) 2017(2)
- 지정한 작업번호에 대한 작업을 포그라운드 / 백그라운드 실행으로 옮기는 명령어
  EX) # bg %3 → jobs 명령어로 확인한 작업번호 3의 작업을 백그라운드로 실행하도록 함
- 작업번호 지정없이 실행 시 가장 최근 작업에 대해 명령을 실행

작업 예약 관련 명령어

• **at 명령어**
- 지정한 시간에 지정한 명령을 한번만 실행하는 명령어
- [-l] : 예약 중인 명령어 목록을 출력 = **atq 명령어**와 같은 기능
- [-r] : 지정한 예약된 작업을 삭제 = **atrm 명령어**와 같은 기능
- 접근제어 파일 : **/etc/at.allow**, **/etc/at.deny** (at.allow 파일 우선)

• **crontab 명령어** 2014(2) 2015(1) 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)
- 지정한 시간에 지정한 명령어를 주기적으로 실행하는 명령어
- [-e] : 사용자의 crontab 파일을 편집
- [-l] : crontab 파일의 목록을 출력
- [-r] : crontab 파일을 삭제
- crontab 파일 형식 : [분] [시] [일] [월] [요일] [작업내용]

EX) 30 20 1 * * /usr/bin/ls -l ~fedora > ~fedora/cron.out → 매월 1일 20시 30분에 'ls -l ~fedora' 명령의 실행 결과를 cron.out 파일에 저장한다.

0 12-22/2 * * 1 /usr/bin/ps -ef → 매주 월요일 12시부터 22시 정각 2시간마다 'ps -ef' 명령어를 실행

*/30 0-2 1 * * /usr/bin/ls -l ~guest > ~root/geust.out → 매월 1일 0시부터 2시까지 30분마다 'ls -l ~geust' 명령의 실행 결과를 ~root/guest.out 파일에 저장한다.

- 접근제어 파일 : **/etc/cron.allow**, **/etc/cron.deny** (cron.allow 파일 우선) 2014(1) 2015(2)

4. 소프트웨어 설치 및 관리

4-1. 패키지를 통한 소프트웨어 설치 2015(2)

rpm 명령어 2014(1) 2014(1) 2014(2) 2014(2) 2014(2) 2015(1) 2015(1) 2015(2) 2016(1) 2016(1) 2017(1) 2017(2) 2017(2) 2018(1)

• RPM : Redhat Package Manager의 약자로 레드햇에서 만든 패키지 관리 도구. 확장자가 rpm인 패키지를 관리

• RPM 패키지 이름 구성 2016(2)

패키지 이름	패키지 버전	패키지 릴리즈	아키텍처	확장자
gnome-sudoku-	3.29.2-	3.fc29.	aarch64	.rpm
yum-	3.4.3-	518.fc29.	noarch	.rpm

• 주요 옵션

1) 설치/업그레이드 관련 옵션
- [-i PACKAGE.rpm] : 패키지 설치
- [-U] : 패키지 업그레이드
- [-h] : 해시 기호(#)를 출력하여 진행도를 보기 쉽게 출력
- [-v] : 자세하게 출력
- [--replacepkgs] : 패키지가 설치가 되어있어도 강제로 설치
- [--replacefiles] : 패키지의 파일이 설치가 되어있어도 강제로 설치
- [--nodeps] : 의존성을 무시
- [--oldpackage] : 구버전으로 다운그레이드
- [--rebuilddb] : RPM 데이터베이스 업데이트
- [--force] : [--replacepkgs], [replacefiles], [oldpackage] 옵션을 같이 사용한 것과 동일

2) 질의 관련 옵션 → [-q] 와 같이 사용
- [-a] : 전체 패키지 목록 질의
- [-f FILE] : 지정한 파일이 속한 패키지 질의
- [-i] : 자세한 상세정보 질의
- [-p PACKAGE] : 지정한 패키지의 정보 질의
- [-l PACKAGE] : 지정한 패키지가 설치한 파일 목록 질의
- [-R PACKAGE] : 지정한 패키지의 의존성 질의

3) 삭제 옵션
- [-e PACKAGE] : 지정한 패키지 삭제

4) 검증 옵션
- [-V PACKAGE] : 지정한 패키지 무결성 검사
- 출력결과가 없으면 문제가 없는 것이다.

5	MD5 checksum	D	장치 (Device)
S	파일 크기 (Size)	U	사용자 (User)
L	심벌릭 링크 (Link)	G	그룹 (Group)
T	갱신일 (Time)	M	퍼미션 (perMission)

- 사용 예제
  # rpm -ivh gnome-sudoku-3.29.2-3.fc29.aarch64.rpm → gnome-sudoku 설치
  # rpm -Uvh gnome-sudoku-3.29.2-3.fc29.aarch64.rpm → gnome-sudoku 업그레이드
  # rpm -qa | grep gnome → 전체 패키지 목록 중에 gnome 문자열이 들어간 패키지 출력
  # rpm -qf /usr/bin/gnome-sudoku → gnome-sudoku 파일이 어느 패키지에 속해 있는지 확인
  # rpm -qi gnome-sudoku → gnome-sudoku 패키지의 상세 정보 출력
  # rpm -e gnome-sudoku → gnome-sudoku 패키지 삭제

yum 명령어 / dnf 명령어 2014(1) 2014(2) 2015(1) 2015(2) 2017(1) 2017(2)

- RPM 기반의 패키지 관리 도구 → 레드햇 계열 도구
- rpm 명령은 패키지 의존성을 확인하여 설치를 일일이 해주어야 하지만 yum 명령은 알아서 확인하여 설치까지 할 수 있도록 도와준다.
- DNF는 페도라 22 버전 이후부터 yum을 대체하는 패키지 관리자이다.
- 주요 옵션
  - [list] : 설치된 전체 패키지 목록 출력
  - [list all] : 이미 설치되었거나 설치 가능한 모든 패키지 목록 출력
  - [list available] : 설치 가능한 모든 패키지 목록 출력
  - [list updates] : 업데이트 가능한 패키지 목록 출력 ([list upgrades] 옵션과 같은 기능)
  - [check-update] : 업데이트 가능한 패키지 목록 출력 ([check-upgrade] 옵션과 같은 기능)
  - [list installed] : 설치된 패키지 목록 출력
  - [install] : 지정한 패키지 설치
  - [update] : 지정한 패키지 업데이트 ([upgrade] 옵션과 같은 기능)
  - [info] : 지정한 패키지의 정보 출력
  - [search] : 지정한 문자열이 들어간 패키지 검색
  - [remove] : 지정한 패키지 삭제
  - [-y] : 설치 과정의 모든 질문에 yes로 대답
  - [-v] : 자세하게 출력
- **yumdownloader 명령어** 2018(1)
  - yum 패키지를 설치하지 않고 rpm을 다운로드하는 명령어
  - yum-utils 패키지에 포함
- yum 저장소 디렉터리 : **/etc/yum.repos.d** 2017(1)

dpkg 명령어 2015(1) 2016(2) 2018(1)

- dpkg는 데비안의 패키지 관리 도구. 확장자가 deb인 패키지를 관리
- 주요 옵션
  - [-i PACKAGE.deb] : 지정한 패키지 설치 (.deb 확장자를 가지는 패키지 전체 파일명으로 설치해야 함)
  - [-l] : 설치된 전체 패키지 목록 출력
  - [-L PACKAGE] : 지정한 패키지가 설치한 파일 출력
  - [-I PACKAGE.deb] : (대문자 i) 지정한 패키지의 정보 출력
  - [-P PACKAGE] : 지정한 패키지 삭제
  - [-S FILE] : 지정한 파일이 속한 패키지 질의

apt 명령어 2015(1)

- APT(Advanced Packaging Tool)은 데비안 계열의 패키지 관리도구
- 주요 옵션

1) **apt-get 명령어** 옵션

- [install] : 지정한 패키지 설치
- [update] : 패키지 정보 업데이트
- [upgrade] : 설치되어 있는 모든 패키지를 업데이트
- [--reinstall] : [install] 옵션과 함께 사용하여 지정한 패키지 재설치
  EX) # apt-get --reinstall install gnome-sudoku
- [remove] : 지정한 패키지 삭제 (설정파일은 삭제하지 않음)
- [--purge] : [remove] 옵션과 함께 사용하면 설정파일까지 모두 삭제
  EX) #apt-get --purge remove gnome-sudoku

2) **apt-cache 명령어** 옵션

- [search] : 패키지 검색
- [show] : 지정한 패키지의 정보 출력

4-2. 소스 코드 컴파일

gcc 명령어 2014(1)

- GNU의 C/C++ 컴파일러
- 주요 옵션
  - [-o] : 출력 파일 지정
  - [-c] : 오브젝트 파일 생성 (컴파일만 진행)
  - [-v] : 컴파일 과정을 자세하게 출력

EX) # gcc -o output.out source.c → source.c 파일을 컴파일하여 output.out 실행파일을 생성
gcc -c source.c → source.c 파일의 오브젝트 파일(source.o) 생성
gcc source.c → source.c 파일을 컴파일하여 실행파일을 생성(a.out)

소스파일로 된 패키지 설치하기 2015(1)

- **configure 명령어** 2018(1)
  - 제일 첫번째 단계로, 환경설정을 위한 스크립트
  - Makefile을 생성하는 단계이다.
- **make 명령어**
  - Makefile 파일에 설정된 정보를 읽어서 여러 개의 소스파일을 컴파일하고 링크하여 실행파일을 생성하는 명령어
  - 소스 파일을 컴파일하기 위함
- **cmake 명령어** 2014(2) 2016(2)
  - 멀티 플랫폼으로 사용할 수 있는 make의 대용품
  - make와 달리 유닉스 계열 뿐만 아니라 윈도우 계열의 프로그래밍 도구(Visual Studio 포함)도 지원
- **make install 명령어**
  - 설치를 위한 명령어
  - make 명령어로 생성된 설치파일을 실행하기 위한 과정이다.
- 일반적인 과정 : configure → make → make install 2016(1) 2016(2) 2017(2)
- cmake 과정 : cmake → make install 2015(2)

아카이브/압축 관련 명령어 2016(2) 2017(1) 2018(1)

- **tar 명령어** 2014(1) 2014(2) 2015(1) 2015(2) 2017(1) 2017(2)
  - 파일/디렉터리를 하나로 묶어 아카이브 파일을 생성
  - [c] : 아카이브 파일 생성
  - [x] : 아카이브 파일에서 원본파일을 추출
  - [t] : 아카이브 파일의 내용 출력
  - [r] : 기존 아카이브 파일에 새로운 파일/디렉터리 추가
  - [u] : 아카이브 파일의 내용에 변동사항을 업데이트
  - [v] : 처리중인 파일의 정보를 출력
  - [f] : 아카이브 파일 지정
  - [z] : gzip으로 압축하거나 해제
  - [j] : bzip2로 압축하거나 해제
  - [J] : .xz로 압축하거나 해제

  EX) # tar cvf test.tar a.txt b.txt → a.txt 파일과 b.txt 파일을 test.tar 아카이브 파일로 묶음
  # tar tvf test.tar → test.tar 파일의 내용 출력
  # tar xvf test.tar → test.tar 파일의 원본 파일 추출
- **gzip 명령어**
  - ".gz" 확장자 형태로 압축
  - [-d] : 압축 해제
  - [-l] : 압축 파일의 정보 출력
  - [-v] : 압축 진행 과정을 출력
  - [-9] : 압축률을 최대로 하여 압축 * 1~9까지 지정할 수 있으며 숫자가 클수록 압축률이 크지만 속도가 느리다.
  - [-k] : 기존 파일을 유지하고 압축한다.

  EX) # gzip -v test.tar → test.tar 파일 압축
  - **zcat 명령어** : .gz 로 압축된 파일의 내용을 출력
  - **gunzip 명령어** : .gz 로 압축된 파일의 압축 해제
- **bzip2 명령어**
  - ".bz2" 확장자 형태로 압축
  - gzip 명령어보다 압축률이 높지만 속도가 약간 느리다.
  - gzip 명령어와 옵션이 동일
  - **bzcat 명령어** : .bz2 로 압축된 파일의 내용을 출력
  - **bunzip2 명령어** : .bz2 로 압축된 파일의 압축 해제
- **xz 명령어**
  - ".xz" 확장자 형태로 압축
  - 압축률이 가장 높은 명령어
  - gzip, bzip2 명령어와 옵션이 동일
  - **xzcat 명령어** : .xz 로 압축된 파일의 내용을 출력
  - **unxz 명령어** : .xz 로 압축된 파일의 압축 해제
- **zip 명령어**
  - ".zip" 확장자 형태로 압축
  - 윈도우와 호환되는 압축파일
  - 형식 : # zip [압축파일명.zip] [압축할 파일들] → 기존 압축 명령과 형식이 다름
  - **unzip 명령어** : .zip 으로 압축된 파일의 압축 해제
- **compress 명령어**
  - ".Z" 확장자 형태로 압축
  - **uncompress 명령어** : .Z 로 압축된 파일의 압축 해제
- 압축 명령어들의 압축률 순서 2015(2)
  - xz > bzip2 > gzip > compress

```
-rw-r--r--. 1 root root 696320  8월 13 14:05 test.tar
-rw-r--r--. 1 root root 129449  8월 13 13:48 test.tar.bz2
-rw-r--r--. 1 root root 142657  8월 13 13:38 test.tar.gz
-rw-r--r--. 1 root root 105948  8월 13 13:55 test.tar.xz
-rw-r--r--. 1 root root 142796  8월 13 14:06 test.tar.zip
```

CH.5 장치 관리

1. 장치의 설치 및 관리

1-1. 장치의 설치 및 관리

/proc 디렉터리내 주요 파일 2016(2)

- **/proc/partitions** : 현재 시스템의 파티션 정보를 저장 2015(2) 2016(2)

```
[root@localhost fedora]# cat /proc/partitions
major minor  #blocks  name

  11        0    1048575 sr0
   8        0   20971520 sda
   8        1    1048576 sda1
   8        2   19921920 sda2
   8       16    1048576 sdb
   8       17     512000 sdb1
   8       18     512000 sdb2
   8       32    1048576 sdc
   8       33     512000 sdc1
   8       34     512000 sdc2
   8       48    1048576 sdd
   8       49     307200 sdd1
   8       50     307200 sdd2
   8       51     409600 sdd3
 253        0   17821696 dm-0
 253        1    2097152 dm-1
[root@localhost fedora]#
```

- **/proc/cpuinfo** : CPU 정보를 저장

```
[fedora@localhost ~]$ cat /proc/cpuinfo
processor       :
vendor_id       :
cpu family      :
model           :
model name      :
stepping        :
microcode       :
cpu MHz         :
cache size      :
physical id     :
```

- **/proc/devices** : 현재 시스템이 사용하는 디바이스 정보를 저장

```
[fedora@localhost ~]$ cat /proc/devices
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  7 vcs
 10 misc
 13 input
 14 sound
 21 sg
 29 fb
 99 ppdev
116 alsa
```

- **/proc/filesystems** : 커널이 지원하는 파일 시스템의 정보를 저장 2015(2)

```
[fedora@localhost ~]$ cat /proc/filesystems
nodev   sysfs
nodev   rootfs
nodev   ramfs
nodev   bdev
nodev   proc
nodev   cpuset
nodev   cgroup
nodev   cgroup2
nodev   tmpfs
nodev   devtmpfs
nodev   configfs
nodev   debugfs
```

- **/proc/meminfo** : 시스템의 메모리 정보를 저장 2018(1)

```
[fedora@localhost ~]$ cat /proc/meminfo
MemTotal:        4024764 kB
MemFree:         2136204 kB
MemAvailable:    2441304 kB
Buffers:           30240 kB
Cached:           471252 kB
SwapCached:            0 kB
Active:           925256 kB
Inactive:         372064 kB
Active(anon):     797248 kB
Inactive(anon):    11204 kB
Active(file):     128008 kB
Inactive(file):   360860 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:       2097148 kB
```

- **/proc/swaps** : 스왑 공간 관련 정보를 저장

```
[fedora@localhost ~]$ cat /proc/swaps
Filename                                Type            Size    Used    Priority
/dev/dm-1                               partition       2097148 0       -1
```

- **/proc/mdstat** : RAID나 중복 디스크에 대한 정보를 저장

/dev 디렉터리 주요 장치 파일 2015(1)

- **/dev/lp*** : 프린터 장치 2018(1)
- **/dev/hd*** : IDE 드라이버, ATA 드라이버, …
- **/dev/sd*** : SCSI 드라이버, SATA 드라이버, USB 드라이버, …
- **/dev/tty*** : 터미널 장치

커널 컴파일 2017(1)

- 커널(kernel) : 운영체제의 가장 바깥 부분에 위치하여 사용자 명령어에 대한 처리를 담당
- 커널 컴파일 : 커널 소스를 사용자가 자신에 맞는 커널 환경을 만드는 과정
- 커널 컴파일 과정과 명령어 2014(2) 2016(1)

  1) 커널 소스 설치 2017(2)
  - 커널 소스를 /usr/src 디렉터리에 다운로드 후 압축해제 (보통 **/usr/src/kernels**에 설치됨)
  - 커널 패치 수행

  2) 커널 컴파일 준비(초기화) (**make mrproper 명령어**) 2015(1) 2015(2) 2017(1)
  - 기존에 설정되어 있는 커널 설정 값을 모두 초기화하는 명령어
  - 컴파일 되어 있는 오브젝트 파일(.o 확장자)이 제거된다.

  3) 커널 컴파일을 위한 환경 설정 2014(1) 2014(2) 2017(1)
  - 필요 없는 드라이버/모듈을 제외하거나 포함하는 단계
  - **make config 명령어** : 전통적인 방식
  - **make menuconfig 명령어** : 메뉴 방식의 화면으로 설정 2015(1) 2016(2)
  - **make xconfig 명령어** : X 윈도 환경에서 구현한 방식. 인터페이스가 편리 2017(2)

  4) 커널 컴파일

  (1) **make dep 명령어**: 새로운 커널을 만드는 명령어 (커널 2.6 버전 이후로는 사용 X)

  (2) **make clean 명령어** : 이전에 수행했던 컴파일 과정에서 생성된 목적파일, 커널, 임시 파일 등을 삭제

  * **make distclean 명령어** : 커널 소스를 받은 최초 상태로 되돌리는 명령어 2017(2)

  (3) **make bzImage 명령어** : 압축된 커널 이미지를 생성

  (4) **make modules 명령어** : 커널 환경 설정에서 모듈로 설정한 기능들을 컴파일 2015(2)

  (5) **make modules_install** 명령어 : 컴파일된 모듈을 /lib/modules에 설치

  - 위에 명령어들은 **make all 명령어**로 한번에 수행할 수 있다.

  5) 커널 설치 (**make install 명령어**) 2018(1)
  - 생성한 커널 이미지를 설치
  - /boot 디렉터리에 필요한 파일을 복사하는 과정

• 순서 정리 : 커널 소스 설치 → make mrproper → make config → make clean → make bzlmage → make modules → make modules_install → make install 2016(2)

* **make help 명령어** : 커널 컴파일에 필요한 명령어들을 확인할 수 있다. 2018(1)

커널 모듈 설정 명령어/파일 2016(1) 2017(1)

• 커널 모듈의 설치 위치는 **/lib/modules/[커널 버전]/*** 이다. 2014(1) 2016(2) 2018(1)

• **lsmod 명령어** 2014(2) 2015(1) 2016(1)
- 현재 시스템에 설치된 모듈 목록 출력
- **/proc/modules** 파일 참조

```
[fedora@localhost ~]$ lsmod
Module                  Size  Used by
rfcomm                 77824  4
fuse                  102400  3
nf_conntrack_netbios_ns    16384  1
nf_conntrack_broadcast    16384  1 nf_conntrack_netbios_ns
xt_CT                  16384  1
ip6t_rpfilter          16384  1
ip6t_REJECT            16384  2
nf_reject_ipv6         16384  1 ip6t_REJECT
xt_conntrack           16384  21
ip_set                 36864  0
nfnetlink              16384  1 ip_set
ebtable_nat            16384  1
ebtable_broute         16384  1
bridge                143360  1 ebtable_broute
stp                    16384  1 bridge
llc                    16384  2 bridge,stp
ip6table_nat           16384  1
nf_conntrack_ipv6      20480  12
nf_defrag_ipv6         36864  1 nf_conntrack_ipv6
nf_nat_ipv6            16384  1 ip6table_nat
ip6table_mangle        16384  1
```

• **insmod 명령어**
- 모듈을 커널에 설치하는 명령어
- 형식 : # insmod [옵션] [설치할 모듈(오브젝트 파일)]

• **rmmod 명령어**
- 모듈을 삭제하는 명령어
- [-r] : 의존성이 있는 모듈을 모두 삭제

• **modprobe 명령어** 2014(1) 2014(2) 2015(2) 2016(2) 2018(1)
- 모듈을 설치/삭제하는 명령어
- insmod/rmmod 명령어와 다르게 **modules.dep 파일**을 참조해 의존성 문제를 해결
- [-l] : 모든 모듈 목록 출력
- [-a] : 모듈을 설치. 의존성이 있는 모듈을 함께 설치
- [-r] : 모듈을 삭제. 의존성이 있는 모듈 중 사용되고 있지 않는 모듈을 같이 삭제

• **depmod 명령어** 2014(1) 2016(1) 2017(2)
- 의존성을 검사하여 modules.dep 파일을 갱신하는 명령어
- [-a] : **/etc/modules.conf** 파일에 있는 모든 모듈의 의존성을 갱신
- [-v] : 처리된 모듈의 목록을 출력

• **modinfo 명령어** 2017(2)
- 지정한 모듈의 정보를 출력하는 명령어

- **/etc/modprobe.d 디렉터리** 2015(1) 2015(2) 2017(1)
  - 부팅 시 모듈을 자동으로 적재하기 위한 디렉터리
  - 디렉터리 안에 있는 .conf 파일을 읽어 부팅 시에 자동으로 적재한다.
  - 커널 2.4 버전 이전에는 **/etc/modprobe.conf 파일**을 읽어 자동으로 적재했었다. 2016(1)

```
[fedora@localhost ~]$ ls -l /etc/modprobe.d/
합계 16
-rw-r--r--. 1 root root  223  9월  8  2017 kvm.conf
-rw-r--r--. 1 root root  747  7월 31  2017 lockd.conf
-rw-r--r--. 1 root root 1004  8월 10  2017 mlx4.conf
-rw-r--r--. 1 root root   92  8월 10  2017 truescale.conf
```

- **/lib/modules/[커널 버전]/modules.dep 파일** 2016(2)
  - 모듈의 의존성이 정의되어 있는 파일
  - 파일 형식 : [모듈 이름] : [의존성이 있는 모듈 이름] [의존성이 있는 모듈 이름] …

1-2. 주변 장치 설정 2014(1) 2017(2)

프린터

- 오픈 소스 프린팅 시스템 (CUPS) 2015(1)
  - 애플에서 개발한 오픈 소스 프린팅 시스템
  - 컴퓨터를 프린터 서버로 사용할 수 있게 해주는 모듈 방식의 프린팅 시스템
  - **/etc/cups/printers.conf 파일**
    - ㄴ 프린터 큐 관련 환경 설정 파일
    - ㄴ **lpadmin 명령어**를 이용하거나 웹을 통해 설정 2018(1)
- 인터넷 인쇄 프로토콜(IPP) 포트 번호 : TCP,UDP/631 2016(1) 2016(2) 2018(1)

프린터 관련 명령어 2014(1) 2015(1) 2015(2) 2016(1) 2016(1) 2016(2) 2017(1) 2017(1) 2017(2) 2017(2) 2018(1)

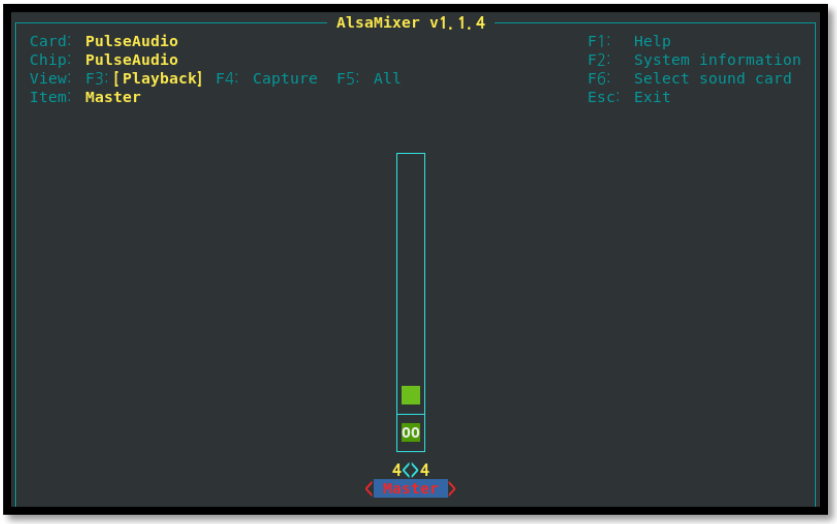
- 기본적으로 [-P] 옵션으로 프린터 지정을 하지 않으면 기본 프린터를 기본값으로 명령 실행
- **/etc/printcap 파일** : 프린터 설정 파일
- **lpc 명령어** : 대화형으로 프린터를 관리(제어)하기 위한 명령어
- **lpq 명령어** : 프린터 큐의 내용 출력
  - System V 계열 : **lpstat 명령어**
- **lprm 명령어** : 프린터 큐의 작업 삭제
  - System V 계열 : **cancel 명령어**
- **lpr 명령어** : 파일 내용을 출력
  - System V 계열 : **lp 명령어**
  - [-# NUM] : 한 페이지를 "NUM"장씩 출력 ([-K] 옵션과 동일)
    - ㄴ System V 계열 : [-n NUM] 옵션
  - [-b] : 배너나 헤더를 출력하지 않음
  - 다른 방법을 통한 출력 방법 : **# cat [출력할 파일] > /dev/lp** 2015(2)

스캐너

- SANE 2015(1) 2015(2)
  - 스캐너 관련 하드웨어를 사용할 수 있도록 해주는 API
  - 관련 소프트웨어 : Xsane(X 윈도 기반), Simple Scan(GNOME), GIMP(GNU, 그림 편집 프로그램) 2016(2)

사운드 카드 2018(1)

- ALSA(Advanced Linux Sound Architecture) 2015(1)
  - 사운드 카드용 장치 드라이버를 위한 API를 제공
  - 리눅스 커널의 구성요소. GPL 기반
  - 관련 소프트웨어 : alsamixer 2016(2)



- OSS(Open Sound System)
  - 사운드를 만들고 캡처하는 인터페이스
  - 초기에는 OSS를 사용했으나 최근에는 ALSA가 더 많이 사용된다.

하드웨어 관련 명령어

- **lscpu 명령어** : CPU 정보 출력
- **lspci 명령어** : 시스템 하드웨어 정보 출력 2018(1)
- **lsscsi 명령어** : SCSI 장치의 목록 출력
- **lsusb 명령어** : usb 장치의 목록 출력

CH.6 시스템 보안

1. 시스템 분석

1-1. 시스템 로그 분석

시스템 로그 파일 종류

• 일반적으로 리눅스에서는 /var/log 디렉터리에서 시스템의 모든 로그를 기록하고 관리한다. 2014(2) 2017(2)

• 로그들은 (r)syslogd에 의해서 관리되며, **/etc/syslog.conf 파일**에서 로그 파일의 저장 위치와 저장 파일 이름을 변경할 수 있다. 2014(1)

• 시스템 로그 / 관련 명령어 정리 2014(1) 2014(2) 2015(1) 2015(2) 2016(2) 2016(2) 2017(2) 2018(1)

로그명	내용	로그경로	명령어	기타
utmp	현재 로그인한 사용자의 정보	/var/run/utmp	finger, w, who	바이너리 파일이므로 명령어를 통해 확인
wtmp	성공한 로그인/로그아웃, 시스템 boot/shutdown 정보	/var/log/wtmp	last, last -F	바이너리 파일이므로 명령어를 통해 확인
lastlog	마지막으로 성공한 로그인 기록	/var/log/lastlog	lastlog, finger	- lastlog -u [계정명] - lastlog -t [일수]
btmp	실패한 로그인 시도에 대한 기록	/var/log/btmp	lastb last -f btmp	- 실패한 모든 로그를 남김 - 바이너리 파일이므로 명령어를 통해 확인
acct/pacct	사용자가 입력한 명령어, 터미널 종류, 프로세스 시작시간 등을 기록	- (생략)	lastcomm	바이너리 파일이므로 명령어를 통해 확인
secure	사용자/그룹 생성, 삭제, 로그인 등 사용자 인증에 대한 정보 기록	/var/log/secure	x	원격 접속(ssh, telnet) 기록, su 명령어 수행 내역, xinetd 인증 기록
message	시스템 운영에 대한 전반적인 메시지 기록	/var/log/messages	x	tcpwrapper, snort 등의 수행 내역 기록
dmsg	리눅스가 부팅될 때 출력되는 메시지 기록	/var/log/dmesg	dmesg	텍스트 파일
boot.log	리눅스가 부팅될 때 파일시스템, 서비스 데몬 체크한 정보 기록	/var/log/boot.log	x	-
Xferlog	FTP 로그 파일	/var/log/xferlog	x	FTP 사용한 내용을 상세하게 기록

로그 관련 명령어 상세 2017(1)

• **last 명령어** 2016(1) 2018(1)

- 성공한 로그인/로그아웃 정보, 시스템 부팅/종료 정보를 출력하는 명령어
- 관련 로그 파일 : /var/log/wtmp
- [-F] : 정보에 대한 시간을 자세히 표시
- [-NUM] : 최근으로부터 NUM개의 정보를 출력
- [-f FILE] : 지정한 파일의 로그 정보 출력
- [-x] : 런레벨의 변동 정보도 같이 출력
- 명령어 뒤에 특정 사용자 계정을 지정하면 해당 사용자 계정에 대한 로그만을 출력

EX) # last -10 reboot → 시스템 부팅/종료에 대한 로그의 최근 10개 출력

• **lastb 명령어** 2014(2) 2015(2) 2017(1)
- 실패한 로그인 시도에 대한 정보를 출력하는 명령어
- 관련 로그 파일 : /var/log/btmp
- last 명령어와 옵션이 같음

• **lastlog 명령어** 2017(2)
- 마지막으로 성공한 로그인 기록을 출력하는 명령어
- 관련 로그 파일 : /var/log/lastlog
- [-u] : 지정한 사용자에 대한 로그만 출력
- [-t DAY] : DAY 일수 이전에 대한 로그만 출력

• **dmesg 명령어** 2016(1) 2018(1)
- 리눅스 부팅 시 출력되는 메시지 기록을 출력하는 명령어
- 관련 로그 파일 : /var/log/dmesg

/etc/(r)syslog.conf 파일 2015(2) 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)

• 파일 내용 구성

```
FACILITY.PRIORITY; ···                ACTION
```

• facility(서비스) 종류
- auth : 인증 관련 시스템 (로그인, su, getty 등)
- authpriv : auth와 같지만 사용자 별로 읽을 수 있는 파일에 기록됨
- kern : 커널 시스템
- cron, ftp, mail, lpr, syslog, ···

• priority(우선순위) : emerg > alert > crit > err > warning > notice > info > debug

• action(앞에 서비스/우선순위에 대한 결과) : 특정 로그 파일을 지정하거나, 터미널/콘솔/특정 유저에게 보낼 수 있다.

• 설정 내용 예제
- `kern.*  /dev/console`
  → kernel에 관련된 모든 로그를 콘솔에 출력
- `*.info;mail.none;auth.none  /var/log/messages`
  → mail, auth 서비스에 대한 로그를 제외한 모든 서비스에 대한 info 레벨이상의 메시지를 messages 파일에 기록
- `*.err  fedora`
  → 모든 서비스에 대한 err 레벨 이상의 메시지를 fedora 사용자의 스크린으로 메시지를 보냄
- `mail.*;mail.!=info  /var/log/maillog`
  → 메일 서비스에 대해 info 레벨을 제외한 모든 메시지를 maillog 파일에 기록

rsyslog 2014(2) 2015(1) 2016(2) 2018(1)
- syslog의 확장 버전으로, TCP/UDP를 이용한 로깅을 지원
- 데이터 베이스 관련 로그 관리도 가능함.
- 데몬은 rsyslogd 이며, 설정파일은 /etc/rsyslog.conf 이다.

로그 순환(logrotate) 2015(2) 2017(1)

- 로그 파일이 지정 용량에 도달했을 때 다른 파일로 대체하고, 로그 파일을 압축하여 관리하는 것
- 통합 로그 설정 (**/etc/logrotate.conf**) 2014(1) 2015(1) 2017(2)

```
# vi /etc/logrotate.conf
weekly
/* 주 단위 로그파일 순환 (daily, monthly) */
rotate 4
/* 순환 로그 파일의 개수 설정 */
create
/* 순환 시 새롭게 로그파일을 생성 */
dateext
/* 로그파일의 확장자로 날짜를 붙여서 생성 */
compress
/* 로그 파일 압축(compress), 압축안함(uncompress) */
include /etc/logrotate.d
/* 해당 디렉터리에 있는 개별 데몬/프로세스 설정 파일을 포함 */
```

- 개별 로그 설정 (/etc/logrotate.d/*) 2016(1)

```
# vi /etc/logrotate.d/httpd
/var/log/httpd/*log {                  /* httpd 디렉터리에 있는 모든 로그에 적용 */
        missingok                      /* 로그파일이 없어도 오류 발생 X */
        notifempty                     /* 로그파일이 비어 있는 경우 순환 X */
        create 0644 root utmp          /* 해당 권한, 소유자, 소유그룹으로 생성 */
}
```

2. 시스템 보안 및 관리

2-1. 시스템 보안 관리

SELinux 2014(1)

- 리눅스 커널에 내장된 보안 모듈로 MAC(강제적 접근 제어)를 수행
- 관련 명령어
  - **getenforce** 명령어 : SELinux의 상태를 확인하는 명령어 2015(2)
    - ㄴ 활성화 – enforcing / 비활성화 – disabled / 테스트 모드(비활성화) – permissive 2017(1)
  - **sestatus 명령어** : SELinux의 정보를 출력하는 명령어
  - **setenforce 명령어** : SELinux의 동작 모드를 변경하는 명령어 (0 – permissive / 1 – enforcing) 2016(2) 2017(2)

2-2. 시스템 보안 관련 명령어

SSH(Secure Shell) 2016(1) 2018(1)

- 암호통신을 이용하여 네트워크상의 다른 컴퓨터에 접속하여 원격으로 명령을 실행/파일 조작을 하는 프로토콜
- 원격 쉘(rsh) 지원, 원격 복사(scp) 지원, sftp 지원
- 포트 번호 : TCP/22
- 명령어 옵션 2016(1)
  - [-l] : 원격 시스템에 로그인 할 사용자 지정
  - [-p] : 원격 시스템에 연결할 포트 지정.

PAM(Pluggable Authentication Modules) 2014(2)

- 리눅스 사용자 인증의 핵심으로, 중앙통제 기능을 한다.
- PAM 구성 파일(/etc/pam.d/*) 문법
  - [type] [control] [module-path] [module-arguments]
- 구성 토큰 2014(1) 2015(1) 2018(1)
  (1) type : PAM에 어떤 타입의 인증이 사용될 것인지를 알려줌
    - account : 사용자가 해당 서비스에 접근이 허용되었는지, 패스워드 기간이 만료되었는지 결정
    - auth : 주로 패스워드를 사용하지만, 보다 정교한 방법을 통해 사용자가 맞는지를 결정
    - session : 사용자가 인증 받기 전/후에 해야 할 것을 나타냄
  (2) control : 이 모듈이 동작하지 않는다면 PAM에게 무엇을 해야 할지 알려주는 것
    - requisite : 이 모듈을 이용하는 인증이 실패하는 경우, 즉시 인증을 거부하도록 함
    - required : 인증이 거부되기 전에 PAM이 이 서비스에 등록된 다른 모듈들을 요구해도 인증을 실패할 경우, 인증을 거부하도록 함
    - sufficient : 이전에 요청되어진 모듈이 실패하더라도 이 모듈에 의해 인증이 성공할 경우, PAM은 인증을 승인
  (3) module-path : PAM에게 어떤 모듈을 사용할 것인지, 어디에 위치하여 있는지 알려줌
  (4) module-arguments : 모듈에게 전달되는 인수
- **pam_tally2 명령어** 2015(1)
  - 사용자들이 인증의 실패에 대한 정보 출력 / 관리를 하는 명령어
  - 패스워드 인증 실패 횟수 초과 시 명령어 사용
    # pam_tally2 -u [계정 이름] -r

sudo 명령어 2014(2) 2018(1)

- 관리자가 사용자에게 특정 명령어에 대해 루트 권한으로 실행할 수 있게 해주는 명령어
- /etc/sudoers 파일에서 특정 명령어에 대해 sudo 명령어를 사용할 수 있도록 관리할 수 있다.

tripwire 명령어 2017(1)

- 원본 파일의 무결성을 점검하는 도구
- MD5, SHA, CRC-32 등의 다양한 암호화 함수를 제공
- 먼저 시스템에 존재하는 파일들에 대해 DB를 만들어 저장한 후, 생성된 DB와 비교하여 점검한다.

COPS(Computer Oracle and Password System) 2014(1)

- 리눅스 시스템에 대한 보안 문제를 알려주는 도구
- /etc/passwd, /etc/group, SUID 파일 등에 대한 점검을 수행
- 파일, 디렉터리 및 장치 파일에 대한 퍼미션 점검 수행

3. 시스템 백업

3-1. 백업 정책 수립 2014(1) 2014(2)

백업의 종류 2015(1)

- 데이 제로 백업(A Day-zero Backup) : 시스템을 설치한 후 사용자들이 시스템을 사용하기 전에 시스템을 백업하는 것
- 풀 백업(A Full Backup) : 주기적으로 시스템을 백업하는 것
- 증분 백업(An Incremental Backup) : 특정한 이벤트 후 또는 주기적으로 이전의 백업 후 변경된 파일들만 백업하는 것
- 단순 백업 : 첫 백업 때 풀 백업을 진행한 후, 그 다음부터 변경분 백업을 수행하는 것
- 다단계 백업(Multilevel Backup) : 큰 규모나 중요한 시스템의 백업을 할 때 수행하는 것

2-2. 백업 관련 명령어

cpio 명령어 2014(2) 2016(1) 2017(1)

- 많은 양의 데이터를 테이프 드라이브에 백업하기 위한 명령어
- 네트워크를 통한 백업 / 증분 백업을 지원하지 않는다.
- 주요 옵션
  - [-c] : ASCII 형태로 헤더 정보를 읽고 쓴다.
  - [-i] : 아카이브에서 파일 추출
  - [-o] : 아카이브 생성
  - [-v] : 진행 과정을 자세하게 출력

  EX) # ls *.conf | cpio -ocv > config.bak → 모든 .conf 파일을 백업

dump 명령어 2014(1) 2015(1)

- 파일 시스템 전체를 백업하는 명령어
- 점진적인 백업 기능 : 이전 백업 이후 변경된 파일들에 대해 백업 수행 가능
- 장점
  - 여러 개의 테이프에 백업 가능
  - 어떤 타입의 파일도 백업 가능
  - 증분 백업 가능
  - 결함을 가진 파일들도 다룰 수 있음
- 단점
  - 모든 파일 시스템은 개별적으로 dump 되어야 함 (각 파티션도 개별적으로 dump)
  - NFS 파일 시스템을 dump할 수 없다. (로컬 파일 시스템만 dump 가능)
- 주요 옵션
  - [-0~9] : 0 – 전체 백업 / 1,2,… - 부분 백업
  - [-f] : 지정한 파일명으로 백업 파일 생성

restore 명령어 2014(2) 2016(2) 2017(2)

- 복구를 위한 명령어
- 주요 옵션
  - [-i] : 대화형으로 복원 수행
  - [-f] : 장치 이름 지정
  - [-r] : 백업 대상 전체를 복원

기타 백업 관련 명령어

- tar 명령어를 이용한 백업 2015(2)c
  - [-g] : 증분 백업을 위한 옵션
  - 36 페이지 참고
- dd 명령어를 이용한 백업 2014(1) 2015(1) 2015(2) 2016(1) 2018(1)
  - 파티션이나 디스크 단위로 백업할 때 사용
  - 29 페이지 참고
- **rsync 명령어** 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)
  - 파일을 동기화하는 명령어. 원격지에 있는 파일들도 동기화할 수 있다.
  - cp, ftp, rcp 명령어보다 동기화 기능이 뛰어남
  - 형식 : # rsync [옵션] [동기화할 원본] [동기화될 위치]
  - [-a] : 아카이브 모드 (여러 옵션을 묶어 놓은 옵션)
  - [-v] : 진행 과정을 자세하게 출력
  - [-z] : 동기화 파일을 압축

3과목 네트워크 및 서비스의 활용

CH.7 네트워크 서비스

1. 웹 관련 서비스

1-1. 웹 관련 서비스의 이해

웹(Web) 2014(1)

- HTML(HyperText Markup Language) : 웹 페이지를 만들기 위한 마크업 언어
- HTTP 프로토콜이 사용되고 W3C에서 주관하여 개발
- 주요 웹 브라우저 종류 2015(1)
  - chromium : 구글 크롬으로 더 잘 알려져 있는 브라우저. 속도가 빠르고 다양한 플러그인이 존재
  - FireFox(파이어폭스) : 모질라(Mozilla) 재단에서 개발한 게코 엔진 기반의 오픈 소스 웹 브라우저 2018(1)
  - Opera(오페라) : 노르웨이에서 개발된 블링크 엔진 기반의 웹 브라우저
  - Safari(사파리) : 애플에서 개발한 Webkit 엔진 기반의 웹 브라우저. IOS를 위한 브라우저이다.

HTTP(HyperText Transfer Protocol)

- 웹에서 웹페이지를 가져오기 위한 프로토콜
- 요청 메소드(method) 2014(1) 2014(2) 2016(2)
  - GET 방식 : 요청 데이터에 대한 인수를 URL에 포함시켜 전송하는 방식
  - POST 방식 : HTTP Body 영역에 소켓을 이용하여 전송하는 방식
- 상태 코드 2014(1) 2014(2) 2015(2) 2016(1) 2016(2) 2017(2) 2018(1)

200	OK	요청 정상 처리
301	Moved Permanently	요청된 정보의 위치가 영구적으로 변경됨을 알림
302	Found	요청된 정보의 위치가 일시적으로 변경됨을 알림
400	Bad Request	잘못된 요청으로 처리 불가
403	Forbidden	접근 권한이 없음
404	Not Found	요청한 페이지 없음
500	Internal Server Error	내부 서버 오류

1-2. 웹 관련 서비스의 운영

웹 서버 종류 2015(1)

- Apache, IIS, Nginx 등이 있으며, 그 중 Apache가 가장 많이 쓰이고 있다.
- Nginx (엔진엑스) 2017(2)
  - 비동기 이벤트 기반 구조
  - 웹 서버, 리버스 프록시 및 메일 프록시 기능을 지원
- Apache (아파치) 2015(2)
  - 오픈 소스 소프트웨어 그룹인 아파치 소프트웨어 재단에서 개발하였고, 현재 가장 많이 쓰이고 있는 웹 서버
  - 리눅스, 유닉스 계열 뿐만 아니라 윈도우에서도 운용 가능

Apache 웹 서버

- 설치 방식 2017(1)
  - DSO(Dynamic Shared Object) : 처음 컴파일 이후 모듈 추가 시 다시 컴파일할 필요 없음
    - └ /usr/local/apache/modules 디렉터리에 모듈이 생성 2015(1)
  - SO(Static Object) : 모듈 추가 시 새로 컴파일 해주어야 함
- MPM(Multi-Processing Module) 2014(2) 2015(1) 2017(1)
  - 아파치가 받아들인 요청을 처리하기 위해 자식 프로세스에게 분배하는 방식

- Perfork 방식과 Worker 방식이 존재
  (1) Prework 방식 : 하나의 자식 프로세스 당 하나의 쓰레드를 사용하는 방법
  (2) Worker 방식 : 하나의 자식 프로세스 당 여러 개의 쓰레드를 사용하는 방법

• 소스 파일로 설치 시 configure 옵션 2017(2) 2018(1)
- [--prefix=DIRECTORY] : DIRECTORY에 지정한 디렉터리에 아파치를 설치한다
- [--enable-mods-shared=MOUDLE_LIST] : 지정 모듈을 동적 모듈로 컴파일
- [--enable-so] : DSO 기능 활성화

• 서버 환경 설정 (**httpd.conf**) 2015(2) 2015(2) 2016(1) 2016(1) 2016(1) 2016(2) 2017(1) 2017(1)

```
# vi httpd.conf
ServerType standalone                    // 서버 타입 설정 (Standalone / inetd)
ServerRoot "/etc/httpd"                  // 웹서버의 홈 디렉터리 설정
PidFile "run/httpd.pid"                  // 웹서버 가동시 자신의 PID를 기록할 파일
Listen 80                                // 웹서버 데몬의 리스닝 포트 설정
ServerToken Prod                         // 헤더의 Server 필드를 통해 제공할 정보 수준
ServerAdmin admin@shionista.com          // 웹문서 로딩 에러 발생시 에러 페이지에 표시
Timeout 600                              // 아무런 작업을 하지 않는 세션 유지 시간
MaxClients 50                            // 동시 연결 가능한 최대 클라이언트 수
KeepAlive On                             /* 클라이언트와 연결된 작업 프로세스가 계속해서 요청을
                                         처리할지(On), 새로운 프로세스가 처리할지(Off) 설정 */
MaxKeepAliveRequest 100                  // KeepAlive On 일 경우, 요청 작업의 최대 개수
KeepAliveTimeout 5                       // KeepAlive On 일 경우, 요청이 없는 세션 유지 시간
HostnameLookups Off                      // DNS에 대한 역검색 여부
DocumentRoot "/usr/local/apache/htdocs"  // 웹 문서(html, php 파일)의 기본 디렉터리 설정
LoadModule foo_module libexec/mod_foo.so
                                         //DSO를 사용하기 위해 모듈을 지시하는 설정

<Directory "/">                          // 특정 디렉터리를 제어할 때 사용하는 태그
        Options -Indexes -FollowSymLinks
                                         /* -Indexes : 디렉터리 인덱싱/리스팅 제한
                                         -FollowSymLinks : 심벌릭 링크 접근 제한 */
        DirectoryIndex index.php         // 해당 디렉터리 접근 시 가장 먼저 클라이언트에게  전달할 문서 지정
</Directory>

<IfModule mod_userdir.c>
        UserDir public_html                                  // 개인 홈페이지 디렉터리 설정
        AddType application/x-httpd-php .php .html           // php, html 확장자를 php 타입으로 실행
        AddType application/x-httpd-php-source .phps         // phps 확장자는 php 소스를 확인

</IfMoudle>
```

• 가상 호스트 환경 설정 파일 (**httpd-vhosts.conf**) 2017(2)
- 하나의 IP 주소에 여러 도메인을 사용하는 경우에 설정한다.

• **htpasswd 명령어** 2016(2) 2017(1)
- 특정 페이지를 제한하기 위해 인증을 요구하고 그 인증에 대한 관리를 하는 명령어
- 제한하고 싶은 아파치 내 디렉터리로 이동하여 **.htaccess 파일**을 만들어야 한다.
- [-c] : 지정한 디렉터리(페이지)에 사용자 계정 및 패스워드를 생성
  EX) # htpasswd -c /var/www/html/.htpasswd admin → /var/www/html/ 디렉터리에 대한 admin 계정, 패스워드를 생성
- 처음 추가 이후에는 [-c] 옵션 없이 추가한다.
- [-D] : 지정한 디렉터리(페이지)에 생성한 사용자 계정을 삭제

- **apachectl 명령어** 2017(2)
  - [start] : 아파치 서버 시작
  - [stop] : 아파치 서버 중지
  - [restart] : 아파치 서버 재시작
  - [graceful] : 현재 연결을 끊지 않은 상태로 재시작 (설정을 변경한 경우에 사용)

APM

- Apache, PHP, MySQL을 묶어 APM이라고 부름
- PHP(Power Hypertext Preprocessor) 2014(2) 2018(1)
  - 동적 웹페이지를 만들기 위한 프로그래밍 언어
  - PHP5 설치 시 생성되는 모듈 : libphp5.so
  - PHP 작성 시 <? ~ ?> 태그를 사용한다.
- MySQL 2016(2)
  - 오픈 소스의 관계형 데이터베이스 관리 시스템
  - **mysql_install_db** : 최초 설치 후 기본 DB 생성할 때 사용하는 명령어

1-3. 웹 관련 서비스의 운영

데이터베이스(Database, DB)

- 개념
  - 통합된 데이터(Integrated Data) : 원칙적으로 똑같은 데이터가 중복되어 있지 않지만, 의도적인 중복이 있을 수 있다.
  - 저장된 데이터(Stored Data)
  - 운영 데이터(Operational Data) : 어떤 기능을 수행하기 위해 반드시 유지해야 할 데이터가 존재
  - 공용 데이터(Shared Data) : 데이터가 하나의 어떤 것에 한정되어 지지 않고 여러 존재들이 해당 데이터를 공동으로 이용할 수 있다.

- MySQL
  - 많이 사용되는 RDBMS(관계형 데이터베이스 관리 시스템)
  - 오픈 소스 프로그램이며, 다양한 OS를 지원
- Oracle DB
  - 오라클 사에서 개발한 상업용 RDBMS
  - 유닉스/리눅스 환경에서 가장 많이 사용됨
  - 대량의 데이터 처리에 특화
- PostgreSQL
  - 객체-관계형 DB 관리 시스템(ORDBMS)
  - 특징으로는 객체 생성의 유연함, 상속 기능 등이 있다.
- SQLite
  - 임베디드 RDBMS
  - 일반적인 RDBMS에 비해 대량의 데이터 처리에는 적합하지 않음
  - 데이터를 저장하는 데에 하나의 파일만을 사용
- NoSQL (Not Only SQL) 2017(1)
  - 전통적인 RDBMS보다 덜 제한적인 일관성 모델을 이용.
  - RDBMS가 아닌 키(key)와 값(value) 형태로 저장되고, 키를 통해 데이터 관리/접근
  - 대용량 데이터 처리, 데이터 분산 처리, 빠른 읽기/쓰기 속도의 장점을 가지고 있다.
  - 종류 : Key-value 기반(Riak, …) / Document 기반(Mongo DB, …) / Big Table 기반(Hbase, …) / …
- Mongo DB 2014(1)
  - NoSQL이며 Document 기반의 DB이다.
  - 오픈 소스 프로그램이며, NoSQL의 가장 유명한 DB이다.
  - 각 객체의 구조가 뚜렷하며, RDBMS처럼 고정된 Schema 및 복잡한 JOIN이 없다.

CGI(Common Gateway Interface) 2014(2)
- 외부의 프로그램을 실행시켜 그 결과를 HTML로 돌려줌
- CGI 프로토콜이 단순하여 사용하기 쉽다.
- CGI 스크립트는 어떤 언어로도 코딩될 수 있으나 프로세스의 생성과 초기화에 많은 시간이 소요됨

Apache Tomcat 2015(1)
- 웹 서버와 연동하여 실행 가능한 자바 환경을 제공하여 자바 서버 페이지(JSP)와 자바 서블릿이 실행할 수 있는 환경을 제공하는 웹 어플리케이션 서버(WAS)
- 오픈 소스 프로그램

2. 파일 관련 서비스
2-1. SMB 서버 관리
삼바(Samba) 2014(1) 2018(1)
- Windows 클라이언트에서 유닉스 서버에 있는 파일/프린터를 공유할 수 있게 해주는 오픈 소스 소프트웨어
- SMB 프로토콜을 유닉스 환경에 구현한 프리웨어. GPL 라이선스를 따르고 있다.
  * SMB(Server Message Block) : 삼바 표준 프로토콜
  * CIFS(Common Internet File System) : 네트워크를 위한 SMB 프로토콜의 확장된 버전

SMB 서버 환경 설정 (**smb.conf**) 2014(1) 2014(2) 2015(1) 2015(2) 2016(2) 2017(1) 2017(2) 2018(1)

```
# vi smb.conf
[global]                                    // 서버 전체 환경 설정
        workgroup = fedora                  // 작업 그룹 이름
        hosts allow = 192.168.0. 192.168.1. 127.
        // 접근 가능한 호스트 정의
        // 여기서는 192.168.0.0/24 192.168.1.0/24 127.0.0.0/8 에 대해서만 접근을 허용
        security = user                     // 클라이언트가 삼바 서버에 접속할 때 인증 레벨을 부여
        /*
        - user : 기본적인 보안 정책. 아이디/패스워드 방식으로 서버에 접근
        - share : 아이디/패스워드 인증을 거치지 않음
        - server : 기본적으로 user 모드와 동일하지만 현재 시스템이 아닌 SMB 서버를 지원하는 다른 서버를 통해 처리하는 방식
        - domain : Windows NT 도메인에서 처리하는 방식
        */
        log file = /var/log/samba/%m.log // 로그 파일 위치 지정 (%m 은 삼바 클라이언트의 NetBIOS 이름이다.)


[homes]                                     // 각 계정의 사용자 홈 디렉터리 설정
        comment = SMB Home Directories
        browseable = no                     // 공유 이름을 브라우저에 표시 유무
        writable = yes                      // 쓰기 권한

[public]                                    // 공개적으로 접근 가능한 디렉터리를 설정
        path = /home/samba                  // 공유할 디렉터리 지정
        valid users = fedora test33         // 공유 디렉터리에 로그인할 수 있는 사용자 지정
        writable = yes
        write list = @staff                 // 쓰기가 가능한 사용자 지정
                                            // 여기서는 staff 그룹(@staff)의 사용자만 쓰기를 허용
```

관련 명령어/파일

- **testparm 명령어** : smb.conf 파일을 점검하는 명령어 2014(2) 2016(2) 2017(2)
- **smbclient 명령어** 2015(1) 2016(1) 2017(1) 2018(1)
  - windows 공유 폴더에 접근하기 위한 명령어
  - 형식 : # smbclient //[HOSTNAME | IP]/[SHARE_NAME(디렉터리)]
    # smbclient ₩₩₩₩[HOSTNAME | IP]₩₩[SHARE_NAME(디렉터리)]
  - [-U] : 로그인할 SMB 사용자 이름 지정
    EX) # smbclient //192.168.x.x/share -U smbuser → smbuser 계정으로 192.168.x.x 호스트의 share 폴더에 접근
  - [-L] : 지정한 윈도우 host의 공유 폴더를 확인
- **smbpasswd 명령어** 2015(2)
  - 삼바 접근을 위한 계정을 관리하는 명령어
  - [-a] : 계정 추가
  - [-d] : (disable) 지정한 계정을 사용하지 못하도록 설정
  - [-e] : (enable) 사용 중지된 계정을 다시 사용할 수 있도록 설정
  - [-x] : 지정한 계정을 삭제
  - [-n] : 지정한 계정의 패스워드를 삭제 (패스워드 없이 로그인 가능)
  - 옵션 없이 사용 시 지정한 계정의 패스워드를 변경
- **pdbedit 명령어** 2016(1)
  - [-L] : 삼바에 등록된 모든 계정의 정보를 출력 (USERNAME : UID : GROUP)

2-2. NFS 서버 관리

NFS(Network File System)

- 네트워크 파일 시스템. 네트워크 상의 다른 컴퓨터에 있는 파일이나 파일 시스템을 공유하기 위한 시스템
- RPC(원격 프로시저 호출)을 사용
- 서버 환경 설정 (**/etc/exports 파일**) 2014(1) 2014(2) 2015(1) 2015(2) 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)
  - 형식 : [마운트 디렉터리] [마운트 허용 클라이언트] ([옵션])
  - **ro** : 공유된 자원을 읽기만 가능
  - **rw** : 공유된 자원을 읽기/쓰기 가능
  - **root_squash** : root 권한으로 접근 시 nobody 사용자로 변경하여 마운트
  - **no_root_squash** : root 권한으로 접근 가능
  - **all_squash** : 모든 사용자의 권한을 nobody 사용자로 변경하여 마운트
  - **no_all_squash** : root를 제외한 사용자에 대해 같은 UID를 가진 사용자에게 동일한 권한으로 마운트
  - **sync** : 파일 시스템 내 데이터 변동 시 즉시 동기화 처리
  - **async** : 파일 시스템 내 데이터 변동 시 비동기적으로 처리
  EX) /home/nfs 192.168.10. (no_root_squash) → /home/nfs 디렉터리에 192.168.10.0/24 네트워크 내 클라이언트가 root 권한으로 마운트 가능 하도록 설정

관련 명령어

- **showmount 명령어** 2014(1) 2014(2) 2016(2)
  - NFS/NIS의 마운트 정보를 출력
  - [-a] : 모든 원격 마운트 정보 출력
  - [-e] : export된(공유되거나 내보낸) 디렉터리의 정보 출력
    EX) # showmount -e 192.168.10.9 → 192.168.10.9의 NFS 서버의 export된 정보 출력
- **exportfs 명령어** 2015(2) 2017(2)
  - NFS 서버를 다시 시작하지 않고 공유정보를 수정하거나 출력하는 명령어
  - [-a] : /etc/exports 파일을 읽음
  - [-r] : /etc/exports 파일을 다시 읽음
  - [-v] : 현재 공유 목록을 출력

```
[root@localhost /]# exportfs
/smb_test       169.254.231.130
[root@localhost /]#
[root@localhost /]# exportfs  -v
/smb_test       169.254.231.130(ro,sync,wdelay,hide,no_subtree_check,sec=sys,root_squash,no_all_squash)
[root@localhost /]#
```

- **rpcinfo 명령어** 2015(1)
  - RPC 정보를 출력하는 명령어

2-2. FTP 서버 관리

/(File Transfer Protocol)
- 네트워크상의 컴퓨터들 간에 파일을 교환하기 위한 프로토콜
- 포트번호 : TCP/20(data), TCP/21(control) 2016(2)
- 종류 : vsftpd, proftpd, …

vsftpd(Very Secure FTP Daemon)
- 보안이 강화된 FTP 데몬. 많은 배포판에서 기본 FTP로 채택된 데몬이다.
- 서버 환경 설정 (**/etc/vsftpd/vsftpd.conf 파일**) 2014(1) 2014(2) 2015(1) 2015(2) 2017(1) 2017(2) 2018(1)

# vi vsftpd.conf	
anonymous_enable=YES	// 익명 사용자의 접속 허용 유무
local_umask=022	// 로컬 계정 사용자들의 umask 값 설정
idel_session_timeout=1200	// idle(유휴) 시간에 대한 타임아웃 값 설정
max_clients=10	// 최대 FTP 접속 클라이언트 수 설정
max_per_ip=2	// 한 호스트가 동시에 접속할 수 있는 IP 수 설정
tcp_wrappers=YES	// tcp_wrapper 적용 여부 설정
chroot_list_enable=YES	/* 특정 사용자에 대해 자신의 홈 디렉터리를 루트 디렉터리로 인식하도록 설정
chroot_list_file=/etc/vsftpd.chroot.list	// chroot() 기능 관련 설정을 적용할 사용자를 등록한 파일 지정
chroot_local_user=YES	/* chroot() 기능을 사용하여 자신의 홈 디렉터리를 상위 디렉터리로 이동하지 못하게 설정(YES) */
ls_recurse_enable=YSE	// FTP 서버 내에서 ls 명령어의 [-R] 옵션 허용 유무
listen=YES	// YES = standalone 방식 / NO = xinetd 방식
listen_port=21	// standalone 방식일 경우, 리스닝 포트번호 설정

- **/etc/vsftpd/ftpuser 파일** 2016(1)
  - ftp 서버 접근을 제한할 계정 목록을 저장

proftpd

- 서버 환경 설정 (**/etc/proftpd.conf 파일**) 2017(1)

```
# vi proftpd.conf
ServerName ":::: TEST ProFTP ::::"      // FTP 서버 이름을 설정
ServerType standalone                   // 데몬 구동 방식 설정(standalone/xinetd)
Port 21                                 // 리스닝 포트번호 설정
Umask 022                               // 로컬 계정 사용자들의 umask 값 설정
MaxInstances 10                         // 최대 FTP 접속 클라이언트 수 설정
User nobody                             // FTP 실행 시 사용할 사용자 명 설정
Group Nobody                            // FTP 실행 시 사용할 그룹 명 설정
TimeoutIdle 1200                        // idle(유휴) 시간에 대한 타임아웃 값 설정

<Directory /*>                          // 지정한 디렉터리에 대한 설정 (여기서는 루트 아래 전체)
        AllowOverwrite on               // 사용자가 전송하는 파일이 서버에 있을 때 Overwrite 허용 유무
</Directory>

<Anonymous ~ftp>                        // 익명 사용자를 위한 디렉터리에 대한 설정
        RequireValidShell off           /* /etc/shells 파일에 정의되지 않은 쉘을 사용하는 사용자
                                        접근 허용 여부 설정 (off - 익명 사용자 접속 가능) */
        MaxClients 5                    // 익명 사용자에 대한 최대 접속 클라이언트 수 설정
</Anonymous>
```

3. 메일 관련 서비스

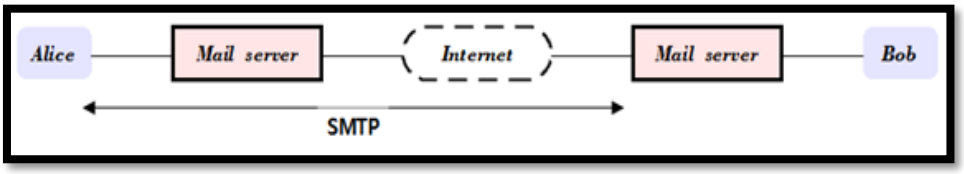
3-1. 메일 관련 서비스의 이해

메일 서비스

- 기본 컴포넌트 3가지 2016(1) 2017(1)
  - MUA(Mail User Agent) : 사용자들이 메일을 송/수신하기 위한 클라이언트 에이전트
  - MTA(Mail Transfer Agent) : 메일 서버로 메일을 전송하기 위한 서버/클라이언트 에이전트
  - MDA(Mail Delivery Agent) : 수신된 메일을 해당 사용자에게 전달해주는 에이전트
- 포워딩(forwarding) 방법
  - virtusertable 파일 설정을 통한 포워딩
  - aliases 파일 설정을 통한 포워딩
  - .forward 파일을 통한 포워딩 : 일반 계정 사용자가 자신의 홈 디렉터리에 만들어 설정 2015(2)

메일 관련 프로토콜 2014(1) 2015(1) 2015(2)

- SMTP(Simple Mail Transfer Protocol) 2014(2) 2016(2)
  - MTA 클라이언트와 서버를 구성하는 프로토콜 (TCP/25)
  - 송신자와 송신자의 메일 서버 사이, 메일 서버와 메일 서버 사이에서 사용된다.



- POP3(Post Office Protocol)
  - 메일 서버로부터 메일을 수신하기 위한 서버/클라이언트 프로토콜 (TCP/110)
  - 메일 서버로부터 메일을 가져온 후 서버에서 메일을 삭제함
- IMAP4(Internet Mail Access Protocol)
  - POP3와 비슷한 기능이지만 더 많은 기능을 포함하고 복잡함 (TCP/143)
  - 메일 서버로부터 메일을 가져와도 서버에 메일이 유지됨.

메일 관련 프로그램

- MTA 프로그램 2015(1) 2015(2) 2017(2) 2018(1)
  - Sendmail : SMTP를 기반으로 한 메일 전송 프로그램. 리눅스 배포판에 기본적으로 설치됨
  - Qmail : Sendmail에 비교해 보안이 뛰어나며 모듈 방식으로 편리한 기능
  - Postfix : IBM 라이선스를 따르는 오픈소스 프로그램
- MUA 프로그램
  - Thunderbird(썬더버드) : 모질라에서 만든 메일 클라이언트 프로그램 2017(2)
  - Evolution(에볼루션) : GNOME의 메일 관리 프로그램
- MDA 프로그램
  - Procmail : 메일을 필터링하는 기본적인 프로그램 2016(2)
  - SpamAssassin : 아파치 재단에서 개발한 메일 필터링 프로그램. 펄(perl)로 제작됨.
- 보안 프로그램 2016(1)
  - PGP(Pretty Good Privacy) : PEM에 비해 보안성은 떨어지지만 구현이 쉽다.
  - PEM(Privacy Enhanced Mail) : 높은 보안성, 복잡한 구현 방식으로 잘 쓰이지 않음
- 기타 관련 프로그램
  - dovecot : POP3와 IMAP4 역할을 수행하는 프로그램 2016(1)

sendmail 환경설정 파일 2014(2)

- **/etc/mail/sendmail.cf 파일**
  - Sendmail의 핵심. 메일 송/수신 시 이 파일을 해석하여 실행한다.
  - **m4 명령어**로 생성할 수 있다. 2014(1) 2015(2) 2017(2) 2018(1)
    EX) # m4 sendmail.mc > snedmail.cf
- sendmail.cf 파일 항목 2015(1) 2017(1) 2017(2)
  - **Cw** : 호스트 지정
  - **Fw** : 파일 지정
  - **Dj** : 특정 도메인을 강제 지정

- **/etc/mail/local-host-names 파일** 2014(1) 2016(1)
  - 메일 서버에서 사용하는 호스트(도메인)을 등록하는 파일

- **/etc/mail/access 파일**
  - 각종 접근 제어 설정이 저장되는 파일
  - **makemap 명령어** : /etc/mail/access 파일 편집 후에 DB 파일(access.db)을 만드는 명령어 2014(2) 2017(1)
    EX) # makemap hash /etc/mail/access < /etc/mail/access
- access 파일 항목 2016(2) 2017(1)
  - RELAY : relay 허용 * Relay : 중계
  - OK : 무조건 허용
  - REJECT : relay 차단 (주로 스팸 서버의 IP를 차단)
  - DISCARD : relay 없이 폐기 (어떠한 답신도 보내지 않음)
  - 501 : 지정된 메일 주소와 일치하는 모든 메일의 수신 차단
- access 파일 예제 2014(2) 2018(1)

# vi access		
Connect:192.168.10.9	OK	/* 192.168.10.9 호스트로 접속하는 클라이언트의 메일 허용 */
Connect:localhost	RELAY	// localhost로 접속하는 클라이언트의 RELAY 허용
From:add@spam.com	REJECT	/* add@spam.com에서(발신) 오는 메일을 거절하고 거절 답신 보냄 */
From:root@spam.co.kr	DISCARD	/* root@spam.co.kr에서(발신) 오는 메일을 거절하고 거절 답신을 보내지 않음 */
To:log@shionista.com	OK	// log@shionista.com으로(수신) 오는 메일을 허용

• **/etc/aliases 파일** 2014(2) 2015(1) 2016(2) 2017(1)
- 특정 ID로 들어오는 메일을 여러 호스트에게 전달할 때 사용하는 파일 (작은 규모의 메일링 리스트)
- 사용자가 다른 메일 계정 (별칭)을 사용할 수 있도록 할 수 있다.
- **newaliases 명령어** : /etc/aliases 파일의 변동 사항을 적용 2014(1) 2016(2)
  * sendmail -bi 명령어와 같은 기능 2017(2) 2018(1)

• **/etc/mail/virtusertable 파일** 2014(1) 2015(2) 2018(1)
- 가상 메일 사용자의 설정이 저장되는 파일
- access 파일과 마찬가지로 **makemap hash 명령어**로 DB 파일을 만들어 주어야 함

EX) webmaster@server.shionista.com admin → 해당 메일 주소로 오는 메일을 admin 계정으로 수신
webmaster@test.shionista.com test → 해당 메일 주소로 오는 메일을 test 계정으로 수신

메일 관련 명령어

• **mailq 명령어** 2015(1) 2016(1)
- 메일 큐 목록(**/var/spool/mqueue**) 출력 (sendmail -bp 명령어와 같은 기능)
- [-v] : 자세하게 출력
- [-Ac] : /etc/spool/submit.cf 에 지정된 메일 큐 목록(/var/spool/clientmqueue)을 출력

4. 기타 서비스

4-1. 슈퍼 데몬 관리

xinetd 개념

• 기존의 inetd를 대체하는 보안이 강화된 오픈 소스 슈퍼 데몬
• TCP Wrapper 와 유사한 접근 제어 기능을 갖는다.
• RPC 요청에 대한 지원이 미비하지만, 포트맵(Portmap)으로 해결 가능
• standalone 과의 비교 : 12 페이지 참고 2014(1) 2015(1)
• 주요 기능 2016(1)
- DoS 공격에 대한 효과적인 억제
- 로그 파일 크기 제한
- IP 주소 당 동시 접속 수 제한
- TCP/UDP 및 RPC 서비스들에 대한 접근제어
• 설정 파일 구조 (**/etc/xinetd.d/***) 2014(1) 2014(2) 2015(1) 2015(2) 2015(2) 2016(2) 2017(1) 2017(1) 2017(2) 2018(1)

vi /etc/xinetd.d/telnet

```
service telnet {
        disable = no                            // 서비스 사용 설정
        socket_type = stream                    // tcp = stream , udp = dgram
        wait = no                               // 요청을 받은 후 즉시 다음 요청 처리(no)
        user = root                             // 실행할 사용자 권한 설정
        server = /usr/sbin/in.telnetd           // 서비스 실행 파일 경로
        log_type = FILE /var/log/xinetd.log     // 로그를 기록할 파일 경로 (FILE 선택자 사용 시)
        log_on_failure += USERID                // 로그인 실패 시 로그에 기록할 내용
        no_access = 10.0.0.0/8                  // 접속을 거부할 IP 대역
        only_from = 192.168.10.0/24             // 접속을 허용할 IP 대역
        cps = 10 30                             /* 들어오는 접속 수 제한. 지정한 접속 수 초과할 시
                                                지정한 시간 동안 서비스가 비활성화됨 */
        instances = 5                           // 동시에 작동할 수 있는 최대 개수
        access_time = 08:00-17:00               // 접속 허용 시간대
}
```

TCP Wrapper 2014(2) 2016(2) 2018(1)

- 슈퍼 데몬에 의해 수행되는 호스트 기반의 네트워킹 ACL(Access Control List) 시스템
- **/etc/hosts.allow 파일**과 **/etc/hosts.deny 파일**에 정의된 호스트 정보를 기준으로 접근통제
- 접근 통제 파일 참조 순서 : /etc/hosts.allow → /etc/hosts.deny → 두 파일에 없으면 모든 접근 허용

4-2. DNS 관리

DNS 개념 2015(2) 2016(1)

- DNS(Domain Name System) : 사람이 식별하기 쉬운 도메인 이름을 컴퓨터가 식별하기 위한 네트워크 주소(IP) 간의 변환을 수행하기 위한 시스템 (TCP,UDP/53)
- 데몬 이름 : named 2015(1)
- 기본 DNS 서버 주소에 대한 정보는 **/etc/resolv.conf 파일**에 저장되어 있다. 2014(1)

- Recursive 네임 서버 (Cache DNS 서버) 2014(2) 2016(2) 2017(1)
  - 서버에 질의가 들어오면 자신의 캐시에 저장된 정보 또는 반복적 질의를 통해 그 결과를 호스트에게 응답해주는 네임서버
  - 반복적 질의(Iterative Query) : Recursive 네임 서버가 각 네임서버(Authoritative 네임서버)로 질의하는 방식
  - 재귀적 질의(Recursive Query) : 호스트가 Recursive 네임서버로 질의할 때 사용되는 방식
- Authoritative 네임서버 (DNS 서버)
  - 특정 도메인에 대한 정보를 관리하면서 해당 도메인에 대한 질의에만 응답해주는 네임서버
  - 존(Zone) : 네임서버가 관리하는 도메인 영역
  - 존 파일(Zone File) : 관리 도메인에 대한 정보를 담고 있는 파일. 이 파일을 읽어 질의에 응답한다.

- DNS 서버는 마스터(master) 네임서버와 슬레이브(Slave) 네임서버로 구분된다.
  * master - slave OR Primary - Secondary
  - 존 전송(Zone Transfer) : 마스터에 있는 원본 존 데이터를 슬레이브가 동기화 하는 작업

DNS 서버 설정

- DNS 레코드 종류 2014(1) 2015(2) 2018(1)

A (Address)	- 도메인에 대한 IPv4 주소 질의 - **AAAA** : IPv6 주소에 대한 질의
ANY	- 도메인에 대한 모든 레코드 질의 - 요청 대비 응답이 큰 레코드 타입
TXT	도메인에 대한 텍스트 정보 질의
SOA (Start Of Authority)	존(Zone)의 기본 속성 정보 질의
MX (Mail Exchanger)	도메인의 메일서버 질의
NS (Name Server)	도메인의 네임서버 질의
PTR (Pointer)	- IP에 대한 도메인 정보 질의 - Reverse Zone 파일에 설정된 PTR 레코드 질의
CNAME (Canonical Name)	호스트의 별명을 정의

• 네임서버 환경 설정 **(/etc/named.conf)** 2014(1) 2014(2) 2015(1) 2015(2) 2016(1) 2016(1) 2016(2) 2016(2) 2017(1) 2017(1) 2017(2) 2017(2) 2018(1)

vi /var/named.conf

```
acl "allow_user" {192.168.10.50; 192.168.10.51; 192.168.10.52; };

options {
        directory   "/var/named";          // zone 파일들의 위치 지정
        allow-query         {any};         /* 네임서버에 질의할 수 있는 대상 지정
#       allow-query         {localhost};   (any : 전체 / localhost : 내부 네트워크) */
#       allow-query         {allow_user};
        recursive   no;                    // 재귀적 질의 허용 여부
        forward     only;                  /* 자신에게 들어온 질의 요청을 다른 서버로
        forwarders          {192.168.11.10 }; 넘기는 옵션 */
};

zone "shionista.com" IN{
        type        master;                // master 네임서버임을 의미
        file        "shionista.com.db";    /* 관리하는 도메인의 리소스 레코드 정보를
};                                         담고 있는 Zone 파일명을 의미 */

zone "10.168.192.in-addr.arpa" IN{         /* 리버스 도메인. in-addr.arpa 도메인의
        type        master;                하위 도메인으로 구성한다. */
        file        "shionista.com.db.rev";
};

zone "." IN {
        type        hint;                  // 루트 도메인 type은 hint로 지정한다.
        file        "named.ca";
};
```

- 주석 처리 구문은 "#", "//", "/* ~ */" 을 사용 2015(2) 2017(2)

• 존파일 설정 **(/var/named/*)** 2014(1) 2014(2) 2014(2) 2015(1) 2016(1) 2016(2) 2018(1) 2018(1)

vi /var/named/shionista.com.db

```
/* [host_name] [TTL] [class] [record_type] [data] 으로 구성 */
$1800
@       IN      SOA     ns.shionista.com.   master.shionista.com.(  /* 존의 속성을 정의하는 부분
                                                                    뒷부분은 관리자 메일 주소 */
                        2018082517          ;serial                 // 존 파일 수정 시간
                        21600               ;refresh                // master - slave 동기화 시간
                        900                 ;retry                  // 동기화 실패 시 재시도 대기
                        604800              ;expire                 // 유효기간
                        43200               ;minimum                // TTL과 같은 의미
                        )
        IN      NS      ns.shionista.com.
        IN      MX      10 mx1.shionista.com.       /* 앞에 숫자(10,20)은 우선순위를
        IN      MX      20 mx2.shionista.com.       나타낸다. 낮을수록 우선순위가 높다. */
NS      IN      A       192.168.10.5
MX      IN      A       192.168.10.10
www     600     IN      A       192.168.10.3
www1    IN      CNAME   www
www2    IN      CNAME   www
```

• 역(reverse) 존 파일 설정 2015(1)

vi /var/named/shionista.com.db.rev

```
                                ~~~생략~~~
6           IN          PTR     ns.shionista.com
7           IN          PTR     ns.shionista.com
```

DNS 관련 명령어 2014(1)

• **named-checkconf 명령어** : 네임서버 환경 설정 파일(/etc/named.conf)을 검사 2014(2) 2016(1)

• **named-checkzone 명령어** : 존 파일(/var/named/)을 검사 2017(2) 2018(1)

• **rndc 명령어**
- 네임서버 제어 명령어 (구 ndc 명령어)
- [stop] : 네임서버 중지
- [status] : 네임서버 상태 정보 출력
- [reload] : 존 파일을 다시 로드
- [flush] : 네임서버의 캐시 제거

• **nslookup 명령어** 2014(2)
- DNS 관련된 각종 정보를 확인할 수 있는 명령어
- [server IP_ADDRESS] : 질의할 DNS 서버 지정
- [set type = RECORD] : 질의할 레코드 유형 지정

4-3. 프록시 관리

프록시(proxy) 개념 2014(2) 2016(2) 2017(1)

• 프록시 서버 : 클라이언트가 자신을 통해 다른 네트워크 서비스에 간접적으로 접속할 수 있게 해주는 서버

• 서버는 프록시 서버를 클라이언트로 인식하고 클라이언트는 프록시 서버를 서버로 인식하게 된다.

• 사용 목적
- 접근 정책을 만들어 유해/악성 사이트로의 접근 제한 (방화벽 기능)
- 익명으로 페이지에 접근 (보안성)
- 캐시를 사용하여 인터넷을 빠르게 사용하기 위해
- 우회 목적

squid 프록시 서버 2016(1)

• 유닉스 계열에서 작동하며 GPL 라이선스를 따르는 오픈 소스 프록시 서버 소프트웨어

• 서버 환경 설정 (**squid.conf 파일**) 2014(1) 2015(2) 2017(2) 2018(1)

vi httpd.conf

```
http_port           3128                        // 서비스 포트 지정
cache_mem           8MB                         // 제공할 캐시 메모리 용량 지정
cache_dir           /var/spool/squid 100 16 256 /* 캐시가 저장되는 경로, 크기 지정
                                                (크기 100MB, 하위 16개 그 다음 단계
                                                하위 256개의 디렉터리까지) */
cache_access_log    /var/log/squid/access.log   // 캐시에 접속한 로그를 기록할 파일 지정
acl localhost src 192.168.10.0/255.255.255.0    // 별칭 지정
http_access allow localhost                     // 접근 제어 부분
http_access deny all
```

4-4. NIS 관리

NIS 개념 2014(1) 2015(1)

• NIS(Network Information System) : Sun Microsystems 사의 클라이언트 서버 디렉터리 서비스 프로토콜

• 시스템의 중요한 파일의 정보들을 네트워크상에 있는 모든 호스트에 배포하는데 사용하는 데이터베이스 액세스 기능을 제공하며 RPC 기반이다.

• 하나의 NIS 서버에서 다수의 서버에 대한 사용자 인증 기능을 수행할 수 있다.

• 초기에는 YP(Yellow Pages)라고 불림

• NIS/NIS+는 보안에 취약(root 권한 획득)하지만 꼭 사용해야 할 경우 덜 취약한 NIS+을 사용하는 것이 바람직

• NFS와 연동하여 효율적인 네트워크 서버 환경을 구축할 수 있다. 2016(2)

• 관련 서비스에는 ssh, telnet, samba 등이 있다. (dns는 X) 2016(1) 2017(2)

NIS 관련 데몬 2014(1) 2014(2) 2015(2) 2016(1) 2016(2) 2017(1)

• NIS는 RPC를 사용하기 때문에 RPC 호출을 처리하는 데몬을 반드시 구동시켜야 함

• 초기에는 포트맵(portmap)이 그 역할을 수행하였으나 현재는 **rpcbind 데몬**을 사용

• **ypserv** : NIS 서버 데몬

• **ypbind** : NIS 클라이언트 데몬

• **ypxfrd** : NIS 서버 데몬. NIS 서버와 클라이언트 간의 매핑 속도를 향상시켜주는 데몬 2017(2)

• **rpc.yppasswd** : NIS 암호 업데이트 데몬

• **rpc.ypupdated** : 맵을 수정하는 데몬

NIS 관련 파일

• **passwd.byname** : 사용자 계정 관련 정보가 저장된 파일 2014(2) 2016(2)

• **hosts.byname** : 호스트 관련 정보가 저장된 파일

• **/var/yp** : 관련 맵 파일이 위치하는 디렉터리 2014(1) 2015(2) 2016(2)
  - 설정 내용 수정 시 이 디렉터리에서 make 명령어로 갱신
  - 맵 파일은 /var/yp/[도메인명] 으로 생성됨

• **/etc/yp.conf** : 환경 설정 파일 2015(2) 2016(1)

```
# vi yp.conf
server nis.shionista.com
ypserver nis.shionista.com
domain shionista.com
```

• **/etc/sysconfig/network** : 네트워크 관련 항목들을 설정하는 파일
  - NIS 도메인 이름을 부팅 시에도 적용하기 위해 이 파일에 도메인을 등록
  - "NISDOMAIN=[도메인 이름]"을 추가 해 주면 된다.

NIS 관련 명령어

• **ypcat 명령어** 2015(1)
  - NIS 클라이언트에서 사용하는 명령어. NIS 서버의 DB인 맵파일의 내용을 출력
  - passwd.byname 파일 내용을 확인할 때 사용
    EX) # ypcat passwd.byname

• **ypwhich 명령어** 2014(2) 2015(2) 2017(2)
  - NIS 클라이언트에서 사용하는 명령어. NIS 서버의 이름과 관련 맵파일을 출력

4-4(추가). 책에 없는 LDAP

LDAP(Lightweight Directory Access Protocol) 개념 2014(1) 2014(2) 2017(2) 2017(2)

- IP 프로토콜 기반으로 디렉터리 서비스를 조회하고 수정하는 프로토콜
- X.500 DAP를 기반으로 경량화(Lightweight)하여 만들어짐
- NIS와 같은 네트워크 기반 인증 관련 프로토콜(NIS, LDAP, Active Directory, …)이다. 2015(1) 2016(1) 2018(1)
- 트리 구조로 구성되어 있으며 논리나 계급등의 기준으로 조직화되어 있음

LDAP 엔트리 2014(1) 2015(1) 2015(2) 2016(1) 2016(2) 2017(1) 2018(1)

- **O**(Organization) : 최상위 조직 (회사 이름)
- **OU**(Organizational Unit) : 그룹 (부서 이름)
- **C**(Country) : 국가
- **ST**(State or Province) : 주 (우리나라에서는 도)
- **CN**(Common Name) : 이름
- **SN**(Sir Name) : 이름에서 성
- **DC**(Domain Component) : 도메인
- **DN**(Distinguished Name) : 고유 이름

4-5. DHCP 관리

DHCP(Dynamic Host Configuration Protocol) 개념

- 호스트의 IP 주소를 동적으로 할당해주기 위한 프로토콜
- IP를 자동으로 할당해주기 때문에 편리하고 IP 충돌 예방이 가능
- ARP 프로토콜 사용 2017(1)

DHCP 환경 설정 (**/etc/dhcp/dhcpd.conf 파일**) 2014(1) 2014(2) 2015(1) 2015(2) 2016(1) 2016(2) 2017(1) 2017(2) 2018(1)

- **range** : DHCP 서버가 호스트에게 할당해 줄 수 있는 IP 범위 지정

```
~ range 100.100.100.50 100.100.100.100
```

- **option routers** : 게이트웨이 주소 지정

```
~ option routers 100.100.100.1
```

- **option subnet-mask** : 서브넷 마스크 지정

```
~ option subnet-mask 255.255.255.0
```

- **fixed-address** : 특정 호스트에게 할당할 고정 IP 지정

```
~ host shionista {
~   hardware ethernet AA:BB:CC:DD:EE:FF
~   fixed-address 100.100.100.51
~ }
```

5. 책에 없는 네트워크 관련 서비스

5-1. 가상화 기술

가상화 개념

- 가상화 기능 2015(2) 2016(1) 2018(1)
  - 공유(Sharing) : 여러 개의 가상 자원들이 하나의 동일한 물리적 자원과 연결됨
  - 단일화(pooling) : 여러 개의 물리적 자원으로 하나의 가상 자원 생성하여 전체 용량을 증가시키고 활용과 관리를 단순화
  - 에뮬레이션(emulation) : 물리적 자원에 없는 기능을 처음부터 있던 것처럼 가상 자원에서 구현할 수 있다.
  - 캡슐화(encapsulation)/절연(Insulation) : 가상 자원들과 물리적 자원들 간의 상호 매핑은 가상 자원들 또는 사용자들에게 영향을 주지 않고 물리적 자원을 교체할 수 있다.

- 가상화 효과 2016(2)
  - 자원 활용률 증가 : 물리적 자원과 자원 풀에 대한 동적인 공유 가능
  - 향상된 보안 : 분리/격리로 데이터와 서비스에 대하여 통제되고 안전한 접근 제공
  - 가용성 증가 : 사용자에게 영향을 주지 않으면서 물리적 자원의 변경 가능
  - 확장성 증가 : 물리적 자원의 구성 변경 없이 가상화 자원의 확장 가능
  - 향상된 프로비저닝 : 물리적 단위에 상관없이 가상화 자원을 빠르게 할당/제공 가능 2015(2) 2016(1)

전가상화와 반가상화 2015(1)

- 전가상화 2018(1)
  - 전체 하드웨어를 모두 가상화
  - 게스트 OS의 수정 없이 다양한 OS를 지원
  - CPU의 VT(Virtual Technology) 기술 사용으로 성능 저하 발생
  - 대표적인 기술로 KVM, VMware 등이 있다.
- 반가상화 2014(1)
  - 전체 하드웨어를 가상화 하지 않고, 게스트 OS 커널의 일부 수정 필요
  - 게스트 OS가 직접 하드웨어를 제어하지 않고 하이퍼바이저(Hypervisor)를 호출하여 제어
    * 하이퍼바이저 : 다수의 OS를 하나의 컴퓨터 시스템에서 가동할 수 있게 하는 소프트웨어
  - 별도의 툴이 필요
  - 대표적인 기술로 XEN 등이 있다.

XEN 2014(1) 2017(1) 2017(2) 2018(1)

- 대표적인 전가상화/반가상화 오픈소스 하이퍼바이저
- 전가상화의 느린 속도를 개선하기 위해 개발되었다.
- 베어메탈(bare metal) 방식 : OS가 설치되어 있지 않는 하드웨어
- 도메인(domain)
  - XEN 환경에서 동작하고 있는 가상머신을 의미
  - XEN을 사용하기 위해 도메인 0 이라는 특수한 도메인을 사용
  - 도메인 0 : 다른 도메인을 제어할 수 있는 특권을 가짐
  - 도메인 U : 특권이 없는 나머지 도메인들을 의미
- 전가상화 구성 시에는 QEMU 기반으로 동작
- CPU 전가상화/반가상화를 지원
- **xm 명령어**를 통해 도메인을 관리/제어할 수 있음 2015(1)

KVM (Kernel-based Virtual Machine) 2014(2) 2017(1) 2017(2)

- 리눅스 커널을 하이퍼바이저로 변환하기 위한 전가상화 방식의 가상머신
- 리눅스 커널 2.6.20부터 커널 모듈에 포함되어 있는 기능
- 사용하는 CPU에서 HVM(Hardware Virtual Machine) 기능을 제공해야 함
- QEMU
  - KVM을 사용하기 위한 에뮬레이터
  - 반가상화 방식의 하이퍼바이저이기 때문에 KVM은 반가상화, 전가상화 모두 지원한다고 할 수 있다.
- CPU의 반가상화는 지원하지 않는다.
- 게스트 OS의 가상화 디스크 이미지가 저장되는 디렉터리 : **/var/lib/libvirt/images** 2014(2)

Docker 2015(1) 2016(2) 2017(1)

- 리눅스 기반의 컨테이너 가상화 기술
- 반가상화 방식보다 더 경량화된 방식
- 하이퍼바이저나 게스트 OS 없이 서버 운영에 필요한 프로그램과 라이브러리만 격리해서 설치/사용 가능
  EX) 하나의 물리적 서버에 다수의 웹 서버 운영

기타 가상화 관련 내용

- 다양한 하이퍼바이저를 통합 관리하기 위한 플랫폼 2015(2) 2017(1)
  - OpenStack, CloudStack, OpenNebula 등
- CPU의 가상화 지원 여부는 **/proc/cpuinfo 파일**을 참조 2015(2) 2016(1)
- 관련 명령어
  - **virt-top 명령어** : 가상머신별 자원 사용률을 실시간으로 모니터링하는 명령어 2014(2) 2015(1)
  - **virtsh 명령어** : CUI 기반의 가상머신을 관리하는 명령어 (생성, 재시작, 종료 등) 2014(1) 2015(2) 2016(1) 2017(2)
  - **virt-manager 명령어** : GUI 기반의 가상머신 관리자 프로그램 2014(1) 2016(2) 2018(1)
- **libvirtd 데몬** : 가상화 기술을 위한 데몬 2014(1) 2015(1) 2016(2)

5-2. NTP 관리

NTP 개념 2018(1)

- NTP(Network Time Protocol) : 네트워크를 통해 네트워크 상의 컴퓨터 시간을 동기화 시키는 프로토콜
- NTP는 계층구조를 가지며, 각 계층을 stratum이라고 한다. 2014(1)
- 포트번호 : TCP/123

NTP 계층 구조

- Stratum 0 : 세슘 원자시계, GPS, 표준 주파수 방송국 등
- Stratum 1 : 계급 0 에 동기화 시킨 시간 서버
- Stratum 2 : 여기서부터 계층적 트리 구조를 형성

NTP 관련 명령어/파일

- **ntpdate 명령어** : NTP 서버를 이용하여 시간을 동기화시키는 명령어
- **ntpq -p 명령어** : 피어들의 상태를 요약해서 출력 2015(1) 2017(2)
- **/etc/ntp.conf 파일** : ntp 서버 설정 파일 2016(1)

5-2. VNC 관리

VNC 개념 2015(1) 2015(2)

- VNC(Virtual Network Computing) : 원격으로 다른 컴퓨터를 제어하는 그래픽 데스크톱 공유 시스템
- 거의 모든 OS에서 사용할 수 있는 오픈소스 소프트웨어
- 포트번호 : 기본적으로 5900번이고 디스플레이 넘버를 더해서 사용 2016(1)

VNC 관련 명령어/파일

- **vncpasswd 명령어** : VNC 서버 패스워드를 변경하는 명령어
- **vncserver 명령어** : VNC 서버 가동
  - [-kill] : VNC 서버 종료
  EX) # vncserver -kill : 1 → VNC 1번 디스플레이 종료
- **vncviewer 명령어** : 클라이언트에서 VNC Viewer 실행하는 명령어 2016(2)
- **/etc/sysconfig/vncservers 파일** 2014(2) 2017(2) 2018(1)

# vi vncservers	
VNCSERVERS="1:root 2:shionista"	// 디스플레이 번호 : 계정명
VNCSERVERS[1]="-geometry 1024x768"	// 1번 디스플레이에 대한 설정 (-geometry : 해상도 설정)
VNCSERVERS[2]="-geometry 1024x768"	// 2번 디스플레이에 대한 설정