



# 리눅스 명령어

2007. 10. 10

안효창

# Contents

- 1 시스템 관리의 기초
- 2 온라인 매뉴얼
- 3 셸
- 4 프로세스
- 5 기본 명령어

# 시스템 관리의 기초

## ■ 시스템 시작 시 작동 순서

- 시스템 부팅
- 시스템 초기화
- 로그인 프롬프트
- 사용자 계정명 입력
- 패스워드 입력
- 셸(Shell) 부여
  - 기본적으로 bash셸이 구동됨.
  - root 사용자는 # , 일반사용자는 \$
- 명령어 입력
- 셸에 의해 해석된 명령어는 커널에 전달
- 커널에 의해 하드웨어(시스템) 제어
- 결과 내용은 표준출력(모니터) 장치를 통해 출력

### • 부팅메세지

- /var/log/messages 에 저장
- dmesg명령으로 확인

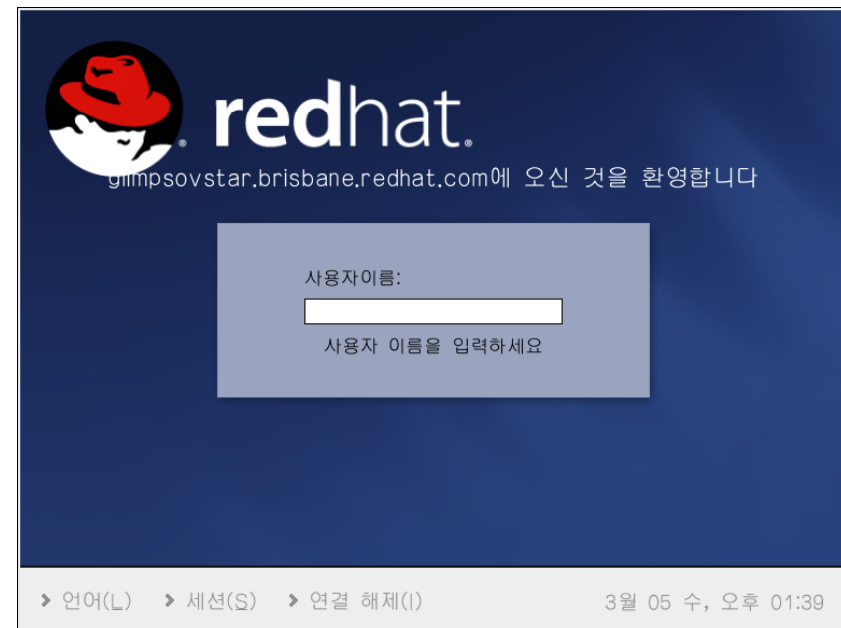
# 시스템 관리의 기초

- 텍스트 로그인 / 그래픽 로그인

```
Red Hat Linux release 8.0 (Psyche)
Kernel 2.4.18-14 on an i686

localhost login: root
Password:
Last login: Thu Mar 13 18:11:33 on :0
[root@localhost root]#
```

텍스트 모드로 설치 시

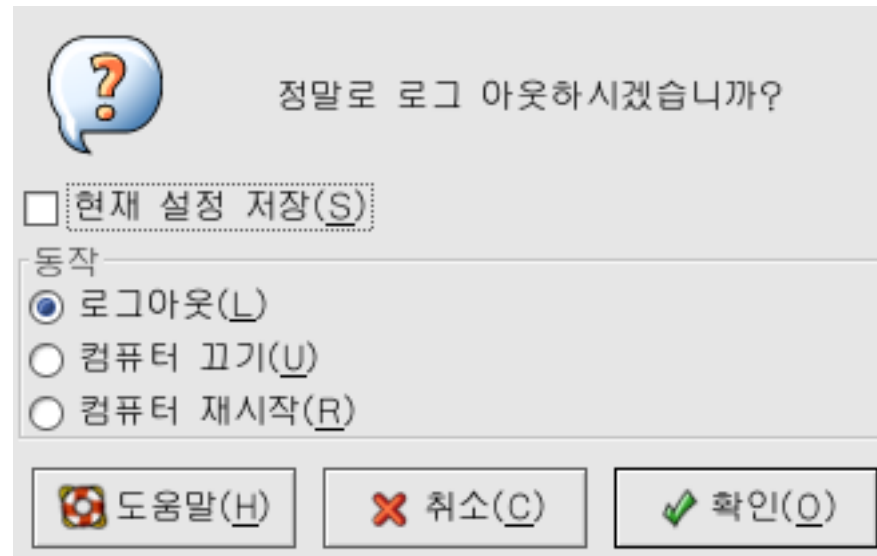


그래픽 모드로 설치 시

# 시스템 관리의 기초

- 텍스트 로그아웃 / 그래픽 로그아웃

# logout



시작 >> 로그 아웃

# 시스템 관리의 기초

- 가상콘솔
  - 기본적으로 6개 제공
  - Ctrl + Alt + F1 ~ F6 : 텍스트 모드
  - Ctrl + Alt + F7 : X윈도우로 전환
  - X윈도우 → 콘솔 : Ctrl + Alt + Backspace
  - X윈도우에서 터미널 사용
    - 바탕화면 >> 오른쪽 마우스 >> 터미널

# 시스템 관리의 기초

- ROOT 계정 (관리자)
  - 시스템 설치 시 자동으로 생성되면, 설치 시 password 입력
  - 시스템 관리를 위한 모든 권한을 가지고 있음
  - 막강한 권한 만큼 사용시 주의를 요구함
- 일반 계정 (사용자)
  - 시스템 관리자에 의해 생성됨
  - 시스템 관리자가 초기 패스워드 설정
  - 사용 목적에 따른 제한된 시스템 사용
  - 개인 사용 환경 변경 및 저장 가능
  - root 권한을 얻기 위해서는 su 명령 사용

# 시스템 관리의 기초

- 부팅과 셧다운
  - X윈도우에서 셧다운
    - 시작 >> 시스템 종료
    - 바탕화면 >> 오른쪽 마우스 >> 터미널 >> shutdown 또는 halt 명령
    - X윈도우 로그인 창에서 Shutdown/Halt 아이콘
  - 셧다운은 root사용자만 가능해야 함.
  - 셧다운은 시스템의 정상종료 후, 수행해야 함



# 시스템 관리의 기초

- shutdown 명령어
  - shutdown -h now == halt == init 0
  - shutdown -r now == reboot == init 6
  - shutdown +10 → 10분 후 셧다운 (다른 사용자를 고려)
  - shutdown -c → 셧다운 취소
  - shutdown -r +20 → 20분후 셧다운(다른 사용자에게 정보 보내줌)
  - ctrl + alt + delete → 재부팅 (/etc/inittab 파일에 지정되어 있음)
    - # Trap CTRL-ALT-DELETE
    - ca::ctrlaltdel:/sbin/shutdown -t3 -r now

# 온라인 매뉴얼

## ■ 사용법

- # man [분류번호] <명령어>
  - [space] : 다음페이지
  - [b] : 이전페이지
  - /검색어 [Enter] : 이후 페이지 문자검색
  - ?검색어 [Enter] : 이전 페이지 문자검색
  - [n] : 다음단어로 이동
  - [q] : 종료

# 온라인 매뉴얼

- 영역별 의미
  - LS(?) : man? 페이지로 분류
  - SYNOPSIS : 명령어 사용법
  - DESCRIPTION : 명령어에 대한 간략한 설명 및 사용 가능한 옵션
  - AUTHOR : 작성한 사람
  - REPORTING BUGS : 버그발견 시 연락처
  - COPYRIGHT : 저작권
  - SEE ALSO : 관련 명령어

# 셸

- Shell 이란 ?
  - 명령어 해석기
  - 시스템과 사용자간의 대화창구 == 셸 인터페이스
- 셸의 종류
  - Bourne 셸 → Bourne Again Shell (bash) (\$)
  - C 셸 → 확장 C 셸 (tcsh) (%)
  - Korn Shell, V Shell...
  - 기본기능이나 작동법은 동일

# 셸

- Shell 의 기능
  - 내부명령, 외부명령 실행 → 명령프롬프트
  - 여러 명령 연결 - 파이프(|)
  - 입/출력 리다이렉션 (>,<,>>,<<,2>) : 명령방향 전환
  - 특수문자 해석 및 치환, 명령어 치환
  - 지역/환경변수 관리
  - 스크립트 프로그래밍 언어 → 자동화 작업

# 셸

- Shell 프롬프트
  - # echo SHELL  
→ echo는 뒤에 나오는 문자열 출력
  - # echo \$SHELL  
→ '\$'는 뒷단어를 변수로 인식하고 변수내용 출력
- 메타 문자
  - Shell이 해석하는 특수한 기능을 가진 문자 → \$
  - 메타문자 해석 금지 → \ , " , "  
예) #echo \SHELL

# 셸

- Linux의 기본 Shell
  - GNU Bourne Again Shell (/bin/bash)
    - C 문법을 많이 도용
- 배시의 환경설정 파일
  - 사용자별 환경설정 파일
    - .bash\_logout, .bash\_profile, .bashrc
    - 원본파일 /etc/skel/ 디렉토리에 존재
    - 새로운 사용자 추가 시마다 각 사용자의 홈 디렉토리에 복사됨.
    - “.”으로 시작하는 파일은 숨김 파일을 의미
  - 모든 사용자 공통의 환경설정 파일
    - /etc/profile , /etc/bashrc

# 셸

- 환경설정 파일 인식 순서
  - /etc/profile → ~/.bash\_profile → ~/.bashrc → /etc/bashrc
- 현재 path 관련 변수 확인
  - # echo \$PATH
- path 디렉토리 추가
  - 전체 사용자에게 추가 : /etc/profile 편집
  - 특정 사용자에게 추가 : ~/.bash\_profile 편집
  - 현재 로그에서만 추가 (로그아웃 시 소멸)
    - \$ echo \$PATH
    - \$ PATH=\$PATH:<추가할 디렉토리>



# 셸

- 개인별 별칭(alias) 설정
- 단축명령어 등록
  - ~/.bashrc 편집
    - 예) alias lsa='ls -a' 추가
- 사용자 프롬프트 변경
  - 절대경로 표시 : /etc/bashrc 편집
    - PS1="[Wu@Wh WW] WW\$  
→ PS1="[Wu@Wh W\$PWD] WW\$

# 셸

- 로그아웃시의 실행
  - ~/.bash\_logout 편집
- 도스키 및 자동 완성
  - 도스키
    - 기존사용 명령어 다시 불러오기
    - 화살표 사용
  - 자동완성
    - 일부 앞 글자만 입력 후 [TAB] 키 사용
    - 디렉토리 이동 시 편리함

# 프로세스

- ps 명령어
  - 현재 프로세스 상태 확인
  - 추천 옵션 : # ps auwx
  - 상태(STAT) 의미

# 프로세스

## ■ kill 명령어

- 프로세스를 종료할 때 사용하는 명령
- 종료버튼을 눌러도 종료되지 않을 때 사용
- ps 명령어로 프로세스번호 확인 후 → # kill <pid>
- 옵션
  - # kill -KILL(9) <pid> : 하드웨어적 종료 (가장 강력)
  - # kill -TERM(15) <pid> : 소프트웨어적 종료 (소프트웨어에 따라서 받아들일 수도 있고, 그렇지 않을 수도 있음)
  - # kill -HUP(1) <pid> : 데몬의 경우 종료후, 다시 시작(사실은 프로세스를 종료하는 것은 아니고, 코드 및 데이터를 refresh 하는 역할)
  - # kill -2 <pid> : 포그라운드에서 [ctrl]+[c]를 누르는 효과
  - # kill -3 <pid> : 포그라운드에서 [ctrl]+[W]를 누르는 효과

# 프로세스

- killall 명령어 : 데몬 이름을 직접 입력하여 종료
- pstree 명령어 : 프로세스를 트리 형식으로 보여줌 (종속성 확인)

# 기본 명령어

- ls 명령어
  - 도스의 “dir”과 같은 역할
  - 해당 디렉토리에 있는 파일의 목록을 보여줌
  - 사용 예
    - # ls : 현재 디렉토리의 파일목록
    - # ls /usr/bin : /usr/bin/ 디렉토리의 목록
    - # ls -a : 현재 디렉토리의 목록(숨김 파일 포함)
    - # ls -l : 현재 디렉토리의 목록을 자세히 보여줌
    - # ls \*.txt : 확장자가 txt인 목록을 보여줌
    - # ls -l /usr/bin/a\*  
: /usr/bin/ 디렉토리에 있는 목록 중 앞 글자가 ‘a’인 것의 목록을 자세히.
    - # man ls 로 확인

# 기본 명령어

- cd 명령어
  - 디렉토리 이동 명령
  - 사용 예
    - # cd : 사용자의 홈 디렉토리로 이동
    - # cd ~john → john 사용자의 홈 디렉토리로 이동 (대개 /home/john/)
    - # cd .. : “..”은 현 디렉토리의 부모 디렉토리를 의미
    - # cd /usr/bin : “/usr/bin” 디렉토리로 이동 (절대경로)
    - # cd /root → cd ../usr/bin : 상대경로로 이동

# 기본 명령어

## ■ mkdir 명령어

- 새로운 디렉토리를 생성
- 생성된 디렉토리는 명령어를 수행한 사용자의 소유가 됨.
- 사용 예
  - # mkdir abc : 현재 디렉토리 아래에 abc라는 디렉토리 생성
  - # mkdir -p def/ghi : 현재 디렉토리 아래에 def 디렉토리 생성하고 그안에 ghi 디렉토리 생성

## ■ rmdir 명령어

- 디렉토리 삭제 명령
- 해당 디렉토리의 삭제권한이 있어야 함.
- 해당 디렉토리는 비어 있어야 함.
- 사용 예
  - # rmdir abc



# 기본 명령어

## ■ touch 명령어

### ■ 사용 예

- # touch abc.txt

파일이 없을 경우엔 abc.txt라는 빈파일 생성

abc.txt가 있을 경우엔 파일의 수정날짜를 변경시킴

## ■ rm 명령어

### ■ 파일이나 디렉토리 삭제 명령어

### ■ 해당 파일이나 디렉토리에 삭제권한이 있어야 함

### ■ 사용 예

- # rm abc.txt : 그냥 삭제

- # rm -i abc.txt : 삭제시 확인

- # rm -r abc : 디렉토리 삭제

- # rm -rf abc : abc디렉토리와 그 하부를 강제로 전부 삭제

# 기본 명령어

- mv 명령어
  - 파일과 디렉토리의 이름변경이나 위치이동 때 사용
  - 사용 예
    - # mv aaa bbb ccc ddd : 이동. ddd는 디렉토리.
    - # mv abc.txt www.txt : 이름 변경
- cp 명령어
  - 파일이나 디렉토리 복사 명령어
  - 새로 복사한 파일은 사용자의 소유가 됨
  - 사용 예
    - # cp abc.txt cba.txt
    - # cp -r abc cba : 디렉토리 복사

# 기본 명령어

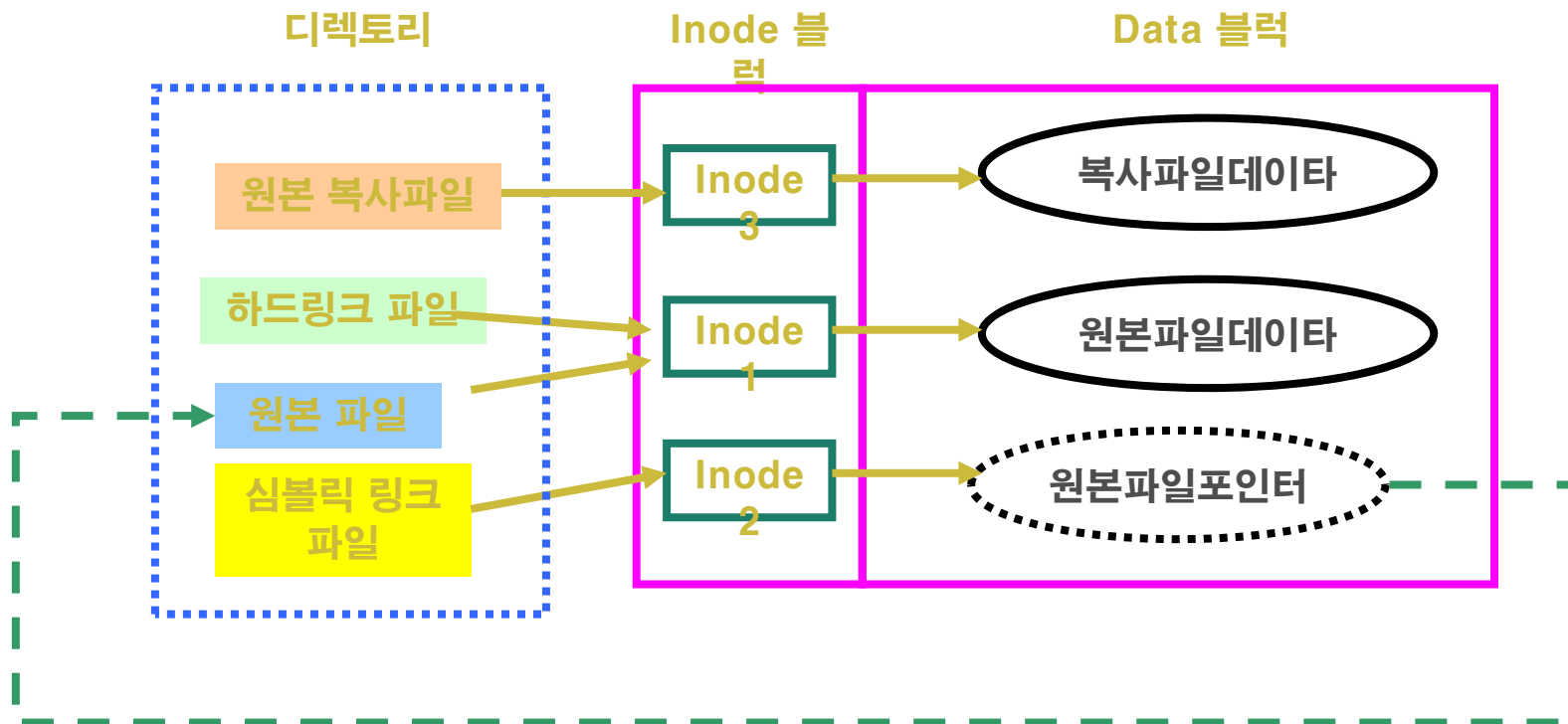
- cat 명령어
  - 텍스트로 작성된 파일을 화면에 출력
  - 사용 예
    - # cat abc.txt
- more 명령어
  - 텍스트로 작성된 파일을 화면에 페이지 단위로 출력
  - [space]키는 다음 페이지, [b]는 앞 페이지
  - 사용 예
    - # more abc.txt

# 기본 명령어

- ln 명령어
  - Windows의 “바로가기 아이콘”과 비슷한 것을 만들어줌
  - 심볼릭 링크와 하드링크가 있음
  - 사용 예
    - # ln -s aaa.txt bbb.txt : 심볼릭 링크
    - # ln aaa.txt bbb.txt : 하드 링크

# 기본 명령어

- 복사, 하드링크, 심볼릭 링크



# 기본 명령어

## ■ 파일의 링크

### ■ 심볼릭 링크

- Windows의 바로가기 아이콘 과 비슷함
- 주로 사용자가 많이 사용하는 링크임
- 긴 디렉토리명을 이동 시 유용함
- 원본을 다른 곳으로 이동시키면 사용 못함
- 새로운 inode 생성
- 원본자체를 가리키는 포인터 역할 뿐이므로, 원본 수정 시 당연히 수정된 파일이 보임
- 원본의 삭제 시 링크가 끊김
- 파일시스템이 달라도 관계없음

# 기본 명령어

- 하드링크
  - 동일한 inode 사용
  - 원본 수정시 하드링크도 수정됨
  - 원본 삭제시 살아 있음
  - 원본의 이동이나 삭제시 남아 있음
  - 동일한 파일시스템에만 적용됨
  - 파일복사와 차이는 공간을 차지하지 않는 것임

# 기본 명령어

- 파이프와 필터
  - 파이프 (pipe : 명령어의 연결통로)
    - 앞 프로그램의 결과를 뒤 프로그램의 입력값으로 전달해주는 역할
    - 메타문자 : |
    - 한가지 명령어로 부족할 때 다른 명령과 조합
    - 사용 예
      - # ls -al /usr/bin | more
      - # ls /usr/bin | sort | less → 알파벳순으로 정렬



# 기본 명령어

## ■ 필터

- 표준 입력으로부터 자료를 읽어 간단한 처리후 표준출력으로 보내는 프로그램
- grep, tail, wc, sort, grep, awk, sed 등
- 파이프와 동시에 사용을 많이 한다.
- 사용 예
  - # grep abc.txt hello → abc.txt에서 hello라는 글자가 있는 행 출력
  - # grep -v abc.txt hello → hello라는 글자 외의 행 출력
  - # ps aux | grep xfs → 현재 프로세스 중 xfs 라는 프로세스 출력
  - # rpm -qa | ^X → 현재 설치된 rpm 중, 맨 앞글자가 'X'인 패키지명 출력

# 기본 명령어

- tee : 파이프 중간에 사용하여 입력을 출력으로 보내기 전에 파일로 기록
  - 사용 예
    - # ls -al /bin | tee bin\_list | sort

# 기본 명령어

- 표준 입출력과 리다이렉션
  - 표준입력은 키보드, 표준출력은 모니터로 되어 있음
  - 입출력을 파일로 하고자 할 때 주로 사용
- 표준출력 Redirection
  - 표준출력장치(모니터)의 방향을 파일로 전환
  - 메타문자 : > , >> (append)
  - 사용 예
    - # ls /usr/bin > list\_file
    - # ls /usr/bin >> list\_file
    - # cat /dev/cdrom > "win2000.iso"

# 기본 명령어

- 표준오류 Redirection
  - 메타문자 : `2>` , `2>>`
  - 사용 예
  - `# gcc -c hello.c 2> error.txt` → 오류 미발생 시 파일생성 안됨
- 입력 Redirection
  - 메타문자 : `<` , `<<`
  - `# sort < abc.txt > sorted.txt`
  - `# cat << end` → end문자를 만나면 입력종료



■ *Any Questions ?*

