도스에서 리눅스로 - 신속하게!

저자 : Guido Gonzato <*Guido@ibogfs.cineca.it* <mailto:Guido@ibogfs.cineca.it>> 1996년 4월 26일 1.1

이 만 용 <*geoman@nownuri.nowcom.co.kr* <mailto:geoman@nownuri.nowcom.co.kr>> 1996년 11월 30일

머리말 ====== 이 미니 하우투는 386 이상의 컴퓨터에 대한 유닉스 클론인 리눅스로 막 빠져들었 으며 리눅스로의 전환하려고 결심한 모든 도스 사용자들(조금 있으면 전직 도스 사용자가 될)을 위한 글이다. 본인이 처음 PC 에 리눅스를 설치했을 때 본인은 그 즉시 운영을 해보고 싶었다. 그리고 이는 누구나 원하는 일이라고 본다. 아마도 여러분은 도스가 유닉스에 대한 절름발이격이라고 할 수 있다는 것을 알고 있을 것이다. 오랫동안 도스를 사용해 온 사람으로서 본인은 이미 두 운영체제 간의 유사한 점을 약간 알고 있다. : 화일시스템, 디렉토리, 환경 변수, 프로그램 실행, .BAT 화일 등에 대한 개념들. 이 문서의 목적은 여러분이 갖고 있는 도스 지식을 리눅스 환경에서 최대한 활용할 수 있도록 돕기 위함이다. 여러분의 배우자가 잠시 없거나 아니면 둘이 같이 잠들 기 전에 몇 페이지 읽어볼 수 있게 PC 가까운 곳에 놓아두도록 하라. 이 문서가 리눅스에 대한 입문서는 아니라는 사실을 강조하고 싶다. 초보자들에게는 Matt Welsh 씨의 "Linux Installation and Getting Started"를 한 부 복사해서 보길 권장한다. 문서는 sunsite.unc.edu 또는 미러링 사이트에서 /pub/Linux/docs/LDP 에서 찾을 수 있다. 상당히 잘 씌여졌으며 이해하기 쉽게 되어 있다. 많은 하우투 문서, FAQ 문서 또한 구할 수 있다. 이 문서는 그 문서들과의 중복된 또 다른 문서이기 보다는 단지 본인처럼 200 페이지 이상 읽는 것보다는 그냥 20 페 이지 정도 읽어보길 좋아하는 사람들을 위한 것이다.

목 차

제 1 절 소 개

1.1 리눅스가 여러분에게 맞는가?

가장 좋은 컴퓨터 또는 가장 좋은 운영체제란 없다. : 그것은 여러분이 어떤 일을 하는가에 달려있다. 예를 들어 본인은 맥을 좋아하지 않는다(흐 신성모독인가?) 분명히 사용하기는 편하지만 어떤 용도로는 단연 PC 를 선호한다. (맥에서 dir/s/w >lst 와 같은 일을 한다고 생각해보라) 물론 많은 사람들에게 있어 맥은 그야말로 가장 좋은 컴퓨터이다. 마찬가지로 나는 리눅스가 최고의 OS 라고는 생각 치 않는다. 굳이 말하자면 윈95 보다는 훨등하게 뛰어나지만 많은 사람들이 윈95를 더 선호하고 있다고 할 수 있다.

유닉스 계열은 프로그래머/과학자에게 집중하는 반면 맥과 윈도우즈는 비기술적인 일반사용자에게 촛점을 맞추고 있다. 따라서 여러분이 일반적으로 리눅스로부터 컴파일러, 네트워크, 과학 소프트웨어를 필요로 하거나 여러분 스스로 무엇을 뚝딱 뚝딱 만들거나 고치는 일(말하자면 해킹)을 싫어하지 않는다면 리눅스로부터 엄청난 혜택을 누릴 수 있다. 만약 주로 상용 소프트웨어를 필요로 하거나 명령을 타이핑하는 것을 싫어한다면 여러분은 리눅스를 미워할 것이다.

여러분에게 달렸다. 어찌 되었든 리눅스와 도스/윈도우즈는 같은 컴퓨터에 공존할 수 있다는 것만 기억하라.

1.2 좋다. 좀 더 얘기해다오.

여러분은 PC 에 리눅스를 인스톨하였다. 설치 지시에 따라 여러분 자신에게 계정을 내주었을 것이다.(그렇지 않다면 섹션 6.1 을 보라) 그리고 리눅스를 운영 중일 것이다. 어떤 이는 여러분에게 원하는 모든 것이 거기 있다고, 그리고 단지 로그인 해서 해보기만 하면 된다고 말했을 것이다. 사용자명과 패스워드를 넣고 여러분은 스크린을 보며 생각하게 될 것이다: "흠... 이제 뭘 하지?" 자, 절망하지 말라. 여러분은 DOS 로 하던 일 아니 그 이상을 지금 당장 할 수 있다. 리눅스 대신 도스를 띄웠다면 여러분은 다음과 같은 일들을 할 것이다.

1. 프로그램 실행과 화일 생성, 카피, 보기, 지우기, 이름변경

2. CD, MD, RD 그리고 DIR 하기

3. 플로피 포맷하고 화일 카피하기
4. 여러분의 AUTOEXEC.BAT 와 CONFIG.SYS 고치기
5. 여러분만의 .BAT 화일이나 QBASIC 프로그램 짜기
6. 나머지 1%

여러분은 리눅스에서도 도스에서와 비슷한 방법으로 이런 일을 할 수 있다는 사실을 알게 되면 기뻐할 것이다. 도스에서 보통의 사용자들은 100 개 이상의 명령 중 거의 얼마 사용하지 않고 있다. 마찬가지로 리눅스에 대해서도 그렇다고 할 수 있다. 만약에라도 위에서 제시한 목록에 사용된 도스 명령어를 모른다면 지금 포기하라. 본인은 여기서 디렉토리가 무엇이며 PATH 변수가 무엇인지에 대해서는 설명하지 않 는다.

더 나아가기 전에 몇 가지 지적할 만한 사항이 있다면:

1) 도스와는 달리 리눅스는 자체 보안 메커니즘을 가지고 있다. 여러분이 PC 를 켜 고 리눅스를 실행시키면 여러분을 시스템에 알리는 "로그인"을 해야 한다. 왜냐하면 유닉스는 다중 사용자 운영체제이기 때문이다. 게다가 일반적인 사용자는 시스템을 실제로 소유할 수 없다. 로그인시 로그인명이 root 인 사용자만이 그런 권력을 지닌다(이 녀석이 바로 시스템 관리자이다. 여러분이 여러분의 PC 에서 작업한다면 여러분이 root 이다) 화일과 디렉토리는 사용자에 관련된 허가권을 가지고 있고 따라서 일반 사용자는 접근조차 안되는 경우도 있다. 반면 도스에서는 언제든 여러분의 하드디스크를 몽땅 지우는 일이 가능하다.

2) 일단 작업을 마친 후 여러분은 PC 의 스위치를 그냥 내려선 절대 안된다. 대신 알맞는 셧다운 과정을 따라야 한다.(CTRL-ATL-DEL 을 누르고 시스템이 내부를 점검하고 OK 라고 말하고 나면 스위치를 내린다) 어쨌든 리눅스는 잘못된 프로그램 이 여러분의 컴퓨터를 멈추게 하는 일은 극도로 불가능한 정마로 믿을 만한 운영체 제이다. - 따라서 리셋 키는 필요치 않다.

3) you are strongly encouraged to experiment, play, try by yourself: it

3) 여러분 스스로 실험해보고 즐기기 바란다. 해가 되지 않는다. 프롬프트 상에서 (일반적으로는 $, root 의 경우 #) 도움을 얻을 수 있다. $ help (상당히 논리적이다) 또는 다음과 같이 어떤 명령에 대한 정보를 얻는다. $ man <명령> which, if you have installed the man pages, will invoke the manual ("man") 여러분이 맨 페이지를 설치했다면 <명령>에 관련된 매뉴얼("man") 페이지를 볼 수 있다. 또한 다음과 같이 해도 된다. $ apropos <명령> 또는 $ whatis <명령> 4) 범례:

a) <...>은 명시되어야 할 무엇을, 반면에 [...]은 생략가능한 것을 나타낸다. 예를 들어:

$ tar -tf <file.tar> [> redir_file] file.tar 는 명시되어야 하지만 redir_file 에 대한 리다이렉션은 선택적이다. b) 지금부터 RMP 는 더 많은 정보를 위해 맨페이지를 읽어보라는 뜻으로 사용한다. 이제 여러분은 다음 섹션을 알아볼 차례이다.

제 2 절 화일 처리하기

2.1 화일:사전 지식

Linux has a file system - meaning by that "the structure of directories and 리눅스는 화일 시스템이라는 것을 가지고 있다. 이는 도스에서와 유사하게 "디렉토 리와 화일의 구조"를 의미한다. 화일은 특수한 규칙을 따르는 화일명을 가지고 있고 디렉토리 안에 저장되며 어떤 것은 실행화일이며 대부분은 명령행 스위치(swtich, 옵션)을 가지고 있다. 또한 여러분은 와일드 카드문자, 리다이렉션, 파이프 등을 사용할 수 있다. 큰 차이는 없다.

- 첫번째 차이점 : 도스에서는 화일명이 소위 8.3 형식을 따른다. 예를 들어 NOTENOUG.TXT. 리눅스에서는 이보다 낫다. 리눅스를 인스톨할 때 ext2 또는 umsdos 같은 화일시스템(여러분은 지금 정확한 의미를 알지 못한다)을 사용한다면, 하나 이상의 도트(.)를 사용하면서 더 긴 이름을 적을 수 있다. 예를 들어 This_is.a.VERY_long.filename 본인이 대소문자를 같이 사용했다는 사실에 주목하기 바란다. 사실...

- 두번째 차이점 : 대문자와 소문자가 구별된다. 따라서 FILENAME.tar.gz 와 filename.tar.gz 은 2 개의 다른 화일이다. 명령도 마 찬가지이다. ls (도스에서의 DIR 과 같다) 명령을 치면 현재 디렉토리에 대한 화일 목록을 얻지만 LS 라고 치면 에러 메세지가 나올 것이다.
- 세번째 차이점 : 만약 피리어드 문자(.)로 화일명을 사용하면 그 화일은 숨겨져 있다고 간주된다. 예: .I.am.a.hidden.file 은 ls 명령으로 나타나지 않는다.
- 네번째 차이점 : 프로그램에 대하여 .COM, .EXE 그리고 배치화일에 대하여 .BAT 등의 강제적인 확장 규정은 없다. 실행화일은 ls -F 명령을 내렸을 때 화일명 뒤에 스타 문자(*)가 표시된다.

  예: $ ls -F letter_to_Joe Cindy.jpg cjpg* I_am_a_dir/ my_1st_script*

  cjpg* 와 my_1st_script* 은 실행가능한 "프로그램"이다. 도스에서는 .BAK가 백업 화일이지만 리눅스에서는 보통 틸드문자( )로 끝난다.
- 다섯번째 차이점 : 도스 프로그램의 스위치는 /스위치이만 리눅스에서는 -스위치 이다. 예를 들어 dir /s 은 ls -R 과 같다. PKZIP 또는 ARJ 과 같은 많은 도스 프 로그램은 유닉스 스타일의 스위치를 사용한다.

2.2 화일 : 도스로부터 리눅스로 명령 번역

왼쪽에는 도스 명령어가 오른쪽에는 해당 리눅스 명령이 있다.

```
COPY:   cp
DEL:    rm
REN:    mv
TYPE:   more, less, cat
리다이렉션과 연결 연산자: < > >> |
와일드카드: * ?
nul:    /dev/null
prn, lst:       /dev/lp0; lpr
- 예 -
DOS                                     Linux
C:\GUIDO>copy joe.txt joe.doc           $ cp joe.txt joe.doc
C:\GUIDO>copy *.* total                 $ cat * > total
C:\GUIDO>copy fractals.doc prn          $ cat fractals.doc | lpr
C:\GUIDO>del temp                       $ rm temp
C:\GUIDO>del *.bak                      $ rm *~
C:\GUIDO>ren paper.txt paper.asc        $ mv paper.txt paper.asc
C:\GUIDO>type letter.txt                $ more letter.txt
C:\GUIDO>type letter.txt                $ less letter.txt
C:\GUIDO>type letter.txt > nul          $ cat letter.txt >/dev/null
        n/a                             $ more *.txt *.asc
        n/a                             $ cat section*.txt
```

주목:

- * 는 리눅스에서 더욱 똑똑하게 행동한다: - * 은 숨겨진 화일을 제외한 모든 화일과 매치된다. .* 는 모든 숨겨진 화일과 매치된다. *.* 은 . 과 그리고 뒤에 다른 문자들이 있는 경우와 매치된다.

  - p*r 은 peter, piper와 매치된다. *c* 은 picked, peck 에 매치된다.
- more 를 사용할 때 <스페이스>를 누르면 앞으로 전진하고 q 또는 CTRL-C 는 종료할 때 쓴다. less는 더욱 직관적이고 화살표키를 사용한다.

- UNDELETE 기능이란 없다. 따라서 지우기 전에 두 번 생각해야 한다.
- 도스의 < > >> 에 덧붙여 리눅스는 에러메세지(stderr)를 리다이렉트하기 위한 2> 이라는 것을 가지고 있다.
- 리눅스에는 또 다른 와일드 카드 문자가 있는데]. $-$ $[abc] * a,b,c.$ $-$ $[I - N] * I,J,K,L,M,N.RENAME.ren * .xxx * .yyy.$

2.3 프로그램 실행: 멀티태스킹과 세션

프로그램을 실행시키기 위해서는 도스에서처럼 이름을 입력하면 된다. 만약 프로그 램이 PATH(섹션 4)에 포함된 디렉토리(섹션 2)에 저장되어 있다면 실행될 것이다. 예외: 도스와는 달리 리눅스에서는 현재 디렉토리가 PATH에 포함되어 있지 않는 한 현재 디렉토리에 있는 프로그램을 실행되지 않는다.

여러분의 프로그램이 만약 <프로그램>이라면 ./<프로그램> 이런 식으로 실행해야 현재 디렉토리에 있는 것이 실행된다. 전형적인 명령어 라인은 다음과 같다.

$ command -s1 -s2 ... -sn par1 par2 ... parn <input >output

where -s1, ..., -sn are the program switches, par1, ..., parn are the 여기서 -s1, ..., -sn 은 프로그램 스위치이고 par1, ..., parn 은 프로그램에 대한 전달인수이다. 몇 개의 명령을 한 줄에 써줄 수도 있다.

$ command1 ; command2 ; ... ; commandn

프로그램을 수행시키는 방법으로 충분하지만 한 발짝 더 나아가는 것은 쉽다. 리눅스를 사용하는 중요한 이유 중 하나는 멀티태스킹 운영체제라는 사실이다. 몇 개의 프로그램을(이제부터는 프로세스라고 부르겠다) 동시에 수행시킬 수 있 다는 것이다. 프로세스를 백그라운드에서 돌리고 다른 작업을 계속 해나갈 수 있다. 게다가 리눅스는 여러 개의 세션을 가지고 있다. 마치 동시에 여러 개의 컴퓨터를 가지고 있는 것과 같다.

- 세션 1 부터 8 까지 전환을 하려면

  $ <ALT-F1> ... <ALT-F8>.
- 현재의 세션을 닫지 않고 다른 세션을 시작하려면

  $ su - <로그인명> 예: $ su - root 여러분이 디스크를 마운트(섹션 3)할 때 매우 유용하다. 오로지 루트만이 마운트를 할 수 있기 때문이다.
- 세션을 마치기 위해서는:

  $ exit
- 포그라운드에서 프로세스를 시작하려면:

  $ progname $-switches][parameters][¡input][¿output](\&).\$progname[-switches][parameters][¡input][¿output]\&$

- 얼마나 많은 프로세스가 있는지 알아보려면: $ ps 현재 실행 중인 프로세스의 목록을 보여준다. 각각은 PID 라는 숫자를 가지고 구분 된다.
- 프로세스를 죽이기: $ kill <PID> 프로세스를 어떻게 끝내야 할 지 모를 때 프로세스를 죽일 필요가 있다 ;-) 때때로 프로세스는 다음과 같이 해야만 죽을 때가 있다. $ kill -15 <PID> $ kill -9 <PID> 이에 덧붙여 셸(COMMAND.COM 에 해당; 많은 사람들이 bash 를 사용하고 있다) 은 여러분이 프로세스를 멈추게 하거나 잠시 중지시키는 일, 프로세스를 백그라운 드로 보내는 일, 그리고 다시 백그라운드에서 포그라운드로 옮기는 일을 가능케 해준다. 이러한 맥락에서 프로세스는 "작업(jobs)"이라고 불린다.
- 얼마나 많은 작업이 있는지 보려면: $ jobs 여기서 작업은 셸이 PID 가 아니라 셸이 프로세스에 부여한 작업 번호로 구분한다.
- 포그라운드의 프로세스 중지하기: $ <CTRL-C> (항상 그렇게 되는 것은 아니다)

- 포그라운드에 실행 중인 프로세스 일시 정지시키기: $ <CTRL-Z> (마찬가지이다)
- 일시정지된 프로세스를 백그라운드로 보내기: $ bg <job>
- 프로세스를 포그라운드로 되돌리기: $ fg <job>
- 다시 프로세스를 죽이기 $ kill %<job>

  where <job> may be 1, 2, 3, ... Using these commands you can format a 여기서 <job>은 1, 2, 3 이런 숫자이다. 이러한 명령어를 사용하여 여러분은 디스 켓을 포맷하고 화일을 zip 압축하고 프로그램을 컴파일하고 동시에 다른 화일의 압축을 풀어낼 수 잇다. 그리고 또 여러분의 명령을 기다리는 프롬프트가 나온다. 도스에서 해보라! 그리고 얼마나 퍼포먼스가 차이나는지 윈도우즈에서도 해보기 바란다.

제 3 절 디렉토리 사용하기

3.1 디렉토리 : 사전 지식

We have seen the differences between files under DOS and Linux. As for 여러분은 도스와 리눅스 간의 화일의 차이점을 알아보았다. 디렉토리에 관한 한, 도스에서 루트 디렉토리는 \ 이지만 리눅스에선 / 이다. 마찬가지로 도스에서는 디렉토리 구분자가 \ 이지만 리눅스에서는 / 이다.

화일 패스에 대한 예로서:

```
DOS:    \PROGRAMS\C++\SOURCES\HELLO.CPP
Linux:  /home/guido/papers/geology/mid_eocene.tex
```

As usual, .. is the parent directory, . is the current directory. Remember 통상 .. 은 부모 디렉토리이고 . 는 현재 디렉토리이다. 리눅스에서는 여러분이 원한다고 해서 어떤 곳으로든 cd 하거나 또는 rd, md 할 수 없다. 모든 사용자는 시스템 관리자에 의해 부여된 자기 자신의 디렉토리에서 시작한다. 예를 들어 집에 있는 PC 에서 본인의 홈 디렉토리는 /home/guido이다.

3.2 도스에서 리눅스로의 명령 전환

```
DIR:    ls, find, du
CD:     cd, pwd
MD:     mkdir
RD:     rmdir
- 예 -
DOS                                     Linux
C:\GUIDO>dir                            $ ls
C:\GUIDO>dir file.txt                   $ ls file.txt
C:\GUIDO>dir *.h *.c                    $ ls *.h *.c
C:\GUIDO>dir/p                          $ ls | more
C:\GUIDO>dir \*.tmp /s                  $ find / -name "*.tmp"
C:\GUIDO>cd                             $ pwd
        n/a - see note                  $ cd
        n/a - see note                  $ cd ~
        n/a - see note                  $ cd ~/temp
C:\GUIDO>cd \other                      $ cd /other
C:\GUIDO>cd ..\temp\trash               $ cd ../temp/trash
C:\GUIDO>md newprogs                    $ mkdir newprogs
C:\GUIDO>md \progs\turbo                $ mkdir /progs/turbo
```

```
C:\GUIDO>rd newprogs                  $ rmdir newprogs
C:\GUIDO>rd \progs\turbo              $ rmdir /progs/turbo
```

주목: 틸드문자 는 여러분의 홈 디렉토리에 대한 단축글자이다. cd 또는 cd 라고 하면 여러분이 현재 어디에 있든지 홈 디렉토리로 이동한다. cd /tmp 라고 하면 여러분은 /home/여러분의 홈디렉토리/tmp 로 보내준다. 이해하겠는가?

제 4 절 플로피, 하드디스크, 등등

4.1 플로피 디스크 사용

You have never thought about it, but the DOS command FORMAT A: does a lot 도스 명령어 FORMAT A: 는 사실 보기보다 더 많은 일을 한다는 것에 대해서는 생각 치 못했으리라. 사실 FORMAT 명령을 하면 다음과 같은 일이 벌어진다.

1. 물리적으로 디스크를 포맷
2. A:\ 디렉토리를 생성한다. (즉 화일 시스템 만들기)
3. 사용자에게 사용할 수 있도록 해준다 (디스크 마운트하기)

각 단계가 리눅스에서는 구분되어 있다. 다른 형식도 가능하지만 MS-DOS 형식의 플로피를 만들거나 사용할 수 있다. 가장 많이 쓰이는 형식은 ext2 이다. 디스크를 준비하는 방법에 대해 알아보자.(root 로 세션을 열어야 한다)

- 표준 1.44 메가 플로피 디스크(A:) 포맷하기: # fdformat /dev/fd0h1440
- 화일 시스템 만들기: # mkfs -t ext2 -c /dev/fd0h1440

  또는 MS-DOS 화일 시스템을 만들기 위해서는 # mformat a: 디스크를 사용하기 전에 여러분은 그것을 마운트해야 한다.
- 디스크를 마운트하기 위해서는 # mount -t ext2 /dev/fd0 /mnt 또는 # mount -t msdos /dev/fd0 /mnt Now you can address the files in the floppy. When you've finished, before 이제야말로 플로피에 화일을 사용할 수 있다. 작업이 끝난 후에는 디스켓을 꺼내기 에 앞서 언마운트를 해야 한다(* 매우 중요 *)
- 디스크를 언마운트하기 위해서는: # umount /mnt 이제 디스크를 꺼내도 좋다. 예전에 사용된 것이 아닌 포맷되지 않은 디스켓에 대 해서만 fdformat과 mkfs를 사용하라. 드라이브 B: 를 사용하고 싶으면 예제에서 fd0h1440과 fd0 대신 fd1h1440과 fd1 을 사용한다.

여러분은 지금 A:, B: 를 다루는데 사용하는 것은 /mnt 이다. 예를 들면:

```
DOS                            Linux
C:\GUIDO>dir a:                $ ls /mnt
C:\GUIDO>copy a:*.* \docs\temp $ cp /mnt/* /docs/temp
C:\GUIDO>copy *.zip a:\zip     $ cp *.zip /mnt/zip
C:\GUIDO>a:                    $ cd /mnt
A:\>                           /mnt$
```

말할 것도 없이 플로피는 여러분이 장치라고 생각하는 것 중 하나일 뿐이다. 예를 들어 다른 하드디스크나 시디롬 드라이브를 원한다고 치자. 장치 목록을 보기 위해서는 /dev 디렉토리의 목록을 살펴보라. 시디롬 마운트 방법은 다음과 같다.

mount -t iso9660 /dev/cdrom /cdrom

제 5 절 시스템 다루기

5.1 시스템 초기화 화일들

도스에서 중요한 화일 2 개 AUTOEXEC.BAT와 CONFIG.SYS는 부팅 시 시스템을 초기화 하고 PATH나 FILES와 같은 환경 변수를 세팅하고 프로그램이나 배치 화일을 실행 시키기도 한다. 리눅스에서서는 여러 개의 초기화 화일들이 있는데 그 중 몇 개는 여러분이 제대로 알지 못한 상태에서는 수정하지 않은 것이 좋다. 어쨌든 어떤 것이 가장 중요한 것인지 말해주겠다.

화일	주의사항
/etc/inittab	지금은 건드리지 말라.
/etc/rc.d/*	설정 확일

PATH 또는 다른 환경변수를 세팅하기 원한다면 또는 로그인 메세지를 바꾸거나 로그인 후 프로그램을 수행시키기 위해서는 다음 화일을 살펴보라.

화일	주의사항
/etc/issue	로그인 전의 메세지 세팅
/etc/motd	로그인 후의 메세지 세팅
/etc/profile	PATH 등의 변수 세팅
/home/your_home/.profile	여러분이 원하는 것은 무엇이든지...

If the latter file exists (note that it is a hidden file), it will be read 마지막 화일(숨겨진 화일임에 주목)이 존재하면 로그인 후 그 안에 있는 명령이 수행된다.

```
예 - 여기에 소개한는 .profile 을 살펴보라.
        # 나는 주석이다.
        echo Environment:
        printenv | more   # DOS에서의 SET 명령과 비슷
        alias d='ls -l'   # 별명이라는 것이 무엇인지 이해할 수 있을 것이다
        alias up='cd ..'
        echo "현재 패스는 "$PATH
        echo "즐거운 하루가 되시길, "$LOGNAME
```

여러분이 예상했다시피 PATH와 LOGNAME 은 환경 변수이다.

5.2 프로그램 초기화 화일들

Under Linux, virtually everything can be tailored to your needs. Most 리눅스에서서는 사실 모든 것을 여러분의 요구에 맞출 수 있다. 대부분의 프로그램에 는 여러분이 조작할 수 있는 보통 .prognamerc 과 같은 이름의 초기화 화일을 여러분의 홈 디렉토리에 가지고 있기 마련이다. 여러분이 수정을 가할 첫번째 대상들이다.

- xinitrc: X 윈도우를 초기화하기 위하여 startx 에 의해 사용
- fvwmrc: 윈도우 관리자 fvwm 에 의해 사용, 샘플 화일은 /usr/lib/X11/fvwm 디렉 토리에 있는 system.fvwmrc 이다.
- xfm/*: X 용 화일 관리자 xfm 에서 사용
- Xdefault: X 용 터미널 에뮬레이터인 rxvt 에서 사용

이 모든 것과 나중에 닥치게 될 모든 것에 대해서는 RMP.

제 6 절 약간의 프로그래밍

6.1 스크립트 짜기 : .BAT 화일 만들기

긴 명령어에 대한 단축 명령으로서 .BAT 화일을 사용한 적이 있다면(본인은 많이 그러했다) profile 이나 .profile 안에다(위의 예 참고) 적절한 alias 행을 넣 어주기만 하면 된다. 하지만 .BAT 이 좀 더 복잡해진다면 셸을 통해 사용할 수 있 는 스크립트 언어를 애호하게 될 것이다. 더 나은 것은 아니더라도 QBASIC 만큼이 나 셸은 강력하다. 변수는 물론 while, for, case, if ... then ... else 등의 제어문이 있고 또 다른 많은 기능들이 들어있다.

다른 실제적인 프로그래밍에 대한 좋은 대안이라고도 할 수 있다. 도스에서는 .BAT 에 해당하는 스크립트를 짜기 위해선 명령 지시문을 포함하는 표준 아스키 화일을 쓰고 저장한 다음, 수행가능하게 만들어주면 된다.

$ chgmod u+x <scriptfile>

실행하기 위해서는 그냥 이름을 타이핑한다. 한 마디 경고. 시스템 편집기는 vi 가 있는데 약간 배우기 어려울 것이다. 여러분이 어렵다고 투덜댈 것이라고 확신하며 본인 또한 제대로 알지 못하므로 여기서 설명하지는 않겠다. Matt Welsh씨의 "Linux installation..." 페이지 109쪽 을 보라. joe 나 X 용 이맥스 등을 선택하는 것도 좋다. 다음과 같이만 말해두면 될 것 같다.

- 텍스트를 입력하기 위하여 i 를 타이핑한 후 텍스트를 입력한다. - 저장하지 않고 끝내기 위해서는 ESC 키를 한 번 누르고 :q! 하면 된다. - 저장하고 끝내기 위해서는 ESC 키를 한 번 누르고 :wq 하면 된다.

Writing scripts under Linux is such a vast subject it would require a book 리눅스에서 스크립트 짜는 일은 그것만으로도 한 권이 책이 될 만큼 방대한 분량의 주제이다. 그리고 더 깊이 들어가지는 않을 것이다. 단지 여러분에게 유용한 몇 가 지 예들에서 여러분이 기본적인 규칙을 발견할 수 있도록 제시하고자 한다.

```
예 1 : first_script
        #!/bin/sh
        # 나는 주석문이다.
        # sh가 거기 있어야 하므로 첫번째 줄을 고치지는 말라.
        echo "오늘은 ‘date‘"
        echo "제 이름은 "$0
        echo "당신은 저에게 다음과 같은 "$#" 인수를 전달하셨습니다: "$*
        echo "첫번째 전달인수는 "$1
        echo "뭔가 감을 잡으셨습니까?"
예 2: 2exe
        #!/bin/sh
        echo "$1" 을 실행가능케 만들고 있습니다... "
        chmod u+x $1
예 3: backup
        #!/bin/sh
        echo "~/bak로 화일을 카피 중입니다"
        for name in $*
        do
          cp ${name} ~/bak
        done
예 4: fmta
        #!/bin/sh
        echo "오로지 루트만이 포맷할 수 있습니다"
        fdformat /dev/fd0H1440
        mkfs -t ext2 -c /dev/fd0H1440
```

```
        echo "디스크가 포맷되었습니다."
예 5: mnta
        #!/bin/sh
        echo "오토지 루트만이 마운트할 수 있습니다"
        mount -t msdos /dev/fd0 /mnt
        echo "작업을 마치고 나서 꼭 umount 하십시요."
```

6.2 여러분 혼자 C 를 배우기

리눅스에서 GW/BASIC 같은 것을 기대하지는 않을 것이다. 그렇지 않은가? 유닉스 계열에서 시스템 언어는 C 이며 C 를 사랑하기도 하고 C 를 미워하기도 한다. 다른 많은 언어들(포트란, 파스칼, LISP, 베이식... 하지만 터보 파스칼 같은 것은 없다)이 또한 사용가능하다.

여러분이 C 를 안다고 가정하고 여기 터보 C++ 이나 다른 도스용 C 에 길들여져 있 을 여러분에게 몇 가지 사항에 대해 안내를 하고자 한다. 리눅스에서의 C 컴파일러는 gcc 이며 보통 도스용에서 볼 수 있는 그러한 화려한 면들은 없다. 통합환경(IDE), 온라인 도움말, 통합 디버거 등은 찾아볼 수 없다. 단지 투박한 명령행 방식의 컴파일러이지만 매우 강력하고 효율적이다. 여러분의 일반적인 hello.c 를 컴파일하기 위해서는 다음과 같이 한다.

```
$ gcc hello.c
```

결과는 a.out 이라는 실행화일이다. 실행화일의 이름을 다르게 주기 위해서는

```
$ gcc -o hola hello.c
```

라이브러리와 함께 프로그램을 링크하려면 -l<arg> 스위치를 붙인다. 예를 들어 수학 라이브러리를 링크하는 경우에는

```
$ gcc -o mathprog mathprog.c -lm
```

(The -l<arg> switch forces gcc to link the library /usr/lib/lib<arg>.a; so (-l<arg> 스위치는 gcc 로 하여금 /usr/lib/lib<arg>.a 를 링크하도록 명한다. 따라서 -lm 은 /usr/lib/libm.a 를 링크하게 된다)

자그마한 프로그램에는 괜찮다. 하지만 여러 개의 소스로 구성된 프로그램을 만들 때는 make 유틸리티를 써야 할 것이다. 여러분이 수학식 파서(parser)를 만들었다 고 치자. 소스 화일의 이름은 parser.c 라고 하고 parse.h와 xy.h 2 개의 헤더 화일 을 포함한다고 치자. 여러분이 calc.c 라는 프로그램에서 parser.h 를 다시 포함하 고 parser.c 의 루팅을 이용하려고 할 것이다. 좀 복잡하긴 하다. calc.c 를 컴파 일하려면 어떤 일을 해야 하는가?

여러분은 makefile 이라는 것을 작성해야 한다. 그 makefile 은 소스와 오브젝트 화일과의 연관성을 말해준다. 예제를 보면,

```
# 이것은 calc.c 를 컴파일하기 위한 makefile 이다.
# 적절한 곳에서 <탭>키를 누르기 바란다.
calc: calc.o parser.o
<탭>gcc -o calc calc.o parser.o -lm
# calc 는 calc.o, parser.so 이 2 개의 오브젝트 화일에 의존한다.
calc.o: calc.c parser.h
<탭>gcc -c calc.c
# calc.o 은 2 개의 화일에 의존한다.
parser.o:  parser.c parser.h xy.h
<탭>gcc -c parser.c
```

```
        # parser.o 는 3 개의 화일에 의존한다.
        # makefile 의 끝
makefile 로 저장하고 컴파일하기 위하여 다음과 같이 실행한다.
        $ make
또는 calc.mak 라고 저장한 다음, 다음과 같이 한다.
        $ make -f calc.mak
물론 관련된 문서를 꼭 읽어보기 바란다. (RMP)
이것 말고도 여러분은 C 함수에 대한 도움을 그 즉시 얻을 수 있는데 다음과 같이
하면 된다. 예를 들어,
        $ man printf
```

제 7 절 나머지 1 퍼센트

7.1 루트로부터 벗어나기

어떤 일을 할 때마다 루트로 로그인하는 것은 매우 안좋다. 매우 위험하며 오로지 시스템 관리할 때만 사용해야 한다. 여러분에게 새로운 계정을 부여할 때는 루트로 로그인한 후에 다음과 같이 명령한다.

 # adduser

물어오는 질문에 하나씩 답변한다. 이해하지 못하는 질문에 대해서는 그냥 <엔터> 를 치면 기본값이 사용된다.

7.2 가상 메모리 만들기

Although Linux can theoretically run with only 2 megs of RAM, the more you 리눅스가 이론적으로는 램 2 메가에서도 작동한다고는 하지만 램이 많을 수록 더 많은 일을 할 수 있다. 최소한 8 메가는 갖추어야 X 윈도우 시스템을 운영할 수 있다. 추가로 가상메로리를 만들고자 한다면 다음과 같이 한다.

```
        # dd if=/dev/zero of=/swapfile bs=1024 count=8192
        # mkswap /swapfile 8192
        # sync
        # swapon /swapfile
```

/etc/rc.d/rc.local 의 마지막 줄에서 매번 부팅할 때마다 스왑화일을 작동시켜주 기 바란다.

7.3 tar와 gzip 사용하기

유닉스에서는 화일을 한데 모으고 압축하는데 사용되는 많은 프로그램들이 있다. tar 은 화일을 한데 모으는데 사용된다. PKZIP가 비슷하다고 할 수 있지만 압축을 행하지는 않는다. 단지 한 데 모을 뿐이다. 새로운 저장화일을 만들려면,

$ tar -cvf <archive_name.tar> <file> [file...]

저장화일로부터 화일을 빼내려면: $ tar -xpvf <archive_name.tar> [file...]

저장화일의 내용을 보려면: $ tar -tf <archive_name.tar> | less

더 이상 쓰이지지는 않는 compress나 그와는 별도로 gzip 을 사용하여 화일을 압축 할 수 있다. $ compress <file> 또는 $ gzip <file> 확장자가 .Z(compress) 또는 .gz(gzip)인 화일을 생성한다.

이 프로그램들은 한 번에 하나의 화일만 압축한다. ([역자 주: 이는 틀린 말이다. gzip * 이라고 하면 현재 디렉토리에 있는 모든 화일을 압축한다) 압축을 풀려면 다음과 같이 한다.

$ compress -d <file> 또는 $ gzip -d <file> 나머지는 관련 매뉴얼 페이지를 읽어보기 바란다.

arj, zip, unzip(PK??ZIP 호환) 유틸리티 또한 사용가능하다. .tar.gz 또는 .tgz 확장자를 가진(tar 로 묶고 gzip 으로 압축된 화일)은 도스에 서의 .ZIP 화일만큼이나 널리 쓰인다.

.tar.gz 화일의 내용을 살펴보려면 다음과 같이 한다. $ gzip -dc <file.tar.gz> | tar tf - | less

7.4 어플리케이션 설치

Most Linux applications are distributed as .tar.gz archives, specifically 많은 리눅스 어플리케이션들은 .tar.gz 의 형태이며 특별히 다음과 같이 하여 / 디렉토리에서 압축을 풀어서 설치하곤 한다.

 # gzip -dc <file.tar.gz> | **tar xvf -** 이렇게 하면 압축된 당시와 똑같이 제대로 된 디렉토리에 설치될 것이다. 괜찮지 않은가? 슬랙웨어 배포판의 사용자(모르긴 하지만 다른 배포판의 사용자도) pkgtool 과 같은 편리한 프로그램을 가지고 있을 것이다. 다른 패키지들은 자기 자신만의 인스톨 방법을 가지고 있다. 문서를 읽어보기 바란 다. 이외에도 어떤 패키지들은 C 또는 C++ 소스 화일로 배포되며 따라서 바이너리로 컴파일해야 한다. 대부분의 경우 다음과 같이 하면 된다.

 # make 여러분에게는 대부분의 배포판에 들어있는 gcc 컴파일러가 필요하다. 패키지를 인스톨하는 일은 루트(root)의 업무라는 사실을 기억하기 바란다.

7.5 필수불가결의 묘수

명령 완성 : <탭>을 누르면 여러분을 위해 자동으로 화일명을 완성해준다. 예를 들어 gcc this_is_a_long_name.c 라고 쳐야 할 때 여러분은 단지 gcc thi<탭> 이렇게 하면 된다. (thi라는 이름으로 시작하는 다른 화일들이 있다면 구별이 가능 한 만큼 타이핑한 다음 <탭>을 누르면 된다.) 백스크롤 : <쉬프트>+<PGUP>(회색 키)을 누르면 몇 페이지만큼 뒤로 스크롤한 화 면을 보여준다. 그 양은 여러분이 갖고 있는 비디오 메모리량과 관계된다. 그리고 다음과 같이 해주면,

$ script <script_file>

이 이후에는 똑같은 명령을 내리기 전까지 모든 화면의 내용이 script_file 로 저 장되어 나중에 이용할 수 있게 된다. 화면 재정리 : 만약 바이너리 화일을 more 나 cat 하면 여러분의 화면이 온통 이상한 문자로 가득 차 버릴 것이다. 이 문제를 해결하기 위해서는 보이지 않더라 도 다음과 같이 타이핑한다.

$ reset

또는 echo CTRL-V ESC c RETRUN 을 순서대로 쳐준다. 커널로부터의 메세지 : 부팅 메세지를 비롯하여 커널이 여러분에게 말하는 바를 보기 위해서는 /var/adm/messages 를 살펴보면 된다.

7.6 유용한 프로그램들

First of all, where to find them. Since you all know how to surf the Net 무엇보다도 프로그램들을 어디서 찾을 것인가? 네트워크를 서핑하는 방법 그리고 archie 나 ftp 사용법을 알고 있을 것이므로 리눅스에서 중요한 다음 3 가지 사이 트를 알려주겠다.

sunsite.unc.edu `<http://sunsite.unc.edu>`, TSX-11.mit.edu, nic.funet.fi. 그것보다는 가까운 미러사이트를 이용하기 바란다.

at

은 특정 날짜/시간에 프로그램을 실행시킬 수 있도록 해준다. delete-undelete 는 그 이름이 의미하는 바를 해준다.

df

는 하드디스크의 용량 등 정보를 보여준다.

dosemu

은 전부는 아니더라도 몇몇 도스 프로그램을 실행시킬 수 있도록 해준다. 그리고 심지어 윈도우즈 3.x 도 제한적을 실행된다.

file

<filename> 은 여러분에게 화일이 어떤 형식인지를 알려준다. (아스키 화일 인지 실행화일인지 저장 화일인지 등)

fine

(섹션 2.2 참고)는 가장 강력하고 유용한 명령 중 하나라고 할 수 있다. 몇 가지 속서와 매치되는 화일을 찾아서 원하는 조치를 취할 수 있다. $ find <directory> <expression> <expression>은 검색할 기준 또는 행위를 포함한다. 예를 들어:

$ find . -type l -exec ls -l {} \; 위의 예는 심볼릭 링크를 모두 찾아서 그것이 지칭하는 바를 보여준다.

$ find / -name "*.old" -ok rm {} \; 위에서는 주어진 패턴(*.old)에 맞는 화일을 찾아서 여러분에게 지울 것인지 물어 보고 지운다.

$ find . -perm 755 화일허가권(755) 즉 실행화일을 모두 찾아준다.

$ find . -user root

root

사용자가 소유한 화일을 모두 보여준다. 이것말고도 정말 다양한 사용법이 있으므로 매뉴얼을 읽어 보기 바란다. grep 은 화일 내의 텍스트 패턴을 찾아준다. 예로는,

$ grep -l "geology" *.tex *.tex 화일에서 geology 라는 단어를 포함한 줄을 보여준다. RMP.

gzexe

는 압축 실행화일을 만들어준다. 즉 실행시 자동으로 압축이 풀려서 실행되는 화일을 만들어준다.

joe

는 뛰어난 편집기이다. jstar 라고 시작하면 워드스타나 도스 에디터와 같은 키보드 조합을 써서 편집을 할 수 있다.

lpr

<file>은 백그라운드에서 화일을 출력한다. 프린팅 상황을 보기 위해서는 lpq 를 사용한다.

mc

는 매우 유용한 화일 관리자이다.

pine

은 뛰어난 전자우편 프로그램이다.

sudo

allows users to perform some of root's tasks (e.g. formatting and sudo 는 일반 사용자로 하여금 몇몇 루트의 작업을 할 수 있게 해준다.(예를 들어 디스크 포맷이라든가 마운트 등... 나머지에 대해서는 RMP)

zcat

and zless are useful for viewing gzipped text files without ungzipping zcat 과 zless 는 압축을 풀 필요 없이 gzip으로 압축된 텍스트 화일을 보는데 사용 된다. 다음과 같이 사용할 수 있다.

$ zless textfile.gz $ zcat textfile.gz | lpr

7.7 일반적인 확장자와 관련 프로그램들

여러분은 상당히 많은 화일 확장자를 경험하게 될 것이다. 약간은 특별한 것들을 제외하고(예를 들어 폰트 등...) 여기 어떤 게 어떤 것인지 목록을 적어두었다.

- 1 ... .8: 맨페이지. man 을 구하면 된다.
- arj: arj 형식의 저장화일
- dvi: output file produced by TeX (see below). xdvi to visualize it; dvips
- dvi: TeX에 의해 만들어진 출력 화일(아래를 보라). xdvi 는 그것을 보여준다. dvips는 그것을 포스트 스크립트 .ps 화일로 만들어준다.
- gif: 그래픽 화일. seejpeg이나 xpaint 사용
- gz: gzip 형식의 저장화일
- info: info 화일 (맨페이지의 또 다른 형태) info 를 사용한다.
- jpg,.jpeg: 그래픽 화일. seejpeg 을 사용한다.
- ps: 포스트스크립트 화일. 이것을 보기 위해서는 gs 그리고 ghostview 를 구하면 좋다.
- tex: text file to submit to TeX, a powerful typesetting program. Get the
- tex: 강력한 조판 프로그램인 TeX 용 텍스트 화일. tex 용 패키지를 구하면 된다. 많은 배포판에 이미 들어있다.
- texi: texinfo 화일(cp. .info). texinfo 를 구하라.
- xbm,.xpm,.xwd: 그래픽 화일. xpaint를 구하라.
- zip: zip 으로 만든 저장화일. zip와 unzip을 구한다.
- Z: compress로 압축한 저장화일

7.8 일단 여기까지....

축하한다! 여러분은 유닉스에 관하여 지금까지 몇 가지 사항을 알아보았고 써볼 준비가 되어있다. 물론 아직까지도 상당히 제한된 지식이라는 것을 기억하고 편하게 사용할 수 있도록 리눅스에서 많은 연습이 필요하다. 하지만 대부분 많은 어플리케 이션만을 다루고 작업하기만을 원한다면 여기 소개한 것만으로도 충분하리라. 아직까지도 그렇게 많은 것을 알고 있지는 않지만 매일매일 리눅스를 잘 사용하고 있다. I'm sure you'll enjoy using Linux and will keep learning more about it - 본인은 여러분이 리눅스를 즐겁게 사용하고 더 많은 것을 계속 배울 것이라고 확신 한다. - 모두들 그러 듯... 또한 여러분이 도스로는 절대 되돌아가지 않으리란 것을 장담한다. 이 글을 읽는 3, 4 명의 독자들에게 내 글이 도움이 되었길 바란다.

7.9 성명서

"From DOS to Linux - Quick!"은 Guido Gonzat 씨에 의해 1996년 4월에 씌여졌다 <*Guido@ibogfs.cineca.it* `<mailto:Guido@ibogfs.cineca.it>`>. "Linux Installation and Getting Started"의 저자 Matt Welsh씨, "Linux frequently asked questios with answers"의 저자 Ian Jackson 씨, "Linux - Il sistema operativo FREE SOFTWARE per persnoal computer 386/486"의 저자인 Giuseppe Zanetti씨, 그리고 내게 전자우편으로 많은 제안을 해준 많은 사 람들에게 감사한다. 그리고 특히 리눅스를 우리에게 가져다준 리누스 토르발즈와 GNU에 감사한다.

 어떤 요청이나 제안, 비난 등... 마음놓고 본인에게 연락하기 바란다. 전자우편을 사용할 수 없다면 그냥 느려터진 우편으로 보내도 좋다. Guido Gonzato - Via Monte Ortigara 19/a, 37127 Verona - Italy

삶을 즐기시길... Guido =8-)