레드햇 리눅스 킥스타트 하우투

Martin Hamilton <martinh@gnu.org> v0.2, 1999년 1월 11일

이 규 성 <pgclks@hitel.net> 1999년 5월 31일

이 하우투는 다수의 동일한 리눅스 박스에 빠르게 리눅스를 설치할 수 있는 레드햇 리눅스 킥스타트를 사용하는 방법에 대하여 간략히 설명한다. 어느 정도 숙련된 사용자들을 위해 자신만의 킥스타트 설치 과정을 수정하는 법과 자신만의 RPM 패키지를 만들기 위한 지름길을 제공한다.

목 차

제 1 절 저작권

이 문서의 저작권은 Martin Hamilton에게 있다. 이 문서는 *GPL* <http://www.gnu.org/copyleft/gpl.html>에 의거하여 자유롭게 배포될 수 있다.

제 2 절 홈페이지

만일 이 문서를 리눅스 하우투 미러 사이트나 CD-ROM을 통해 입수하였다면 *킥스타트 하우투 홈페이지* <http://wwwcache.ja.net/dev/kickstart/>를 방문하여 새로운 버전이 있는지 확인해 보라.

제 3 절 소개

레드햇 리눅스 5.x 버전은 아직까지는 잘 알려지지 않고 문서화가 잘 되어 있지 않은 킥스타트라는 기능을 제공한다. 이 기능은 다음과 같은 레드햇 리눅스 설치의 대부분 또는 모든 것을 자동으로 처리해 준다.

- 언어 선택
- 네트웍 설정과 배포판 소스 선택
- 키보드 선택
- 부트 로더 설치 (예 lilo)
- 디스크 할당과 화일시스템 생성
- 마우스 선택
- X 윈도우 시스템 서버 설정
- 시간대 선택
- (초기) root 패스워드 설정
- 설치할 패키지들

눈썰미가 있는 레드햇 사용자라면 여기까지 보고 위의 내용이 수동 설치시 필수적인 주된 단계라는 것을 알아차릴 것이다. 킥스타트는 여러분들이 일반 적으로 키보드로부터 입력할 정보들을 설정화일에 넣음으로써 설치과정을 스크립트화할 수 있다.

하지만 기다려라 - 아직 많은 기능이 있다 :-)

일반적인 설치과정을 마쳤다면 킥스타트는 또한 실행하고자하는 쉘 수준의 명령어를 지정할 수 있게 해준다. 이는 레드햇 리눅스 배포본에서는 얻을 수 없는 많은 공개 프로그램들과 법적인 이유로 레드햇 리눅스에 포함될 수 없는 ssh와 PGP 암호화 체계등과 시스템이 보다 완벽하게 작동하도록 하는 소프트웨어들을 자동적으로 설치할 수 있음을 의미한다.

제 4 절 준비 사항

킥스타트 설치로의 접근방법으로는 두가지가 있다 - 하나는 단순히 킥스타트 구성 화일을 레드햇 부트 플로피에 복사하는 방법이다. 또 다른 한가지의 방법은 레드햇 부트 플로피를 사용하여 킥스타트 구성 화일을 네트웍으로 이용하는 방법이다.

두가지 경우 모두 다음을 만족해야 한다:

1. 인텔 (i386) 계열 머신 - 이 글을 쓰는 지금까지는 이들 머신에서만 작동하는 것으로 알려져 있다.
2. 킥스타트 설정 화일 - 설정에 대해서는 다음 섹션에서 다룰 것이다.
3. 레드햇 부트 디스크 - updates 디렉토리에서 버그 및 드라이버가 업데이트된 부트디스크를 이용할 수 있다.
4. 사용할 IP 주소를 위한 DNS 항목 - 선택항목이지만 머신의 도메인 이름을 입력받기 위해 설치를 멈추고 입력대기 상태가 될 수 있다.

만일 설정 화일을 네트웍을 통해서 가져오길 원한다면, NFS를 통해 공유를 해야 한다. 이 방법이 현재 지원되는 유일한 접근 방법이다. 설정 화일은 레드햇 배포본 자체를 가져올 다른 NFS 서버를 지정할 수 있게 해준다.

킥스타트 설치를 위해 저장된 특정한 IP 주소를 사용하는 것처럼 머신에 고정 IP 주소를 설정할 수 있다. 또는 설정 화일에 IP 주소를 적어주기를 원하지 않는다면 BOOTP/DHCP 서버를 이용하여 IP 주소를 받아오도록 할 수 있다. 예로 *동적 IP 주소 할당 기능을 가진 CMU BOOTP 서버* `<ftp://ftp.ntplx.net/pub/networking/bootp>`는 주어진 범위안에서 자동으로 새로운 주소를 할당할 것이다.

NFS 및 BOOTP/DHCP에 관한 자세한 정보는 부록 A에 있다.

제 5 절 부트 플로피 설정하기

요점을 말하자면 여러분들이 해야 할 것은 킥스타트 구성 화일을 레드햇 부트 플로피에 /ks.cfg라는 이름으로 복사하는 것이다. 예를 들자면

```
mcopy ks.cfg a:
```

그러나 레드햇 부트 플로피는 거의 사용가능 공간이 없기 때문에 킥스타트 구성 화일을 위한 공간을 만들기 위해서는 다른 화일을 지워야 한다는 것을 알게 될 것이다. 필자는 SYSLINUX 부트 로더에 의해 보여지는 다양한 메세지를 지움으로써 내게 필요한 충분한 공간을 얻을 수 있었다. 예를 들자면

```
mdel a:\*.msg
```

또다른 방법으로는 자신이 갖고 있지 않는 하드웨어용 드라이버를 지우는 방법이 있다 - 이 문서의 부트 플로피 수정에 관련된 섹션을 보기 바란다.

또한 SYSLINUX 구성 화일인 syslinux.cfg 화일을 편집해야 한다. 이 화일 또한 레드햇 부트 플로피의 최상위 디렉토리에 있다. 예를 들어 다음과 같은 syslinux.cfg 내용은 머신이 부팅하자마자 지체없이 킥스타트 모드로 진입 하게 할 것이다.

```
default ks
prompt 0
label ks
  kernel vmlinuz
  append ks=floppy initrd=initrd.img
```

아마도 거의 대부분이 부트 및 보충 플로피를 가장 최근의 것을 기반으로 만들고자 할 것이다. 가장 최근의 이미지는 각 지역의 레드햇 미러 사이트의 updates/i386 디렉토리에서 구할 수 있다. 구버전의 이미지는 버그가 있거나 하드웨어 드라이버를 제대로 지원하지 않을 수 있다.

제 6 절 킥스타트 설정 화일

설정 화일에는 3개의 주 섹션이 있다:

1. 시스템 정보, 예를 들면 디스크 분할 및 네트웍 설정
2. 설치할 레드햇 패키지
3. 설치-후-쉘명령

우리가 여기서 다루지 않는 부분이 있을 수도 있지만 작동은 할 것이다. 좀더 자세한 정보를 원한다면 레드햇 배포본 씨디롬 또는 레드햇 미러 사이트의 루트 디렉토리 아래의 misc/src/install/ks.samp와 doc/README.ks 등의 킥스타트 설정 샘플을 참고하기 바란다.

6.1 시스템 정보

필자가 사용한 사용가능한 지시자는 다음과 같다:

lang

언어 설정. 예를 들어 영어의 경우

```
lang en
```

network

네트웍 설정. 예를 들어 BOOTP/DHCP를 사용하려면

```
network --bootp
```

nfs

NFS 서버와 설치할 매체의 위치. 예를 들어

```
nfs --server chicken.swedish-chef.org /mnt/cdrom
```

chicken.swedish-chef.org를 NFS 서버로 사용하고 레드햇 배포본을 /mnt/cdrom에 마운트한다.

keyboard

키보드 타입 선택. UK 키보드를 사용한다면

```
keyboard uk
```

zerombr

마스터 부트 레코드(MBR) 내용 비우기 - MBR에 존재하는 시스템 부트 로더를 지운다.

clearpart

존재하는 파티션 지우기 - 예를 들어 설치에 앞서 디스크 상의 파티션을 모두 지우려면 다음과 같이 해준다.

```
clearpart -all
```

part

디스크 분할. 예를 들어 500MB의 루트 화일 시스템을 만들기 위해서는

```
part / --size 500
```

install

레드햇 리눅스를 새롭게 설치한다.

mouse

사용하는 마우스 설정. 예를 들어 PS/2 또는 "bus mouse" 호환 기종이라면

```
mouse ps/2
```

timezone

시간대 설정. 예를 들어 영국에서 시간대를 설정한다면

```
timezone --utc Europe/London
```

rootpw

초기 root 패스워드 설정. 이 때의 패스워드는 이미 암호화된 패스워드가 사용된다.

```
rootpw --iscrypted XaacoeGPmf/A.
```

lilo

LILO 부트 로더 설치. MBR에 설치하려 한다면

```
lilo --location mbr
```

%packages

설치할 패키지 - 아래를 보기 바란다.

%post

설치 후 실행할 쉘 명령 - 아래를 보기 바란다.

킥스타트가 레드햇 배포판을 찾을 디렉토리는 플랫폼을 위한 레드햇 배포판 트리구조를 갖고 있는 RedHat이라는 하위디렉토리를 갖고 있어야 한다. 위의 예제에서는 다음과 같은 화일과 디렉토리를 볼 수 있다.

```
/mnt/cdrom/RedHat
/mnt/cdrom/RedHat/base
/mnt/cdrom/RedHat/contents
/mnt/cdrom/RedHat/i386
/mnt/cdrom/RedHat/instimage
/mnt/cdrom/RedHat/RPMS
/mnt/cdrom/RPM-PGP-KEY
```

네트웍으로 설치하지 않고 CD-ROM으로 설치를 하려한다면 그 내용은 다음과 같이 될 것이다.

```
RedHat
RedHat/base
RedHat/contents
RedHat/i386
RedHat/instimage
RedHat/RPMS
RPM-PGP-KEY
```

다양한 아키텍쳐를 위한 레드햇 배포판을 가지고 있다면 각각의 배포판은 서브 디렉토리 아래에 같은 화일을 갖고 있음을 알아두어야 할 것이다.(예를 들어 NFS 서버상에서 아키택쳐별 버전이 하나 이상이어서 하나의 CD-ROM에 담을 수 없을 경우)

```
alpha/RPM-PGP-KEY
i386/RPM-PGP-KEY
sparc/RPM-PGP-KEY
```

i386/Redhat/i386과 같은 아키텍처/Redhat/아키텍처의 형태를 갖는 화일이 있어야 한다.

자신만의 암호화된 패스워드를 만들고 싶다면, Perl을 이용하면 매우 쉽다.

```
% perl -e 'print crypt("schmurrdegurr", "Xa") . "\n";'p
```

다음은 테스트는 해보지 않았지만 논의할 만한 가치가 있는 다른 옵션이다:

cdrom

네트웍 설치가 아닌 CD-ROM 설치

device

장치에 대한 정확한 지정 예;

```
device ethernet 3c509 --opts "io=0x330, irq=7"
```

device의 또다른 값으로는 SCSI 컨트롤러를 위한 scsi와 각종 CD-ROM 드라이브를 위한 cdrom이 있다.

upgrade

새로 설치하지 않고 기존의 설치된 시스템을 업그레이드

xconfig

X 윈도우 서버 설정, 그래픽 카드와 모니터 설정

```
xconfig --server "Mach64" --monitor "tatung cm14uhe"
```

마지막의 항목에 대해서는 깊이 있게 연구하지 않았다. 왜냐면 킥스타트를 적용할 머신의 콘솔에서 X를 실행하려 하지 않았기 때문이다. 킥스타트에서 xconfig를 실행한다는 것은 다소 색다르다고 들었다. 하지만 Xconfigurator를 통해서 명령행으로 같은 기능을 사용할 수 있다. 따라서 이 부분은 후-설치- 스크립트에서 하는 것이 가장 좋을 것이다.

다음은 킥스타트 설정 화일의 처음부분의 모든 항목을 입력했을 때의 설정 파일의 내용이다.

```
lang en
network --static --ip 198.168.254.253 --netmask 255.255.255.0
  --gateway 198.168.254.1 --nameserver 198.168.254.2
nfs --server chicken.swedish-chef.org /mnt/cdrom
keyboard uk
zerombr yes
clearpart --all
part / --size 500
part swap --size 120
install
mouse ps/2
timezone --utc Europe/London
rootpw --iscrypted XaacoeGPmf/A.
lilo --location mbr
```

레드햇 문서중의 몇몇 문서는 실무에서 실제로 작동하지 않는 `network` 명령의 옵션에 대해 설명하고 있다는 것을 알아두기 바란다: `network --option`. 올바른 사용법은 network 뒤에 –static이나 –bootp 또는 –dhcp를 사용하는 것이다. BOOTP와 DHCP는 다른 코드를 사용하므로 두 옵션이 다르다는 것을 알아두기 바란다.

part 지시자에 지정한 파티션 크기보다 늘일수 있도록 해주는 매개변수인 –grow를 추가할 수도 있다. 파티션 이름 다음에 –grow를 추가하는 것이다.

6.2 설치할 패키지

킥스타트 설정 화일의 패키지 섹션의 시작은 `%packages` 지시자로 시작을 한다. 이 지시자에는 두가지 형태의 패키지 구분자가 따라온다. 버전과 플랫폼 정보를 제외한 설치될 패키지의 이름과 그룹이름으로 주어지는 설치될 패키지의 그룹의 형태가 그것이다.

다음은 킥스타트 설정 화일을 위한 패키지 선택의 예이다.

```
%packages
@ Base
netkit-base
bind-utils
ncftp
rdate
tcp_wrappers
traceroute
cmu-snmp
```

그렇다면 이 그룹들은 무엇인가? 레드햇 배포판의 최상위 디렉토리의 base/comps 화일에 기본으로 정의된 그룹이 있다. 다음은 글을 쓰고 있는 현재까지의 그룹의 목록이다.

- Base
- Printer Support
- X Window System
- Mail/WWW/News Tools
- DOS/Windows Connectivity
- File Managers
- Graphics Manipulation
- X Games
- Console Games
- X multimedia support
- Console Multimedia
- Print Server
- Networked Workstation
- Dialup Workstation
- News Server

- NFS Server
- SMB (Samba) Connectivity
- IPX/Netware(tm) Connectivity
- Anonymous FTP/Gopher Server
- Web Server
- DNS Name Server
- Postgres (SQL) Server
- Network Management Workstation
- TeX Document Formatting
- Emacs
- Emacs with X windows
- C Development
- Development Libraries
- C++ Development
- X Development
- Extra Documentation

이것은 수동 설치를 할 때 볼 수 있는 다양한 설정과 관련이 있다는 점을 알 수 있을 것이다. 한 그룹을 선택했을 때 또 다른 그룹을 설정하게 되면 중복이 되는 패키지가 있다는 점과 이러한 중복의 문제를 일으키지 않고 여러 그룹들을 설치할 수 있다는 점을 알아두기 바란다. comps 화일에 나열된 각 그룹의 항목은 다음과 같은 형태를 갖는다.

```
0 Extra Documentation
sag
lpg
howto
faq
man-pages
end
```

첫 줄의 그룹이름 앞의 1은 기본 설치되는 그룹을 나타낸다. 자신만의 그룹을 생성하거나 기존의 그룹을 편집하여 재정의함으로써 리눅스 설치를 최적화할 수 있다.

6.3 설치-후-쉘명령

이 기능이 아마도 무엇보다도 훌륭한 기능이리라 생각된다. 수동으로 설치시 이와 같은 기능이 없을 것이다. 여기에 디스크 분할, 패키지 설치등의 주된 설치 과정 후에 실행되어질 쉘 수준의 명령을 지정할 수 있다.

이 섹션의 시작은 킥스타트 설정화일에서 %post 라는 지시자로 구별된다. 지시자 다음에는 새로 설치한 리눅스 시스템에 설치된 유틸리티를 이용할 수 있다. 예를 들면

```
%post
ln -s /etc/rc.d/init.d /etc/init.d
ln -s /etc/rc.d/rc.local /etc/rc.local
ln -s /usr/bin/md5sum /usr/bin/md5
ln -s /usr/bin/perl /usr/local/bin/perl
chmod ug-s /bin/linuxconf
mkdir /var/tmp/tmp
perl -spi -e 's!image=/boot/vmlinuz-.*!image=/boot/vmlinuz!' /etc/lilo.conf
rm /etc/rc.d/rc*.d/*sendmail
```

또한 다음과 같이 입/출력 리다이렉션을 사용할 수 있다:

```
cat << EOF >>/etc/passwd
squid:*:102:3500:Squid Proxy:/usr/squid:/bin/bash
EOF

cat << EOF >>/etc/group
cache:x:3500:
EOF
```

run-time 시작 스크립트 수정:

```
cat << EOF >>/etc/rc.local
echo 8192 > /proc/sys/kernel/file-max
echo 32768 > /proc/sys/kernel/inode-max

[ -x /usr/sbin/sshd ] && /usr/sbin/sshd
[ -x /usr/sbin/cfd ] && /usr/sbin/cfd

EOF
```

crontab 내용 설정 :

```
cat << EOF >/tmp/crontab.root
# Keep the time up to date
0,15,30,45 * * * * /usr/sbin/ntpdate -s eggtimer 2>&1 >/dev/null
# Recycle Exim log files
1 0 * * * /usr/exim/bin/exicyclog
# Flush the Exim queue
0,15,30,45 * * * * /usr/exim/bin/exim -q
EOF

crontab /tmp/crontab.root
rm /tmp/crontab.root
```

자신이 만든 다른 RPM 화일도 설치할 수 있다:

```
rpm -i ftp://chicken.swedish-chef.org/rpms/squid.rpm
rpm -i ftp://chicken.swedish-chef.org/rpms/ssh.rpm
rpm -i ftp://chicken.swedish-chef.org/rpms/exim.rpm
rpm -i ftp://chicken.swedish-chef.org/rpms/cfengine.rpm
rpm -i ftp://chicken.swedish-chef.org/rpms/linux.rpm

ssh-keygen -b 1024 -f /etc/ssh_host_key -N ""
depmod -a
```

실행하고자 하는 명령을 담은 자신만의 RPM을 만듦으로써 같은 효과를 얻을 수 있다는 점을 알아두기 바란다. - 문서의 후반부에서 언급될 것이다. 신중히 선택한 이름을 주어 자신이 만든 RPM을 처음으로 설치(예: 'aaa'로 시작하는 이름)하거나 맨 마지막(예: 'zzz'로 시작하는 이름)에 설치되도록 할 수 있다.

root crontab 목록을 만드는 비교적 수월한 방법은 crontab 내용을 화일로 만들어 /etc밑의 cron.hourly, cron.daily, cron.weekly, cron.monthly의 이름의 디렉토리 아래에 위치시키는 것이라는 점을 알아두기 바란다.

자신만의 RPM을 만드는 것에 대한 자세한 정보는 부록 B에서 얻을 수 있다.

제 7 절 자동 설치

평상시처럼 레드햇 부트 플로피로 설치할 머신을 부팅하여, `SYSLINUX` 프롬프트에서 `RETURN`을 누르지 말고 `linux ks`를 입력하라.

운이 좋다면, 위에서 한 내용이 입력해야 할 전부이다.

만일 레드햇 부트 플로피를 위에서 말한대로 수정을 했다면 아무것도 할 일이 없을 것이다.

레드햇 설치과정에서 수행되는 일반적인 단계를 자동화시켰지만, 킥스타트가 다음에 무엇을 어떻게 해야할 지 혼돈스러울 경우엔 대화상자가 나타나게 될 것이다. 거의 대부분의 경우가 네트웍 인터페이스가 자동으로 인식되지 않는 경우로 IRQ와 I/O 주소를 위해 입력대기 상태가 될 것이다. 킥스타트는 ISA 버스 카드에 대한 도움을 얻으려 하지만 PCI 버스 카드는 자동으로 인식 한다.

킥스타트가 무슨 일을 하고 있는지는 평상시처럼 가상콘솔을 바꿈으로 해서 볼 수 있다.

- Alt-F1 - 설치 대화상자
- Alt-F2 - 쉘 프롬프트
- Alt-F3 - 설치 로그 (설치프로그램으로 부터의 메시지)
- Alt-F4 - 시스템 로그 (커널등으로부터의 메시지)
- Alt-F5 - 기타 메시지

제 8 절 부트/보충 디스크 마운트하기

레드햇 부트 디스크 boot.img는 부팅을 위한 프로그램인 `SYSLINUX`를 이용한 MS-DOS 포맷이다. 보충 디스크 supp.img는 리눅스의 `ext2` 화일시스템으로 되어있다. 자신의 리눅스 커널이 loopback 화일시스템을 지원한다면 이들 두개의 화일들을 여러분들의 화일시스템에 마운트하여 내용을 고쳐볼 수도 있다:

```
# mkdir -p /mnt/boot /mnt/supp
# mount -o loop -t msdos boot.img /mnt/boot
# mount -o loop supp.img /mnt/supp
```

이제 부트 디스크와 보충 디스크에 포함된 화일들을 각각 /mnt/boot /mnt/supp 디렉토리 아래에서 볼 수 있으며 수정을 할 수도 있다. 잠깐! `mount`의 옛 버전은 `-o loop` 옵션을 처리하지 못할 수도 있다. 이런 경우엔 각 화일들을 loopback 장치로 설정하는 `losetup`을 이용해야 할 것이다.

```
# losetup /dev/loop0 boot.img
# mount -t msdos /dev/loop0 /mnt/boot
```

보충 디스크(supplementary disk)에 있는 ext2 화일시스템을 마운트 할때는 -t ext2를 사용해야 할 경우도 있다. 하지만 요즘 리눅스 배포본을 사용하면 이 점에 대해서는 걱정하지 않아도 될 것이다.

물론 여기저기 많이 손대고 싶지 않다면 플로피 디스크 이미지를 손대기보다 실제 사용하는 플로피 디스크를 편집할 수도 있다. 만일 시간에 쫓긴다면 원본 플로피 디스크의 읽기/쓰기와 관련한 지체현상을 초래함이 없이 디스크 이미지를 편집할 수 있기 때문에 루프백 장치를 사용하는 것도 좋을 것이다.

제 9 절 레드햇 설치관리자(RedHat Installer) 수정

만일 설치 과정 자체를 수정하려 한다면 소스 코드를 레드햇 CD-ROM이나 각 지역의 레드햇 미러사이트에서 찾을 수 있다. 소스 코드는 i386 배포본의 최상위 디렉토리의 misc/src/install 디렉토리 아래에 있다.

레드햇 부트 디스크를 살펴보면 리눅스 커널인 vmlinuz 이외에도 큰 용량의 initrd.img등의 화일이 있음을 알 수 있다.

```
- -rwxr-xr-x   1 root     root         559 May 11 15:48 boot.msg
- -rwxr-xr-x   1 root     root         668 May 11 15:48 expert.msg
- -rwxr-xr-x   1 root     root         986 May 11 15:48 general.msg
- -rwxr-xr-x   1 root     root      968842 May 11 15:48 initrd.img
- -rwxr-xr-x   1 root     root        1120 May 11 15:48 kickit.msg
- -r-xr-xr-x   1 root     root        5352 May 11 15:48 ldlinux.sys
- -rwxr-xr-x   1 root     root         875 May 11 15:48 param.msg
- -rwxr-xr-x   1 root     root        1239 May 11 15:48 rescue.msg
- -rwxr-xr-x   1 root     root         402 May 11 15:48 syslinux.cfg
- -rwxr-xr-x   1 root     root      444602 May 11 15:48 vmlinuz
```

예상을 했겠지만 initrd.img 화일은 화일로 저장된 ext2 화일 시스템이다. 이 화일은 높은 압축율로 압축되어 있다. 압축을 푼 다음 화일을 마운트할 수 있다.

```
# gzip -dc /mnt/boot/initrd.img >/tmp/initrd.ext2
# mkdir /mnt/initrd
# mount -o loop /tmp/initrd.ext2 /mnt/initrd
```

이 화일시스템에서 가장 중요한 부분은 부트 디스크에 포함된 커널 모듈들의 모음이다. 만일 새 버전의 드라이버를 포함시키려 한다면 모듈을 포함시켜 정적으로 컴파일된 커널로 교체하든지 아니면 모듈의 모음을 교체해야 한다. 어쩌면 공간을 마련하기 위해 다른 모듈을 삭제해야 하는 경우도 있다.

모듈들의 모음은 modules/modules.cgz라는 형태로 존재한다. 이 화일을 어떻게 다루어야 할지 의아해 할 것이다. 이 화일은 압축된 cpio 화일이다. 믿거나 말거나. 실제로 RPM은 내부적으로 cpio를 사용한다. 이 화일은 다음과 같은 순서로 압축을 풀고 cpio를 해제한다.

```
# gzip -dc /mnt/initrd/modules/modules.cgz >/tmp/modules.cpio
# cpio -itv < modules.cpio > modules.listing
# mkdir modules
# cpio -idumv <../modules.cpio
```

내가 알기로는 요즘 많이 사용되는 리눅스 배포본에서는 압축된 화일시스템을 곧바로 접근할 수 있는 방법이 없다. 만일 더 좋은 방법이 있다면 알려주기를 바란다.

내용을 변경하려 한다면 알아두어야 할 것이 있다.

1. archive를 재생성하기 위해 cpio를 사용한다. archive를 재생성하는 것은 이 문서를 읽는 여러분들의 연습을 위해 남겨두기로 한다.

2. 위의 archive를 압축하기 위해 `gzip`을 사용한다.
3. 위의 화일을 /mnt/initrd 또는 initrd.img 화일을 풀어놓은 디렉토리에 복사한다.
4. /mnt/initrd 를 언마운트한다.
5. `gzip`을 이용하여 새로운 initrd.img를 다시 압축한다.
6. 압축된 화일을 부트 디스크 이미지에 넣는다. - 우리가 사용한 예에서는 /mnt/boot/initrd.img
7. 부트 디스크 이미지를 언마운트 한다.

최종적으로 마침내 수정된 부트 디스크 셋업을 사용하여 새로운 부트 플로피를 만들 수 있다.

```
# cat boot.img >/dev/fd0
```

제 10 절 자주 질문되는 질문과 답변/개선희망사항 리스트

Q: 킥스타트 설치후에 부팅을 할 수 없다. BIOS에서 `Missing operating system` 메시지를 낸다.

A: 루트 화일 시스템이 있는 파티션이 부팅할 수 없다는 것 같군요. `fdisk`를 사용해서 부팅가능 상태로 바꿔라.

Q: 플로피로 부팅후 다음과 같은 메시지가 나온다: `Error opening files for kickstart copy: File exists.`

A: boot.img와 supp.img의 최근 버전을 사용하라 - 레드햇 미러 사이트에서 구할 수 있다. 레드햇 5.1을 위한 화일의 옛버전은 버그가 있었다.

Q: 눈에 띄는 패치(업데이트 RPM)를 자동으로 적용할 수 있나용? 어떻게?

A1: 설치하고자 하는 RPM을 설치가 실행되는 디렉토리인 RPMS 디렉토리로 복사하고 이전 버전의 RPM들을 제거한다. 그리고 RedHat/base/hdlist를 새로운 RPM에 맞도록 업데이트시켜라. 위 절차를 자동으로 해주는 Eric Doutreleau의 스크립트는 부록 C를 보면 된다. 스크립트를 실행한다면 후에 genhdlist를 실행하는 것을 잊지 말기 바란다.

A2: 다음의 펄 스크립트를 사용해보라: *patchup* `<http://wwwcache.ja.net/dev/patchup/>`. 이 스크립트는 시스템에 설치된 RPM과 지정한 디렉토리에 있는 RPM을 비교하여 업데이트가 필요한 RPM을 알려준다. 이 스크립트는 업데이트를 해야한다고 판단되면 설치도 할 수 있다.

A3: *rpm2hml* `<http://rufus.w3.org/linux/rpm2html/>` 은 A2(patchup)의 좀 더 강력한(12MB의 C 코드와 한 페이지 분량의 펄) 버전을 가지고 있다.

Q: 모든 클라이언트를 위한 설치 서버상의 설정화일을 IPADDR-kickstart를 시도한 후의 대체시스템으로 사용할 수 있나용?

A1: BOOTP/DHCP의 '부트 화일' 파라미터인 bf를 화일이름으로 설정하라.

A2: /etc/bootptab에 `bf=/kickstart/ks.cfg` 레코드를 적절히 추가하라.

Q: 잘못 되었을 때의 유연성은 있나용? 예를 들자면 CD-ROM의 배포본을 찾을 수 없다면 다른 위치를 입력하도록 알려줄 수 있나용?

A: ?

Q: 패키지를 제외시키려면 어떻게? - 예를 들어 sendmail을 따로 떼어 놓는 등..

A: sendmail 없는 **BASE** 패키지를 다시 만들어라.

Q: /etc/rc.d 아래의 실행레벨 스크립트가 부팅시 자동으로 시작되게 할 수 있나용?

A: chkconfig 유틸리티를 이용해서 부팅시에 자동적으로 실행될 서비스를 설정할 수 있게 해준다. 설치후스크립트 섹션에서 실행할 수 있다. 예를 들어 ypbind 서비스를 실행레벨 3, 4, 5에서 실행하려면:

```
chkconfig --level 345 ypbind on
```

이처럼 하면 ypbind를 345 레벨에서 시작할 것이다.

Q: %post 섹션에서 쉘 명령을 실행할 때 결과 출력물을 메인 화면에 출력하지 않고 또 다른 가상 콘솔로 출력할 수 있나용? 쉘 명령 섹션에서 open을 사용해 가능할까요?

A: 다음과 같이 하면 된다:

```
exec >/dev/tty5
```

Q: 화일 시스템 생성코드는 배드블럭을 검사하는지용?

A: 화일시스템 생성 출력물이 보여지는 가상콘솔로 바꿔보면 'read-only' 테스트가 수행된다는 말을 보지 못할 것이다. 위 질문에 대한 답은 '검사를 하지 않는다'인것 같다.

Q: 내 머신을 다른 머신과 다르게 설정할 수 있나용?

A: 호스트와 관련된 부분을 킥스타트 설정의 스크립트 섹션으로 옮길 수 있다. 예를 들자면 머신에 RPM만 설치하는 식으로... 설정 화일의 패키지 섹션에 조건 설치 기능을 넣는 것도 유용하다. 예를 들자면 아키텍쳐 또는 호스트명/도메인명/IP주소에 따라 전환되는 기능을 넣는 것이다.

Q: 레드햇 5.1과 5.2사이에 변화가 있나용?

A1: 설치관리자(installer)상의 많은 변화가 있지만 대부분 버그수정이나 내부적인 개선이다. 두 버전의 misc/src/install 디렉토리에 대하여 diff -rcs를 수행한 결과를 보면 킥스타트엔 아무 영향이 없다.

A2: 레드햇 5.2에서는 분명히 bootpd에 대한 자동 IP 할당/DHCP 패치가 포함되었으나 사용법을 담고 있는 문서를 빼먹었다.

Q: (어떻게) 특정 파티션이나 파티션들을 지울 수 있지용? 예를 들자면 /home은 놔두고 / 만 설정하는 등..

A: 할 수 없음.

Q: 여러개의 장비들을 건너 뛰면서 파티션을 설정할 수 있나용? 예를 들자면 / 는 sda에 두고 /home 은 sdb에 두는 식으로..

A: 파티셔닝 툴로 첫번째 드라이브로만 접근할 수 있는 것 같다.

Q: 기존의 파티션을 마운트 테이블에 포함되도록 지정할 수 있나용? 아니면 새로 생성되는 파티션만 파티션 테이블에 포함되도록 지정할 수 있나용?

A: ?

Q: mkkickstart를 실행한 하면 화일이 어디에 생성되나용?

A: 화일을 만들지 않는다 - 표준출력(화면)으로 킥스타트 설정을 보여준다. (역자註 : 화일로 만들라면 리다이렉션을 쓰면 되갔지용? e.g. >)

Q: 4번 가상콘솔(Alt-F4)에서 Unable to load NLS charset cp437(nls_cp437) 이란 메세지가 나온다. 이것이 뭔 말인가용? 걱정되네요..

A: Joliet 익스텐션으로 구워진 CD-ROM을 마운트하려 한것 같네요. 이론적으로 CD-ROM상의 화일이름이 깨질 수 있어서 리눅스상의 화일이름을 올바르게 인식하지 못할 수 있다고 한다. 실제적으로는 아무 문제도 없는듯 하다.

Q: 왜 X 윈도우 시스템이 설치되지용? 패키지 목록에 넣지 않았는데요..

A: XFree86-VGA16 RPM은 기본 요소이다. 따라서 base class 정의를 변경하지 않았다면 항상 설치가 된다.

Q: 설치-후-스크립트에서, 플로피의 제한된 툴로는 할 수 없는 일들을 지금까지 설치한 패키지를 이용할 수 있나용?

A: 킥스타트 설정에 Perl을 함께 설치하도록 설정했다면 거의 모든 것이 약 5줄 정도로 가능하다. :-)

제 11 절 도움주신 분들

chkconfig에 대한 정보와 SYSLINUX 설정화일 수정, 배포판 서버의 RPM을 업데이트하는 펄 스크립트를 작성해준 Eric Doutreleau에게 감사를 드린다. 자세한 조사를 해준 Robert Kaminsky에게 감사를 드린다. 의견과 질문을 보내준 Piete Brooks, Flavia Regina Munhoz, Tom Toffoli, Bob Robbins, Charlie Brady, Ragen Herrington에게도 감사를 드린다.

제 12 절 부록 A - BOOTP/DHCP와 NFS 설정하기

도대체 BOOTP와 DHCP가 무엇인지 궁금하다면, 이에 대한 자세한 정보는 *DHCP 웹 사이트* `<http://www.dhcp.org/>`에서 얻을 수 있다. NFS는 NFS 하우투에 따로 자세히 문서화되어 있고, DHCP 미니 하우투 역시 문서화되어 있다. 필자는 주제에 대해 깊이있게 다루지 못하더라도 여러분들이 알기 쉽게 충분히 자세히 적으려고 노력했다.

우리가 다루고 있는 BOOTP/DHCP + NFS 설정에서 킥스타트 설정화일은 IPADDR가 설치할 머신의 IP 주소일 때 BOOTP/DHCP 서버에서 /kickstart/IPADDR-kickstart 로부터 설치하고자 하는 머신에서 NFS로 공유할 수 있어야한다. 예를 들자면 198.168.254.254의 머신을 위해서는 /kickstart/198.168.254.254-kickstart가 필요하다.

BOOTP/DHCP의 응답으로 `bf` 매개변수를 줌으로써 이 위치를 무시할 수 있다. 이를 이용해 다른 머신들과의 NFS 공유를 해제하는 것이 가능할 수도 있다.

기존의 리눅스 박스와 어떤 디렉토리들을 NFS로 공유하고자 한다면 다음과 같은 내용을 갖는 /etc/exports을 만들기 바란다:

```
/kickstart *.swedish-chef.org(ro,no_root_squash)
/mnt/cdrom *.swedish-chef.org(ro,no_root_squash)
```

만일 DNS에서 사용하고자 하는 IP 주소를 등록하지 않았다면 NFS 서버나 RPC portmapper로부터 찾을 수 없다는 메세지를 얻게 될 것이다. 이런 경우에는 IP 주소/넷마스크를 설정화일에 써줌으로써 해결할 수 있다. 예들 들어

```
/kickstart 198.168.254.0/255.255.255.0(ro,no_root_squash)
```

그리고 /etc/hosts.allow 화일에는 다음과 같이 적어준다:

```
ALL: 194.82.103.0/255.255.255.0: ALLOW
```

이와 같이 하는 이유는 NFS와 관련된 데몬의 일부 또는 전부에 대한 접근을 제어하는데에 TCP wrapper를 사용하기 때문이다. /etc/exports 화일의 문법은 다른 유닉스의 문법과 다른 경향이 있다는 점을 알아두기 바란다. 리눅스 배포판으로 제공되는 NFS 서버는 유닉스에서 제공되는 서버보다 매우 폭넓은 선택사항을 제공한다.

만일 킥스타트 설정 화일에 root의 패스워드가 포함되어 있거나 NFS로 export 된 디렉토리가 극비에 부쳐야 할 정보를 담고 있다면, 이 정보를 가능한 적은 수의 사람들에게도 노출시키지 않도록 해야한다. 이는 NFS 공유 접근권한을 잘 지정해줌으로써 가능하다. 예를 들자면 전체 도메인과 공유를 하기 보다는 특정한 호스트나

서브넷을 지정해주는 것이다. 킥스타드 설치를 위해 특정한 IP 주소를 사용하지 않아도(free) 무방하지만 나중에 변경을 해주던지 또는 BOOTP/DHCP를 통해 IP 주소를 받아올 수 있도록 재설정을 해주어야 한다.

대부분의 NFS 서버에서는 (유닉스의 종류에 따라 rpc. 가 붙은) mountd 와 nfsd에게 SIGHUP를 보냄으로 해서 /etc/exports 화일이 변경되었음을 알려 주어야 한다. exportfs라는 프로그램 혹은 스크립트가 있어서 다음과 같이 실행한다.

```
# exportfs -a
```

머신이 부팅될 때 NFS 기능을 활성화시키지 않았다면 디렉토리는 자동으로 export되지 않을 것이다. 다시 부팅하거나 root로 다음 명령을 실행하라:

```
# portmap
# rpc.nfsd
# rpc.mountd
```

어떤 시스템에서는 rpc. 접두어가 없는 경우도 있다. 요즘 대부분의 유닉스 배포본에서는 이들 프로그램들은 /usr/sbin 디렉토리 또는 /usr/libexec 밑에서 찾을 수 있다. su 를 사용해서 루트의 권한을 얻었다면 검색 가능한 디렉토리 경로에 없을 수도 있다. 또한 portmap 프로그램은 솔라리스같은 시스템에서는 rpcbind로 불리고, 어떤 버전의 nfsd는 서버를 실행하기 위한 절차를 지정하는 명령행 인수를 필요로 하기도 하고, 또한 biod라 불리는 또다른 데몬을 실행시켜야 할 필요가 있을 수도 있다. 위의 예제는 거의 대부분의 리눅스 시스템에서 사용가능하다.

만일 앞에서 언급한 DHCP와 동적 주소지정 방식을 사용하는 CMU BOOTP 서버를 사용하고 있다면 /etc/bootptab(/etc/bootptab은 BOOTP/DHCP 설정을 담고있는 화일)의 내용은 다음과 같을 것이다.

```
.dynamic-1:ip=198.168.254.128:T254=0x30:T250="ds=198.168.254.2:
dn=swedish-chef.org:sm=255.255.255.0:gw=198.168.254.1:
dl=0xFFFFFFFF":
```

(중략)

위의 내용은 새 머신을 만나게 되면 198.168.254.128에서 시작하여 다음 48개 주소를 동적으로 부여하게 된다.(48의 16진 표현=30) 각 클라이언트는 250번 이내의 값을 가지게 된다. 위의 경우엔 다음과 같이 설정된다:

- DNS 서버 ds를 198.168.254.2로
- 도메인 이름 dn을 swedish-chef.org로
- 서브 넷 마스크 sm을 255.255.255.0로
- 디폴트 게이트웨이 gw를 198.168.254.1로
- IP 사용기한 dl(IP주소를 얼마나 오랫동안 사용하나)을 "forever"로

이 서버의 동적 주소할당을 지원하지 않는 많은 다른 버전은 주소를 제대로 할당받지 못하는 것 같다. 이러한 점때문에 각각의 설치될 머신들의 하드웨어 주소(일반적으로 이더넷)를 /etc/bootptab에 적어주어야 한다. /etc/bootptab 내용은 다음과 같을 것이다:

```
bork.swedish-chef.org:ip=198.168.254.128:ha=0000E8188E56:
  ds=198.168.254.2:dn=swedish-chef.org:sm=255.255.255.0:
  gw=198.168.254.1:dl=0xFFFFFFFF":
```

(중략)

ha 파라미터는 설치될 머신의 하드웨어 주소와 관련된다는 점을 알아두기 바란다.

제 13 절 부록 B - 자신만의 RPM 만들기

RPM 패키지 포맷은 이미 Ed Bailey가 쓴 Maximum RPM에 문서화가 잘 되어 있다. Maximum RPM은 *RPM WWW 사이트* <http://www.rpm.org/>에서 다운로드 받을 수 있으며 유명 서점에 가면 구할 수 있다. 다음에 설명하는 내용은 마음이 시간에 쫓기는 사람들을 위한 힌트에 지나지 않는다.

RPM 패키지는 spec 화일로부터 만들어진다. spec 화일(킥스타트 설정 화일과 비슷한 양식을 가진다)은 패키지를 만들기 위해 수행되는 절차들로 이루어져 있다. - 소스와 다양한 플랫폼 지정과 컴파일전에 패치를 적용하는 등등.. 일단 만들어지고 설치되어진 바이너리 RPM은 여러분이 관계된다고 지정하는 화일과 디렉토리로부터 만들어 진다. RPM은 어떤 화일과 디렉토리가 주어진 패키지와 관련이 있는지 알 수 없으므로 RPM에게 알려줘야 한다는 점을 알아 두기 바란다.

다음은 *Squid WWW 캐쉬 서버* <http://squid.nlanr.net/> RPM의 SPEC 화일의 내용이다:

```
Summary: Squid Web Cache server
Name: squid
Version: 1.NOVM.22
Release: 1
Copyright: GPL/Harvest
Group: Networking/Daemons
Source: squid-1.NOVM.22-src.tar.gz
Patch: retry-1.NOVM.20.patch
%description
This is just a first attempt to package up the Squid Web Cache for easy
installation on our RedHat Linux servers

%prep
%setup
%build
configure --prefix=/usr/squid
perl -spi -e 's!#( -DALLOW_HOSTNAME_UNDERSCORES)!$1!' src/Makefile
make

%install
make install

%files
/usr/squid
```

다음은 RPM을 만드는 방법이다:

```
% mkdir -p SOURCES BUILD SRPMS RPMS/i386
% cp ~/squid-1.NOVM.22-src.tar.gz SOURCES
% cp ~/retry-1.NOVM.20.patch SOURCES
% rpm -ba squid-1.NOVM.22+retry-1.spec
```

이렇게 하면 자동적으로 BUILD 디렉토리 밑에 소스 코드를 풀고 패치(많은 옵션이 있으니 자세한 내용은 Maximum RPM을 참고하기 바란다)를 적용하게 될 하위디렉토리를 생성할 것이다. 이제 RPM은 자동으로 `configure`, `make`를 실행하여 설정을 하고 패키지를 만들고 `make install`을 이용해 설치를 하고 /usr/squid 디렉토리 아래에 패키지를 설치할 것이다. 맨 마지막 과정은 Squid 소프트웨어의 바이너리 RPM을 구성할 것이다.

패키지를 풀고 만들고 설치하는 과정에서 마음대로 쉘 명령을 넣을 수 있다. 예를 들어 Squid의 컴파일시의 변수중 하나인 `perl`을 호출할 수 있다.

최종적으로 바이너리 RPM은 RPMS 디렉토리의 플랫폼 지정 서브디렉토리인 i386 디렉토리 아래에 놓이게 된다. 이 경우 squid-1.NOVM.22-1.i386.rpm 화일이 생성된다. 화일이름은 spec 화일의 `Name`, `Version`, `Release` 정보와 더해서 하드웨어 플랫폼 이 경우에는 i386을 연결시켜 만들어지는 점을 알아두기 바란다. 자신만의 RPM을 만들 경우 무식하게 길고 고통스러운 이름을 갖지 않게 하려면 이점을 명심해 두기 바란다.

전체 소프트웨어 패키지를 리빌드할 필요없이 RPM을 화일을 만들 수 있다는 점 또한 알아둘 만하다. 예를 들면,

```
Summary: Linux 2.0.36 kernel + filehandle patch + serial console patch
Name: linux
Version: 2.0.36+filehandle+serial_console
Release: 1
Copyright: GPL
Group: Base/Kernel
Source: linux-2.0.36+filehandle+serial_console.tar.gz
%description
This is just a first attempt to package up the Linux kernel with patches
for installation on our RedHat Linux servers

%prep
echo

%setup
echo

%build
echo

%install
echo

%post
/sbin/lilo

%files
/lib/modules/2.0.36
/boot/vmlinuz
```

이 경우 간단히 /boot/vmlinuz 화일과 /lib/modules/2.0.35 디렉토리의 내용과 설치할 머신에 패키지를 설치한 후에 /sbin/lilo 명령을 실행하는 RPM 화일을 만들었다. 만일 더 깔끔한 spec 화일을 작성하는 방법이 있다면 알려주기 바란다.

제 14 절 부록 C - 자신이 만든 RPM을 배포본에 포함시키기

다음은 Eric의 업데이트된 RPM을 레드햇 배포판에 넣는 스크립트이다.

```
#!/usr/bin/perl
#
$redhatdir="/cdrom/i386";
$rpmdir="/cdrom/i386/RedHat/RPMS/";
$updatedir="/cdrom/updates/";
@OTHERDIR=($updatedir);
foreach $dir (@OTHERDIR)
```

```
        {
        print "update for $dir\n";
        system(" find $dir -name \"*.rpm\" -exec cp {} $rpmdir \\; ");
        }
chdir($contribdir) || die "peux pas aller dans $contribdir $!\n";
system("chmod -R 755 $redhatdir");
chdir($rpmdir) || die "problem to go in $rpmdir $!\n";
#
# remove the old file
#
opendir(DIR,'.');
@package=grep(/\.rpm$/,readdir(DIR));
foreach $file (@package)
        {
        $file =~ /(.*)\-([\d+|\.]+\w*)\-(\d+)\.[i386|noarch].*/;
        $nom=$1;
        $version=$2;
        $buildvers=$3;
        if ($NOM{$nom})
                {
                $version2=$VERSION{$nom};
                $buildver2=$BUILDVERS{$nom};
                $file2=$FILE{$nom};
                $nom2=$NOM{$nom};
                if ( $version2 gt $version )
                        {
                        print "$file2 is newer than $file\n";
                        unlink($file);
                        }
                else
                        {
                        if ( $version2 lt $version )
                                {
                                print "$file is newer than $file2\n";
                                unlink($file2);
                                $VERSION{$nom}=$version;
                                $BUILDVERS{$nom}=$buildvers;
                                $FILE{$nom}=$file;
                                $NOM{$nom}=$nom;
                                }
                        else
                                {
                                if ( $buildver2 > $buildvers )
                                        {
                                     print "$file2 : $buildver2 est mieux que $file : $buildvers\n";
                                     unlink($file);
                                        }
                                else
                                        {
                                   print "$file2 : $buildver2 is older than $file : $buildvers\n";
                                     unlink($file2);
                                     $VERSION{$nom}=$version;
                                     $BUILDVERS{$nom}=$buildvers;
                                     $FILE{$nom}=$file;
                                     $NOM{$nom}=$nom;
```

```
                                                }
                                        }
                                }
                        }
                else
                        {
                        $VERSION{$nom}=$version;
                        $BUILDVERS{$nom}=$buildvers;
                        $FILE{$nom}=$file;
                        $NOM{$nom}=$nom;
                        }
                }

        # we do the hard thing here
        #
        system("$redhatdir/misc/src/install/genhdlist $redhatdir");
```