The Loopback Root Filesystem HOWTO

Andrew M. Bishop

amb@gedanken.demon.co.uk

도운이

김현종

polsci@hitel.net

v1.1, 24 September 1999

본 문서는 루프백 디바이스를 이용해 별도의 파티션 분할작업 없이 도스 파티션에 리눅스 시스템을 설치하는 방법을 설명합니다. 또, 이 기술의 다른 응용도 다룰 것입니다.

1. 소개

1.1. 저작권

1.2. Revision History

Version 1.0.0

Initial Version (June 1998)

Version 1.0.1-1.0.3

Slight Modifications, kernel version changes, typos etc. (1998 - July 1999)

Version 1.1

Added Copyright Information and Re-Submitted (September 1999)

2. 루프백 디바이스와 램디스크의 원리

우선 루프백 파일시스템을 루트디바이스로 설정하는 원리 몇가지를 설명하겠습니다.

2.1. 루프백 디바이스

리눅스의 루프백 디바이스는 가상의 디바이스로서 실제의 미디어 디바이스와 똑같이 취급됩니다.

일반적인 실제 디바이스의 예로는 `/dev/hda1`, `/dev/hda2`, `/dev/sda1` 등의 하드디스크 파티션들이나 `/dev/fd0` 와 같은 플로피 디바이스등을 들 수 있습니다. 이런 디바이스들은 모두 파일및 디렉토리 구조를 담는 디바이스들입니다. 이들을 사용하려면 특정 파일시스템(ext2fs, msdos, ntfs 등등)으로 포맷한 후 마운트해야 합니다.

루프백 파일시스템이란 어떤 파일시스템의 파일 한개를 마치 하나의 디바이스처럼 인식시킨 것입니다. 이 파일은 다른 디바이스와 마찬가지로 포맷되고 마운트될 수 있습니다. 이렇게 하기위해서는 먼저 이 화일이 `/dev/loop0` 나 `/dev/loop1` 등과 같은 특수한 디바이스와 연결되어야 합니다. 그 다음에야 새로운 가상의 디바이스로 마운트될 수 있는 것입니다.

2.2. 램디스크 디바이스

리눅스에서 파일시스템으로 마운트 가능한 가상디바이스가 또 한 종류 있는데 바로 램디스크 디바이스입니다.

램디스크란 별다른 물리적 장치를 지칭하는 것이 아니라, 바로 메모리의 일부를 디스크로 인식시킨 것입니다. 이 용도로 할당되는 메모리는 디스크로 스왑되지 않으며 디스크 캐쉬에 남게됩니다.

램디스크는 램디스크 디바이스인 `/dev/ram0` 나 `/dev/ram1` 등에 기록함으로서 즉각 만들수 있습니다. 이것 역시 루프백 디바이스와 같은식으로 포맷할 수있고 마운트도 가능합니다.

부팅시 램디스크를 이용할 경우(리눅스 배포본이나 복구용 디스크에서 주로 사용됩니다), 디스크 이미지(디스크의 전체 내용을 하나의 파일로 만든 것)가 부트플로피상에 압축된 상태로 위치하고 있습니다. 부팅이 시작되면 커널에 의해 이것이 자동적으로 압축이 풀리면서 램디스크로 로딩된 후 마운트되게 됩니다.

2.3. 초기 램디스크 디바이스(Initial Ramdisk)

리눅스의 초기 램디스크(**Initial Ramdisk**) 디바이스는 루프백 디바이스를 루트 파일시스템으로 사용하는데 필요한 중요한 메카니즘입니다.

초기 램디스크가 사용되면, 파일시스템 이미지는 메모리로 복사된 후 마운트되고 그 화일들을 액세스 할 수 있게 됩니다. 램디스크상의 특정 프로그램(`/linuxrc`)이 실행된 후 종료하면, 이제 다른 디바이스가 루트 파일시스템으로서 마운트되게 됩니다. 기존의 램디스크 역시 계속 존속하지만 이것은 `/initrd` 라는 디렉토리로 마운트되거나 아니면 `/dev/initrd` 디바이스를 통해 계속 사용할 수 있습니다.

이런 방식은 다소 이상하게 보일 것입니다. 정상적인 부트과정이라면 지정된 루트파티션을 마운트해서 그대로 실행되면 될텐데 왜 이렇게 램디스크를 먼저 마운트 해서 동작시킨 후 그다음 진짜 루트파티션을 마운트할까요? 그 이유는 초기 램디스크의 옵션을 사용하면 진짜 부트 과정을 시작하기에 앞서 루트파티션을 변경할 수 있기 때문입니다.

2.4. 루트 파일시스템

루트 파일시스템이란 부팅이 끝난후 / 디렉토리로 마운트되는 디바이스를 말합니다.

루트 파일시스템은 다른 모든 파일들을 포함하게 되는 관계로 다소 복잡한 면이 있습니다. `rc` 스크립트 파일들을 부팅할 때, 이들은 `/etc/rc.d` 나 `/etc/rc?.d` 등에 위치합니다. 이는 `/etc/init` 프로그램의 버전에 따라 조금씩 다릅니다.

시스템이 일단 부트된 후에는 루트파티션을 언마운트하거나 변경하는 것이 불가능합니다. 왜냐하면 다른 모든 프로그램이 이미 이 루트파티션을 사용하고 있기 때문입니다. 이때문에 초기 램디스크 이미지가 유용한 것입니다. 초기 램디스크를 이용하면 최종적인 루트 파티션을 부트시작시의 것과 달리할 수 있습니다.

2.5. 리눅스의 부트과정

부팅시 초기 램디스크의 동작과정을 아래에 순서대로 서술하겠습니다.

1. 커널이 메모리로 로드된다. 이 일은 `LILO` 나 `LOADLIN` 등의 부트로더가 담당한다. 이 과정에서 `Loading...` 이라는 메시지를 볼수 있을 것이다.
2. 램디스크 이미지가 메모리로 로드된다. 이 일 역시 `LILO` 나 `LOADLIN` 이 담당한다. 여기서도 `Loading...` 이라는 메시지가 나타날 것이다.

3. 커널이 초기화된다. 이 과정에서 명령행의 옵션들이 해석되고 램디스크를 루트디바이스로 설정하게 된다.
4. 초기 램디스크상의 `/linuxrc` 라는 프로그램이 실행된다.
5. 루트 디바이스가 커널 파라메터에 설정된 대로 변경된다.
6. 초기화 프로그램인 `/etc/init` 가 실행되어 사용자가 설정한 부트과정을 수행한다.

이상에서 초기 램디스크를 사용한 부팅과정을 간략히 설명하였습니다. 간략한 설명이었지만 커널이 어떻게 기동되고 초기 램디스크가 어디에서 사용되는지를 설명하는데는 충분합니다.

3. 루프백 루트 디바이스 만들기

이제 루프백 디바이스를 만드는 일반적인 원리를 설명하겠습니다.

3.1. 필요한 것들

루프백 루트 디바이스를 만들려면 다음 몇가지가 필요합니다.

- 작업에 사용할 리눅스 시스템
- 대형 파일들을 목표 파티션인 DOS 파티션으로 복사할 수단

가장 중요한 것은 우리의 작업에 사용할 리눅스 시스템입니다. 루프 디바이스는 오직 리눅스에서만 제작할수 있습니다. 이 리눅스 시스템은 커널 컴파일이 가능해야 합니다.

일단 루프백 디바이스가 만들어지면 이것은 매우 덩치큰 파일 한개가 됩니다. 필자는 80 MB 짜리 파일 한개를 사용했습니다. 하지만 X 터미널을 돌릴 생각이라면 이걸로도 부족하며 훨씬 많은 용량이 필요합니다. 이 파일은 최종적으로는 DOS 파티션으로 복사될 것입니다. 따라서 이 작업을 위해서는 네트웍을 쓰든지 아니면 상당수의 플로피 디스켓이 필요할 것입니다.

필요한 소프트웨어는 다음과 같습니다.

- `LOADLIN` 버전 1.6 이상
- 루프백 디바이스를 지원하는 버전의 `mount`
- 필요한 옵션들을 지원하는 버전의 커널

이 모든 것들은 최근의 리눅스 배포본에는 기본적으로 들어있습니다.

3.2. 리눅스 커널을 만들기

필자는 리눅스 커널 버전 2.0.31 을 써서 루프백 디바이스를 만들었습니다. 다른 버전들도 물론 됩니다만 최소한 다음의 옵션들은 가지고 있어야만 합니다.

필요한 커널 옵션들은 다음과 같습니다.

- 램 디스크 지원(`CONFIG_BLK_DEV_RAM`).
- 초기 램디스크 지원(`CONFIG_BLK_DEV_INITRD`).
- 루프 디바이스 지원(`CONFIG_BLK_DEV_LOOP`).
- fat 파일시스템 지원(`CONFIG_FAT_FS`).
- msdos 파일시스템 지원(`CONFIG_MSDOS_FS`).

처음의 두가지는 램디스크 디바이스와 초기 램디스크 디바이스를 위한 것입니다. 그다음 것은 루프백 화일시스템을 위한 것입니다. 마지막 두개는 도스 파티션을 마운트하는데 필요한 것입니다.

모듈을 사용하지 않는 커널을 만드는게 가장 쉬운 방법입니다. 하지만 원하신다면 필자는 해보지 않았지만 모듈을 쓸수도 있을 것입니다. 만일 모듈이 사용된다면 최소한 위의 옵션들만은 모듈로 설정되서는 안되며 커널내에 포함시켜야 합니다.

커널 버전에 따라서는 커널 패치가 필요할 수도 있습니다. 루프백 디바이스를 루트 파일시스템으로 사용할수 있게 해주는 패치는 무척 간단합니다.

- 버전 2.0.0 이전의 커널; 필자는 이에 대해서는 모릅니다.
- 버전 2.0.0 부터 2.0.34 의 커널; 아래의 2.0.x 용 커널 패치가 필요.
- 버전 2.0.35 부터 2.0.x 의 커널; 커널 패치가 필요없음.
- 버전 2.1.x 의 커널; 2.1.x 의 정확한 버전에 따라 아래의 2.0.x 패치나 2.2.x 패치를 적용시켜야 함.
- 버넌 2.2.0 부터 2.2.10 의 커널; 아래의 2.2.x 용 커널 패치가 필요.

2.0.x 커널의 `/init/main.c` 파일에 아래처럼 수정된 한 라인을 덧붙여야 합니다. 이 라인은 `"loop", 0x0700` 입니다.

```
static void parse_root_dev(char * line)
{
int base = 0;
static struct dev_name_struct {
const char *name;
const int num;
} devices[] = {
{ "nfs",     0x00ff },
{ "loop",    0x0700 },
{ "hda",     0x0300 },

...
```

```
{ "sonycd",  0x1800 },
{ NULL, 0 }
};

...

}
```

2.2.x 커널들은 /init/main.c 파일에 다음 세줄이 추가되어야 합니다. 추가되는 라인들은 "loop", 0x0700 과 그 위아래 한줄씩들입니다.

```
static struct dev_name_struct {
const char *name;
const int num;
} root_dev_names[] __initdata = {
#ifdef CONFIG_ROOT_NFS
{ "nfs",     0x00ff },
#endif
#ifdef CONFIG_BLK_DEV_LOOP
        { "loop",    0x0700 },
#endif
#ifdef CONFIG_BLK_DEV_IDE
{ "hda",     0x0300 },

...

{ "ddv", DDV_MAJOR << 8},
#endif
{ NULL, 0 }
};
```

일단 커널 설정이 끝나면 컴파일해서 zImage 파일을 만들어야 합니다(make zImage 명령). 컴파일이 끝나면 이 파일은 arch/i386/boot/zImage 에 위치할 것입니다.

3.3. 초기 램디스크 디바이스 만들기

초기 램디스크는 루프백 디바이스를 사용해 아주 쉽게 만들수 있습니다. 아래의 명령들을 실행시키기 위해서는 루트 권한을 가져야 합니다. 루트의 디렉토리(/root)에서 실행한다고 가정하겠습니다.

```
mkdir /root/initrd
dd if=/dev/zero of=initrd.img bs=1k count=1024
mke2fs -i 1024 -b 1024 -m 5 -F -v initrd.img
mount initrd.img /root/initrd -t ext2 -o loop
cd initrd
```

```
[create the files]
cd ..
umount /root/initrd
gzip -c -9 initrd.img > initrdgz.img
```

위의 단계들을 해설하면 다음과 같습니다.

1. 초기 램디스크를 위한 마운트 포인트를 만들어 줍니다. 디렉토리만 만들어 주면 됩니다.
2. 필요한 크기의 빈 화일을 만듭니다. 필자는 1024 KB 를 사용하는데 사용자의 필요에 따라 다소 가감해도 좋습니다(크기를 결정하는 것은 제일 마지막 파라메터입니다).
3. 빈 파일 위에 ext2 파일 시스템을 만듭니다.
4. 루프백 디바이스를 사용해 마운트 포인트에 파일을 마운트합니다.
5. 마운트된 루프백 디바이스로 이동합니다.
6. 필요한 파일들을 만듭니다(자세한 것은 아래에).
7. 마운트된 루프백 디바이스에서 빠져나옵니다.
8. 디바이스를 언마운트합니다.
9. 나중에 사용할 수 있도록 압축된 버전으로 만듭니다.

초기 램디스크에 들어갈 내용물

램디스크에 넣어두어야 할 파일들은 될수있는 한 적은게 좋습니다. 하지만 필요한 명령어들을 수행할 수는 있어야 겠지요.

- `/linuxrc` 는 msdos 파일 시스템을 마운트하기 위한 스크립트입니다(아래를 보세요).
- `/lib/*` 프로그램들에 필요한 동적 링커와 라이브러리들
- `/etc/*` 동적 링커에 사용되는 캐쉬(반드시 필요하다고 단정할 수는 없지만 링커의 투덜거림을 없앨수 있습니다).
- `/bin/*` 쉘 해석기 (`ash` 가 `bash` 보다 크기가 작아서 선호됩니다. `mount` 과 `losetup` 프로그램들은 각각 DOS 디스크와 루프백 디바이스들을 다룰 때 필요합니다.
- `/dev/*` 필요한 디바이스들. `ld-linux.so` 를 위해서는 `/dev/zero` 가 필요하고, msdos 디스크를 마운트하기 위해서는 `/dev/hda*` 이 필요하며, 루프백 디바이스를 위해서는 `/dev/loop*` 이 필요합니다.
- `/mnt` msdos 디스크를 마운트할 빈 디렉토리

필자가 사용하는 초기 램디스크의 내용은 아래와 같습니다. 파일시스템의 오버헤드까지 합쳐 약 800 KB 정도가 됩니다.

```
total 18
drwxr-xr-x   2 root     root         1024 Jun  2 13:57 bin
drwxr-xr-x   2 root     root         1024 Jun  2 13:47 dev
```

```
drwxr-xr-x   2 root     root         1024 May 20 07:43 etc
drwxr-xr-x   2 root     root         1024 May 27 07:57 lib
-rwxr-xr-x   1 root     root          964 Jun  3 08:47 linuxrc
drwxr-xr-x   2 root     root        12288 May 27 08:08 lost+found
drwxr-xr-x   2 root     root         1024 Jun  2 14:16 mnt

./bin:
total 168
-rwxr-xr-x   1 root     root        60880 May 27 07:56 ash
-rwxr-xr-x   1 root     root         5484 May 27 07:56 losetup
-rwsr-xr-x   1 root     root        28216 May 27 07:56 mount
lrwxrwxrwx   1 root     root            3 May 27 08:08 sh -> ash

./dev:
total 0
brw-r--r--   1 root     root       3,   0 May 20 07:43 hda
brw-r--r--   1 root     root       3,   1 May 20 07:43 hda1
brw-r--r--   1 root     root       3,   2 Jun  2 13:46 hda2
brw-r--r--   1 root     root       3,   3 Jun  2 13:46 hda3
brw-r--r--   1 root     root       7,   0 May 20 07:43 loop0
brw-r--r--   1 root     root       7,   1 Jun  2 13:47 loop1
crw-r--r--   1 root     root       1,   3 May 20 07:42 null
crw-r--r--   1 root     root       5,   0 May 20 07:43 tty
crw-r--r--   1 root     root       4,   1 May 20 07:43 tty1
crw-r--r--   1 root     root       1,   5 May 20 07:42 zero

./etc:
total 3
-rw-r--r--   1 root     root         2539 May 20 07:43 ld.so.cache

./lib:
total 649
lrwxrwxrwx   1 root     root           18 May 27 08:08 ld-linux.so.1 -
> ld-linux.so.1.7.14
-rwxr-xr-x   1 root     root        21367 May 20 07:44 ld-linux.so.1.7.14
lrwxrwxrwx   1 root     root           14 May 27 08:08 libc.so.5 -
> libc.so.5.3.12
-rwxr-xr-x   1 root     root       583795 May 20 07:44 libc.so.5.3.12

./lost+found:
total 0

./mnt:
total 0
```

유일하게 복잡한 단계는 `dev` 의 디바이스들에 관한 것입니다. `mknod` 프로그램을 써야 이 디바이스들을 만들수 있는데, 현재 시스템의 `/dev` 에 있는 디바이스들의 파라메터를 그대로 베끼면 됩니다.

/linuxrc 파일

아래의 예에서는 msdos 파티션으로서 `/dev/hda1` 을 마운트합니다. 마운트에 성공하면 `/dev/loop0` 에 `/linux/linuxdsk.img` 를 지정하고 `/dev/loop1` 에 `/linux/linuxswp.img` 을 지정합니다.

```
#!/bin/sh

echo INITRD: Trying to mount /dev/hda1 as msdos

if /bin/mount -n -t msdos /dev/hda1 /mnt; then

   echo INITRD: Mounted OK
   /bin/losetup /dev/loop0 /mnt/linux/linuxdsk.img
   /bin/losetup /dev/loop1 /mnt/linux/linuxswp.img
   exit 0

else

   echo INITRD: Mount failed
   exit 1

fi
```

첫번째 디바이스 `/dev/loop0` 는 루트 디바이스가 되고 두번째 디바이스 `/dev/loop1` 은 스왑 공간이 됩니다.

부팅종료후 루트권한이 아닌 일반 사용자가 도스파티션에 쓰기가능하려면 대신 `mount -n -t msdos /dev/hda1 /mnt -o uid=0,gid=0,umask=000,quiet` 을 써야 합니다. 이것은 도스 파티션에 대한 모든 액세스를 루트권한으로 변환시켜 퍼미션 문제를 해결합니다.

3.4. 루트 디바이스 만들기

당신이 사용할 루트 디바이스는 `linuxdsk.img` 이라는 파일입니다. 위의 초기 램디스크 제작과 똑같은 방법으로 만들면 되는데, 다만 이 파일은 훨씬 큽니다. 원하는 어떤 리눅스 시스템이라도 이 디스크위에 설치할 수 있습니다.

가장 쉬운 방법은 현재의 리눅스 시스템을 그대로 이리로 복사해넣는 것입니다. 다른 방법으로는 이 위에다 새로운 리눅스 시스템을 인스톨하는 것입니다.

여기까지 다 완료했다면 이제 다음의 몇가지 자잘한 수정을 해주어야만 합니다.

`/etc/fstab` 파일은 반드시 루트파티션과 스왑파티션이 초기램디스크로 셋업된 루프 디바이스들임을 기록하고 있어야만 합니다.

```
/dev/loop0      /       ext2   defaults 1 1
/dev/loop1      swap    swap   defaults 1 1
```

이는 진짜 루프 디바이스가 사용될 때 커널이 루트 디바이스의 위치를 혼동하지 않도록 해줍니다. 또한 스왑 공간이 보통의 일반적인 스왑 파티션처럼 사용되게 해줍니다. 이들 이외의 또다른 루트 디바이스나 스왑 파티션에 대한 지정이 있어서는 안됩니다.

리눅스가 가동된 후 도스파티션을 읽으려면 몇가지 추가적인 작은 수정이 필요합니다.

/initrd 이라는 디렉토리를 만들어 주십시오. 이것은 루프백 루트 파일시스템이 일단 마운트 된 후 초기 램디스크가 마운트되는 위치입니다.

/DOS 같은 심볼릭 링크를 만들어 진짜 도스 파티션이 마운트 되는 /initrd/mnt 로 링크시키십시요.

디스크들을 마운트시키는 스크립트 한줄을 rc 파일에 추가하세요. 이는 mount -f -t msdos /dev/hda1 /initrd/mnt 같은 명령이면 됩니다. 이것은 도스 파티션을 가짜(fake)로 마운트시키는 것인데 이렇게 하면 모든 프로그램들(가령 df)이 도스 파티션이 여기에 있는 것으로 알게됩니다. 만일 당신이 /linuxrc 파일내에서 다른 옵션들을 썼다면 필히 여기에도 그것들을 사용해야 합니다.

루트 디바이스상에 커널이 있을 필요는 전혀 없습니다. 왜냐하면 커널은 이미 그전에 로드됐기 때문이다. 하지만 당신이 커널 모듈을 사용한다면 모듈들을 루트디바이스에 포함시켜야 합니다.

3.5. 스왑 디바이스 만들기

당신이 사용할 디바이스는 linuxswap.img입니다 이 스왑 디바이스는 극히 간단하게 만들 수 있습니다. 위에서처럼 빈 파일 하나를 초기 램디스크용으로 만든 후 mkswap linuxswap.img 해서 초기화시키십시오.

스왑 공간의 크기는 설치한 시스템으로 당신이 무슨 일을 할것인가에 따라 다릅니다. 하지만 필자는 8 MB 부터 당신이 가진 램용량 사이의 값을 추천하겠습니다.

3.6. MSDOS 디렉토리 만들기

사용될 파일들을 도스파티션상으로 옮겨야 합니다.

C:\LINUX 라는 도스 디렉토리에 필요한 파일들은 다음과 같다.

- LINUXDSK.IMG 루트 디바이스가 될 디스크 이미지.
- LINUXSWP.IMG 스왑 공간

3.7. 부트 플로피 만들기

사용될 부트 플로피는 단순한 보통의 도스포맷 플로피 한장입니다.

이것은 도스상에서 format a: /s 명령으로 만듭니다.

이 디스크 위에 아래와 같은 AUTOEXEC.BAT 파일과 커널, 압축된 초기 램디스크, LOADLIN 실행화일을 복사해 넣습니다.

- AUTOEXEC.BAT 도스의 자동실행화일
- LOADLIN.EXE LOADLIN 실행화일
- ZIMAGE 리눅스 커널
- INITRDGZ.IMG 압축된 초기 램디스크 이미지

AUTOEXEC.BAT 파일은 아래의 한줄을 포함해야만 합니다.

```
\loadlin \zImage initrd=\initrdgz.img root=/dev/loop0 ro
```

이것은 사용할 커널이미지, 램디스크 이미지, 초기 램디스크가 끝난후의 루트 디바이스, 읽기전용으로 마운트되는 루트 파티션을 지정한 것입니다.

4. 시스템의 부팅

이 새로운 루트 디바이스로 부트하기 위해서는 위와같이 준비된 플로피 디스크를 PC 에 넣고 부팅합니다.

다음과 같은 일이 일어남을 볼수 있을 것입니다.

1. DOS boots
2. AUTOEXEC.BAT starts
3. LOADLIN is run
4. The Linux kernel is copied into memory
5. The initial ramdisk is copied into memory
6. The Linux kernel is started running
7. The `/linuxrc` file on the initial ramdisk is run
8. The DOS partition is mounted and the root and swap devices set up
9. The boot sequence continues from the loopback device

이상과 같이 완료되면 이제 부트 플로피를 제거하고 리눅스 시스템을 사용하면 됩니다.

4.1. 발생할 수 있는 문제들

위의 과정의 각 단계에서의 실패에 대해 그 의미와 점검해야 바를 설명하겠습니다.

도스 부팅은 `MS-DOS Sarting ...` 이라는 메시지가 화면에 뜸으로서 쉽게 알수 있습니다. 만일 이런 메시지가 화면에 나타나지 않는다면 플로피디스크가 부트가능하게 되어있지 않았거나 PC 가 플로피 디스크 부팅이 안되도록 설정되었기 때문입니다.

`AUTOEXEC.BAT` 파일내의 명령어들이 실행될 때는 디폴트로 화면에 나타나도록 되어있습니다. 우리의 경우 `LOADLIN` 한줄이 나타날 것입니다..

`LOADLIN` 이 실행되면서 화면에 출력되는 작업이 두가지 있습니다. 우선 커널이 메모리로 로드되는 것이고, 그 다음으로 램디스크가 메모리로 복사되는 것입니다. 이 두 작업 모두 `Loading...` 메시지를 출력합니다.

커널이 스스로의 압축을 풀다가 만약 커널 이미지가 손상된 것을 발견하면 **crc** 에러를 내게 됩니다. 이상이 없다면 그다음으로 초기화 과정을 차례로 수행하게 되는데 각 과정마다 친절히 점검상황을 알려줍니다. 초기 램디스크 디바이스의 로딩역시 이 단계에서 나타납니다.

이제 새로운 루트 디바이스의 정상적인 부트과정이 속개되는데 이 역시 친절히 진행상황을 알려줍니다. 읽기쓰기 용으로 루트디바이스를 마운트해서 문제가 생길수 있다면, `LOADLIN` 명령어 라인 옵션인 '`ro`'을 써주면 이를 방지할 수 있습니다. 그밖의 일어날 수 있는 문제로는 부트과정시 루트 디바이스를 어디에서 찾아야 하는지를 모르는 경우인데, 이는 필시 `/etc/fstab` 에 관련된 당신의 실수일 것입니다.

부트 과정이 완료되면 이제 문제가 생길수 있는 부분은 프로그램들이 도스파티션의 마운트되었는지 아닌지를 모르는 경우 뿐입니다. 이는 앞에서 설명한 가짜(fake) mount 명령을 사용하는 것이 좋은 아이디어가 됩니다. 이를 쓰면 도스 디바이스상의 파일들을 쉽게 액세스할 수 있습니다.

4.2. 참고 문서

필자가 최초의 루프백 루트 파일 시스템을 만들때 이용한 문서들:

- 리눅스 커널 소스. 특히 `init/main.c`
- 리눅스 커널 문서. 특히 `Documentation/initrd.txt` 와 `Documentation/ramdisk.txt`.
- `LILO` 문서
- `LOADLIN` 문서

5. 기타 루프백 루트 디바이스가 유용한 곳

일단 도스 파티션상에서 하나의 파일로 된 파일시스템을 부팅하는 원리를 알았다면, 이제 여러가지로 응용해 볼 수 있습니다.

5.1. 도스 하드디스크만으로 설치하기

부트 플로피를 써서 도스 하드디스크 상의 파일로부터 리눅스를 부팅할 수 있다면, 당연히 하드디스크를 써서도 부팅시킬 수 있습니다.

`AUTOEXEC.BAT` 를 손봐서 메뉴방식으로 부트하도록 해둔 후, 메뉴 중 하나에 `LOADLIN` 을 선택할 수 있게 한다면 부팅과정이 훨씬 더 빨라집니다. 다른 것은 모두 동일합니다.

5.2. LILO 로 부팅되는 설치

`LOADLIN` 은 리눅스 커널을 부팅시킬수 있는 여러 프로그램중 하나일 뿐입니다. `LILO` 역시 똑같은 기능을 하지만 이것은 도스가 필요없습니다.

이경우 도스 포맷된 플로피디스크 대신 ext2fs 포맷된 것을 써야합니다. 그밖의 커널과 디스크상의 초기 램디스크에 관한 세세한 사항들은 아주 유사합니다.

필자가 `LOADLIN` 방법을 선택한 이유는 `LILO` 에 주어야 하는 옵션들이 조금 더 복잡했기 때문입니다. 또한 `LOADLIN` 쪽이 도스환경에서 읽힐수 있기 때문에 초심자들에게는 플로피디스크가 무엇인지 파악하는데 유리합니다.

5.3. VFAT / NTFS 설치

필자는 NTFS 에서도 이 방법을 시도했으며, 아직껏 문제는 없었습니다. 리눅스 커널 2.0.x 버전에서는 아직 NTFS 파일시스템 드라이버가 기본으로 설정되어있지 않습니다. 하지만 다음의 패치를 쓸수 있습니다. http://www.informatik.hu-berlin.de/~loewis/ntfs/. 버전 2.2.x 대에서는 커널에 NTFS 드라이버가 기본으로 포함되어 있습니다.

VFAT 이나 NTFS 를 쓸 경우의 유일한 변경사항들은 초기 램디스크의 사용에 있습니다. `/linuxrc` 는 msdos 가 아니라 vfat 이나 ntfs 타입의 파일시스템을 마운트하도록 변경되어야 합니다.

필자가 아는 한에는 VFAT 파티션이라고 이 방법이 안될 이유가 없습니다.

5.4. 파티션을 다시 잡지않고 리눅스를 설치하기

보통, PC 상에 표준적인 리눅스 배포본을 설치할 때는 플로피 디스크로부터의 부팅과 디스크의 파티션을 다시잡는 과정을 거쳐야 합니다. 이 과정을 빈 루프백 디바이스와 스왑 파일을 만드는 부트플로피로 대체할 수 있습니다. 이렇게 하면 보통과 똑같이 설치를 계속하지만 파티션이 아닌 루프백 디바이스위에 설치가 되는 것입니다.

이 방법은 `UMSDOS` 를 이용한 설치방법 대신 사용할 수 있습니다. 그러면 도스 파일시스템의 할당단위인 32 KB 대신 ext2 파일시스템의 최소할당단위인 1 KB 를 할당할 수 있어 디스크를 보다 효율적으로 사용할 수 있습니다. 이것은 VFAT 과 NTFS 포맷된 디스크에도 사용할 수 있습니다.

5.5. 부트 불가능한 디바이스로부터의 부팅

이 방법을 쓰면 일반적으로 부팅 불가능한 디바이스로부터 리눅스 시스템을 부팅할 수 있습니다(역주 : 다소 오해의 소지가 있을지 모르겠는데 여기서 필자가 뜻하는 바는 루트파티션을 다양하게 택할 수 있다는 의미입니다. 기본적으로 플로피나 하드, 혹은 최근의 시디롬 부팅 등의 부팅가능한 매체는 반드시 필요합니다).

- CD-Rom
- Zip Disks
- Parallel port disk drives

분명 그밖에도 가능한 디바이스들이 많이 있습니다. NFS 루트 파일시스템은 이미 커널내에 옵션으로 포함시킬수 있도록 되었지만 그 대신 여기서 설명한 방법을 써도 됩니다.