McCune01.fm Page 1 Friday, November 10, 2000 4:31 PM

CHAPTER

1

HAVING LINUX AND WINDOWS ON THE SAME PC

1.1 Partitions

There is no need to get rid of Windows to run Linux. In fact, there are many ways to run both of them on the same PC. Each operating system has its own strengths and weaknesses, so often having both on the same PC can be an advantage.

Before the actual installation, we need to go over some basics of Linux and Windows such as partitions and filesystems.

A partition is a way of sectioning off space on a hard drive. Most PCs have their hard drive partitioned into one large drive. It doesn't have to be this way. Drives can be divided into several partitions. This is often done to separate the programs from the data and also for storing multiple operating systems on the same drive.

The first section of a hard drive contains information on the partitions, including where the start and end of each partition is located. It also contains the location of the boot loader, which starts loading the operating system. Each operating system has its own boot loader. Windows 3x, 95, and 98 use `IO.SYS` and `DOS.SYS`, Windows NT uses `NTLOADER`, and Linux uses `LILO`. There are also commercial and shareware boot loaders, such as Norton System Commander, that are designed to make it easier to boot with multiple configurations and multiple operating systems.

1.2 Filesystems

There are also several different filesystems used by Linux and Windows. A filesystem is simply a way of organizing files on a partition. Windows uses FAT, FAT16, FAT32 (VFAT), and NTFS (NT Filesystem). The native filesystem for Linux is ext2, although it supports many other filesystems.

FAT is the original filesystem used by DOS. It is an eight-bit filesystem and will support partitions of up to 32 MB. This was no problem in the early 1980s, when most PCs didn't even have hard drives.

FAT supports the following file attributes:

- Read-only—When set, the file can't be deleted or changed.
- Archive—Determines whether a file has been changed. This is used by many backup programs.

- Hidden—The file doesn't show up in the directory contents.
- System—Used for system files. System files are treated differently by the operating system.

Later, as hard drives came into use, the 32 MB limitation of FAT became a burden and an improved FAT16 replaced it. FAT16 increased the available size of the filesystem to 2 GB. Other than the filesystem size, FAT16 is essentially the same as the original FAT filesystem. FAT16 is supported by DOS 4.0 and greater, all versions of Windows, and all current versions of Linux.

With Windows 95 release 2, Microsoft introduced FAT32. This increased the size of the filesystem to 2 terabytes, which is larger than any hard drives currently available for PCs. It is also faster and more robust than FAT16.

NTFS is the native filesystem for Windows NT and 2000. Like FAT32, it also supports 2-terabyte filesystem sizes, but the boot partition is currently limited to 7.8 GB. For some files such as database files, NTFS can support up to 16 exabytes. NTFS offers better reliability and security than any FAT-based filesystem.

The reliability factors are beyond the scope of this book, but NTFS security considerations need to be covered. First of all, everything in the filesystem has an owner. By default, the user who creates an object (anything in the filesystem is an object) is the owner. The owner has full rights to the object unless the rights are taken away.

There are also groups, which contain users. Three special group accounts are: administrator, everyone, and guest.

- The administrator account has all rights to the filesystem. This account can change, create, and delete all objects as well as change the rights of other accounts.
- Everyone is a group that includes all the user accounts on the system. This account is used to change the rights for every user on the system.
- The guest account is a default account with minimal rights. It is often used for accounts such as FTP access accounts, which only need access to a few specific files.

Files in NTFS have the same attributes as files in the FAT filesystem: read-only, hidden, system, and archive. Each user and group can also be assigned rights to objects in the NTFS filesystem. The rights that can be assigned are:

- List folder contents—Shows up in a directory listing.
- Read—Can read the contents of the object.
- Read and execute—Can read and execute the object.
- Write—Can change or delete the object.
- Modify—Can change the rights on the file.
- Full control—Has all of the above rights.

There are three settings for the rights: allow, deny, and inherited.

- Allow—Allows rights on the object.
- Deny—Takes away rights on the object.
- Inherited—If neither allow nor deny is specified, the object will inherit the rights of the directory above it.

To view the rights of an object on NTFS, right-click on the object and choose Properties. Then select the Security tab.

Ext or ext2 is the native filesystem for Linux partitions. Ext is the original filesystem for Linux and ext2 is an improved version of it. Objects (such as files, directories, and devices) in Linux support three properties: read, write, and execute.

- Read—If set, allows the object to show up in a directory listing and be read.
- Write—If set, allows the object to be written and deleted.
- Execute—If set, allows the object to be executed. This must also be set for directories.

An object has three sets of rights: owner, group, and everyone.

- Owner—The user who created the file, unless it is changed.
- Group—The group that owns the file is the group to which the owner belongs, unless it is changed.
- Everyone—The right for all other users on the system.

To view the rights of an object, type `ls -l <object name>` . For example, to find the rights of `index.txt`, type `ls -l index.txt`. The output of this command is as follows:

```
-rwxrwxr--   1 root    root     6230 Dec 21 00:12 index.txt
```

Let's examine what this output means, starting with the first character:

- First character—This is a special attribute such as a directory, link, or device driver. A link in Linux is similar to a shortcut in Windows.
- Next three characters (`rwx`)—The owner has read, write, and execute properties.
- Next three characters (`rwx`)—The group has read, write, and execute properties.
- Next three characters (`r--`)—Everyone has the read property.
- `1 root`—The owner's ID number and name.
- `root`—The group name, which is also `root`.
- `6230`—The size of the file in bytes.
- `Dec 21 00:12`—The last date and time the file was modified.
- `index.txt`—The filename.

You can change the rights of a file with `chmod`. The owner and group can be changed with `chown`.

There is one special account in Linux: `root`. The root account is created automatically when Linux is installed and it has full rights to all objects in the filesystem.

1.3 Partition Naming

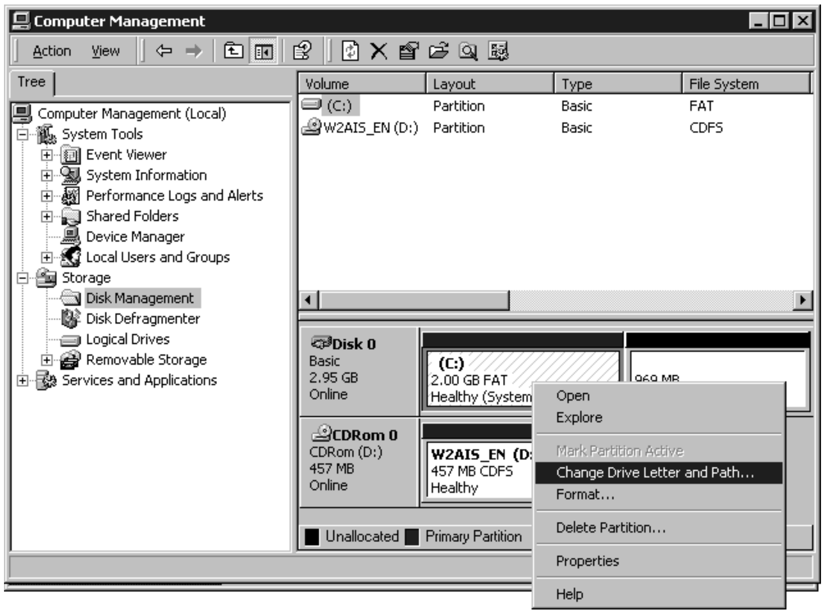
Linux and Windows have different ways of naming partitions. Windows simply assigns each partition a letter starting with C. Letters A and B are reserved for floppy drives, since the first PCs came with two floppy drives. The remaining drive letters are assigned as follows:

1. The first primary partition on each drive.
2. The volumes inside the extended partitions on each drive.
3. The remaining primary partitions on each drive.
4. The CD-ROM drive.

For example, if you had two hard drives each with two primary partitions with two volumes in extended partitions on each drive, they would be named as follows:

C: The first primary partition on the first drive.
D: The first primary partition on the second drive.
E: The first extended partition on the first drive.
F: The second extended partition on the first drive.
G: The first extended partition on the second drive.
H: The second extended partition on the second drive.
I: The second partition on the first drive.
J: The second partition on the second drive.
K: The CD-ROM drive.

These drive letters can't be changed in Windows 3x and 9x, but they can easily be changed in Windows NT and 2000.



Volume	Layout	Type	File System
(C:)	Partition	Basic	FAT
W2AIS_EN (D:)	Partition	Basic	CDFS

Figure 1.1 Disk Administration for Windows 2000.

For Windows NT, go to Start -> Programs -> Administrative Tools -> Disk Administrator. For Windows 2000, go to Start -> Programs -> Administrative Tools -> Configure Your Server. From here, choose File Server -> Open Computer Management, then choose Storage -> Disk Management. This allows you to change the partitions and drive letters. You can change the drive letter by simply right-clicking on the drive and then choosing Change Drive Letter and Path. You can then add, edit, or delete the drive. One note, though: You cannot change your boot partition. This is good because the system won't boot if you do!

Linux treats partitions differently. The first two letters denote the type of drive, the next letter is the drive letter, and the last character is the partition number of the drive. There are four main drive types: IDE (`hd`), SCSI (`sd`), ESDI (`ed`), and RAID (`md`, `rd`, or `ida`). For example, `hda1` is an IDE drive (`hd`), it is the first IDE drive (`a`), and the first partition (`1`). `sdc5` would represent a SCSI drive (`sd`), the third SCSI drive (`c`), and the fifth partition (`5`).

1.4 Linux and Windows 95/98

First, let's go over what happens when Linux and Windows 9x boot up.

1.4.1 Booting Linux

When Linux boots, it loads the `LILO` program, which stands for LInux LOader. This then loads the kernel, which is the core of the operating system. Finally, modules are loaded from the `/etc/rc.d` directory. Actually, the Linux boot process is a little more complicated than this, but this description is good enough for our purposes.

The `LILO` program is configured by using the `/etc/lilo.conf` file. A typical `lilo.conf` would look like this:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.13-4mdk
    label=linux
    root=/dev/hda5
    read-only
```

Let's go over the lines one at a time:

- `boot`—This is the device that contains the boot files.
- `map`—This is the location of the map file. The map file is a binary file containing disk parameters for the system. The default is `/boot/map`.
- `install`—This is the file that is installed as the boot sector. The default is `/boot/boot.b`.
- `prompt timeout`—This is how long it waits before booting in tenths of a second. This allows time to enter boot parameters manually. In the above example, it is five seconds (`50` tenths of a second). If you have a multiple boot system, pressing <SHIFT> will bring up the boot choices. You can set up to 16 different boot configurations.

- `image`—This is the kernel. The parameters below are kernel parameters:
  - `label`—The name that shows up on the boot menu.
  - `root`—The location of the filesystem.
  - `read-only`—The filesystem is mounted read-only so that it can be checked for errors with `fsck`. It is then remounted as read/write.

These aren't the only parameters for `lilo.conf`.

1.4.2 Troubleshooting LILO

`LILO` loads the four letters in "LILO" as it goes through the four stages of loading. This can be helpful in troubleshooting. If `LILO` stops while loading, the letters displayed tell where it failed:

- (<nothing>)—No part of `LILO` has been loaded. `LILO` either isn't installed or the partition on which its boot sector is located isn't active.
- `L <error> ...`—The first-stage boot loader has been loaded and started, but it can't load the second-stage boot loader. The two-digit error codes indicate the type of problem. (See the section titled "Disk Error Codes" in Appendix A.) This condition usually indicates a media failure or a geometry mismatch.
- `LI`—The first-stage boot loader was able to load the second-stage boot loader, but could not execute it. This can either be caused by a geometry mismatch or by moving `/boot/boot.b` without running the map installer.
- `LIL`—The second-stage boot loader has been started, but it can't load the descriptor table from the map file. This is typically caused by a media failure or by a geometry mismatch.
- `LIL?`—The second-stage boot loader has been loaded at an incorrect address. This is typically caused by a subtle geometry mismatch or by moving `/boot/boot.b` without running the map installer.
- `LIL-` —The descriptor table is corrupt. This can either be caused by a geometry mismatch or by moving `/boot/map` without running the map installer.
- `LILO`—All parts of `LILO` have been successfully loaded.

If Linux doesn't load, it will give an error code. The meanings of the error codes are listed in Appendix A.

1.4.3 Booting Windows 9x

The way Windows boots is slightly different. The partition points to the boot sector, which loads the two text files `config.sys` and `autoexec.bat`. It then loads the Windows equivalent of the Linux kernel, `win.com`. The configuration file for `win.com` is the Registry, which consists of two binary files: `system.dat` and `user.dat`. While `config.sys` and `autoexec.bat` are text files, Registry files need to be edited with `regedit`.

There are several ways to install Linux and Windows 3x/9x on the same machine. They are listed below, starting with UMSDOS. Before anything is done, back up your hard drive and run `scandisk` on the hard drive to correct any errors. This will save you a lot of trouble down the road.

1.4.4 UMSDOS

The easiest way to install Linux on a Windows 9x machine is to use UMSDOS, which allows Linux to co-exist with Windows on a FAT or FAT32 partition. It allows Linux to boot directly from a command prompt on the FAT partition. On the plus side, you avoid re-partitioning the hard drive or disturbing the existing Windows installation. On the minus side, you do lose the security and robustness of Linux's native ext2 filesystem.

If you plan on using UMSDOS, it is best to choose a distribution with an automated UMSDOS install. Most modern distributions have a UMSDOS install. One Linux distribution designed to easily install as UMSDOS is Phat Linux. It is a hefty download, so unless you have a fast connection and a lot of time, you might want to buy the CD-ROM. As a bonus, you also get technical support with the purchase.

Phat Linux was originally put together by two high school kids who wanted an easy way to install Linux. To install it, unzip the files to the `\phat` directory and run `linux.bat`. This should start booting up into Linux. If it hangs during the bootup, it may be necessary to reboot, press F8 on bootup, and choose the Safe Mode command prompt. After the bootup, there will be a menu that lets you choose the configuration to set up the video, sound, and networking. The `linux.bat` is only three lines, as shown below:

```
loadlin vmlinuz initrd=ramdisk.gz mem=96M
echo Linux failed to load.
pause
```

The first line actually loads Phat Linux. The command is broken down below:

- `loadlin`—This is the loader for UMSDOS.
- `vmlinuz`—This is the Linux kernel.
- `initrd`—Points to a memory image. Phat Linux uses a compressed memory image to start the operating system.
- `ramdisk.gz`—A compressed file that contains the complete Linux filesystem. This can also be a directory.
- `Mem`—The amount of memory in megabytes.

Phat Linux is based on Red Hat with a KDE desktop, so anyone familiar with Red Hat should be able to use Phat Linux.

UMSDOS can easily be installed manually since support for it has been built into the Linux kernel for some time. Most distributions will set up everything for you, but knowing how it works will help with troubleshooting.

The default directory in which to install UMSDOS is `\linux`. This acts like the root directory for Linux. Under this are the standard Linux directories of `bin`, `etc`, `lib`, `root`, `sbin`, `tmp`, `usr`, and `var`.

Then there is the problem of swap space. Normally, Linux creates a separate swap partition. This has the advantage of being fast and robust. With UMSDOS, however, it is usually better to create a swap file. To create a swap file, type the following:

```
dd if=/dev/zero bs=<block size> count=<number of blocks> of=/<swap
file name> mkswap /<swap file name> <swap file size in bytes>
sync
swapon /<swap file name>
```

Then add the following to your `/etc/fstab`:

```
/<swap file name>   swap    swap    defaults
```

To determine the block size, run `chkdsk` and it will list the number of bytes in each allocation unit. For example, if our block size is 2048 and we want to create a 16 MB swap file called `swap`, we would enter the following commands:

```
dd if=dev/zero bs=2048 count=8 of=/swap
mkswap /swap 16384
sync
swapon swap
```

Then we would add the following to the `/etc/fstab`:

```
/swap swap swap defaults
```

1.4.5 Booting with UMSDOS

`loadlin` is the loader for UMSDOS. From a DOS prompt, its command would look like this:

```
loadlin <Linux kernel> root=<root for UMSDOS>
```

For example, if the kernel is located at `c:\linux\boot\vmlinuz` and UMSDOS is installed in `c:\linux`, the command for `loadlin` would be:

```
loadlin c:\linux\boot\vmlinuz rw root=c:\linux
```

The `rw` option tells `loadlin` to load the Linux filesystem with the read and write options. You can also load it read-only (`ro`) if needed.

You can also copy `loadlin.exe` and `vmlinuz` to a bootable DOS floppy if you want to use a boot floppy to load Linux. This will save you the trouble of configuring the boot loader.

1.4.6 Managing UMSDOS Filesystems

UMSDOS puts a file called `--linux-.---` in each directory. This file stores the extended attributes for the Linux files. As we discussed above, DOS only supports read, write, hidden, archive, and system attributes. Linux also has user, group, and executable attributes. These are stored in the `--linux-.---` file.

The `--linux-.---` file is maintained with the `umssync` utility. This utility will create the `--linux-.---` file if it doesn't exist. If the file does exist, `umssync` will update Linux attributes stored in the file. It is a good idea to run this utility often to keep the information up-to-date. The following line can be added to `cron` jobs or `rc.d` (see your user manual for an explanation of how to do this):

```
/sbin/umssync -r99 -c -i+ <root of Linux file system>
```

The `-c` option will only update existing `--linux-.---` files and not create new files. Directories without the `--linux-.---` file in them are ignored by Linux.

1.4.7 Working with DOS and UMSDOS

Files created by DOS are invisible to Linux unless `umssync` is used. If you try to create a file in Linux with the same name as a DOS file, it will say that the file already exists. Other than that, there are no problems with running DOS and UMSDOS on the same partition. You can even use a DOS defragmentation utility on the partition without affecting the UMSDOS filesystem.

1.5 Setting up Linux and Windows 3x/9x on Separate Partitions

Before working with partitions on any drive, make a good backup of all current partitions and have a DOS boot floppy. If done right, the existing files should be unharmed, but there is always a chance of something going wrong.

The easiest way to load Linux on a separate partition is to load it onto a different drive. If I have a primary IDE drive on the first IDE port with Windows 98 on it (`hda1`) and I put a primary drive as the second IDE port for Linux, it would be `hdb`. During the install, I would simply install a Linux partition and swap drive on `hdc`. The only choice I have left then is whether to boot with `LILO` or `loadlin`.

1.5.1 Using LILO to Dual Boot

To use `LILO`, install `LILO` onto `hda`. The installation program will usually ask you where to install `LILO`. You will need to check the documentation for instructions on how to install `LILO`, since it varies with different installation programs.

Next, you need to edit `/etc/lilo.conf`. It will look something like this:

```
boot=/dev/hdc
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz-2.2.13-4mdk
    label=linux
    root=/dev/hdc5
    read-only
```

Add this to the end of `lilo.conf`:

```
other=/dev/hda1
table=/dev/hda
label=Windows98
```

This will give you the choice of "linux" or "Windows98" on bootup.

1.5.2 Using loadlin to Boot Up

Of course, `loadlin` can also be used to load Linux from the second drive. First put `loadlin` and `vmlinuz` onto your Windows partition. During bootup, press F8 and choose the Safe Mode command prompt. In the above example, we would run `loadlin` from the root of C: with the following parameters:

```
    loadlin vmlinuz root=/dev/hdc5 ro
```

This would load the kernel `vmlinuz`, set the Linux root to partition 5 on the primary drive on the second IDE port (`hdc5`), and set the root to read-only (`ro`). Change these settings to match the drive on which that Linux was installed.

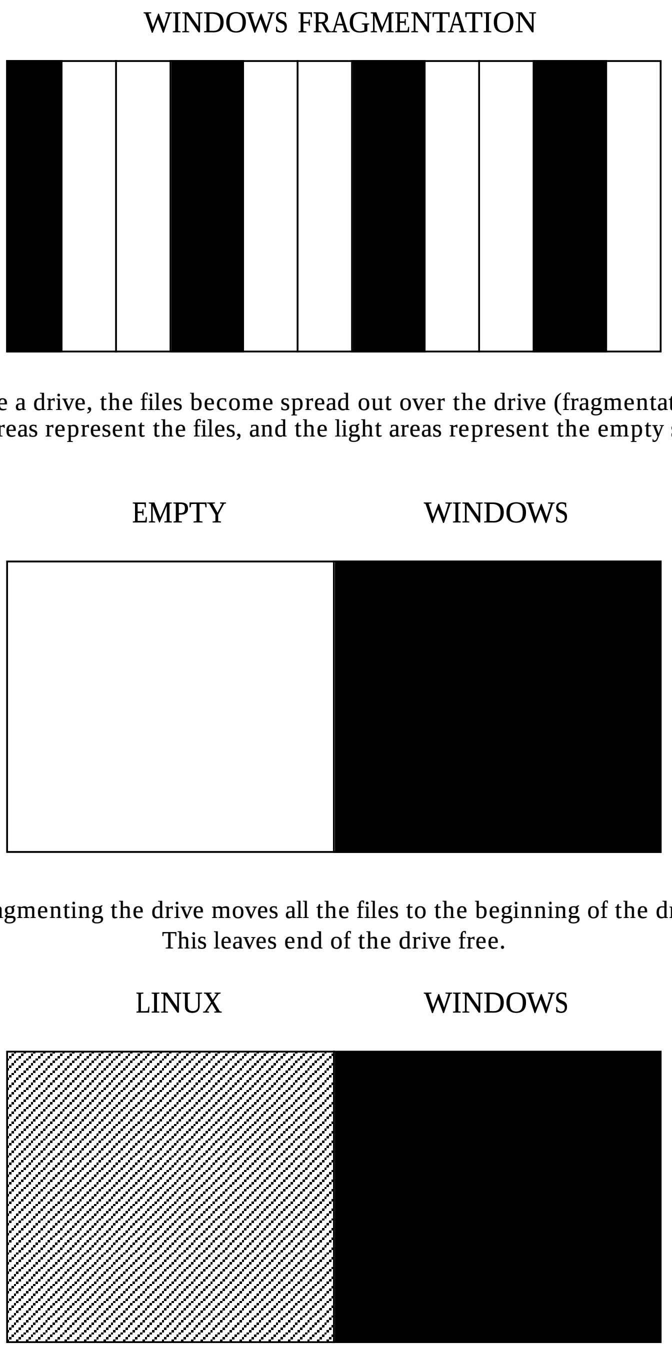
1.6 Partitioning an Existing Hard Drive

Although the best way to load Linux on a separate partition is to load it onto its own hard drive, not everyone has more than one hard drive. With hard drives selling for about $100 these days, it is less of a problem, but if you must, load Linux onto a separate partition on the same hard drive as Windows. There are several programs that allow you to manipulate partitions. If you have 200–1200 MB available on your hard drive, you can create a Linux partition out of the free space.

The first thing to do before manipulating the partitions is to back up your hard drive. Next, run `scandisk` on the drive in thorough mode to make sure there are no errors that could cause problems. Then defragment the hard drive with the Windows `defrag` program. This will move all the files to the beginning of the drive and free up space at the end of the drive for a new partition.

Before we repartition the drive, keep in mind that the boot partition of both operating systems must be within the first 1024 cylinders of the hard drive. This is within the first 504 MB on most hard drives. This is a limitation of PC hardware that dates back to when the maximum size of hard drives was 504 MB.

Once we have backed up and checked the drive, we need to decide how to partition the drive: the hard way or the easy way. Let's start with the hard way.



As you use a drive, the files become spread out over the drive (fragmentation). The dark areas represent the files, and the light areas represent the empty space.

Defragmenting the drive moves all the files to the beginning of the drive. This leaves end of the drive free.

We can then create a partition at the end of the drive, and install Linux on that partition.

Figure 1.2 Disk partitioning.

1.6.1 fdisk

If you have all your data backed up (THIS WILL DESTROY ALL DATA ON THE DRIVE), you can use `fdisk` to re-partition the drive. First, boot to DOS in Safe Mode. Then run `\windows\fdisk`. Choose Delete Partition. Then choose Extended. Then delete the primary partition. Once this is done, use `fdisk` to create a DOS partition.

Once you have created the DOS partition, install Windows and all your applications on it. Then restore your data. This is a lot of work and will probably take several hours. There is an easier way.

1.6.2 Resizing Existing Partitions

There are several programs that will resize a partition. There is the commercial program Partition Magic that has a nice graphical interface, and it works with all PC partitions. It is well worth its price. Partition Magic is available at `http://www.powerquest.com`.

Many Linux installation programs allow the hard drive to be re-partitioned during installation. Instructions should be included in your distribution's documentation. If your distribution doesn't allow this, you can do it manually with FIPS.

1.6.3 FIPS

If you don't mind a text-based interface, there is FIPS, which is a DOS program that resizes partitions. It is on most Linux distribution CD-ROMs under the subdirectory `/dosutils`. Otherwise, you can get a copy of it at: `http://www.igd.fhg.de/nashaefe/fips/`

Compared to many other programs, FIPS has few command-line switches. Those it has are listed below:

- `-t` or `-test`—This doesn't write anything to disk.
- `-d` or `-debug`—This will write errors to the file `FIPSINFO.DBG`. It can help in diagnosing problems.
- `-h` or `-help` or `-?`—The help text.
- `-n<num>`—The drive number to split.

1.6.4 Restrictions of FIPS

There are some restrictions on what you can do with FIPS. The first is that your hard drive must support using INT 13 for low-level disk access. There are a few older Adaptec drive controllers that don't support INT 13, but almost all other drives do, including all newer Adaptec drive controllers.

FIPS also will not work with FAT12. FAT12 is used on partitions that are smaller than 10 MB. This shouldn't cause any problems since it would be useless to split a 10 MB partition anyhow.

You can only split standard FAT and FAT32 partitions. FIPS will not work on extended partitions, NTFS, HPFS (Os2's filesystem), Linux, or any other non-FAT partition.

FIPS will not work with disk managers such as OnTrack. You must uninstall OnTrack, which requires deleting the partition and re-installing (see “fdisk” above).

Also, don’t reduce the original partition to less than 4085 clusters. A FAT partition needs at least 4085 clusters. There is FAT12 for smaller partitions.

And lastly, you can’t create a new partition on a drive if it already has four partitions. This is because FAT only supports four partitions to a disk.

There are a few special situations that require extra steps to re-partiton. If you are using Windows 3.1, any disk compression software (Stacker, DoubleSpace, or SuperStor), or disk mirroring software (Image or Mirror), see the section below entitled “Special Situations with FIPS”.

1.6.5 Using FIPS

If your partition doesn’t have any of the restrictions listed above, FIPS can be used to resize your partition. There are three things that need to be done before you resize the partition:

1. Back up your data—Something may go wrong.
2. Run `scandisk` or `chkdsk /f`—This will correct errors on the drive.
3. Run `defrag`—This will free up space at the end of the drive.

If there is no space available at the end of the drive, check for hidden or system files by typing:

```
    cd \
```

Then type one of the following:

- `dir /a:h /s`—Searches for hidden files.
- `dir /a:s /s`—Searches for system files.

First, check to see what these files do so that you don’t delete any important files. Don’t, for instance, delete or move the `io.com` or `dos.com` or your system will not boot!

For example, the programs IMAGE and MIRROR create the files `image.idx` and `mirror.fil`. These files are used to recover a corrupted disk and they are created each time the system boots. To delete these files, first change the attributes with the commands:

```
attrib -r -s -h image.idx
attrib -r -s -h mirror.fil
```

You can then delete the file normally (`del image.idx` or `del mirror.fil`). Next, make a bootable floppy and copy the files `RESTORRB.EXE`, `FIPS.EXE`, and `ERRORS.TXT` to this disk. You can make a bootable floppy with `format a: /s` or `sys a:`. Some PCs are set so that they won’t boot off a floppy. Consult your computer’s documentation (or a local computer guru) for instructions on how to enable booting from a floppy.

Now that the disk is prepared, boot from the floppy you just created. At the prompt, type `FIPS`. You can exit at any time by pressing <CTRL> `c`. `FIPS` will first try to detect the operating system you are using. Since we booted off a DOS floppy, there should be not problem. It will next try to detect your hard disks. Then it will read the partitions on each drive and display the partition table such as the one shown below (from `FIPS.DOC`) :

```
     |        |     Start      |      |      End       | Start  |Number of|
Part.|bootable|Head Cyl. Sector|System|Head Cyl. Sector| Sector |Sectors  |  MB
-----+--------+----------------+------+----------------+--------+---------+----
1    |    yes |   0  148      1|   83h|  15  295     63|  149184|   149184|  72
2    |     no |   1    0      1|   06h|  15  139     63|      63|   141057|  68
3    |     no |   0  140      1|   06h|  15  147     63|  141120|     8064|   3
4    |     no |   0    0      0|   00h|   0    0      0|       0|        0|   0
```

This is a lot of information. The most important data is the number of megabytes. FIPS will next check the root sector for errors. If you have more than one partition, you will be asked which partition to split. Once you choose a partition, FIPS will show the drive information:

```
Bytes per sector: 512
Sectors per cluster: 8
Reserved sectors: 1
Number of FATs: 2
Number of rootdirectory entries: 512
Number of sectors (short): 0
Media descriptor byte: f8h
Sectors per FAT: 145
Sectors per track: 63
Drive heads: 16
Hidden sectors: 63
Number of sectors (long): 141057
Physical drive number: 80h
Signature: 29h
```

FIPS will then check the drive for errors and free space. If it exits with an error message, make sure you did all of the preparation steps above.

If there are no errors, FIPS will show the size of the original and new partitions. Use the left and right cursor (arrow) keys to change the size of the two partitions. Once you have them at the desired size, press <ENTER>.

FIPS will then recheck the new partition to make sure it is empty. It will show the changes to be made to the partition. You may press `r` to re-edit or `c` to continue. It will then ask if you want to write the changes to disk. Answer `y` to save the changes and FIPS will exit. Reboot the machine to save the changes.

After rebooting, use `scandisk` to check both partitions to make sure they are okay. If there are errors, you can restore the partition by rebooting with the DOS disk and running `restorrb`.

The new partition is a standard DOS partition when first created. When you install Linux, you can use the installation program to delete the new partition and create a Linux partition. Just be sure you delete the correct partition!

Booting Linux to a new partition is no different than booting it on a new hard drive. See the previous sections on dual booting with Linux and Windows 3x/9x.

1.6.6 Special Situations with FIPS

Windows 3.1. If you are using Windows 3.1, you must delete the swap file before splitting the drive. To do this, go the Control Panel (in the Main folder) and choose 386 enhanced. Then choose the Virtual Memory, and change to none. After the drive is split, you can turn the swap file back on.

Stacker, SuperStor, DoubleSpace, and other Disk Compression Programs. These programs create a compressed volume on any disk with a compressed file on it. Then they move all the files to the compressed volume and rename the volumes. The uncompressed volume is typically C: and it contains the boot files and compression program. The compressed volume is D: and it contains the compressed file.

Splitting this drive can be tricky. Be sure to get a good backup because if the compressed file is damaged, the whole drive becomes unreadable. The following steps should allow you to add a new partition to a compressed drive:

1. Make sure you have enough free space on the compressed drive to create the new partition.
2. Use the disk checking software that comes with the drive to check for errors.
3. If you are running Windows 3.1, remove the swap file.
4. Use the disk compression utilities to decrease the size of the compressed volume.
5. Defragment the compressed volume (D:).
6. Use FIPS and split the compressed volume (D:)

If the compressed volume can't be defragmented, try the following steps:

1. Copy your disk defragmentation utility (for example, `defrag.exe` or `diskorg.exe`) and `attrib.exe` (in `C:\DOS`) to the bootable floppy drive.
2. Boot with the floppy and remove all the hidden and system attributes from the files on the compressed drive (D:). Use `dir /a:h` and `dir/a:s` to find the hidden and system files.
3. Defragment the compressed partition (D:).

After this is done, you should be able to split the drive.

1.6.7 NTFS Partitions

Unfortunately, FIPS and UMSDOS don't work on NTFS partitions. You either have to delete the partition and re-create it or use a commercial partition utility such as Partition Magic. Partition Magic is available in most large computer stores. Information on it is available at `http://www.powerquest.com`.

Before manipulating NTFS partitions, make a rescue disk, which saves important system information to a diskette so it can be restored if something goes wrong. For Windows NT, use Start -> Run, type in `rdisk`, and choose Make Emergency Repair Disk. For Windows 2000, go to Start -> Accessories -> System Tools -> Backup. On the Welcome tab is a button for Emergency Repair Disk. Follow the instructions.

Windows NT and 2000 don't use `fdisk`. They have their own partitioning program called Disk Administrator. During the install, you are asked how big to make the partition. By default, the partition will be the entire drive. NT partitions are limited to 4GB on the boot partition and 7.5GB on all other partitions. Windows 2000 doesn't have this limitation.

Once there is free space on the drive, install Linux on the free space. If Linux and NT are on the same drive, you will want to use NT as the boot loader and install `LILO` on the Linux partition. Most distributions allow you to specify where to install the boot loader. See your distribution's documentation and help files for information on this.

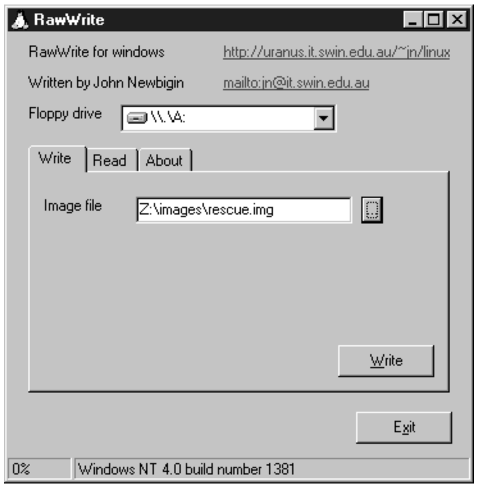
If you have Linux and NT on separate drives, you can use `LILO`. See the section below titled "Using LILO".

Before we start manipulating the NTFS partition, we need to download BootPart by G. Vollant. This program can create and manipulate NTFS boot sectors. Go to `http://www.winimage.com/bootpart.htm`.

1.6.8 Using NT's Boot Loader

The boot process for Windows NT is different than Windows 3x/9x. The main difference we need to be concerned with is that Windows NT uses a file instead of the partition's boot record to load the operating system. This means that we will have to make a boot file for Linux if we want to use Windows NT's boot loader to load Linux.

To create a Linux boot file, we need to copy the boot sector of the Linux partition to a file. To do this, boot from a Linux rescue diskette and use `dd` to copy the boot sector to a file. If you don't have a rescue diskette, you can create one in Windows by running the program `rawwrite.exe`, which can be found in the `\dosutils` directory on a Linux installation CD-ROM. Once the program is open, choose the Write option, and then choose the `rescue.imq` file in the `\images` directory on the Linux installation CD-ROM (Figure 1.3).


RawWrite

RawWrite for windows http://uranus.it.swin.edu.au/~jn/linux

Written by John Newbigin mailto:jn@it.swin.edu.au

Floppy drive \\.\A:

Image file Z:\images\rescue.img

0% Windows NT 4.0 build number 1381

Figure 1.3 Use RawWrite on your Linux installation CD-ROM to copy the rescue image to the A: drive.

Boot off the rescue diskette and copy the boot partition to the floppy as follows:

```
dd if=<Linux partition> of=<name of file> bs=512 count=1
```

Let's go over the command step by step:

- `if`—The location at which to start copying. In this case, we are starting at the beginning of the Linux partition.
- `of`—The output file's name.
- `bs`—The block size. The boot sector is 512 bytes.
- `count`—The number of blocks copied. We are only copying one block—the boot sector.

Let's use an example of having Linux on `hda2` (the second partition on the first hard drive) and let's name the file `boot.lnx` (the Linux boot sector):

```
dd if=/dev/hda2 of=boot.lnx bs=512 count=1
```

Next, copy the file `boot.lnx` to a DOS formatted floppy:

```
mcopy boot.lnx a:
```

`mcopy` is part of the mtools programs. For more information go to `http://www.tux.org/pub/tux/knaff/mtools`. You could also use these commands to copy `boot.lnx` to a floppy:

```
mount -t msdos /dev/fd0 /mnt/floppy
cp boot.lnx /mnt/floppy
umount /mnt/floppy
```

Now, reboot into NT, log in as administrator, and copy `boot.lnx` from A: to the root of C:. Then, edit `boot.ini`. First, take off the system and read-only attributes:

```
attrib -s -r c:\boot.ini
```

Next, edit `boot.ini` with notepad and add a line for booting to Linux:

```
[boot loader]
  timeout=30
  default=multi(0)disk(0)rdisk(0)partition(1)\WINNT
  [operating systems]
  multi(0)disk(0)rdisk(0)partition(1)\WINNT="Windows NT Workstation
     Version 4.0
  multi(0)disk(0)rdisk(0)partition(1)\WINNT="Windows NT Workstation
     Version 4.0 [VGA mode] /basevideo /sos
  C:\BOOT.LNX="Linux"
```

The last line will load Linux. When you are done editing and saving `boot.ini`, change the attributes back:

```
attrib +s +r c:\boot.ini
```

When you boot up NT, you will get the following menu, which allows you to select the system to boot:

```
OS Loader V4.00
   Please select the operating system to start:
   Windows NT Workstation Version 4.0
   Windows NT Workstation Version 4.0 [VGA mode]
   Linux
```

If you change the boot sector of Linux, you must make a new `boot.lnx`. This usually only happens when you upgrade the Linux kernel.

All these steps can be done automatically with BootPart. Simply run the BootPart program and choose the location of the Linux partition. It will edit the `boot.ini` and create the Linux boot file automatically.

1.6.9 Using LILO

Windows NT requires its own master boot record on the drive. To boot NT from `LILO`, NT must be loaded on a separate drive. To use `LILO` to boot to NT, Linux must be on the first drive and NT on the second drive.

If we install `LILO` on the first partition, we must modify `lilo.conf`. Most Linux installation programs allow you to create a `LILO` menu item for another OS (operating system). Check your manuals to see if it is supported. If not, manually edit the `lilo.conf`, which will look something like this:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
default=linux
keytable=/boot/us.klt
prompt
timeout=50
message=/boot/message image=/boot/vmlinuz
label=linux
root=/dev/hda5
read-only
```

If Windows NT is on the second drive, add the following to the end of `lilo.conf`:

```
other=/dev/hdb1
table=/dev/hda
loader=/boot/any_d.b
label=WindowsNT
```

This is what the added lines mean:

- `other`—This points to the first partition on the second hard drive (`dev/hdb1`).
- `table`—This is where the drive table is. This is required by `LILO`.
- `loader=/boot/any_d.b`—Required when not booting from the primary drive.
- `label`—The name for the section.