7 Disk Subsystems

With disks and memory, quite a bit can go wrong, but the good news is that normally a problem can be pinpointed very quickly. Aside from hardware failures and disk corruption giving you headaches, the threat of a virus must always be considered as well. The core of system operation revolves around its data, which normally resides on a hard disk. Whether or not you have fault-tolerant disks in place on your servers, odds are that you cannot afford to have such protection on all of your managed workstations. Disk failures are inevitable. This chapter will help you to quickly diagnose the source of a disk failure and in turn repair the problem.

In addition to describing the multitude of disk management and diagnostic tools at your disposal, this chapter also shows you the most common disk errors and faults, and proven procedures for resolving them.

Windows Server 2003/XP Disk Architecture 101

Before jumping headfirst into the pool of disk troubleshooting, let's first test the water. Resolving faults and repairing disk-related problems does require some knowledge of the basics. In this section, you will not read the dry information on the guts of hard disks (a proven tranquilizer for IT professionals). Instead, this section quickly reviews Microsoft disk terminology, including

- Basic and dynamic disks
- Master Boot Record
- Boot sector
- boot.ini file
- Master File Table
- GPT disks

Basic and Dynamic Disks

Basic disks are the disk type that you have come to know and love when working with Microsoft products. Basic disks are compatible with all Microsoft operating systems and are divided into logical partitions. Dynamic disks, on the other hand, are only compatible with post–Windows 2000 operating systems and are divided into volumes.

While basic disks are limited in functionality, dynamic disks offer a great deal of flexibility in their configuration options. With dynamic disks, you can configure the following software-based disk solutions.

- *Spanned volume*—A single volume that spans two or more physical disks, with data written to the first disk until it is full and then written to the next disk.
- *Striped volume (RAID 0)*—Data stored on two or more physical disks, with data reads and writes going to all disks simultaneously.
- *Mirrored volume (RAID 1)*—Data stored on two or more disks that mirror each other for fault tolerance. If one disk fails, all data can be recovered from the second disk.
- *Striped volume with parity (RAID 5)*—Data stored on three or more physical disks, with data reads and writes going to all disks simultaneously. Unlike RAID 0, RAID 5 volumes can withstand the loss of a single disk.

Whenever possible, you should elect to invest in a hardware RAID solution, which offers better performance and recoverability. Since this reference is dedicated to software troubleshooting, this chapter addresses diagnosing and recovering basic and dynamic disks. As far as RAID configurations are concerned, you will see in this chapter how to recover failed dynamic disks. For hardware RAID troubleshooting and recovery procedures, you should be able to find plenty of information from your systems' RAID controller vendors.

While you can convert a basic disk to a dynamic disk, you cannot convert dynamic disks to basic disks. Once you make a disk dynamic, reverting it to a basic disk requires you to back up its data, delete the disk's volumes, convert it to a basic disk, and then restore the original data.

Master Boot Record

The Master Boot Record (MBR) is an entry located on the first sector on the first partition of a hard disk that contains information on its partition structure and a small amount of executable code that is loaded into RAM at startup. A portion of the executable code contained in the MBR is the location of the boot sector. The fact that the MBR contains the initial information for your system to start makes it a very vulnerable target. A corrupt MBR can easily shut down a system, and thus diagnosing it and repairing the MBR is an important aspect of disk subsystem troubleshooting.

Boot Sector

The boot sector is located on the next sector in the first partition, or on the first sector of subsequent disk partitions. The boot sector contains information on how the computer can “boot” the operating system, and thus is just as critical as the MBR in terms of system vulnerability. On Windows systems, the boot sector locates information on operating systems by reading the boot.ini file, which is covered next.

The boot.ini File

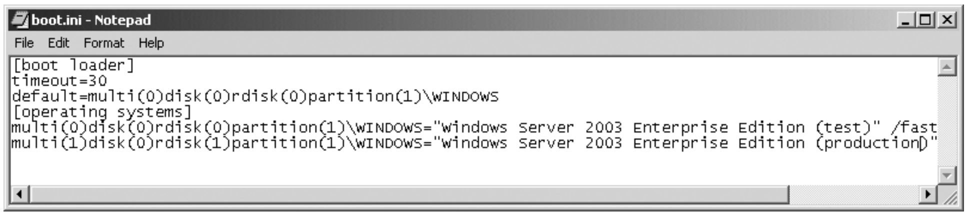
The boot.ini file is a file located on a computer’s system partition that contains information on where to find operating systems on the boot partitions. If you’re scratching your head and wondering about a boot file on a system partition, you’re not alone. Believe it or not, this really is not analogous to cars driving on parkways and parking on driveways.

Here’s why. Your computer starts by reading data on its system partition because it is looking for information on where to find operating systems. The boot.ini file that is on the system partition contains that information. When your computer boots up, it is loading an operating system. Since operating systems are needed to boot, they are found on the boot partition. Of course, if your operating system is installed on the C drive, then your boot and system partitions are actually on the same partition, and thus you have absolutely nothing to ponder!

The most important information found in boot.ini files are Advanced RISC (Reduced Instruction Set) Computing (ARC) paths. The ARC paths tell your system where to find operating systems. ARC paths only cause confusion when they need to be manually edited, and unfortunately these times are often when a system has crashed and you need to recover as quickly as possible. One of the most frequent needs to edit an ARC path comes when the primary drive in a software mirror fails. When this happens, the ARC path in the boot.ini must be edited to tell the computer that the operating system must be loaded from the second disk in the mirror and not the first. With hardware mirroring, this problem does not occur, because both disks in the mirror are seen by the system software as a single physical disk.

Figure 7-1 shows a typical boot.ini file. In this example, the file references two operating systems, one for test and the other for production. What is not indicated in the file is that the second OS, “Windows Server 2003 Enterprise Edition (production),” is mirrored with a second physical disk for fault tolerance. If the disk containing the second OS fails, you need a way to tell the system to boot from the disk that the second operating system is mirrored with. This means that you have to edit the boot.ini file.

In the example, assume that the system contains four SCSI disks, with OS 1 on disk 1 (SCSI ID 0), and OS 2 mirrored on disk 2 (SCSI ID 1) and disk 3 (SCSI ID 2). If OS 2



```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows Server 2003 Enterprise Edition (test)" /fast
multi(1)disk(0)rdisk(1)partition(1)\WINDOWS="Windows Server 2003 Enterprise Edition (production)"
```

Figure 7-1. Sample boot.ini file showing two operating systems

failed, you would need to end the second line in the [operating systems] portion of the boot.ini file to read as follows:

```
multi(1)disk(0)rdisk(2)partition(1)\WINDOWS=”Windows Server 2003
Enterprise Edition (production)” /fastdetect
```

If you notice, the only item that was changed was the `rdisk` value. This told the system that the operating system could be found on the third disk in the SCSI chain. While having the answer to the solution is fine, it is more important to understand how the answer was reached.

For starters, all ARC path entries begin with either `multi` or `scsi`. Notice in the example that even though SCSI disks were installed on the system, `multi` was used. This is because you only use `scsi` in place of `multi` when the system boots from SCSI disks whose associated SCSI controller’s BIOS is disabled. If the SCSI BIOS is enabled, you always use `multi`. To summarize:

- Use `multi` for all IDE disks and SCSI disks when their controller’s SCSI BIOS is enabled.
- Use `scsi` for SCSI disks whose controller’s SCSI BIOS is disabled.

Now that `multi` and `scsi` are out of the way, let’s focus on the next two entries: `disk` and `rdisk`. The easiest way to remember the usage of these two entries is with the “4-5 Rule.” There are four letters in `scsi` and `disk` and five letters in `multi` and `rdisk`. This means that whenever `scsi` is the first term in an ARC path, you edit the `disk` parameter, and when `multi` is listed, then you edit the `rdisk` value. Values for `rdisk` begin with 0 and go up based on physical disk location. Keep in mind that this may not have any relation to SCSI IDs—for example, if you have four disks with IDs 0, 2, 4, and 6. The disk with SCSI ID 6 (logically the fourth disk) would have an `rdisk` value of 3 (the fourth logical value). If you had operating systems on three IDE disks (primary master, primary slave, secondary master), you would use `rdisk` values of 0, 1, and 2, respectively.

The first three entries get the system to a physical disk. The last entry, `partition`, simply tells the system the logical partition on the disk where the OS resides. Unlike the other values, partition values begin at 1 instead of 0. So an OS on the second physical disk on the second partition would have these values: `rdisk(1)partition(2)`. Several switches may accompany operating system entries in the boot.ini file; while not completely relevant from a troubleshooting perspective, they are listed in Table 7-1 to satisfy your curiosity. Relevant switches for troubleshooting purposes are explained in the `bootcfg` command section of this chapter.

Table 7-1. ARC Path Operating System Switches

Switch	Purpose
`/3GB`	For applications designed to take advantage of additional address space, this option is used to tell x86-based systems to allocate 3GB of virtual address space for applications and 1GB for OS kernel and executive components.
`/basevideo`	Tells the OS to boot using standard VGA colors (640 by 480, 16 color), which is useful if the system hangs or the screen locks after you install a new video driver.
`/baudrate=<value>`	Allows you to specify a baud rate to be used for kernel debugging. The default is 9600, but you can change the rate up to 115200. This switch is used in conjunction with the `/debug` switch.
`/bootlog`	This switch enables bootup events to be written to the %systemroot%\bootlog.txt file, which may be helpful for troubleshooting startup problems.
`/bummemory=<value>`	This option is used to specify a value of memory (in megabytes) that cannot be used by the OS. This option is normally used to test for problems caused by a lack of available memory.
`/crashdebug`	This switch is useful if you experience continual stop errors on a system. With this switch, the kernel debugger is loaded with the OS and activates when a stop event is encountered.
`/debug`	This switch loads and activates the kernel debugger as soon as the OS starts.
`/debugport=<port>`	This switch is used to indicate the COM port to be used for kernel-level debugging. The default value is `COM1`, but you can specify other ports, such as COM2 or 1394.
`/fastdetect[=<port>]`	This default switch turns off serial and bus mouse detection in the Ntdetect.com file. If you specify an optional port, such as COM1 or COM2, the system will detect components attached to that port during startup.

continued

Table 7-1. ARC Path Operating System Switches, continued

Switch	Purpose
`/maxmem=<value>`	Allows you to specify a maximum value of memory for use by the OS. While the `/bummemory` switch takes a specific amount of memory away, this switch lets you indicate exactly how much physical memory can be seen and used by the OS and is generally used to diagnose memory-related performance problems.
`/noguiboot`	This option disables the Windows bitmap from appearing while the OS is loading.
`/nodebug`	Disables kernel debugging.
`/numproc=<value>`	Allows you to force a multi-CPU computer to only use the number of CPUs that you specify.
`/pcilock`	On x86 systems, this switch prevents the OS from dynamically assigning IRQs and I/O addresses to PCI devices, leaving that job up to the system BIOS instead.
`/safeboot:<value>`	Makes the system start in Safe Mode. The version of Safe Mode the system starts in is determined by the value specified, which may be `minimal` or `network`.
`/sos`	Causes the name of each device driver to be displayed as it loads, allowing you to clearly see whether a faulty driver is causing a system to not respond or is failing to load.

This section has shown you all you need to properly edit the boot.ini file. In Chapter 5, you learned how to boot the system to the Recovery Console. When you need to edit the boot.ini file after a failure, the easiest method to modify the file is to run the `bootcfg` command from the console. With `bootcfg`, you can modify an OS entry so that its `rdisk` value points to the proper physical disk, for example.

Master File Table

The Master File Table (MFT) is a database that stores attribute information for each file and folder on a disk. Since this database is continually written to and read from, it is extremely important that the file reside on contiguous disk space. Otherwise, performance would significantly suffer. To prevent MFT fragmentation, NFTS automatically reserves 12.5% of the total disk space for the MFT. This space is known as the MFT Zone. Still, if the MFT grows beyond this allocation, which is extremely rare, the MFT can become fragmented, and disk performance will suffer. If you have a volume that consists mainly of very small files, then MFT fragmentation is a real possibility, and you should consider expanding the allocated size of the MFT Zone. Information on how to increase the size of the MFT Zone is found in the section on `fsutil behavior` later in this chapter.

GPT Disks

With Itanium-based systems, a new partition style exists that replaces the MBR format. On Itanium systems, a Globally Unique Identifier (GUID) Partition Table (GPT) format can be employed, which offers support for much larger partition sizes (128 partitions per disk, 18 exabytes per partition). With MBR disks, the maximum supported volume size is 2 terabytes.

While these numbers may initially seem extreme, they are certainly very realistic considering the exponential growth of storage requirements. Think back to the days when the 100MB storage plateau was surpassed for hard disks. We were all naive to think that you can never fill up a 100MB disk. From a troubleshooting perspective, you do not need to worry about how GPT disks operate under the hood. However, when you view them in Disk Management, they are indexed as GPT disks, and traditional disks are displayed as MBR disks.

Be careful not to use traditional MBR disk tools on GPT disks, such as running `fdisk /mbr`. Doing so may damage the file system to the point that the system will not start.

Disk Troubleshooting Tools

In this section, you will be shown additional tools to add to your toolbox. As was the case with earlier chapters, the tools presented in this section are primarily for advanced troubleshooting and go above and beyond the fundamental disk troubleshooting tools presented in Chapter 4. Following this section, the remainder of the chapter focuses on additional fault isolation methods in disk subsystem troubleshooting as well as how to fix disk problems once they're discovered.

bootcfg

Oftentimes when a disk fails and a system will not boot, technicians realize that they need to edit the boot.ini file to resolve the problem. However, editing the boot.ini file has been a lesson in patience for many, and others look at this task as more of a craft that is learned through years of pain and experience. Regardless of how you look at boot.ini file editing, Microsoft has decided to make your life much easier with `bootcfg`.

The `bootcfg` tool automates editing the boot.ini file, eliminating much of the guesswork that is normally associated with editing the file. There are 11 different ways to run `bootcfg`, each of which will be addressed shortly. Before we get to each command version, first note the switches that each command option has in common. The common command switches are for executing the command to manage a remote system and are shown in Table 7-2.

Table 7-2. bootcfg Common Command Options

Option	Use
`/s <system>`	Used to specify the name or IP address of the remote system you wish to manage.
`/u <user>`	Specifies the name of the domain user under which the command should run.
`/p <password>`	When `/u` is used, specifies the password for the domain user.

The next 11 sections describe the usage of each `bootcfg` version.

The boot.ini file is loaded at startup, so to apply any changes to the boot.ini file, you must restart the system.

bootcfg /addsw

The `bootcfg /addsw` command allows you to configure loading options, such as maximum allowable RAM for a specific operating system entry in the boot.ini file. The syntax for `bootcfg /addsw` is:

```
bootcfg /addsw /id <OSNumber> [/s <system>] [/u <domain\user>]
[/p <password>] [/mm <MaxRAM>] [/bv] [/so] [/ng]
```

The `bootcfg /addsw` options are described in Table 7-3.

Table 7-3. bootcfg /addsw Command Options

Option	Use
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/mm <MaxRAM>`	Adds the `/maxmem` switch along with the amount of memory specified (in megabytes) to the OS line number specified with the `/id` switch.
`/bv`	Adds the `/basevideo` switch to the OS line number specified with the `/id` switch. Adding `/basevideo` to an OS entry in the boot.ini file causes the operating system to boot using a standard VGA video driver.
`/so`	Adds the `/sos` switch to the OS line number specified with the `/id` switch. Adding this switch causes the OS to display device driver names as they are loaded when the system boots.
`/ng`	Adds the `/noguiboot` switch to the OS line number specified with the `/id` switch. This switch hides the Windows progress bar that appears while the system boots.

Here are two examples of using `bootcfg /addsw`:

- To configure the second OS referenced in the boot.ini file to use 128MB of the available 512MB of RAM for testing purposes:
  `bootcfg /addsw /id 2 /mm 128`
- To set the first OS in the boot.ini file to boot using standard VGA video:
  `bootcfg /addsw /id 1 /bv`

bootcfg /copy

The `bootcfg /copy` command is used to duplicate an operating system reference line in the boot.ini file. For example, if only one OS was referenced, you could use `bootcfg /copy` to duplicate the reference and cause the boot.ini file to list two operating systems. Once the line is duplicated, you can then use other `bootcfg` commands to edit the OS entry; otherwise, the new entry will point to the same operating system as the original entry, leaving you with two entries that reference the same operating system.

Here is the syntax for `bootcfg /copy`:

```
bootcfg /copy /id <OSNumber> [/s <system>] [/u <domain\user>]
[/p <password>] [/d <description>]
```

Table 7-4 describes the command options for `bootcfg /copy`.

Table 7-4. bootcfg /copy Command Options

Option	Use
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/d <description>`	Provides a description for the new operating system entry in the boot.ini file.

Since all this command does is duplicate a boot.ini OS entry, there are not many options. To duplicate the first OS reference in the boot.ini file and have users see it referenced as “Test OS” in the boot menu, you would run this command:

```
bootcfg /copy /id 1 /d "Test OS"
```

bootcfg /dbg1394

This option is primarily a concern of developers and not for systems administrators and help desk staff. With `boofcfg /dbg1394`, 1394 port debugging is configured for the operating system specified. The primary advantage of 1394 port debugging is that it offers a substantial performance advantage of using debugging tools through a standard serial port.

The syntax for `boofcfg /dbg1394` is:

```
bootcfg /dbg1394 <on|off|edit> /id <OSNumber> [/s <system>]
[/u <domain\user>] [/p <password>] [/ch <channel>]
```

The `boofcfg /dbg1394` options are explained in Table 7-5.

Table 7-5. bootcfg /dbg1394 Command Options

Option	Use
`on`	Adds the `/dbg1394` switch to the OS line number specified with the `/id` switch. Adding `/dbg1394` to an OS entry in the boot.ini file enables 1394 remote debugging support for that OS.
`off`	Removes the `/dbg1394` switch from the OS line number specified with the `/id` switch, thus disabling 1394 remote debugging support.
`edit`	Allows you to change the port and baud rate settings for the specified OS entry based on the values provided in the `/ch` switch.
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/ch <channel>`	Allows you to specify the channel to use for debugging. Allowable values are any integer between 1 and 64. This switch cannot be used in conjunction with the `off` option.

To turn on IEEE 1394 port debugging on the first operating system referenced in a system's boot.ini file, you would run this command:

```
bootcfg /dbg 1394 on /id 1
```

bootcfg /debug

This command is used to configure standard debugging (via serial port) to an OS entry in the boot.ini file. With serial communications, unlike with IEEE 1394, you need to specify a baud rate and a COM port for the system to use for communication. This syntax for `bootcfg /debug` is:

```
bootcfg /debug <on|off|edit> /id <OSNumber> [/s <system>] [/u
<domain\user>] [/p <password>] [/port <COM Port>] [/baud <baud rate>]
```

The command options are described in Table 7-6.

Table 7-6. bootcfg /debug Command Options

Option	Use
`on`	Adds the `/debug` switch to the OS line number specified with the `/id` switch. Adding `/debug` to an OS entry in the boot.ini file enables standard remote debugging support for that OS.
`off`	Removes the `/debug` switch from the OS line number specified with the `/id` switch, thus disabling standard remote debugging support.
`edit`	Allows you to change the port and baud rate settings for the specified OS entry based on the values provided in the `/port` and `/baud` switches.
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/port <COM Port>`	Adds the `/port` switch to the OS line number specified with the `/id` switch. This is used to indicate which COM port should be used for remote debugging. Valid `COM Port` values are `COM1`, `COM2`, `COM3`, or `COM4`.
`/baud <baud rate>`	Adds the `/baud` switch to the OS line number specified in the `/id` switch. This is used to indicate the baud rate to be used for debugging. Valid baud rate values are 9600, 19200, 38400, 57600, or 115200.

Here is an example of removing debugging information from the boot.ini file on the remote system BigBox:

```
bootcfg /debug off /id 1 /s BigBox /u awl\administrator /p password
```

bootcfg /default

The `bootcfg /default` command is used to specify an operating system listed in the boot.ini file as the default OS. The syntax for this command is:

```
bootcfg /default /id <OSNumber> [/s <system>] [/u <domain\user>]
[/p <password>]
```

As with other `bootcfg` command versions, the `/id` switch is used to specify the operating system line in the boot.ini file to designate as the default. So to make the third OS listed in a local system's boot.ini file the default OS, you would run `bootcfg /default /id 3`.

bootcfg /delete

This command is used to delete an operating system reference in the boot.ini file by removing its associated entry in the [operating systems] portion of the file. The syntax for `bootcfg /delete` is:

```
bootcfg /delete /id <OSNumber> [/s <system>] [/u <domain\user>]
[/p <password>]
```

When you run the command, you use the `/id` switch to indicate the OS reference line to delete, so to delete the second OS reference, you would run `bootcfg /delete /id 2`.

bootcfg /ems

This command allows you to change the redirection configuration of the Emergency Management Services (EMS) console (W2K3 servers) to a remote computer. When you run this command, a `redirect-Port#` entry is added to the [boot loader] section of the boot.ini file, and a `/redirect` switch to the specified operating system in the [operating systems] portion of the file.

Here is the syntax for `bootcfg /ems`:

```
bootcfg /ems <on|off|edit> /id <OSNumber> [/s <system>] [/u
<domain\user>] [/p <password>] [/port <COM Port>] [/baud <baud rate>]
```

The command options are described in Table 7-7.

Table 7-7. bootcfg /ems Command Options

Option	Use
`on`	Enables remote output for the OS line number specified with the `/id` switch. When this parameter is used, the `/redirect` switch is added to the OS number specified, and redirect settings are added to the [boot loader] file section based on the value you specify with the `/port` switch.
`off`	Disables EMS remote output on the OS specified.
`edit`	Allows you to change current EMS port settings (set with the `/port` switch) for an OS you specify.
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/port <COM Port>`	Indicates which COM port should be used for redirection. Valid `COM Port` values are `COM1`, `COM2`, `COM3`, `COM4`, or `BIOSSET`. When `BIOSSET` is used as the `COM Port` value, EMS gets the valid COM port to use from the system BIOS.
`/baud <baud rate>`	Indicates the baud rate to be used for redirection. Valid `baud rate` values are 9600, 19200, 38400, 57600, or 115200.

To use `bootcfg` to turn on and configure EMS (using COM1 at 115200 baud) on a W2K3 Server (first OS), you would run this command:

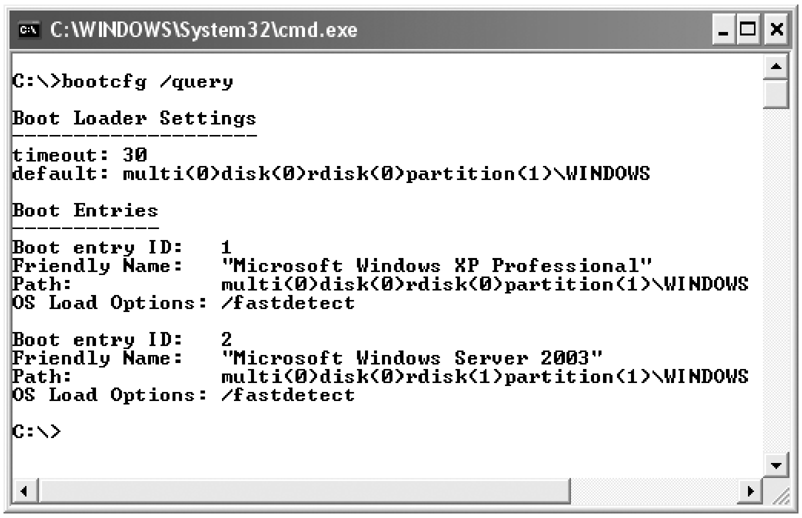
```
bootcfg /ems on /id 1 /port COM1 /baud 115200
```

bootcfg /query

The `bootcfg /query` command allows you to see the [boot loader] and [operating systems] configuration settings in the boot.ini file of a local or remote system. The syntax for this command is:

```
bootcfg /query [/s <system>] [/u <domain\user>] [/p <password>]
```

This is especially useful for troubleshooting since you can quickly check the boot.ini file settings on a remote system. An example of using `bootcfg /query` to check local boot.ini file settings is shown in Figure 7-2.



```
C:\>bootcfg /query

Boot Loader Settings
--------------------
timeout: 30
default: multi(0)disk(0)rdisk(0)partition(1)\WINDOWS

Boot Entries
------------
Boot entry ID:    1
Friendly Name:    "Microsoft Windows XP Professional"
Path:             multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
OS Load Options:  /fastdetect

Boot entry ID:    2
Friendly Name:    "Microsoft Windows Server 2003"
Path:             multi(0)disk(0)rdisk(1)partition(1)\WINDOWS
OS Load Options:  /fastdetect

C:\>
```

Figure 7-2. Checking local boot.ini settings with bootcfg /query

bootcfg /raw

This command is used to add text to the end of an operating system entry in the boot.ini file and replaces any text that had previously existed at the end of the entry. Think of the `/raw` switch as the "catchall" option, allowing you to add any other valid switches to the boot.ini file (shown in Table 7-1) that are not natively supported in the other `bootcfg` commands.

The syntax for `bootcfg /raw` is:

```
bootcfg /raw "<OptionString>" /id <OSNumber> [/s <system>]
[/u <domain\user>] [/p <password>]
```

The command options are described in Table 7-8.

Table 7-8. bootcfg /raw Command Options

Option	Use
`OptionString`	Specifies a string of options, in quotes, to list at the end of the operating system reference line.
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.

Here is an example of using `bootcfg /raw` to turn off serial port mouse detection (`fastdetect`) and enable boot logging on a system:

```
bootcfg /raw "/fastdetect /bootlog" /id 1
```

These options can be removed with the `bootcfg /rmsw` command, which we'll look at next.

bootcfg /rmsw

The `bootcfg /rmsw` command allows you to remove options associated with operating system entries in the boot.ini file. Here is the syntax for `bootcfg /rmsw`:

```
bootcfg /rmsw /id <OSNumber> [/s <system>] [/u <domain\user>]
[/p <password>] [/mm] [/bv] [/ng]
```

The `bootcfg /rmsw` options are described in Table 7-9.

Table 7-9. bootcfg /rmsw Command Options

Option	Use
`/id <OSNumber>`	This switch is required to tell the command which OS reference in the boot.ini file to apply to. Numbering for this switch begins at 1, so to apply the command to the second OS listed in the [operating systems] portion of the boot.ini file, you would enter `/id 2`. You can determine the OS–line number relationship by running `bootcfg /query`.
`/mm`	Removes the `/maxmem` switch from the specified OS reference.
`/bv`	Removes the `/basevideo` switch from the specified OS reference.
`/so`	Removes the `/sos` switch from the specified OS reference.
`/ng`	Removes the `/noguiboot` switch from the specified OS reference.

For an example of using this command, consider a scenario in which you have configured the boot.ini file to display device drivers as a system boots. After watching the system boot and verifying the problem, you replace the faulty driver and do not want all device drivers to be displayed as they load during startup. To stop driver names from appearing during startup, you run the command `bootcfg /rmsw /id 1 /so`.

bootcfg /timeout

When a system is dual booted, the user has a configured amount of seconds to choose an OS before the default OS is loaded. The waiting period is known as the timeout value, which can be modified with `bootcfg /timeout`.

The syntax for `bootcfg /timeout` is:

```
bootcfg /timeout <time> [/s <system>] [/u <domain\user>]
[/p <password>]
```

In the command syntax, `time` is the amount of seconds that you would like the boot menu to appear before the default OS is loaded. The default `time` value is 30 seconds. To change the value to 10 seconds, for example, you would run `bootcfg /timeout 10`.

dmdiag (ST)

The `dmdiag` command is used to quickly retrieve configuration information on a system's hard disks. You can run the command and have its output dumped to a file or displayed on the screen. One of the most useful features of this command is that it displays all the configured mount points on a system as well as any symbolic links on the system. If a computer's storage configuration is unknown to you, running `dmdiag` is a quick way to get brought up to speed.

Here is the complete list of what you will learn after running `dmdiag`:

- Drive letter usage
- A kernel list
- Logical Disk Manager (LDM) file versions
- The LDM size
- A listing of all physical disks and their disk type (basic or dynamic)
- The mount points on the system
- Partition configuration information
- Symbolic links
- The system name and OS version

Tracking down symbolic links can be a tricky process. With `dmdiag`, you can allow you coworkers to believe that you will endure a great deal of stress in locating all the links on a system, while in reality you can have the list dumped to a text file in seconds.

Here is the syntax for `dmdiag`:

```
dmdiag [/v] [/f <filename>]
```

The `dmdiag` command options are described in Table 7-10.

Table 7-10. dmdiag Command Options

Option	Use
`/v`	Used to provide verbose output, which displays all of the configuration data mentioned earlier in this section. Without this switch, very little information is displayed in the output.
`/f`	Causes command output to be dumped to a text file named by the `filename` parameter. If no filename is specified, the file will be called dmdiag.txt and will be placed in the folder from where the command was run.
`filename`	Used with the `/f` switch to provide a path and a filename for the `dmdiag` output file.

To use `dmdiag` to display detailed disk information for a system, you would run `dmdiag /v`. While `dmdiag` is relatively simple in its use and functionality, next you'll see a much more complex series of commands in `fsutil`.

fsutil

The File System Utility (FSutil) command, `fsutil`, is available in several different forms, all of which are covered in this section. This utility allows you to perform a multitude of troubleshooting and administrative tasks on volumes and storage devices. While some `fsutil` commands are very useful, others are specific in their purpose, but knowledge of them may allow you to quickly get out of a troubleshooting jam down the road. Each of the `fsutil` commands is fully explained in the next 11 sections.

Most of the system configuration modifications made by `fsutil` are changes to the Registry. In order for the changes to be applied, you need to reboot the system after running the `fsutil` command.

fsutil behavior

The `fsutil behavior` command allows you to check several FAT and NTFS volume configuration characteristics as well as modify them. Among the configuration settings that you can modify with `fsutil behavior` are

- Support of 8.3 filename conversion
- The last access timestamp for a volume

- Disk quota notification
- Paged pool memory
- The Master File Table size

From a troubleshooting and fault resolution perspective, the last two configuration options listed are the most important. When additional physical RAM is added to a system, Windows does not automatically increase the amount of paged pool memory available. For systems having performance lags resulting from the opening and closing of many files, using `fsutil behavior` to extend the operating system's amount of paged pool memory allocation may improve system performance.

The use of the MFT was explained earlier, in the section on Disk Architecture 101. With NTFS file systems, disk space is automatically allocated for future MFT growth, which is known as the MFT Zone. If the MFT Zone is not large enough, the MFT can grow beyond the disk space reserved for it in the MFT Zone, which would result in a fragmented Master File Table, thus hindering disk performance. This performance problem can be corrected by using `fsutil behavior` to modify a volume's MFT Zone allocation setting.

Now that you have seen some of the most important uses for `fsutil behavior`, let's look at its syntax.

```
fsutil behavior query <disable8dot3 | allowextchar | disablelastaccess
| quotanotify | memoryusage | mftzone>
fsutil behavior set {disable8dot3 <1 | 0> | allowextchar <1 | 0> |
disablelastaccess <1 | 0> | quotanotify <frequency> | memoryusage
<memvalue> | mftzone <zonevalue>}
```

The `query` command option allows you to check the setting of one of the available parameters, while you can use the `set` option to modify an existing setting. With the `set` command option, a value of 1 turns on the option, while a value of 0 turns it off. All other available command parameters and options are described in Table 7-11.

Table 7-11. fsutil behavior Command Options

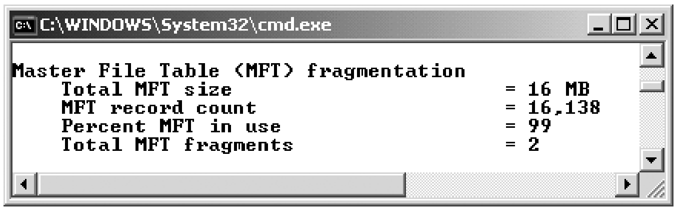
Option	Use
`disable8dot3`	Disables (1) or enables (0) creation of 8.3 character length file-names on FAT and NTFS volumes.
`allowextchar`	Enables (1) or disables (0) the use of characters from the extended character set in short file names on NTFS volumes.
`disablelastaccess`	Disables (1) or enables (0) the use of the last access timestamp for NTFS folders.
`quotanotify`	Allows you to set the frequency in which disk quota violations are written to the system event log.

continues

Table 7-11. fsutil behavior Command Options, continued

Option	Use
`<frequency>`	Used with the `quotanotify` parameter to set the time period in seconds (values of 0 to 4294967295 are valid) in which quotanotify events are written to the system event log. The default is 3600 (one hour).
`memoryusage`	Used to modify the internal cache settings for NTFS paged pool and non–paged pool memory, which may improve disk performance by changing the `memvalue` parameter to 2.
`<memvalue>`	This parameter has two allowable values: 1 (the default) and 2. When the parameter is set to 2, the size of NTFS memory thresholds and lookaside lists is expanded, and additional memory cache is available for file system read operations, thus improving disk performance at the expense of storage space.
`mftzone`	Used to change the volume's MFT Zone setting, which may prevent Master File Table fragmentation that would diminish disk performance. This parameter requires that a `zonevalue` be specified.
`<zonevalue>`	Used with the `mftzone` parameter to specify new MFT Zone configuration settings. Allowable values are 1 (the default) to 4. Each value increment represents one-eighth of the volume's allocated space, so by default, 12.5% of the volume is automatically allocated to the MFT Zone. Changing `zonevalue` to 4 would offer the best read performance, but at the cost of 50% of the available storage space.

When you suspect that MFT fragmentation is the source of a problem, don't just blindly increase the size of the MFT Zone. The easiest way to determine the true size of the MFT, see if it is fragmented, and determine if more disk space must be allocated, is to run the `defrag` command line utility (see Chapter 4) to analyze the volume. To see the MFT portion of the analysis, you must use the verbose (`-v`) switch in the command syntax. So to use `defrag` to check the MFT consumption and fragmentation information on the C drive, you would run `defrag -a -v c:`. The MFT portion of the command output is shown in Figure 7-3.



```
Master File Table (MFT) fragmentation
    Total MFT size                          = 16 MB
    MFT record count                        = 16,138
    Percent MFT in use                      = 99
    Total MFT fragments                     = 2
```

Figure 7-3. Using defrag to check for MFT defragmention

The command output shown in Figure 7-3 shows that the MFT is only consuming 16MB of space on the volume, so in this situation you could eliminate something that is right (MFT fragmentation) in a search for a disk performance problem.

fsutil dirty

With `fsutil dirty`, you can query or set a volume's dirty bit. In Chapter 4 in the section on `chkdsk`, it was mentioned that when a volume's dirty bit is set, `chkdsk` automatically runs the next time the system is restarted.

With `fsutil dirty`, you can query the status of or set the dirty bit for a volume. Here is its syntax:

```
fsutil dirty <query | set> <volumepath>
```

Here are two examples of using `fsutil dirty`:

- To check the dirty bit status on the D drive:
  `fsutil dirty query d:`
- To set the dirty bit and cause `chkdsk /f` to run at the next boot on the E drive:
  `fsutil dirty set e:`

fsutil file

The `fsutil file` command is a very versatile tool that allows you to

- List files by user name (if disk quotas are enabled).
- Create new files of any size for testing purposes (ideal for testing backup performance).
- Set a file's short name.
- Check allocated ranges for a file.

The command syntax for `fsutil file` is:

```
fsutil file createnew <filepath> <size>
fsutil file findbysid <username> <filepath>
fsutil file queryallocranges offset=<offset> length=<length>
<filepath>
fsutil file setshortname <filepath> <shortname>
fsutil file setvaliddata <filepath> <datalength>
fsutil file setzerodata offset=<offset> length=<length> <filepath>
```

As you can see, this command comes in several different versions. The options for each version are described in Table 7-12.

Table 7-12. fsutil file Command Options

Option	Use
`createnew`	Creates a new file (the contents consist of all 0s) with the name and size specified.
`filepath`	Used to specify the complete path to a file, folder, or volume.
`size`	Specifies the size of a file in kilobytes.
`findbysid`	Used to locate files owned by a specific user (only works on volumes with disk quotas enabled).
`username`	Used with the `findbysid` option to specify a user name.
`queryallocranges`	Reports the allocated ranges a file consumes on a volume. This is ideal for determining if a file has sparse regions.
`offset`	Used to indicate the start of the range to set to 0s.
`length`	Specifies the length of the range (in bytes).
`setshortname`	Used to set the short name (8.3) for a file.
`shortname`	Used to specify a file's short name for the `setshortname` parameter. It must follow the 8.3 standard.
`setvaliddata`	Used to configure the valid data length for a file on an NTFS volume.
`datalength`	Specifies the length of data (in bytes) for use with the `setvaliddata` parameter.
`setzerodata`	Used to fill a portion of a file with 0s.

For an example of using `fsutil file`, consider dealing with a user (suppose his name is Chong, who is a member of the Seventies domain) that saved an important file to the network share where disk quotas are enabled. The user needs to locate the file but does not remember what it was named. All that he remembers is that it was saved in the Docshare folder. To quickly locate the file, you can run this command:

```
fsutil file findbysid chong E:\Docshare
```

fsutil fsinfo

The `fsutil fsinfo` command allows you to quickly retrieve information on a system's drives. In particular, you can find out the following information by executing this command:

- A list of drives on the system
- The type of drives on a system (fixed disk, CD-ROM, and so on)
- Disk configuration information (sectors, clusters, MFT Zone parameters)

- Statistical data for a drive (metadata, MFT reads/writes)
- Volume-related information (file system type, disk quotas, Unicode support, case-sensitive filename support)

As with other `fsutil` commands, the `fsinfo` option allows you to quickly retrieve very specific information on a particular volume. The `fsutil fsinfo` syntax is:

```
fsutil fsinfo drives
fsutil fsinfo drivetype <volumepath>
fsutil fsinfo ntfsinfo <rootpath>
fsutil fsinfo statistics <volumepath>
fsutil fsinfo volumeinfo <rootpath>
```

The command parameters are described in Table 7-13.

Table 7-13. fsutil fsinfo Command Options

Option	Use
`drives`	Displays all drives on the system by their access path (drive letter or mount path).
`drivetype`	Displays the type of the drive for the drive specified (CD-ROM, fixed disk, and so on).
`ntfsinfo`	Displays NTFS configuration information (total sectors, total clusters, bytes per sector, bytes per cluster, and MFT Zone configuration).
`statistics`	Lists statistics on read and write data for the specified volume.
`volumeinfo`	Displays configuration information for the specified volume (see Figure 7-4).
`volumepath`	Used with the `drivetype` and `statistics` parameters to specify the path to a logical volume (drive letter, mount path, volume name).
`rootpath`	Used to specify the drive letter (followed by a colon) of a root drive.

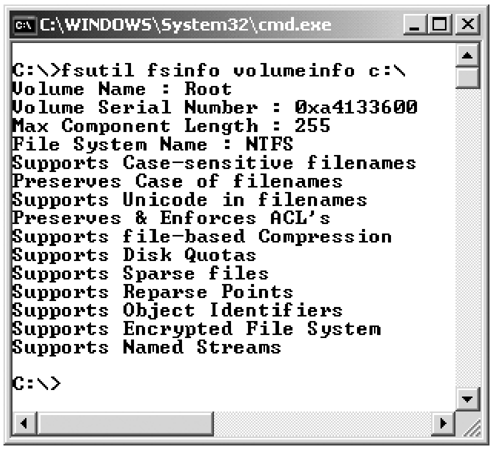
The `fsutil fsinfo` command is a valuable tool for gathering information. Next, you will see an `fsutil` tool that may serve as the perfect Band-Aid in a tough situation.

fsutil hardlink

Suppose that after you have moved files from one volume to another, an application that is hard-coded to find a file in a single location no longer works properly. This is where `fsutil hardlink` is handy. With this tool, you can create multiple logical files in different locations that all reference the same physical file. With hard links, users and applications accessing a particular file in one location can be transparently redirected to another location, where the actual file data is stored.

To create a hard link using `fsutil hardlink`, you would use the following syntax:

```
fsutil hardlink create <newpath> <existingpath>
```



```
C:\>fsutil fsinfo volumeinfo c:\
Volume Name : Root
Volume Serial Number : 0xa4133600
Max Component Length : 255
File System Name : NTFS
Supports Case-sensitive filenames
Preserves Case of filenames
Supports Unicode in filenames
Preserves & Enforces ACL's
Supports file-based Compression
Supports Disk Quotas
Supports Sparse files
Supports Reparse Points
Supports Object Identifiers
Supports Encrypted File System
Supports Named Streams

C:\>
```

Figure 7-4. Querying volume information with fsutil fsinfo volumeinfo

When you run this command, you must specify the path and a name for the new file that will link to the original file, along with the complete path and name of the original file. For example, suppose an application is looking for a file called payroll.dat that was originally located in the C:\data folder but was moved to the E:\accounting\data folder. The quickest was to resolve this problem would be to re-create the payroll.dat file in the C:\data folder as a hard link. This operation would require that you run the following command:

```
fsutil hardlink create C:\data\payroll.dat
E:\accounting\data\payroll.dat
```

fsutil objectid

The `fsutil objectid` command allows you to manage Object Identifiers (OIDs) associated with files, folders, and links. Both the Distributed Link Tracking (DLT) Client service and File Replication Service (FRS) track objects (files, folders, and links) by their associated OIDs, which are 16-byte (32-character) hexadecimal codes that uniquely identify an object on a volume. With `fsutil objectid`, you can create, delete, query, and set OID parameters. Since there is rarely ever an instance where you would need to modify an OID (Microsoft even recommends only doing so at your own risk!), simply keep the usage of this tool in the back of your mind.

Here is the command syntax:

```
fsutil objectid <create | delete | query> <volumepath>
fsutil objectid set <ObjectID> <BirthVolumeID> <BirthObjectID>
<DomainID> <volumepath>
```

These command options are fully described in Table 7-14.

Wolf.C07.qxd 5/28/03 3:11 PM Page 233

Table 7-14. fsutil objectid Command Options

Option	Use
`create`	Creates an OID for the file or folder specified. If one already exists, then this command acts like the `query` command.
`delete`	Deletes a file or folder's OID.
`query`	Displays a file or folder's OID.
`set`	Modifies a file or folder's OID.
`volumepath`	Specifies the complete path to a file, folder, or link.
`ObjectID`	Used with the `set` parameter to provide a specific 32-hex-character OID.
`BirthVolumeID`	Used with the `set` parameter to provide the OID for the volume where the object originally resided.
`BirthObjectID`	Used with the `set` parameter to provide the original OID for the object.
`DomainID`	Used with the `set` parameter to provide the domain OID for the object. This should be 32 0s.

Here is an example of checking the OID for the Windows folder on the C drive:

```
fsutil objectid query C:\Windows
```

Microsoft strongly warns against deleting or modifying Object IDs. Doing so may result in the loss of file data or an entire volume. Modifying an Object ID may also cause problems with the DLT Client service and FRS.

fsutil quota

The `fsutil quota` command allows you to manage disk quota configuration from the command line, as opposed to using Windows Explorer. Suppose that a user is unable to save documents to his home folder, which gets backed up nightly on a network server. You suspect that the user's problem may be the result of his exceeding his quota limit. You can quickly verify your suspicions by running `fsutil quota`.

The syntax for `fsutil quota` is:

```
fsutil quota disable <volume>
fsutil quota enforce <volume>
fsutil quota modify <volume> <threshold> <limit> <domain\user>
fsutil quota query <volume>
fsutil quota track <volume>
fsutil quota violations
```

The options and usage for each `fsutil quota` command version are described in Table 7-15.

Table 7-15. fsutil quota Command Options

Option	Use
`disable`	Disables disk quotas on the specified volume.
`enforce`	Enforces disk quota limits on the specified volume.
`modify`	Changes an existing disk quota or creates a new quota for the specified volume.
`query`	Displays disk quota information for the specified volume.
`track`	Tracks disk usage on the specified volume.
`violations`	Checks the local system and application logs, and displays a list of users that have exceeded quota limits or have reached their quota threshold.
`volume`	Used to indicate the volume to manage.
`threshold`	Used with the `modify` parameter to set a limit (in bytes) at which a user is warned that he or she is approaching the quota limit.
`limit`	Used with the `modify` parameter to set a maximum amount of disk space (in bytes) that can be used by the user.
`domain\user`	Used with the `modify` parameter to specify a user account to which the quota entry applies.

In the example mentioned earlier, you can check for quota violations with `fsutil quota` by running `fsutil quota violations`. If the user had exceeded the quota limit, you could either tell the user to remove some files from the share to free up space, or you could create a custom quota entry for the individual user by running `fsutil quota modify`.

fsutil reparsepoint

The easiest way to think of reparse points is to compare them to shortcuts on your desktop. A shortcut is not a true executable file but is a pointer to one. The same could be said for reparse points, since they provide shortcuts to objects stored at other locations. When you click on a shortcut link, you are transparently redirected to wherever the shortcut points. This is also true with reparse points. A file that contains a reparse point may redirect users and applications to another file or directory without their knowledge.

Reparse points can be tricky in terms of getting rid of them. The mistake that some users make is to delete them right from Windows Explorer. When this is done, sometimes not only is the reparse point deleted, but the actual file or folder that is referenced by the reparse point is deleted as well. This is why `fsutil reparsepoint` is so useful. This tool allows you to query reparse points, letting you determine what true object they reference, and also allows you to cleanly delete reparse points.

The syntax for `fsutil reparsepoint` is:

```
fsutil reparsepoint <query | delete> <filepath>
```

The three options for this command are described in Table 7-16.

Table 7-16. fsutil reparsepoint Command Options

Option	Use
`query`	If a reparse point exists for the file or folder in the specified path, information on its data length, tag value, and GUID is displayed.
`delete`	Deletes a reparse point associated with the file or folder in the specified path.
`filepath`	Used to specify the path to a file or folder containing a reparse point.

The most common use of `fsutil reparsepoint` is to delete reparse point references from files. This may be needed for applications that leave behind reparse points or even from Windows services such as Remote Storage Service (RSS) or Distributed File System (DFS) that may also not cleanly remove reparse point references if they become corrupt. For example, after disabling DFS on a member server in your domain that previously had the role of a standalone DFS root, you notice that when accessing a folder, you are still being redirected to another system. To remove the reparse point from the folder, you run this command:

```
fsutil reparsepoint delete D:\public\sales
```

fsutil sparse

Sparse files are a file system improvement that first appeared in Windows 2000. The idea of sparse files is to save some of the disk space consumed by very large files. Imagine if you had to read a book to someone that contained only 1s and 0s, much as a computer reads data off a hard drive. Suppose that the data in the file being read consisted of a single 1 followed by one million 0s. Instead of writing 100000000 (to spare you the pain of reading many more pages of 0s, we'll stop here), wouldn't it be much easier to simply write 1 [one million more 0s]? This is what sparse files do. Nonmeaningful data (large strings of consecutive 0s) is not allocated as disk space and instead is simply referenced. With sparse file support, not only is disk space saved, but read and write operations run faster as a result.

With `fsutil sparse`, you can

- Identify sparse files.
- Mark a file as a sparse file.
- Scan a file, looking for its ranges of nonzero data.
- Fill a portion of a file with 0s.

The `fsutil sparse` syntax is:

```
fsutil sparse <queryflag | queryrange | setflag | setrange> <path>
[offset] [length]
```

The `fsutil sparse` options are explained in Table 7-17.

Table 7-17. fsutil sparse Command Options

Option	Use
`queryflag`	Determines if a file is set as a sparse file.
`queryrange`	Displays ranges in sparse file that contain nonzero data.
`setflag`	Sets a file to be a sparse file.
`setrange`	Fills the designated sparse file with a range (specified with the `offset` and `length` parameters) with 0s.
`path`	Specifies the complete path to the sparse file. If the path contains spaces, include the path in quotes.
`offset`	Specifies the point within the file to mark as sparse (the beginning of a continuous string of 0s).
`length`	Specifies the length (in bytes) of the region to be marked as sparse.

To set the file E:\Data\Records.dat as a sparse file, you would run the following command:

```
fsutil sparse setflag e:\data\records.dat
```

To then view the file's ranges of nonzero data, you could run this command:

```
fsutil sparse queryrange e:\data\records.dat
```

fsutil USN

The `fsutil USN` command is used to query and manage the NTFS Change Journal on a volume. If you are unaware of the Change Journal, it may be because it is a relatively new (W2K and above) and underpublicized Windows feature. Primarily, the Change Journal is most useful to backup applications mainly because it can streamline the incremental backup process. In the pre–Windows 2000 days, when an incremental backup was performed, the entire volume was scanned, and a list of files that had changed since the last backup was accumulated. This list was the result of either checking the archive bit on each file or checking a file's date-time stamp and comparing it with the time of the last backup. Regardless of how the list was obtained by the application, the problem with this process was that it was both resource intensive and time consuming, neither of which are conditions you wish to see on any application running on your IIS Web server, for example. The other problem with archive bit and date-time stamp scanning was that when a file was renamed or had its access control list (ACL) modified, neither would flip the archive bit or change its date-time stamp. This meant that it would not get backed up with an incremental backup.

All of these problems were remedied with the Change Journal. With the Change Journal, the operating system maintains its own file index and records when changes are made to a file through the use of update sequence numbers (USNs). Even changes to a file's name or ACL are noted in the Change Journal. Backup applications that are truly W2K or higher compliant can interface with the Change Journal. This means that on incremental backups, they do not need to scan the entire volume and instead can simply formulate a list of what needs to be backed up by reading the Change Journal, saving both time and resources. Also, in being able to note changes in filenames and ACLs, incremental backups from Change Journal–aware applications are more accurate.

Besides backup applications, the following Windows services also interact with the Change Journal:

- File Replication Service
- Indexing Service
- Remote Installation Service
- Remote Storage Service

In particular, `fsutil USN` allows you to perform the following administrative tasks on a volume's Change Journal:

- Create a new Change Journal
- Modify an existing Change Journal
- List Change Journal entries between a low and high USN value
- Query general Change Journal information (capacity, record information)
- Display USN data for a given file

These tasks can be performed using the following syntax:

```
fsutil USN createjournal <MaxSize> <AllocationDelta> <VolumePath>
fsutil USN deletejournal <flags> <VolumePath>
fsutil USN enumdata <FileRef> <LowUSN> <HighUSN> <VolumePath>
fsutil USN queryjournal <VolumePath>
fsutil USN readdata <FilePath>
```

As you can see, there are several variations of this command. The options for each command version are explained in Table 7-18.

Table 7-18. fsutil USN Command Options

Option	Use
`createjournal`	Creates a new Change Journal on the volume specified. If one already exists, the Change Journal is modified with the parameters specified in the command.
`Maxsize`	Used to indicate the maximum size (in bytes) to allocate on the volume for the Change Journal.
`AllocationDelta`	Used to specify the amount of data (in bytes) to be removed from the beginning and added to the end of the Change Journal. In other words, once the Change Journal reaches its maximum size, how much of the oldest data should be purged to make room for new data.
`VolumePath`	Used to specify the drive letter or mount path to a particular volume.
`deletejournal`	Used to disable an active Change Journal.
`flags`	Two flag options are available for the `fsutil USN deletejournal` command. • `/d`—Disables the active Change Journal, with I/O control returned to the system before the operation completes. • `/n`—Disables the active Change Journal, with I/O control returned to the system after the operation completes.
`enumdata`	Itemizes and lists Change Journal data between two specified points.
`FileRef`	Specifies the ordinal file position at which itemization is to begin.
`LowUSN`	Specifies the lower boundary of USN records in the Change Journal that are returned in the command output.
`HighUSN`	Specifies the upper boundary of USN records in the Change Journal that are returned in the command output. Files with USNs equal to or between the `LowUSN` and `HighUSN` values are returned by the command.
`queryjournal`	Used to display Change Journal configuration information, including the first USN, the maximum journal size, and the allocation delta.
`readdata`	Used to display the USN data for a single file.
`Filepath`	Used to specify the complete path to a file to be checked with the `fsutil USN readdata` command.

While this command certainly is complex, there are still troubleshooting purposes for it. Suppose that you do not feel that all of your changed files on a system's D drive are being backed up with an incremental backup. It is possible that the maximum size allocated for the Change Journal is not large enough. In this event, you would either run incremental backups more frequently or increase the size of the Change Journal. Suppose that you wanted to increase the size of the Change Journal to 500MB, to ensure that it will not run out of space, and reallocate old space in increments of 5MB. Before running the command, you would first need to convert both values from megabytes to bytes, which are 524,288,000 and 5,242,880, respectively (1MB = 1,048,576 bytes). You would use these values in the following command:

```
fsutil USN createjournal 524288000 5242880 D:
```

fsutil volume

The `fsutil volume` command allows you to dismount a volume or to check the amount of free space on the volume. The syntax for `fsutil volume` is:

```
fsutil volume <diskfree | dismount> <drivename | volumepath>
```

The command syntax requires that either `diskfree` or `dismount` be specified, along with a drive name or a volume path. These options are further explained in Table 7–19.

Table 7-19. fsutil volume Command Options

Option	Use
`diskfree`	Used to query the amount of free space on the drive or volume specified.
`dismount`	Used to dismount the drive or volume specified.
`drivename`	Used to specify the logical drive on which to run the command.
`volumepath`	Used to specify the logical path to a mount point or volume name representing a logical volume.

The `diskfree` option is ideal for checking for space on a volume when you suspect that it may be out of space. The `dismount` option provides a way to quickly terminate any open processes or user sessions currently accessing the volume. Dismounting a volume may be necessary if you are attempting to run a maintenance application on the volume, and the application cannot continue due to a conflicting process accessing the volume. Dismounting the volume should correct the problem, and the application should execute cleanly the next time you attempt to start it. For example, to dismount the logical volume E, you would run this command:

```
fsutil volume dismount e:
```

ftonline (ST)

While Windows 2000 supported fault-tolerant volumes on basic disks for backward compatibility with Windows NT, Windows XP/W2K3 does not. Prior to upgrading a system to XP/W2K3, you should upgrade any fault-tolerant basic disks to dynamic disks or perform a backup so that you can restore the data to another volume. If a user does not perform either of these tasks and simply upgrades his or her system, then the user will be in a bit of trouble, since the fault-tolerant volume will no longer be accessible. That's where `ftonline` comes into play.

When an upgraded system first boots into an XP/W2K3 OS, fault-tolerant basic disks appear in the Disk Management UI as failed and are not accessible. With `ftonline`, you can temporarily mount and bring the failed fault-tolerant basic disks online so that you can copy or back up their data. Once you reboot the system, the fault-tolerant basic disks will again show a failed status and should be deleted and reconfigured.

Here is `ftonline`'s syntax:

```
ftonline <Driveletter>
```

To see an example of `ftonline` in use, consider an example of a user upgrading his system to Windows XP Professional. The system was upgraded a few years ago from Windows NT to Windows 2000 and contained a fault-tolerant basic disk. When the user complains about lost data access to his E drive after the upgrade, you perform these steps:

1. Run `ftonline E:` on the user's system.
2. Back up the data on the online E drive.
3. Reboot the system.
4. Use Disk Management to delete the fault-tolerant basic disk.
5. Re-create the disk as either a new basic or dynamic disk.
6. Restore the backup data to the new volume.

recover

In Chapter 4, it was mentioned that `chkdsk` can be used to spot and repair bad sectors on a disk. Sometimes sectors cannot be repaired, which often results in a file spanning those sectors being considered lost as well. However, with the `recover` command you can get back the portions of a file that remain on the good sectors of a disk. If the file is an important document and no good backup exists, this command provides you with an alternative with which you can save at least a portion of the lost data.

The `recover` command only works at the filename level and does not support the use of wildcards, so this command can only be executed on a single file at a time. The syntax for `recover` is:

```
recover [drive:] [path] filename
```

When the command is run in the directory in which the corrupted file resides, only the filename needs to be included in the command syntax. Otherwise, you can list the complete path to the filename in the command syntax. For example, if the Word document D:\Free Software\Penguin.doc mysteriously became corrupted on your Windows system, you could run the following command:

```
recover "C:\Free Software\Penguin.doc"
```

to try to get back at least portions of the document. When a large document is lost, most users will agree that anything is better than nothing. After running `recover` on a corrupted file, you many not have much, but at least to the user's delight, you will not have "nothing" either.

Diagnosing and Repairing Boot Sector and MBR Problems

Earlier in this chapter, you learned of the differences between the boot sector and the MBR. On most occasions, it is very easy to spot an MBR problem or a boot sector problem. When there is a problem with the MBR, you will see one of the following messages at startup:

- "Error loading operating system"
- "Invalid partition table"
- "Missing operating system"

If the MBR is good, it will point to the boot sector at startup. If it cannot find the boot sector, one of the following error messages will be displayed:

- "A disk read error occurred"
- "NTLDR is compressed"
- "NTLDR is missing"

Before attempting to repair either an MBR or a boot sector error, first attempt to boot the system with a recent scan disk from your antivirus software. With some MBR and boot sector viruses, the Windows repair tools that will be discussed shortly may cause more harm to the partition tables, so checking for viruses should always be your first course of action.

To resolve an MBR error, the quickest course of action is to follow these steps.

1. Boot the system into the Recovery Console and run `fixmbr` from the command prompt.
2. When warned, press Y to continue.
3. Restart the system.

To repair the boot sector, you would take the following steps.

1. Boot the system into the Recovery Console and run `fixboot <drive letter>:` from the command prompt, specifying the drive letter that contains the system volume.
2. Restart the computer.

While under nearly all circumstances `fixboot` and `fixmbr` will resolve a startup problem, they are not guaranteed. If your startup problem cannot be repaired with `fixboot` or `fixmbr`, then you should do the following.

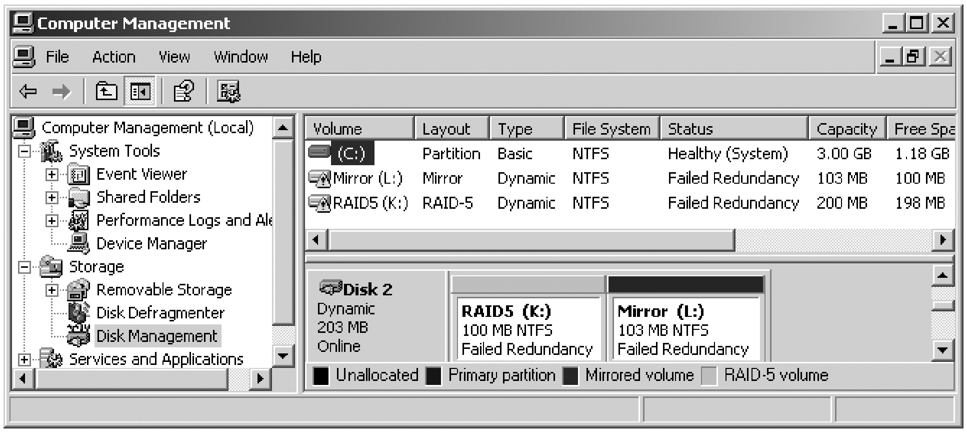
1. Repartition and format the affected drive.
2. Reinstall Windows (if your backup software does not support bare metal restores).
3. Restore necessary data from backup.

This section has shown you a systematic approach for resolving startup problems with the MBR or boot sector. Next you will see how to fix storage problems using the Disk Management tool.

> When you see disk errors at startup, always make sure that there isn't a floppy in the disk drive before spending minutes or even hours troubleshooting a startup problem. This problem runs rampant among users, and the solution to "Remove the floppy from your drive and hit the reset button" has been uttered millions of times by help desk professionals.

Repairing Failed Disks with Disk Management

Disk Management allows you to not only spot disk problems on a system, but repair them as well. In Figure 7-5, the Disk Management display for a system with two disk errors is shown. In the illustration, you can see that Disk Management has placed warning symbols



Volume	Layout	Type	File System	Status	Capacity	Free Spa
(C:)	Partition	Basic	NTFS	Healthy (System)	3.00 GB	1.18 GB
Mirror (L:)	Mirror	Dynamic	NTFS	Failed Redundancy	103 MB	100 MB
RAID5 (K:)	RAID-5	Dynamic	NTFS	Failed Redundancy	200 MB	198 MB

Disk 2
Dynamic
203 MB
Online

RAID5 (K:)
100 MB NTFS
Failed Redundancy

Mirror (L:)
103 MB NTFS
Failed Redundancy

Figure 7-5. Troubleshooting disk faults with Disk Management

next to the two disks that have been labeled as having "failed redundancy," allowing you to quickly spot the faulty disks. In the lower right corner of the display, you can see that the failed redundancy problems with the L and K volumes are both associated with Disk 2, which allows you to quickly diagnose Disk 2 failure as being the likely source of the problem.

Aside from failed redundancy, you may see several other disk status indications listed in Disk Management. Each status is described in Table 7-20.

Table 7-20. Disk Status Indications

Status	Cause	Corrective Action(s)
Foreign	A dynamic disk from one system has been added to another. The OS recognized that a signature resides on the disk, but has no record of it.	Right-click the disk and select Import Foreign Disks.
Missing	A dynamic disk is disconnected, not powered, or corrupted.	1. Repair the hardware or communication problem. 2. Right-click the disk and select Reactivate Disk.
Not Initialized	A disk does not have a valid signature in the MBR or GUID in the GPT.	Right-click the disk and select Initialize Disk.
Offline	A disk is inaccessible due to loss of power, communication, or corruption.	1. Repair the hardware or communication problem. 2. Right-click the disk and select Reactivate Disk.
Online	Normal operation.	None
Online (Errors)	The OS has detected I/O errors with the disk.	1. Run `chkdsk /f` to repair any physical errors to the disk. 2. Check for communication or power problems. 3. Right-click the disk and select Reactivate Disk.
Unreadable	• A dynamic disk's Dynamic Disk Database is corrupted. • A disk is spinning up. • Corruption, hardware failure, or I/O errors have occurred.	1. Repair the problem (if necessary). 2. Right-click the Disk Management icon and select Rescan Disks.

When dynamic disks are configured in a software RAID-based solution, you may encounter other errors. These errors are listed in Table 7-21.

Table 7-21. Dynamic Disk with RAID Configuration Errors

Status	Cause	Corrective Action(s)
Failed	A disk is damaged or corrupted.	1. Repair the hardware or communication problem. 2. Right-click the disk and select Reactivate Disk. 3. If the volume is not listed as healthy, right-click the volume and select Reactivate Volume.
Healthy	Normal operation.	Nothing.
Healthy (At Risk)	I/O errors are continually occurring.	1. Run `chkdsk /f` to repair any physical errors to the disk. 2. Check for communication or power problems. 3. Right-click the disk and select Reactivate Disk.
Healthy (Unknown Partition)	The OS cannot recognize the System ID (if MBR disk) or GUID (if GPT disk).	1. Delete the partition. 2. Re-create the unused space as a new volume. 3. Format the volume.
Unknown	A corrupted boot sector exists.	1. Scan for viruses. 2. Run fixboot from the Recovery Console. 3. If steps 1 or 2 do not fix the problem, repartition and format the disk, and restore lost data from backup.

Now that you have seen how to repair the immediate errors, let's not lose sight of the big picture. With fault-tolerant dynamic disk configurations, the loss of one disk might affect several disks. The remainder of this section focuses on the steps necessary to repair dynamic volumes spanning two or more physical disks.

Failed Spanned or Striped Volumes

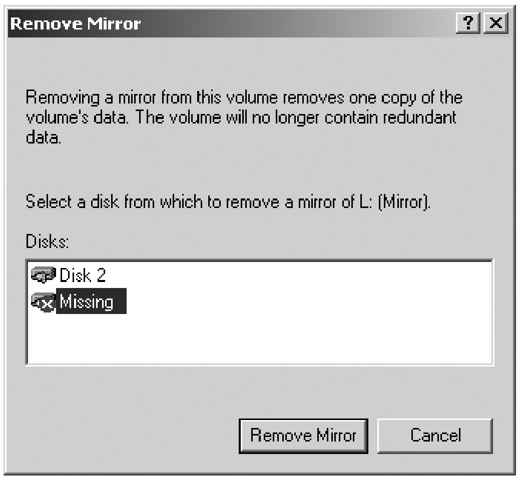
Neither spanned volumes nor striped volumes are fault tolerant. If a disk in one of these logical volumes fails, then you are faced with re-creating the volume from scratch and restoring its most recent data from backup. From Disk Management, you need to right-click the failed volume and select Delete Volume. From that point, you can replace the failed disk, and re-create the new volume and restore from backup. So that lightning does not strike twice, you may want to consider creating the new volume using a fault-tolerant configuration such as RAID 0 or RAID 5.

Failed Mirrored Volumes (RAID 1)

If a disk in a mirrored volume must be replaced, you need to follow these steps

1. Power down the system (if necessary) and replace the failed disk. Then power up the system (if the disk was not hot-swappable).
2. Right-click the failed mirrored volume and select Remove Mirror.
3. When prompted that the volume will no longer contain redundant data, select the missing disk, as shown in Figure 7-6, and click Remove Mirror.
4. When asked if you are sure you want to remove the mirror, click Yes.
5. Right-click the working disk from the original mirrored volume and select Add Mirror.
6. Select the newly installed disk (or an existing disk with equal or more free space) to add to the new mirror and click Add Mirror.

At this point, data from the original disk will be copied to the new disk in the mirrored volume. When the copy is finished, the mirrored volume will show a status of Healthy.


Remove Mirror

Removing a mirror from this volume removes one copy of the volume's data. The volume will no longer contain redundant data.

Select a disk from which to remove a mirror of L: (Mirror).

Disks:
- Disk 2
- Missing

Figure 7-6. Removing the mirror from a failed RAID 1 volume

Failed RAID 5 Volumes

If a disk in a RAID 5 volume must be replaced, you need to take these steps.

1. Power down the system (if necessary) and replace the failed disk. Then power up the system (if the disk was not hot-swappable).
2. Right-click the failed RAID 5 volume and select Repair Volume.
3. Select an available disk as the replacement for the failed disk and click OK.

Windows at this point will regenerate the failed volume. Once it completes, the RAID 5 volume will return to a healthy status.

If you're looking for more disk management and troubleshooting tools, check on the suite of products from Executive Software that are located on the companion CD and documented in Appendix C.

Common Stop Messages

This section shows you the most common disk-related stop messages. Although the notorious blue screen of death (BSOD) certainly makes far fewer appearances on XP/W2K3 systems, it does not disappear altogether. Actually, from a troubleshooting perspective, a BSOD can be your best friend. That is because the stop error that appears on the screen may lead you directly to the problem. The five most common XP/W2K3 disk-related stop error messages are examined in the next five subsections.

Stop 0x00000024 (NTFS File System) or Stop 0x00000023 (FAT File System)

This error tells you that a problem was encountered with the Ntfs.sys file for NTFS file systems or with the File Allocation Table on a FAT file system. This normally indicates that some type of corruption has occurred on the disk. At this point, you should first run `chkdsk <driveletter:> /f` on the system volume. If you are unable to start Windows in order to access the command prompt, then boot into the Recovery Console by booting the system on the Windows installation CD, and run `chkdsk` from there. If physical corruption was found on the disk and resolved by `chkdsk`, your troubles may be over. However, if this is not the first time that the system has experienced this problem, you need to dig a little deeper.

If the problem is repetitive, failing or malfunctioning hardware should be the next components to check. The most common hardware issues related to this type of error are

- A malfunctioning IDE controller
- A bad IDE cable

- A malfunctioning SCSI controller
- A bad SCSI cable
- Improper SCSI termination

Also, don't automatically rule out software as possibly causing the corruption. If software was recently installed on the system, that should be the first place you look. If the following application types are not compatible with the Windows OS version, disk corruption may result:

- Disk defragmenters
- Virus scanners
- Backup utilities

Stop 0x00000050 (Page Fault in Nonpaged Area)

The stop error occurs when requested data cannot be found in memory. This type of page fault is caused by a problem with system memory and is not directly related to a hard disk fault. If RAM was recently added, then that is the likely problem. If this is not the case, then you need to isolate the physical memory that is the cause of the problem. Don't limit the possible faults to this problem to only system RAM. This paging error can also be caused by any of the following being bad:

- Main memory
- L2 cache
- Video RAM

The easiest way to isolate the problem is to replace what is suspected to be bad with a known good component. To eliminate bad video RAM as the cause, replace the video card with a known good card and see if the error no longer occurs. A problem with onboard cache can be isolated by disabling the cache in the system BIOS. After you do this, the system will run noticeably slower, which is normal for a system operating without the use of onboard cache. If the stop error disappears, you will most likely need to replace the motherboard to resolve the problem. Since most organizations have plenty of RAM on hand, whether stored or in other systems, you can quickly replace RAM chips to see if the problem is corrected. If several sticks of RAM are in the system, replace one stick at a time and then reboot the system to see if the problem disappears. With a little patience, you can easily pinpoint the failed hardware that caused this problem.

As with other memory problems, you should not give software a free pass on this problem either. If the hardware checks out OK, or if a new application was recently installed, you should first look at the software. Besides applications, check system services and drivers. This stop error has also been known to appear as a result of the wrong driver being used. For example, an incorrect video driver may cause a video RAM–related paging error. The driver can be updated by running Windows Update, or you can check with the hardware manufacturer for the latest driver.

Stop 0x00000077 (Kernel Stack Inpage Error)

This error message alerts you that a page of kernel data requested from virtual memory could not be found. Since virtual memory is stored on the hard disk, the disks that store virtual memory are where you should focus your troubleshooting efforts. These are the main culprits for this failure.

- Disk corruption—Fix by running `chkdsk /f`.
- Disk hardware failure—Check all of the disk hardware that was mentioned in the section on stop 0x00000024.
- Virus infection—Download the latest virus signatures and run a virus scan on the volume.

Oftentimes, the status code that is listed in the stop error message points you directly to the problem. Table 7-22 lists the most common status codes along with their related problem.

Table 7-22. Stop 0x00000077 Status Codes

Code	Related Problem
0xC000009A (Status Insufficient Resources)	Depleted nonpaged pool resources (very rare).
0xC000009C (Status Device Data Error)	Bad sectors on hard disk.
0xC000009D (Status Device Not Connected)	Loose or disconnected power or data cables, controller or disk configuration problem, or SCSI termination problem.
0xC000016A (Status Disk Operation Failed)	Bad sectors on hard disk.
0xC00000185 (Status IO Device Error)	Loose or disconnected power or data cables, controller or disk configuration problem, SCSI termination, or multiple devices are attempting to access the same resources.

For hardware-related problems, you again need to test and replace the defective component. If disk corruption is indicated, first attempt to fix the problem by running `chkdsk /f` on the bad volume. If disk corruption is excessive, you then need to replace the defective disk.

Stop 0x0000007A (Kernel Data Inpage Error)

This error code indicates that a page of kernel data could not be located in virtual memory (the pagefile). The problems that cause this error are very similar to the ones associated with the previous stop error (0x00000077). With this stop error, you will see the same status codes that were previously listed in Table 7-22. These status codes will give you the information you need to quickly isolate the problem.

The primary culprits behind this error are

- Bad sectors on disk
- Memory hardware (RAM, L2 cache, video RAM)
- Disk cables
- SCSI termination
- A disk controller
- Virus infection

As was mentioned earlier, the easiest way to isolate memory hardware problems is to replace what is suspected to be bad with a known good card or chip from another system. With onboard cache, simply disable it in the BIOS and see if the problem returns.

Stop 0x0000007B (Inaccessible Boot Device)

This error indicates that the operating system has lost access to either the system or boot partition during system startup. This stop error is generally the result of an incorrect storage device driver being installed. For example, after you upgrade the driver for a system's SCSI host bus adapter (HBA) card, this error may appear when the system restarts. If you added new storage devices, such as additional SCSI hard disks to a SCSI bus, the SCSI IDs assigned to the new disks may conflict with existing settings in the boot.ini file. For example, assume that a system's boot partition was located on a SCSI disk with SCSI ID 3, with no other disks attached to the SCSI adapter with BIOS enabled. The operating system's entry in the boot.ini file would look similar to `multi(0)disk(0)rdisk(0)partition(1)`. If the new SCSI disk's SCSI ID was lower than 3, then you would get an Inaccessible Boot Device error. To correct the problem, you could take these actions.

- Edit the operating system entry in the boot.ini file to read `multi(0)disk(0)rdisk(1)partition(1)` (the `rdisk` value is incremented to signify the second logical SCSI disk in the chain).
- Change the SCSI ID of the new disk to a value higher than that of the original disk (ideally 4–6 in this situation, since 7 is normally reserved for the SCSI adapter).

When a boot device is inaccessible, don't rule out taking the error literally. Perhaps a cable is loose or disconnected, the disk's controller is bad, or the disk is corrupted. As always, a virus may be lurking and causing the problem as well. For hardware connection–related troubleshooting, a quick check to see if a physical connection exists is to use the BIOS menu. Use the system BIOS to autodetect IDE drives, and access the SCSI adapter's BIOS to detect the presence of SCSI drives.

You have just seen the most common disk-related stop error messages. Remember, BSODs are not bad by any stretch. In fact, get excited when you see them. They have just made it much easier for you to solve a problem. As always, *never* just turn off the system in disgust when you see the BSOD on your way out the door on a Friday night. Instead, always remember to jot down the stop error message first.

Summary

In this chapter, you were exposed to the tools and techniques for troubleshooting and repairing Windows disk problems. While some of the problems ranged from obvious to obscure, one theme was consistent throughout many of the problem resolution steps. That theme was to restore from backup. This step assumes that the system in question actually was backed up. If a system stores important data, it should be backed up frequently (at least nightly). Sometimes, rebuilding a disk from scratch and restoring data from a backup copy is your only option in correcting a problem. Don't take this option out of the equation when trying to fix a problem. With today's backup networks and Storage Area Networks (SANs), sometimes performing a restore to a previous point in time may be faster than the entire troubleshooting process. Since the mission of the company is the primary motivation for resolving the problem, always go with the resolution that will get the system back up and running the fastest. Also, when time permits, test your backup data by restoring it to a test system. Your job may depend on that backup data, and too many administrators (yours truly, included) have learned the hard way not to instinctively trust backups. Don't make this mistake too.

In the next chapter, you will begin a march through troubleshooting core Microsoft network infrastructure services, starting with DNS.